Evelize Amaral dos Santos
Leonardo Camargo
Luana Boganika
Ricardo Hillbrecht

Projeto Intranet Acadêmica
Módulo Estágio
Módulo Egressos

Curitiba
2006

Universidade Federal do Paraná
Evelize Amaral dos Santos
Leonardo Camargo
Luana Boganika
Ricardo Hillbrecht

Projeto Intranet Acadêmica
Módulo Estágio
Módulo Egressos

Trabalho de Conclusão de Curso apresentado à obtenção do grau de Tecnólogo em Informática, no Curso de Tecnologia em Informática da Universidade Federal do Paraná.

Orientador: Prof.° Mauro José Belli.

MG
004.6
5237

Curitiba
2006

SUMÁRIO

Resumo I

Abstract II

1.0 Introdução 01

1.1 Justificativa 02

1.2 Metodologia 03

2.0 Módulo Estágio 05

2.1 Modelagem do Sistema – Diagramas de Casos de Uso 05

2.1.1 Visão Coordenação de Estágios UFPR 05

2.1.2 Visão Empresa / Visão Aluno 06

2.2 Modelagem de Dados 07

2.2.1 Diagrama de Classes – Módulo Estágio 07

2.2.2 Modelo Físico – Diagrama Relacional – Módulo Estágio 08

3.0 Módulo Egressos 09

3.1 Modelagem do Sistema – Diagrama de Casos de Uso 09

3.1.1 Visão Coordenação do Curso / Visão Egresso 09

3.2 Modelagem de Dados 09

3.2.1 Diagrama de Classes – Módulo Egressos 10

3.2.2 Modelo Físico – Diagrama Relacional – Módulo Egressos 11

4.0 Conclusão 12

5.0 Referências Bibliográficas 13

6.0 Apêndices

6.1 Apêndice 1 - Plano de Projeto 01

6.1.1 Introdução 02

6.1.1.1 Escopo e Propósito do Documento 02

6.1.1.2 Objetivos 02

6.1.1.2.1 Funções Principais 03

6.1.2 Estimativas de Projeto 04

6.1.2.1 Dados históricos usados nas estimativas 04

6.1.2.2 Técnicas de estimativas 04

6.1.3 Riscos do Projeto 04

6.1.4 Recursos do projeto 06

6.1.4.1 Pessoal 06

6.1.4.2 Hardware e Software 06
6.1.4.2.1 Hardware 06
6.1.4.2.2 Software 06
6.1.5 Organização do Pessoal 07
6.1.5.1 Estrutura da Equipe 07
6.1.6 Cronograma – Módulo Estágio e Egressos 08
6.2 Apêndice 2 - Modelagem do Sistema 01
6.2.0 Módulo Estágio 02
6.2.0.1 Descrição dos Casos de Uso 02
6.2.0.1.1 Visão Coordenação de Estágios UFPR 02
6.2.0.1.1.1 Módulo Estágio - Inserir Usuário 03
6.2.0.1.1.2 Módulo Estágio - Convênios 06
6.2.0.1.1.3 Módulo Estágio - Agentes de Integração 09
6.2.0.1.1.4 Módulo Estágio - Cursos 11
6.2.0.1.1.5 Módulo Estágio - Empresas 13
6.2.0.1.1.6 Módulo Estágio - Vagas 15
6.2.0.1.1.7 Módulo Estágio - Usuários 17
6.2.0.1.1.8 Módulo Estágio - Termos de Compromisso 19
6.2.0.1.1.9 Módulo Estágio - Estágio Interno 21
6.2.0.1.1.10 Módulo Estágio - Relatório de Seguro 25
6.2.0.1.1.11 Módulo Estágio - E-mail do Sistema 27
6.2.0.1.2 Visão Empresas 29
6.2.0.1.2 Visão Empresa / Visão Aluno 29
6.2.0.1.2.1 Módulo Estágio - Vagas 30
6.2.0.1.2.2 Módulo Estágio - Termos de Compromisso 35
6.2.0.1.2.3 Módulo Estágio - Relatório de Seguro 37
6.2.0.1.3 Visão Aluno 39
6.2.0.1.3.1 Módulo Estágio - Buscar Vagas 39
6.2.0.1.3.2 Módulo Estágio - Pessoal 41
6.2.1 Diagrama de Telas 44
6.2.1.1 Visão Coordenação de Estágios UFPR 44
6.2.1.1.1 Login 44
6.2.1.1.2 Tela Inicial 44
6.2.1.1.3 Fluxo Inserir Usuário – Selecionando Perfil de Usuário 45

6.2.1.1.4 Fluxo Inserir Usuário – Formulário de Cadastro Parte 1 45
6.2.1.1.5 Fluxo Inserir Usuário – Formulário de Cadastro Parte 2 46
6.2.1.1.6 Fluxo Inserir Usuário – Operação realizada com sucesso 46
6.2.1.1.7 Fluxo Convênios – Pesquisar 47
6.2.1.1.8 Fluxo Convênios – Visualizando Convênio Pesquisado 47
6.2.1.1.9 Fluxo Convênios – Pesquisa por empresas não conveniadas 48
6.2.1.1.10 Fluxo Convênios – Visualização de empresas não conveniadas 48
6.2.1.1.11 Fluxo Convênios – Criar Convênios 49
6.2.1.1.12 Fluxo Agentes de Integração – Visualizar 49
6.2.1.1.13 Fluxo Cursos – Visualizar Cursos Cadastrados 50
6.2.1.1.14 Fluxo Cursos – Inserir Curso 50
6.2.1.1.15 Fluxo Empresas – Pesquisar Razão Social 51
6.2.1.1.16 Fluxo Empresas – Visualizando resultados da pesquisa 51
6.2.1.1.17 Fluxo Empresas – Editar Cadastro 52
6.2.1.1.18 Fluxo Vagas – Selecionando Curso 52
6.2.1.1.19 Fluxo Vagas – Visualizando Vagas do Curso Selecionado 53
6.2.1.1.20 Fluxo Usuários – Buscar Usuário 53
6.2.1.1.21 Fluxo Usuários – Visualizando Resultado da Pesquisa 54
6.2.1.1.22 Fluxo Termos de Compromisso – Busca 54
6.2.1.1.23 Fluxo Termos de Compromisso – Visualizando Resultado da Pesquisa 55
6.2.1.1.24 Fluxo Estágio Interno – Visualizando Vagas Disponíveis 55
6.2.1.1.25 Fluxo Estágio Interno – Editar Vaga 56
6.2.1.1.26 Fluxo Estágio Interno – Preencher Vaga (Buscando Estagiário) 56
6.2.1.1.27 Fluxo Estágio Interno – Selecionando Estagiário 57
6.2.1.1.28 Fluxo Estágio Interno – Formulário de Preenchimento do Termo de Compromisso 57
6.2.1.1.29 Fluxo Estágio Interno – Termo de Compromisso 58
6.2.1.1.30 Fluxo Estágio Interno – Plano de Estágio 58
6.2.1.1.31Fluxo Estágio Interno – Busca Unidade Concedente 59
6.2.1.1.32 Fluxo Estágio Interno – Criar Vaga 59
6.2.1.1.33 Fluxo Estágio Interno – Preenchendo Formulário para Inserir Vaga. 60
6.2.1.1.34 Fluxo Estágio Interno – Visualizando Vaga Inserida 60
6.2.1.1.35 F luxo Estágio Interno – Inserir Requisito 61

6.2.1.1.36 Fluxo Estágio Interno – Procurar Requisitos já cadastrados 61
6.2.1.1.37 Fluxo Estágio Interno – Visualizando Requisitos 62
6.2.1.1.38 Fluxo Estágio Interno – Inserir Horários 62
6.2.1.1.39 Fluxo Estágio Interno – Visualizando Horários 63
6.2.1.1.40 Fluxo Relatório de Seguro – Selecionando Relatório 63
6.2.1.1.41 Fluxo Relatório de Seguro – Relatório de Seguro 64
6.2.1.1.42 Fluxo E-mail do Sistema 64
6.2.1.1.43 Sair 65
6.2.1.2 Visão Empresas 65
6.2.1.2.1 Login 66
6.2.1.2.2 Fluxo Vagas – Visualizar Vagas 66
6.2.1.2.3 Fluxo Vagas – Editar Vaga 67
6.2.1.2.4 Fluxo Vagas – Preencher Vaga – Buscando Estagiário 67
6.2.1.2.5 Fluxo Vagas – Selecionando Estagiário 68
6.2.1.2.6 Fluxo Vagas – Formulário para Novo Termo de Compromisso 68
6.2.1.2.7 Fluxo Vagas – Termo de Compromisso 69
6.2.1.2.8 Fluxo Vagas – Plano de Estágio 69
6.2.1.2.9 Fluxo Vagas – Inserir Vaga 70
6.2.1.2.10 Fluxo Vagas – Visualizando Vaga Inserida 70
6.2.1.2.11 Fluxo Vagas – Inserir Requisito 71
6.2.1.2.12 Fluxo Vagas – Buscando por requisitos já cadastrados 71
6.2.1.2.13 Fluxo Vagas – Visualizando Requisitos Inseridos 72
6.2.1.2.14 Fluxo Vagas – Inserir Horários 72
6.2.1.2.15 Fluxo Vagas – Visualizando Horários Inseridos 73
6.2.1.2.16 Fluxo Termos de Compromisso – Visualizando Termos de Compromisso inseridos 73
6.2.1.2.17 Fluxo Termos de Compromisso – Editar Termo de Compromisso ... 74
6.2.1.2.18 Fluxo Termos de Compromisso – Visualizando Termo de Compromisso alterado 74
6.2.1.2.19 Fluxo Relatório de Seguro – Selecionando relatório 75
6.2.1.2.20 Fluxo Relatório de Seguro – Visualizando relatório 75
6.2.1.2.21 Sair 76
6. 2.1.3 Visão Aluno 76
6. 2.1.3.1 Login 77

6. 2.1.3.2 Tela Inicial 77
6. 2.1.3.3 Fluxo Buscar Vagas – Visualizando Vagas Disponíveis 78
6. 2.1.3.4 Fluxo Buscar Vagas – Não há Vagas 78
6. 2.1.3.5 Pessoal – Editar 79
6. 2.1.3.6 Fluxo Pessoal – Visualizar Habilidades 79
6. 2.1.3.7 Fluxo Pessoal – Procurar Habilidades 80
6. 2.1.3.8 Fluxo Pessoal – Inserir Habilidades 80
6. 2.1.3.9 Fluxo Pessoal – Visualizar Horários 81
6. 2.1.3.10 Fluxo Pessoal – Inserir Horários 81
6. 2.1.3.11 Sair 82
6.2.2 Diagramas de Seqüência 83
6.2.2.1 Login 83
6.2.2.2 Inserir Usuário 83
6.2.2.3Nova Empresa 84
6.2.2.4 Novo Convênio 84
6.2.2.5 Criar termo de Compromisso 85
6.2.2.6 Ofertar Vaga 85
6.2.2.7 Consultar Cursos Vagas 85
6.2.2.8 Buscar Vagas 86
6.2.2.9 Relatório de Seguro 86
6.2.3 Diagramas de Colaboração 87
6.2.3.1 Login 87
6.2.3.2 Inserir Usuário 87
6.2.3.3 Nova Empresa 88
6.2.3.4 Novo Convênio 88
6.2.3.5 Criar Termo de Compromisso 89
6.2.3.6 Ofertar Vaga 89
6.2.37 Consultar Cursos Vagas 90
6.2.3.8 Buscar Vagas 90
6.2.3.9 Relatório de Seguro 91
6.2.4 Módulo Egressos 92
6.2.4.1 Descrição dos Casos de Uso 92
6.2.4.1.1 Visão Coordenação do Curso 92
6.2.4.1.1.1 Módulo Egressos – Registrar Usuário 92

6.2.4.1.1.2 Módulo Egressos – Usuários 95
6.2.4.1.1.3 Módulo Egressos – Questionários 97
6.2.4.1.1.4 Módulo Egressos – Relatórios 100
6.2.4.1.1.5 Módulo Egressos – E-mail do Sistema 103
6.2.4.1.1.6 Módulo Egressos – Pesquisa 105
6.2.4.1.1.7 Módulo Egressos – Alterar Cadastro 106
6.2.4.1.2 Visão Egresso 108
6.2.4.1.2.1 Módulo Egressos – Registrar Usuário 108
6.2.4.1.2.2 Módulo Egressos – Pesquisa 110
6.2.4.1.2.3 Módulo Egressos – Alterar Cadastro 112
6.2.5 Diagramas de Telas 114
6.2.5.1 Visão Egresso 114
6.2.5.1.1 Fluxo Registre-se – Novo Usuário 114
6.2.5.1.2 Login 114
6.2.5.1.3 Tela Inicial 115
6.2.5.1.4 Fluxo Pesquisa – Usuário Respondendo 115
6.2.5.1.5 Fluxo Pesquisa – Usuário Respondendo (Continuação) 116
6.2.5.1.6 Fluxo Pesquisa – Pesquisa Respondida 116
6.2.5.1.7 Fluxo Alterar Cadastro – Usuário Edita Dados Pessoais 117
6.2.5.1.8 Sair 117
6.2.5.2 Visão Coordenação 118
6.2.5.2.1 Login 118
6.2.5.2.2 Tela Inicial 119
6.2.5.2.3 Fluxo Registrar Usuário – Formulário de Cadastro 119
6.2.5.2.4 Fluxo Usuários – Buscar Usuários 120
6.2.5.2.5 Fluxo Usuários – Listando Usuário(s) Encontrado(s) 120
6.2.5.2.6 Fluxo Usuários – Editando Cadastro 121
6.2.5.2.7 Fluxo Questionários – Visualizar 121
6.2.5.2.8 Fluxo Questionários – Visualizando Questionário já inserido 122
6.2.5.2.9 Fluxo Questionários – Editar Questão 122
6.2.5.2.10 Fluxo Questionários – Inserir Questão no questionário existente .. 123
6.2.5.2.11 Fluxo Questionários – Visualizar Questão Inserida 123
6.2.5.2.12 Fluxo Questionários – Inserir Novo Questionário 124
6.2.5.2.13 Fluxo Questionários – Inserir Nova Questão 124

6.2.5.2.14 Fluxo Questionários – Visualizando Estatísticas 125
6.2.5.2.15 Fluxo Relatórios – Seleciona Tipo de Relatório 125
6.2.5.2.16 Fluxo Relatórios – Listando Egressos Por ano de conclusão 126
6.2.5.2.17 Fluxo Relatórios – Listando Egressos Por Nome 126
6.2.5.2.18 Fluxo Email do Sistema 127
6.2.5.2.19 Fluxo Pesquisa – Visualizando 127
6.2.5.2.20 Fluxo Alterar Cadastro – Editando Dados Pessoais 128
6.2.5.2.21 Sair 128
6.2.6 Diagramas de Seqüência 129
6.2.6.1 Inserir Usuário 129
6.2.6.2 Buscar Usuários 130
6.2.6.3 Inserir Questionário 130
6.2.6.4 Relatório 130
6.2.6.5 Responder Questionário 131
6.2.7 Diagramas de Colaboração 131
6.2.7.1 Inserir Usuário 132
6.2.7.2 Buscar Usuário 132
6.2.7.3 Inserir Questionário 132
6.2.7.4 Relatório 133
6.2.7.5 Responder Questionário 133
6.3 Apêndice 3 – Modelagem de Dados 01
6.3.1 Módulo Estágio 02
6.3.1.1 Diagramas de Atividades 02
6.3.1.1.1 Visão Aluno 02
6.3.1.1.1.1 Buscar Vagas 02
6.3.1.1.1.2 Pessoal 03
6.3.1.1.2 Visão Empresa 04
6.3.1.1.2.1 Relatório de Seguro 04
6.3.1.1.2.2 Termos de Compromisso 05
6.3.1.1.2.3 Vagas 06
6.3.1.1.3 Visão Coordenação de Estágios 07
6.3.1.1.3.1 Inserir Usuário 07
6.3.1.1.1.2 Agentes de Integração 08
6.3.1.1.1.3 Convênios 09

6.3.1.1.1.4 Cursos 10
6.3.1.1.1.5 Empresas 11
6.3.1.1.1.6 Vagas 12
6.3.1.1.1.7 Usuários 13
6.3.1.1.1.9 Estágio Interno 14
6.3.1.2 Diagramas de Estados 15
6.3.1.2.1 Classe Aluno 15
6.3.1.2.2 Classe Empresa 15
6.3.1.2.3 Classe Convênio 16
6.3.1.2.4 Classe Vagas 16
6.3.1.2.5 Classe Termo de Compromisso 17
6.3.1.3 Dicionário de Dados 18
6.3.1.3.1 Tabela: AlunoEstágio 18
6.3.1.3.2 Tabela: Cursos 19
6.3.1.3.3 Tabela: Habilidades 19
6.3.1.3.4 Tabela: HabilidadesAluno 19
6.3.1.3.5 Tabela: HabilidadesVaga 20
6.3.1.3.6 Tabela: Pessoa 20
6.3.1.3.7 Tabela: TurnoDisponivel 21
6.3.1.3.8 Tabela: planos_estagio 21
6.3.1.3.9 Tabela: Convênios 22
6.3.1.3.10 Tabela: Empresa 22
6.3.1.3.11 Tabela: EnderecoEmpresa 23
6.3.1.3.12 Tabela: TermoCompromisso 24
6.3.1.3.13 Tabela: Turno 24
6.3.1.3.14 Tabela: Turnodisponivel 25
6.3.1.3.15 Tabela: UsuarioEmpresa 25
6.3.1.3.16 Tabela: Vagas_cursos 25
6.3.1.3.17 Tabela: Vagas 25
6.3.2 Módulo Egressos 27
6.3.2.1 Diagramas de Atividades 27
6.3.2.1 Visão Aluno 27
6.3.2.2 Visão Coordenação 27
6.3.3 Diagramas de Estados 28

6.3.3.1 Classe Aluno 28
6.3.3.2 Classe Questionário 28
6.3.4 Dicionário de Dados 29
6.3.4.1 Tabela: Aluno 30
6.3.4.2 Tabela: EnderecoPessoa 30
6.3.4.3 Tabela: Questionario 30
6.3.4.4 Tabela: Questão 30
6.3.4.5 Tabela: Respostas 31
6.4 Apêndice 4 – Códigos Fonte 01
6.4.0 Módulo Egressos - Código Fonte 02
6.4.1 Estrutura de diretórios 02
6.4.2 tiles-defs.xml 03
6.4.3 struts-config.xml 03
6.4.4 validation.xml 08
6.4.5 validation-rules.xml 10
6.4.6 web.xml 12
6.4.7 package egressos.action 13
6.4.8 AumentarOrdemAction.java 13
6.4.9 BuscarUsuarioAction.java 14
6.4.10 DefinirAtivoAction.java 14
6.4.11 EditarCadastroAction.java 15
6.4.12 EditarEmailAction.java 16
6.4.13 EditarQuestaoAction.java 17
6.4.14 EnviaSenhaAction.java 18
6.4.15 EstatisticasQuestionarioAction.java 19
6.4.16 ExcluirEgressoAction.java 21
6.4.17 ExcluirQuestaoAction.java 21
6.4.18 ExcluirQuestionarioAction.java 22
6.4.19 GerenciarQuestionariosAction.java 23
6.4.20 ListarQuestoesAction.java 23
6.4.21 LogoffAction.java 24
6.4.22 LogonAction.java 25
6.4.23 MainMenuAction.java 25
6.4.24 MostrarRespostasQuestaoAction.java 26

6.4.25 NovaQuestaoAction.java 28
6.4.26 NovoQuestionarioAction.java 28
6.4.27 NovoUsuarioAction.java 29
6.4.28 NovoUsuarioAdminAction.java 30
6.4.29 ObterUsuarioAction.java 30
6.4.30 RegistrarEgressoAction.java 31
6.4.31 RelatorioAnoAction.java 31
6.4.32 RelatorioNomeAction.java 32
6.4.33RelatoriosAction.java 33
6.4.34 ResponderQuestionarioAction.java 33
6.4.35 SalvarCadastroAction.java 34
6.4.36 SalvarQuestaoAction.java 35
6.4.37 SalvarRegistroEgressoAction.java 37
6.4.38 SalvarRespostasQuestionarioAction.java 39
6.4.39 SalvarSenhaAlteradaAction.java 40
6.4.40 SalvarServidorSmtpAction.java 41
6.4.41 WelcomeAction.java 41
6.4.42 package egressos.bean 42
6.4.43 Alternativa.java 42
6.4.44 Aluno.java 42
6.4.45 AlunoDTO.java 43
6.4.46 Constants.java 45
6.4.47 Endereco.java 45
6.4.48 Pessoa.java 46
6.4.49 Questao.java 46
6.4.50 Questionario.java 48
6.4.51 QuestionarioDTO.java 49
6.4.52 Resposta.java 49
6.4.53 RespostaDTO.java 49
6.4.54 ServidorSmtp.java 50
6.4.55 User.java 51
6.4.56 package egressos.dao 51
6.4.57 AlunoDao.java 51
6.4.58 BaseDao.java 55

6.4.59 EnderecoPessoaDao.java 56
6.4.60 PessoaDao.java 57
6.4.61 QuestaoDao.java 59
6.4.62 QuestionarioDao.java 63
6.4.63 RespostaDao.java 65
6.4.64 Servidor_smtp_egressosDao.java 68
6.4.65 UserDao.java 70
6.4.66 package estagio.form 71
6.4.67 AlterarSenhaForm.java 71
6.4.68 DadosPessoaisForm.java 71
6.4.69 LogonForm.java 75
6.4.70 NovaQuestaoForm.java 75
6.4.71 ObterUsuarioForm.java 77
6.4.72 RegistrationForm.java 77
6.4.73 ResponderQuestionarioForm.java 78
6.4.74 ServidorSmtpForm.java 79
6.4.75 package estagio.mail 80
6.4.76 MailManager.java 80
6.4.77 MailMessage.java 80
6.4.78 MailServer.java 81
6.4.79 package estagio.validation 82
6.4.80 Valida.java 82
6.4.81 Diretório pages 83
6.4.82 alterarsenha.jsp 83
6.4.83 buscarUsuario.jsp 83
6.4.84 dadosPessoais.jsp 84
6.4.85 erro.jsp 86
6.4.86 estatisticasQuestionario.jsp 86
6.4.87 footer.jsp 87
6.4.88 gerenciarQuestionarios.jsp 87
6.4. 89 index.jsp 87
6.4.90 informausername.jsp 88
6.4.91 inicial.jsp 88
6.4.92 inserequestao.jsp 88

6.4.93 listarQuestoes.jsp 89
6.4.94 mainMenu.jsp 90
6.4.95 mostrarQuestao.jsp 90
6.4.96 novoservidorsmtp.jsp 91
6.4.97 relatorioano.jsp 92
6.4.98 relatorionome.jsp 93
6.4.99 responderQuestionario.jsp 93
6.4.100 resultadosBuscaUsuarios.jsp 94
6.4.101 sucesso.jsp 94
6.4.102 tiporelatorio.jsp 94
6.4.103 tcc.css 94
6.4.104 Welcome.jsp 95
6.4.105 MóduloEstágios - Código Fonte 96
6.4.105 Estrutura de diretórios 96
6.4.106 tiles-defs.xml 97
6.4.107 struts-config.xml 98
6.4.108 validation.xml 105
6.4.109 validation-rules.xml 108
6.4.110 web.xml 110
6.4.111 Diretório "Pages" 110
6.4.112 buscarcnpj.jsp 110
6.4.113 buscarempresa.jsp 111
6.4.114 buscarestagiário.jsp 111
6.4.115 buscarrequisitos.jsp 111
6.4.116 buscarusuario.jsp 112
6.4.117 buscarvagas.jsp 112
6.4.118 consultarcursosvags.jsp 113
6.4.119 dadospessoais.jsp 114
6.4.120 erro.jsp 116
6.4.121 estilo.css 117
6.4.122 footer.jsp 117
6.4.123 index.jsp 117
6.4.124 inicial.jsp 118

6.4.125 listaragentes.jsp 118
6.4.126 listarconvenios.jsp 118
6.4.127 listarcursos.jsp 119
6.4.128 listarempresas.jsp 119
6.4.129 listarempresassemconvenio.jsp 120
6.4.130 listarhabilidades.jsp 121
6.4.131 listarhabilidadesaluno.jsp 121
6.4.132 listarhabilidadesvaga.jsp 122
6.4.133 listartermos.jsp 123
6.4.134 listartermoscompromisso.jsp 123
6.4.135 listarturnos.jsp 124
6.4.136 listarturnosdisponiveis.jsp 124
6.4.137 listarunidade.jsp 125
6.4.138 listarusuarios.jsp 125
6.4.139 listarvagas.jsp 126
6.4.140 login.jsp 126
6.4.141 mainMenu.jsp 127
6.4.142 novaempresa.jsp 129
6.4.143 novahabilidadealuno.jsp 129
6.4.144 novahabilidadealunogeral.jsp 130
6.4.145 novahabilidadevaga.jsp 130
6.4.146 novahabilidadevagageral.jsp 130
6.4.147 novavaga.jsp 131
6.4.148 novoConvenio.jsp 132
6.4.149 novocurso.jsp 133
6.4.150 novotermocompromisso.jsp 134
6.4.151 novoturno.jsp 135
6.4.152 novoturnodisponivel.jsp 136
6.4.153 pesquisarconvenios.jsp 136
6.4.154 pesquisarempresas.jsp 137
6.4.155 pesquisarempresassemconvenio.jsp 137
6.4.156 pesquisarhabilidadealuno.jsp 138
6.4.157 pesquisartermoscompromisso.jsp 138
6.4.158 selecionarestagiario.jsp 139

6.4.159 selecionarrelatorio.jsp 139
6.4.160 sucesso.jsp 140
6.4.161 tcc.css 140
6.4.162 termocompromisso.jsp 140
6.4.163 tipousuario.jsp 140
6.4.164 Welcome.jsp 141
6.4.165 package estagio.action 141
6.4.166 AlterarSenhaAction.java 141
6.4.167 BuscarEmpresaAction.java 141
6.4.168 BuscarEstagiarioAction.java 142
6.4.169 BuscarHabilidadeAlunoAction.java 143
6.4.170 BuscarRequisitosAction.java 144
6.4.171 BuscarVagasAction.java 145
6.4.172 ConsultarCursosVagasAction.java 146
6.4.173 CriarTermoCompromissoAction.java 148
6.4.174 EditarCadastroAction.java 148
6.4.175 EditarEmailAction.java 150
6.4.176 EditarEmpresaAction.java 150
6.4.177 EditarTermoAction.java 151
6.4.178 EditarVagaAction.java 152
6.4.179 EnviaSenhaAction.java 153
6.4.180 ExcluirAgenteAction.java 154
6.4.181 ExcluirCadastroAction.java 154
6.4.182 ExcluirConvenioAction.java 155
6.4.183 ExcluirCursoAction.java 156
6.4.184 ExcluirEmpresaAction.java 156
6.4.185 ExcluirHabilidadeAction.java 157
6.4.186 ExcluirHabilidadeAlunoAction.java 158
6.4.187 ExcluirHabilidadeVagaAction.java 158
6.4.188 ExcluirTermoAction.java 159
6.4.189 ExcluirTermoCompromissoAction.java 160
6.4.190 ExcluirTurnoAction.java 161
6.4.191 ExcluirTurnoDisponivelAction.java 162
6.4.192 ExcluirVagaAction.java 162

6.4.193 ListarAgentesIntegraçãoAction.Java 164
6.4.194 ListarConveniosAction.java 164
6.4.195 ListarCursosAction.java 165
6.4.196 ListarEmpresasAction.java 165
6.4.197 ListarHabilidadesAction.java 166
6.4.198 ListarHabilidadesAlunoAction.java 166
6.4.199 ListarHabilidadesVagaAction.java 167
6.4.200 ListarTermosAction.java 167
6.4.201 ListarTermosCompromissoAction.java 169
6.4.202 ListarTurnosAction.java 169
6.4.203 ListarTurnosDisponiveisAction.java 170
6.4.204 ListarVagasAction.java 171
6.4.205 LogoffAction.java 172
6.4.206 LogonAction.java 172
6.4.207 NovaHabilidadeAlunoAction.java 173
6.4.208 NovaHabilidadeAlunoGeralAction.java 174
6.4.209 NovaHabilidadeVagaAction.java 175
6.4.210 NovaHabilidadeVagaGeralAction.java 175
6.4.211 NovaVagaAction.java 176
6.4.212 NovaVagaEstagioInternoAction.java 177
6.4.213 NovoConvenioAction.java 177
6.4.214 NovoTermoCompromissoAction.Java 178
6.4.215 NovoTurnoAction.java 178
6.4.216 NovoTurnoDisponivelAction.java 179
6.4.217 NovoUsuarioAction.java 179
6.4.218 PesquisarConveniosAction.java 180
6.4.219 PesquisarEmpresasAction.java 181
6.4.220 PesquisarEmpresasSemConvenioAction.java 182
6.4.221 PesquisarHabilidadeAlunoAction.java 183
6.4.222 PesquisarTermosCompromissoAction.Java 184
6.4.223 PesquisarUnidadeAction.java 184
6.4.224 PesquisarUsuarioAction.java 185
6.4.225 RelatorioSeguroAction.java 186
6.4.226 SalvarCadastroAction.java 187

6.4.227 SalvarConvenioAction.java 188
6.4.228 SalvarCursoAction.java 188
6.4.229 SalvarEmpresaAction.java 190
6.4.230 SalvarHabilidadeAlunoAction.java 190
6.4.231 SalvarHabilidadeAlunoGeralAction.java 191
6.4.232 SalvarHabilidadeVagaAction.java 192
6.4.233 SalvarHabilidadeVagaGeralAction.java 193
6.4.234 SalvarSenhaAlteradaAction.java 194
6.4.235 SalvarServidorSmtpAction.java 194
6.4.236 SalvarTermoCompromissoAction.java 203
6.4.237 SalvarTurnoAction.java 204
6.4.238 SalvarTurnoDisponivelAction.java 205
6.4.239 SalvarVagaAction.java 207
6.4.240 SelecionarRelatorioSeguroAction.java 208
6.4.241 TipoUsuarioAction.java 209
6.4.242 WelcomeAction.java 210
6.4.243 package estagio.bean 210
6.4.244 AgenteIntegracao.java 210
6.4.245 AlunoEstagio.java 212
6.4.246 Constants.java 212
6.4.247 Convenio.java 213
6.4.248 ConvenioDTO.java 213
6.4.249 Curso.java 214
6.4.250 Empresa.java 215
6.4.251 Endereco.java 216
6.4.252 Habilidade.java 216
6.4.253 HabilidadeAluno.java 216
6.4.254 HabilidadeVaga.java 217
6.4.255 Pessoa.java 217
6.4.256 PlanoEstagio.java 218
6.4.257 RelatorioSeguroDTO.java 219
6.4.258 ServidorSmtp.java 219
6.4.259 TermoCompromisso.java 220
6.4.260 TermoCompromissoDTO.java 221

6.4.261 Turno.java 222
6.4.262 TurnoDisponivel.java 222
6.4.263 TurnoDTO.java 222
6.4.264 Usuario.java 223
6.4.265 UsuarioEmpresa.java 223
6.4.266 Vaga.java 225
6.4.267 VagaCurso.java 225
6.4.268 VagaDTO.java 226
6.4.269 VagaSelecionadaDTO.java 227
6.4.270 package estagio.dao 227
6.4.271 AgentesIntegracaoDao.java 227
6.4.272 AlunoEstagioDao.java 227
6.4.273 BaseDao.java 231
6.4.274 ConvenioDao.java 231
6.4.275 CursoDao.java 233
6.4.276 EmpresaDao.java 235
6.4.277 EnderecoEmpresaDao.java 239
6.4.278 EnderecoPessoaDao.java 241
6.4.279 HabilidadesAlunoDao.java 242
6.4.280 HabilidadesDao.java 244
6.4.281 HabilidadesVagaDao.java 246
6.4.282 PessoaDao.java 248
6.4.283 PlanosEstagioDao.java 249
6.4.284 RelatorioSeguroDao.java 251
6.4.285 Servidor_smtpDao.java 256
6.4.286 TermoCompromissoDao.java 257
6.4.287 TurnoDao.java 260
6.4.288 TurnoDisponivelDao.java 262
6.4.289 UsuarioEmpresaDao.java 263
6.4.290 VagaDao.java 265
6.4.291 VagasCursosDao.java 269
6.4.292 package estagio.form 270
6.4.293 AlterarSenhaForm.java 270
6.4.294 BuscarEmpresaForm.java 271

6.4.295 BuscarEstagiarioForm.java 271
6.4.296 BuscarUsuarioForm.java 271
6.4.297 ConsultarCursosVagasForm.java 272
6.4.298 DadosPessoaisForm.java 272
6.4.299 InformaUserNameForm.java 275
6.4.300 LogonForm.java 275
6.4.301 NovaEmpresaForm.java 276
6.4.302 NovaHabilidadeAlunoForm.java 278
6.4.303 NovaHabilidadeForm.java 278
6.4.304 NovaHabilidadeVagaForm.java 279
6.4.305 NovaVagaEstagioInternoForm.java 279
6.4.306 NovaVagaForm.java 280
6.4.307 NovoAgenteIntegracaoForm.java 281
6.4.308 NovoConvenioForm.java 282
6.4.309 NovoCursoForm.java 282
6.4.310 NovoTermoCompromissoForm.java 283
6.4.311 NovoTurnoForm.java 285
6.4.312 PesquisarConveniosForm.java 286
6.4.313 PesquisarEmpresasForm.java 287
6.4.314 PesquisarHabilidadeAlunoForm.java 287
6.4.315 PesquisarTermosCompromissoForm.java 288
6.4.316 SelecionarEstagiarioForm.java 288
6.4.317 SelecionarRelatorioForm.java 289
6.4.318 ServidorSmtpForm.java 289
6.4.319 TipoUsuarioForm.java 290
6.4.320 TurnoDisponivelForm.java 290
6.4.321 package estagio.mail 291
6.4.322 MailManager.java 291
6.4.323 MailMessage.java 292
6.4.324 MailServer.java 293
6.4.325 package estagio.validation 294
6.4.326 Valida.java 294
6.4.327 package jasper 295
6.4.328 teste_seguro.jrxml 295

RESUMO

O presente trabalho apresenta dois módulos que serão parte integrante do sistema de Intranet Acadêmica da Universidade Federal do Paraná; são eles: o Módulo Estágio, que possibilita o gerenciamento e a manutenção do processo de realização de estágios pelos alunos da universidade, bem como formalização de convênios com empresas interessadas; e o Módulo Egressos, que prevê como principal objetivo a manutenção do vínculo Universidade versus Aluno para o Curso de Gestão da Informação, visando um aperfeiçoamento na grade curricular do curso.

ABSTRACT

The present work presents two modules that will be available through the Academic system of Intranet of the Federal University of Paraná. They are: the Trainee Module, that makes possible the management and the maintenance of the process of accomplishment of periods of training for the pupils of the university, as well as formalization of accords with interested companies. And the Egresses Module, that foresees as main objective the maintenance of the bond University versus Pupil of the Course of Information Management, aiming at a perfectioning in the curricular grating of the course.

1.0 Introdução

O processo de desenvolvimento da Intranet Acadêmica da Universidade Federal do Paraná está dividido em 4 módulos principais, sendo eles: Módulo Infra-Estrutura, Módulo Avaliação, Módulo Estágio e Módulo Egressos.

Coube à equipe o desenvolvimento dos dois últimos módulos citados: Estágios e Egressos.

A idéia do Módulo Estágio veio da necessidade da PROGRAD em estabelecer um controle mais efetivo em relação à realização de estágios pelos acadêmicos. O foco principal é gerenciar o processo através do cadastro dos alunos e disponibilização de consulta às vagas, bem como formalização de convênios com as empresas interessadas, as quais poderão inserir novas vagas disponíveis assim que estas surgirem. Seria basicamente como tornar a PROGRAD um agente de integração entre as empresas e a Universidade Federal do Paraná.

O Módulo Egressos é voltado especificamente à Coordenação do Curso de Gestão da Informação, interessada principalmente na manutenção do vínculo aluno versus Universidade após a conclusão do curso. Isto se deu através de levantamentos feitos pela própria coordenação, que constatou a dificuldade de alguns profissionais dessa área, formados na Universidade Federal do Paraná, em conseguir boa colocação no mercado, o que estava ocasionando por vezes uma migração desses profissionais para outras áreas.

Isto é um fator negativo para o curso e como meio de evitar que ocorra foi sugerida uma maneira de reformular a grade curricular, e seria através de uma avaliação feita pelos egressos, onde eles poderiam relatar o que lhes faltou durante o curso e quais as dificuldades encontradas, apontando assim o caminho para o melhoramento do curso.

Na Intranet o módulo consiste em um cadastro dos egressos, onde estes podem atualizar seus dados informando se estão atuando no mercado de trabalho e, através de um questionário previamente elaborado pela coordenação, informar suas principais necessidades avaliando dessa forma o curso concluído.

1.1 Justificativa

O desenvolvimento deste projeto em dois módulos deu-se da necessidade de dois departamentos distintos da Universidade Federal do Paraná: A Coordenação de Estágios (PROGRAD/CGE) e a Coordenação do Curso de Gestão da Informação.

No que diz respeito ao Módulo de Estágio, uma das preocupações da coordenação seria uma forma de controlar os processos de realizações de estágios pelos estudantes da universidade, tendo registrado em um sistema informações referentes às vagas disponíveis, aos termos de compromissos e até mesmo um cadastro de empresas que formalizariam um convênio com a Universidade a fim de integrar os alunos no mercado de trabalho.

Em relação ao Módulo de Egressos, a principal necessidade da Coordenação do Curso de Gestão da Informação seria manter um vínculo com o aluno que concluísse o curso, a fim de reformular a grade curricular do mesmo através da realização de pesquisas.

Através de algumas constatações, como por exemplo, reclamações de egressos em relação à dificuldade de se integrar ao mercado de trabalho, a coordenação viu a necessidade de estabelecer contato com o egresso, para saber se o currículo do curso está atendendo suas necessidades no mercado. Para isso seriam requisitadas informações como: em que empresa está trabalhando e que função exerce, entre outras válidas para um melhoramento ou aperfeiçoamento da grade dentro da gestão da informação, visando formar um profissional que possa realmente atuar na área para a qual estudou e exercendo uma função compatível com sua formação.

1.2 Metodologia

A estruturação de nosso projeto foi embasada nos conceitos de Gerência de Projetos, que prevêem um planejamento do desenvolvimento a ser efetuado, bem como estimativas a respeito do tema abordado e a respeito dos riscos relativos ao completo desenvolvimento do projeto. Também tratam do tempo previsto para cada atividade e à que membro da equipe ela será atribuída, através de um cronograma.

Essa organização torna muito mais fácil o gerenciamento do projeto, visto que o mesmo não é estático, pois pode ir sofrendo necessidade de adaptações durante seu desenvolvimento. Dessa maneira, tendo todo o planejamento documentado, as mudanças são administradas de forma tranqüila, sem maiores empecilhos em relação à forma de como está sendo desenvolvido o trabalho. Maiores detalhes no Plano de Projeto. *(Apêndice I).*

Para a modelagem do sistema utilizamos a metodologia UML (Unified Modeling Language), que tem como objetivo descrever qualquer tipo de sistema, em termos de diagramas orientado a objetos. Naturalmente, o uso mais comum é para criar modelos de sistemas de software, mas a UML também é usada para representar sistemas mecânicos sem nenhum software (RUMBAUGH, James. **UML Guia do Usuário.** Campus.). Para o nosso caso, um software voltado à Web, essa metodologia coube perfeitamente, sendo utilizada na confecção dos diagramas que compõem o sistema.

A linguagem de programação utilizada foi JSP (Java Server Pages), uma tecnologia para desenvolvimento de aplicações com portabilidade de plataforma podendo ser executado em quase todos os Sistemas Operacionais. Permite produzir aplicações que acessem banco de dados, arquivos-texto, captação de informações a partir de formulários, captação de informações sobre o visitante e sobre o servidor, uso de variáveis e loops entre outras coisas. Juntamente utilizamos um framework, o Struts, que auxilia a construção de aplicações para a Web. É construído em Java, e seu coração consiste numa camada de controle flexível baseada nas tecnologias Java Servlets, JavaBeans, ResourceBundles e XML.

O Struts favorece o desenvolvimento de aplicações seguindo o paradigma MVC *(Model-View-Controller).* Ele fornece um componente Controller e se integra a outras tecnologias para oferecer suporte aos componentes Model *(como JDBC, EJB's,etc.)*, e View *(como JSP, XSLT, etc.)* (HUSTED, Ted N.; DUMOULIN, Cedric; FRANCISCUS, George; WINTERFELDT, David. **Struts in Action, Building web applications with the leading Java framework.** Manning Publications Co., 2002.).

No MVC, o fluxo da aplicação é mediado por um controlador central *(Controller)*. O controlador delega requisições para tratadores apropriados, que estão localizados no modelo *(Model)*, que representa a lógica de negócio e o estado da aplicação.

A requisição então é respondida, através do controlador, e apresentada na visão *(View)*, da maneira adequada.

No Struts, essas respostas são orientadas através de mapeamentos, que são carregados através de um arquivo de configuração *(struts-config.xml)*. Isso faz com que não haja dependência entre a visão e o modelo, que auxilia na criação e na manutenção da aplicação.

Como na linguagem JSP as páginas são processadas em um servidor e o cliente visualiza somente o HTML básico, precisamos também de um servidor, e utilizamos o Tomcat.

O software Tomcat, desenvolvido pela Fundação Apache, permite a execução de aplicações para web. Sua principal característica técnica é estar centrada na linguagem de programação Java, mais especificamente nas tecnologias de Servlets e de Java Server Pages (JSP).

Para o sistema gerenciador de banco de dados (SGBD) utilizamos o PostgreSQL versão 8.0.

O PostgreSQL é um banco de dados relacional e orientado a objetos. Seu grande triunfo é possuir recursos comuns à banco de dados de grande porte, o que o deixa apto a trabalhar, inclusive, com operações de missão crítica. Além disso, trata-se de um banco de dados versátil, seguro e gratuito.

É possível utilizar o PostgreSQL em vários sistemas operacionais, em qualquer sistema compatível com as especificações POSIX (Portable Operating System Interface). Segundo informações do site oficial (MICHELAZZO, Paulino. **"A liberdade dos bancos de dados."** Imasters – Software Livre. http://www.imasters.com.br/artigo/3990/livre/a_liberdade_dos_bancos_de_dados. Acesso em 24/04/2006.), o PostgreSQL permite a criação de uma base de dados de tamanho infinito. Cada tabela pode ter até 16 TB (1 terabyte = 1024 gigabytes), sendo que cada linha pode ter até 1,6 TB e cada campo 1 GB. O banco de dados ainda conta com conexões SSL, MVCC, triggers, integridade referencial, entre outros recursos, além de ser compatível com uma série de linguagens, tais como PHP, Python, Java, Perl, etc.

2.0 MÓDULO ESTÁGIO

2.1 Modelagem do Sistema – Diagramas de Casos de Uso

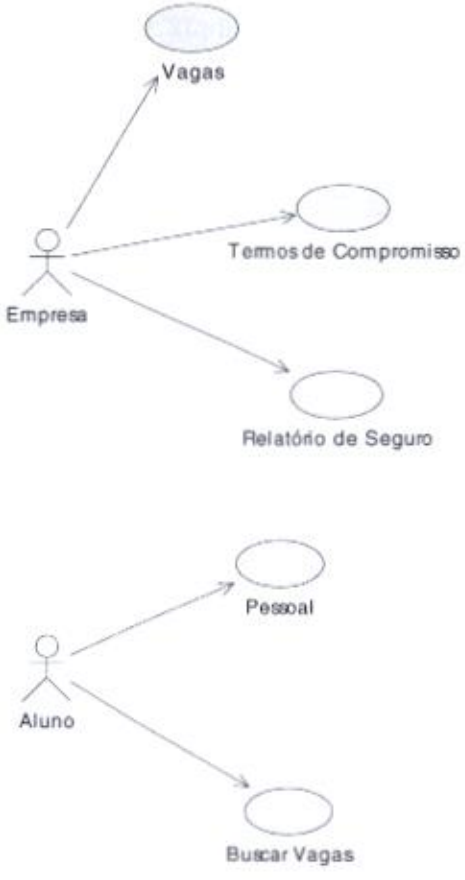
O módulo de estágios está dividido em três visões:

- Visão Coordenação de Estágios da Universidade Federal do Paraná
- Visão Empresa
- Visão Aluno

2.1.1 Visão Coordenação de Estágios Universidade Federal do Paraná

*CGE: Coordenação Geral de Estágios

2.1.2 Visão Empresa / Visão Aluno



A visão completa da modelagem do sistema pode ser adquirida através Descrição dos Casos de Uso e do Diagrama de Telas (*Apêndice 2*), onde também pode-se ter uma idéia real da utilização do sistema através da descrição de possíveis operações a serem realizadas pelo usuário (cenários).

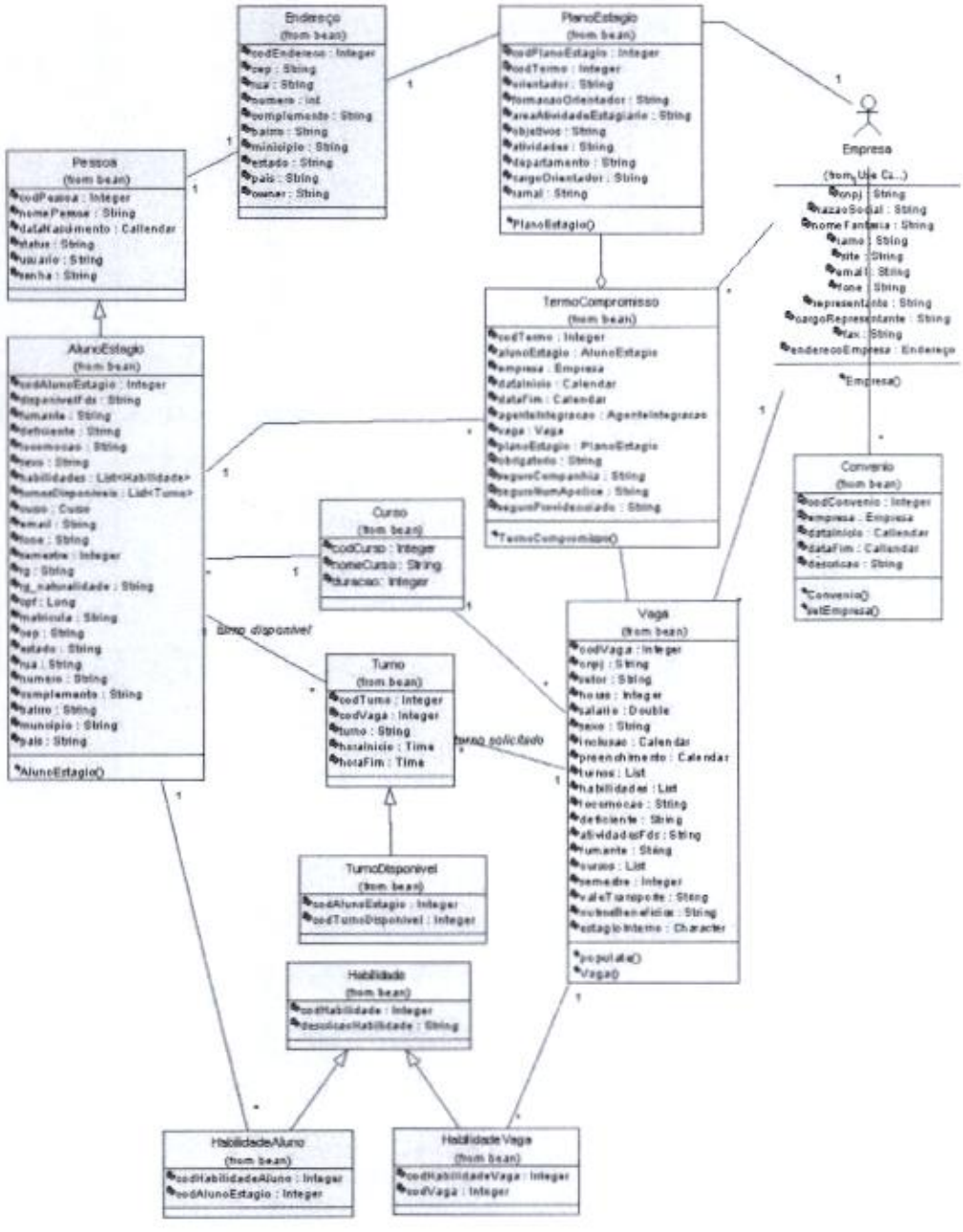
Tais operações também estão descritas nos Diagramas de Seqüência e Colaboração. (*Apêndice 2*) e no Diagrama de Atividades (*Apêndice 3*).

2.2 Modelagem de Dados

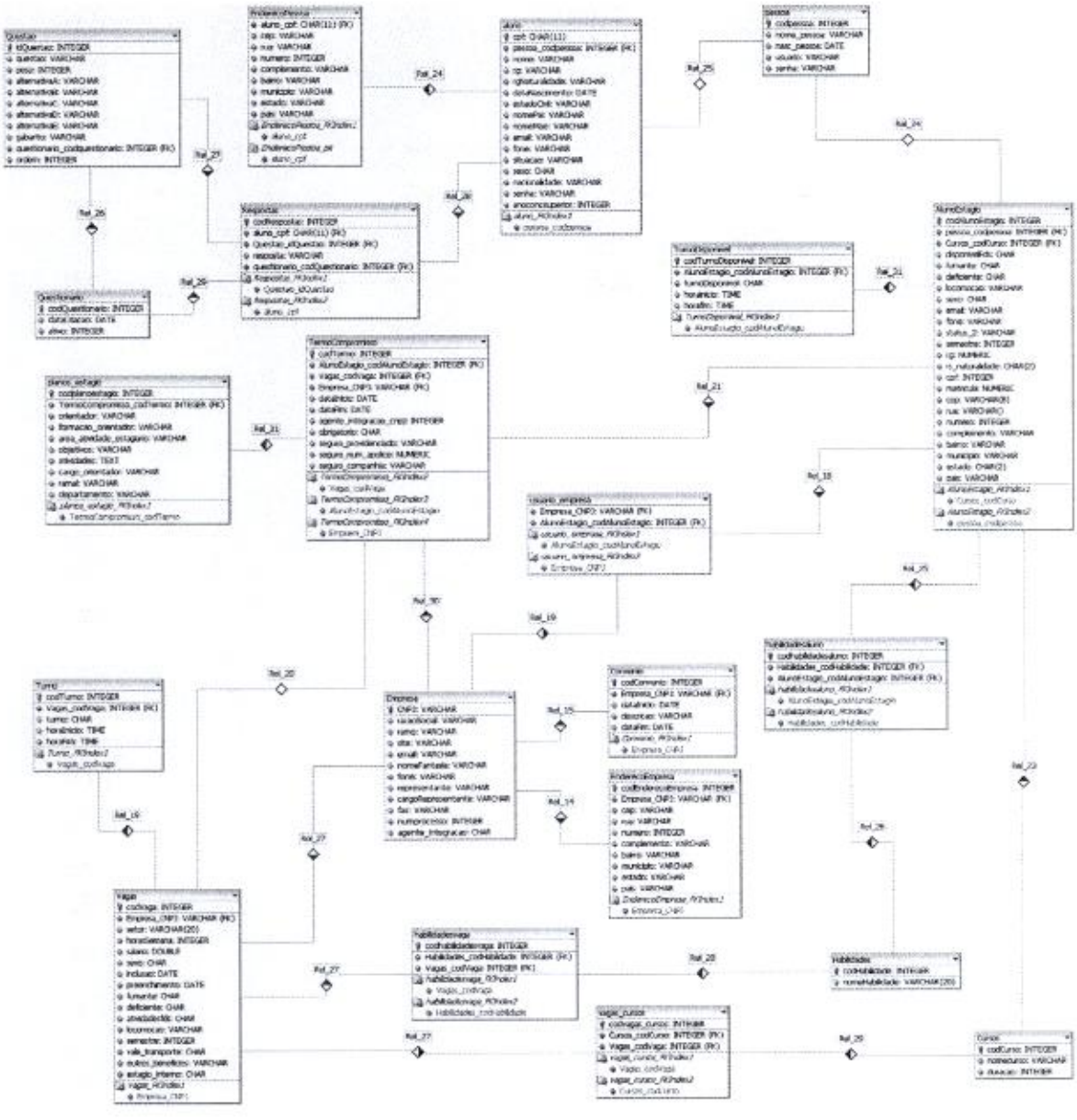
A modelagem dos dados do sistema está integralmente ligada à implementação (*Apêndice 4*), visto que a mesma define como os dados deverão ser tratados na aplicação para criar um sistema de fácil utilização aos usuários, e que dê um retorno satisfatório para o mesmo.

2.2.1 Diagrama de Classes – Módulo Estágio

O comportamento das classes estará descrito no Diagrama de Estados. (*Apêndice 3*).



2.2.2 Modelo Físico: Diagrama Relacional – Módulo Estágio



A modelagem de dados do sistema, bem como o tipo dos dados e suas propriedades podem ser visualizadas no dicionário de dados (*Apêndice 3*).

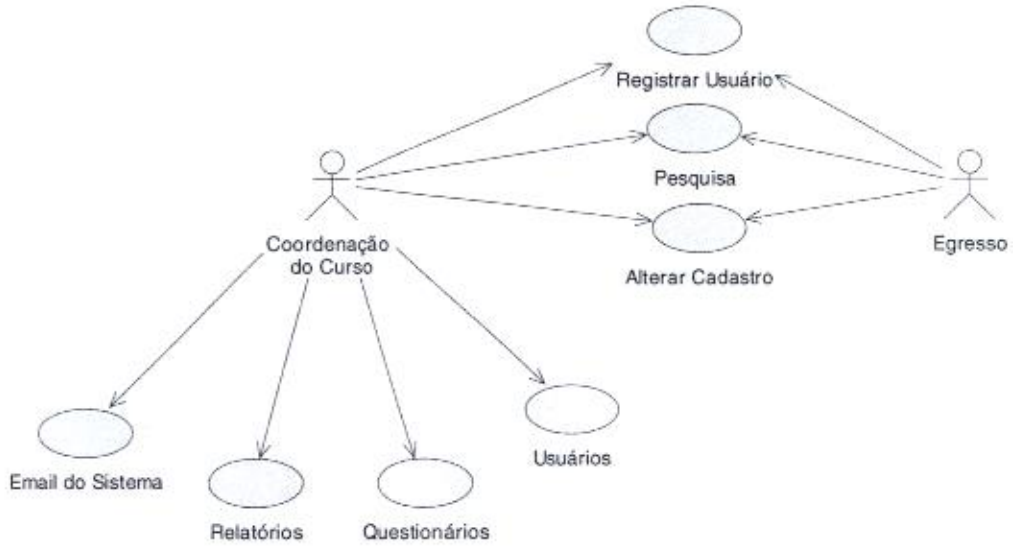
3.0 MÓDULO EGRESSOS

3.1 Modelagem do Sistema – Diagrama de Casos de Uso

O módulo egressos está dividido em duas visões:

- Visão Coordenação do Curso
- Visão Egresso

3.1.1 Visão Coordenação do Curso / Visão Egresso



Na Descrição dos Casos de Uso e no Diagrama de Telas (*Apêndice 2*), têm-se a visualização do real funcionamento do sistema.

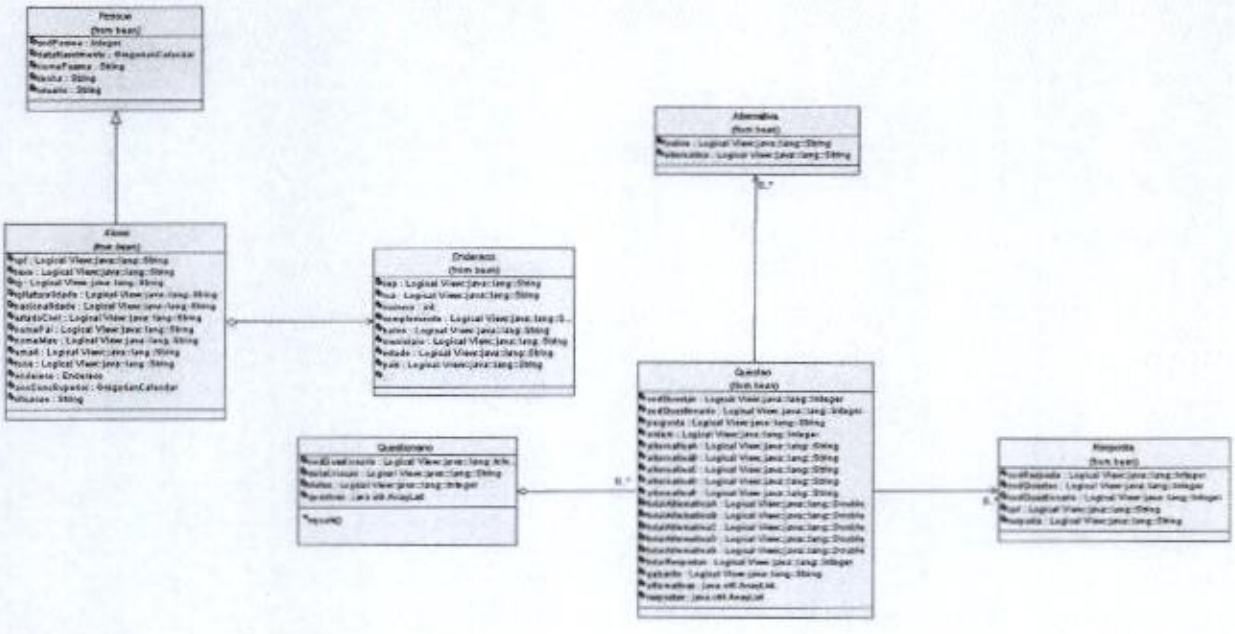
Tais operações também estão descritas nos Diagramas de Seqüência e Colaboração. (*Apêndice 2*) e no Diagrama de Atividades (*Apêndice 3*).

3.2 Modelagem de Dados

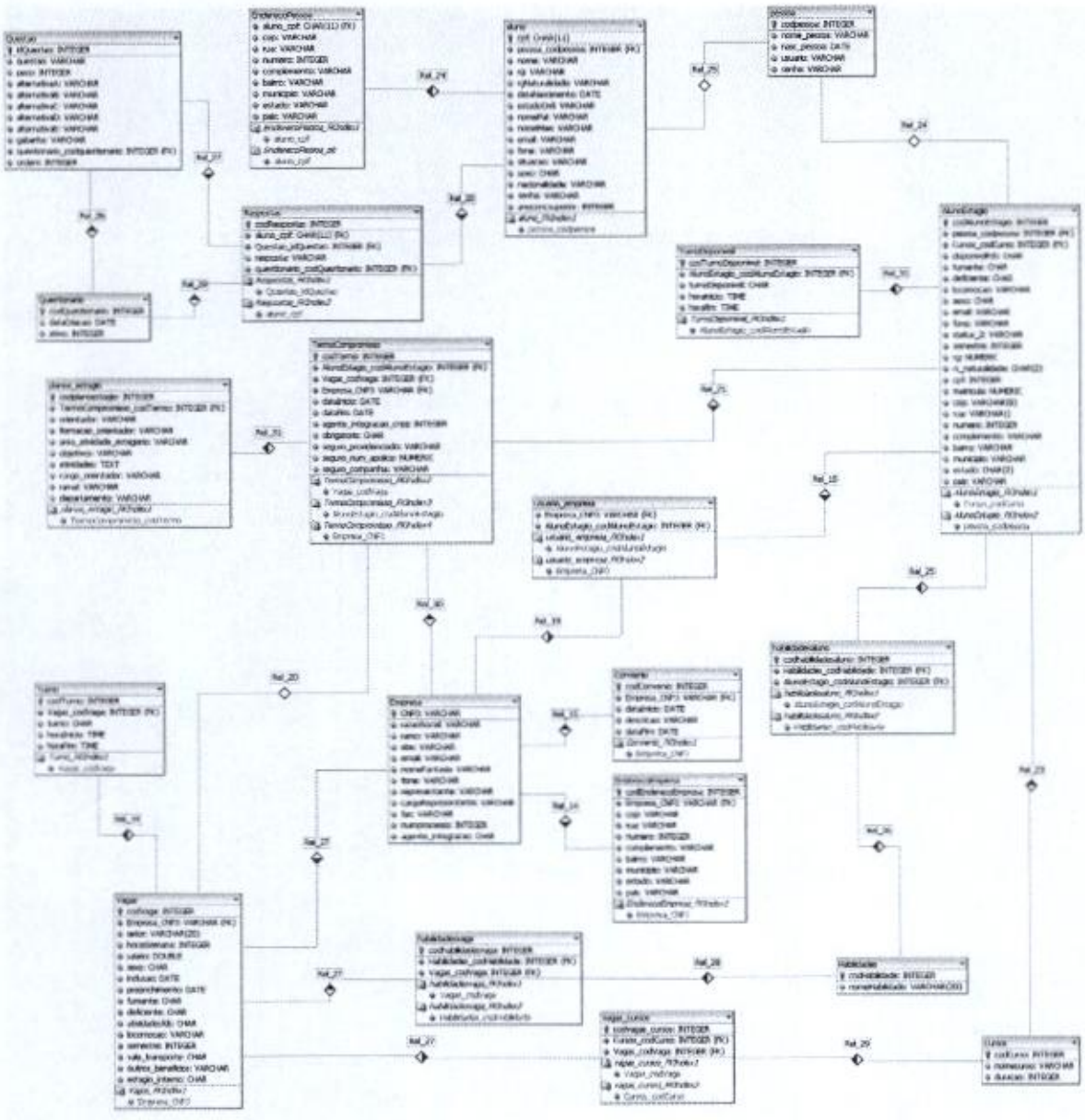
A implementação do sistema (*Apêndice 4*) está refletindo a modelagem dos dados, por isso a importância elevada de tal modelagem, para que o sistema fique consistente.

3.2.1 Diagrama de Classes – Módulo Egressos

O comportamento das classes estará descrito no Diagrama de Estados. (*Apêndice 3*).



3.2.2 Modelo Físico: Diagrama Relacional – Módulo Egressos



A modelagem de dados do sistema, bem como o tipo dos dados e suas propriedades podem ser visualizadas no dicionário de dados (*Apêndice 3*).

4.0 CONCLUSÃO

Ao concluirmos esse projeto pudemos perceber a importância de uma boa estruturação na modelagem e no desenvolvimento de um sistema, para que o mesmo seja implementado de acordo com os requisitos do usuário.

A experiência de trabalhar diretamente com o usuário, em algumas ocasiões, nos proporcionou a vivência de uma prática comum que presenciaremos em nosso dia-a-dia de trabalho como analistas desenvolvedores que pretendemos ser. A adequação e adaptação de alguns itens do sistema são cruciais para o bom desenvolvimento que o analista tem a fazer e a posterior e satisfatória utilização da aplicação feita pelo usuário.

Aprendemos que a modelagem do sistema, bem como a modelagem de dados e o planejamento inicial do projeto a ser seguido são as bases do sistema, e de preferência devem permanecer até a conclusão do desenvolvimento, exceto em casos de alguma alteração nos quais ocorrerão mudanças, porém nunca fugindo à idéia inicial do escopo do projeto. Outro aprendizado importante a citar é que em muitas situações aprendemos também como não fazer, ou seja, guardamos nossos erros para evitar cometê-los novamente no futuro.

Em relação à trabalhos futuros existe a possibilidade da real implementação de nosso projeto na Coordenação de Estágios da Universidade Federal do Paraná, e para isso precisarão ser realizadas modificações para adequação do sistema à algumas necessidades do usuário não previstas na proposta inicial do projeto. São possibilidades a serem estudadas e com certeza interessantes para nós acadêmicos.

5.0 REFERÊNCIAS BIBLIOGRÁFICAS

- HUSTED, Ted N.; DUMOULIN, Cedric; FRANCISCUS, George; WINTERFELDT, David. **Struts in Action, Building web applications with the leading Java framework.** Manning Publications Co., 2002.

- DEITEL, H. M.; DEITEL, P. J.. **Java, como Programar.** 4. ed. Porto Alegre: Bookman, 2003.

- PRESSMAN, Roger S. **Engenharia de software.** São Paulo: Pearson Makron Books, 1995.

- RUMBAUGH, James. **UML Guia do Usuário.** Campus.

- INC, Sun Microsystems. **Java™ 2 Platform Standard Ed. 5.0**. http://java.sun.com/j2se/1.5.0/docs/api/. Acesso em 05/09/2005.

- FOUNDATION, The Apache Software. **Struts**. http://struts.apache.org/struts-doc-1.2.7/userGuide/index.html. Acesso em 05/09/2005.

- CORPORATION, JasperSoft. **Jasper Reposts Documentation.** http://jasperreports.sourceforge.net/documentation.html. Acesso em 15/05/2006.

- GARNIER, Jean-Michel. **"Struts 1.1 Controller UML diagrams".** França. http://rollerjm.free.fr/pro/Struts11.html#3. Acesso em 25/03/2006.

- HOMES, James. **"Reuse Tiles and Simplify UI". Oracle Technology Network.** http://www.oracle.com/technology/oramag/oracle/04-may/o34dev_struts.html. Acesso em 20/09/2005.

- LOWAGIE, Bruno; BAEYENSSTRAAT, Adolf. **"Getting started with iText".** iText Homepage. Bélgica. http://www.lowagie.com/iText/. Acesso em 15/05/2006.

- INC, Sun Microsystems. **"jGuru: Fundamentals of the JavaMail API". Sun Developer Network.** http://java.sun.com/developer/onlineTraining/JavaMail/contents.html. Acesso em 18/03/2006.

- GUJ. **"Java e Desenvolvimento Web"**. http://www.guj.com.br/forums/list.java. Acesso em 02/03/2006.

- FREITAS, Katiúcia. **"Criar uma aplicação utilizando Eclipse, Struts e o FrameWork DBFW4J."** Imasters – Programação Java. http://www.imasters.com.br/artigo/3336. Acesso em 24/04/2006.

- MICHELAZZO, Paulino. **"A liberdade dos bancos de dados."** Imasters – Software Livre. http://www.imasters.com.br/artigo/3990/livre/a_liberdade_dos_bancos_de_dados. Acesso em 24/04/2006.
- ALECRIM, Emerson. **"Banco de dados PostgreSQL e MySQL."** Info Wester. http://www.infowester.com/postgremysql.php. Acesso em 24/04/2006.
- JEVEAUX, MATHEUS. **"Incremente suas aplicações Web com Servlets e Java Server Pages."** Portal Java. http://www.portaljava.com.br/home/modules.php?name=News&file=article&sid=41. Acesso em 25/04/2006.
- CORDEIRO, Pedro Barbosa. **"Tutorial: Utilizando o Rational Rose."** http://www.dsc.ufcg.edu.br/~patricia/LabES/aulas/Tutorial_Rose.doc. Acesso em 20/03/2006.
- DEBONI, José Eduardo Zindel. **"Breve Introdução aos Diagramas da UML."** Voxxel – Conceito em Tecnologia da Informação desde 1988. http://www.voxxel.com.br/pages/introdiauml.html#estados. Acesso em 230/03/2006.
- AQUINO, Renata. **"Indique sites usados nos trabalhos"**. Yahoo! Busca Educação. Brasil. http://br.buscaeducacao.yahoo.com/mt/archives/2005/05/indique_sites_u.html. Acesso em 26/08/2006.
- ALVES, Maria Bernardete Martins; ARRUDA, Susana Margareth. "**ELABORAÇÃO REFERÊNCIAS**". UFSC – New Page 8. Brasil. http://www.bu.ufsc.br/framerefer.html. Acesso em 26/08/2006.
- ZANETTI, Eni Maria de Souza Pinto. **"COMO FAZER REFERÊNCIAS: bibliográficas, eletrônicas e demais formas de documentos".** FUNCAB. Brasil. http://www.funcab.br/Paginas/referencias.htm. Acesso em 26/08/2006.

6.1 Apêndice 1

Plano de Projeto

6.1 PLANO DE PROJETO

6.1.1 Introdução

6.1.1.1 Escopo e Propósito do Documento

Este documento apresenta uma proposta de desenvolvimento de software para atender às necessidades das Coordenações de Estágios e do curso de Gestão da Informação da UFPR como forma de propiciar o acompanhamento da qualidade dos mesmos.

Cabe ressaltar que esta aplicação funcionará em ambiente Intranet, cuja infra-estrutura está em fase desenvolvimento por equipe de profissionais e que a mesma se integrará para permitir a sua utilização com eficiência e eficácia.

O presente documento é constituído pelo detalhamento dos objetivos do projeto, um estudo sobre os riscos que podem afetar seu desenvolvimento, cronogramas e estimativas, organização do pessoal bem como as principais funções que serão desempenhadas pelo mesmo.

6.1.2 Objetivos

Módulo Estágio:

Sistema para gerenciamento de estágios dos alunos da Universidade Federal.

Tem como objetivo principal gerir todo o processo de estágio no âmbito da Universidade Federal do Paraná, controlando desde o convênio com as empresas até os termos de compromissos gerados na efetivação de um contrato de estágio.

Clientes:

Pró-Reitoria de Graduação e Ensino Profissionalizante - PROGRAD

Coordenação Geral de Estágios - CGE

Módulo Egressos:

Sistema de informações de alunos egressos do Curso de Gestão da Informação.

Tem como objetivo principal a manutenção do vínculo aluno - Universidade e também realização de pesquisa de informações para posterior melhoria do currículo do curso.

Clientes:

- Coordenação do Curso de Gestão da Informação - Professor José Simão

Professora Lígia Kraumer

6.1.1.2.1 Funções Principais

Módulo Egressos:

O Módulo Egressos prevê as seguintes funcionalidades:

- Formulário para cadastro dos alunos egressos;
- Formulário para avaliação do curso por parte dos egressos;
- Geração de relatórios para controle.

Módulo Estágio:

O módulo estágios será subdividido em visões, como a seguir:

Visão Coordenação de Estágio:

- Inserção de usuários com opção de cadastramento de membros da Coordenação de Estágios, membros de empresas ou alunos além de opção para editar os cadastros;
- Cadastro e gerenciamento de convênios das empresas;
- Cadastro de empresas;
- Gerenciamento das habilidades/requisitos cadastradas (os);
- Inserção de Cursos;
- Alteração de cadastro pessoal.

Visão Empresa:

- Realizar inclusão de vagas disponíveis para estágio, bem como gerenciamento das vagas preenchidas para controlar as mesmas.
- Visualização e gerenciamento dos Termos de Compromisso;
- Visualização e gerenciamento das vagas disponibilizadas no sistema.
- Alteração de cadastro pessoal.

Visão Aluno:

- Busca de Vagas;
- Alteração de cadastro pessoal, com opções de inserção de habilidades e horários disponíveis para estágio.

6.1.2 Estimativas de Projeto

6.1.2.1 Dados históricos usados nas estimativas

Módulo Egressos:

Documentação em papel armazenada na coordenação do Curso de Gestão da Informação. Essa documentação consiste em formulários onde se têm armazenados nome, telefone e por vezes o e-mail dos alunos egressos.

Módulo Estágio:

Dados dos alunos da Universidade para encaminhamento a vagas de estágio. Esses dados são provenientes do CCE, ou seja, a principio usaremos o modelo de dados do CCE para montar o sistema. Utilizaremos também como base o sistema que a PROGRAD já utiliza para controle de estágios, implementando obviamente as alterações necessárias para que o sistema se torne realmente eficiente aos usuários de ambas as visões.

6.1.2.2 Técnicas de estimativas

Não foram utilizadas técnicas especificas para obtenção das estimativas, porém tanto para o módulo de estágios quanto para o módulo de egressos, foi elaborada uma proposta de projeto baseada nos requisitos levantados junto aos clientes visando a aprovação do plano de projeto para que se inicie o desenvolvimento do mesmo.

6.1.3 Riscos do Projeto

- Desconhecimento da ferramenta de desenvolvimento do projeto:

Os integrantes da equipe não possuem habilidade na utilização das ferramentas de desenvolvimento do sistema, o que poderá ocasionar atrasos no projeto.

Grau : Médio.

- Medidas Preventivas:

Realização de estudo maciço das ferramentas de desenvolvimento individualmente e em grupo.

- Dependência do cliente para aquisição de informações relevantes:

Grau: Elevado.

O bom andamento do projeto e até mesmo a definição da forma de entrada de dados é dependente da aprovação/liberação ou não de acesso aos bancos com informações importantes ao sistema, essa dependência pode ocasionar atrasos, pois poderá atrasar a definição dos planos de ação.

- Medidas Preventivas:

Realizar reunião com o cliente para verificação dos acessos.

- Acesso ao Cliente:

Grau : Elevado.

Devido á política interna o acesso ao cliente pode se tornar difícil ou até inexistente.

- Medidas Preventivas:

Conversas com os responsáveis pelos Núcleos e havendo realmente a indisponibilidade de acesso, atender a demanda de acordo com os conhecimentos pré-existentes do processo.

- Projeto realizado em módulos:

Grau : Elevado.

Como o projeto esta sendo desenvolvido por três equipes, os módulos projetados deverão seguir um padrão de interface para que seja possível sua integração após o término do desenvolvimento.

- Medidas Preventivas:

Boa comunicação entre as equipes é um fator essencial.

- Integração da Equipe:

Grau: Médio

O entrosamento interno da equipe é fator relevante para o desenvolvimento do projeto, fatores externos podem afetar sua execução.

- Medidas Preventivas:

Definir responsabilidades entre os membros

- Fator Tempo:

Grau: Elevado.

O potencial de trabalho a ser realizado fora das dependências da Universidade é um fator de preocupação relevante, pois a responsabilidade com outras atividades pode utilizar tempo que seria necessário para execução do projeto.

- Medidas Preventivas:

Definição de prazos e responsabilidades entre os membros da equipe.

6.1.4 Recursos do projeto

6.1.4.1 Pessoal:

- Equipe do projeto composta por alunos de graduação do curso de Tecnologia em Informática da UFPR, orientados pelo professor Mauro José Belli.

  Alunos(as):

  - Evelize Amaral dos Santos
  - Leonardo Camargo
  - Luana Boganika
  - Ricardo Hillbrecht

- Considera-se também como recursos pessoais do projeto os clientes, pois deles retiraremos as informações necessárias para a construção do sistema:

  - Coordenação do Curso de Gestão da Informação - Professor José Simão e Professora Lígia Kraumer.
  - Pró-Reitoria de Graduação e Ensino Profissionalizante - PROGRAD
  - Coordenação Geral de Estágios - CGE

6.1.4.2 Hardware e Software

6.1.4.2.1 Hardware

- Máquinas situadas no laboratório 3 da Escola Técnica da UFPR com as seguintes configurações: Intel Celeron 1.7 Ghz com memória de 128 Mb.

- Equipamentos pessoais.

6.1.4.2.2 Software

- O sistema será desenvolvido em JSP (Java Server Pages) com banco de dados em PostGreSQL, utilizando as seguintes ferramentas:
  - Rational Rose e DBDesigner para modelagem UML.
  - Ms Project 2003 para o cronograma das datas.
  - NetBeans IDE 4.1 e Dreamweaver MX para o desenvolvimento.
  - PostGreSQL 8.0 para o banco de dados.

6.1.5 Organização do Pessoal

6.1.5.1 Estrutura da Equipe

Todos os integrantes da equipe serão responsáveis pelo projeto como um todo, ou seja, participarão de todas as atividades que o projeto envolve, porém, para facilitar a supervisão e coordenação das tarefas designaremos uma responsabilidade a cada membro:

Evelize Amaral dos Santos: documentação, análise do sistema e testes.

Leonardo Camargo: documentação e pesquisa.

Luana Boganika: análise do sistema , telas e documentação

Ricardo Hillbrecht: cronograma das atividades, implementação.

1.4 Cronograma Módulo Estágio e Egressos

Nome da Atividade	Duração	Início	Fim	A.P.*
Levantamento de Requisitos	12 days	quarta-feira, 4 de maio de 2005	quinta-feira, 19 de maio de 2005	
Reunião Clientes	1 day?	sexta-feira, 20 de maio de 2005	sexta-feira, 20 de maio de 2005	1
Elaborar Casos de Uso	8 days	segunda-feira, 23 de maio de 2005	quarta-feira, 1 de junho de 2005	2
Elaborar Diagrama de Telas	5 days	quinta-feira, 2 de junho de 2005	quarta-feira, 8 de junho de 2005	3
Elaborar Diagrama de Classes	5 days	quinta-feira, 9 de junho de 2005	quarta-feira, 15 de junho de 2005	4
Elaborar DER	6 days	quinta-feira, 16 de junho de 2005	quinta-feira, 23 de junho de 2005	5
Elaborar Diagramas Seqüência	6 days	sexta-feira, 24 de junho de 2005	sexta-feira, 1 de julho de 2005	6
Elaborar Diagramas de Colaboração	1 day?	segunda-feira, 4 de julho de 2005	segunda-feira, 4 de julho de 2005	7
Elaborar Diagramas de Estados	6 days	terça-feira, 5 de julho de 2005	terça-feira, 12 de julho de 2005	8
Elaborar Diagramas de Atividades	8 days	quarta-feira, 13 de julho de 2005	sexta-feira, 22 de julho de 2005	9
Implementar Estrutura do Banco de Dados	5 days	segunda-feira, 25 de julho de 2005	sexta-feira, 29 de julho de 2005	10
Criar Tabelas e seus Relacionamentos	2 days	segunda-feira, 1 de agosto de 2005	terça-feira, 2 de agosto de 2005	11
Codificação	35 days	quarta-feira, 3 de agosto de 2005	terça-feira, 20 de setembro de 2005	12
Implementar classes	5 days	quarta-feira, 21 de setembro de 2005	terça-feira, 27 de setembro de 2005	13
Construir Interfaces	10 days	quarta-feira, 28 de setembro de 2005	terça-feira, 11 de outubro de 2005	14
Adicionar funcionalidade nas telas	30 days	quarta-feira, 12 de outubro de 2005	terça-feira, 22 de novembro de 2005	15
Finalizar Documentação	5 days	quarta-feira, 23 de novembro de 2005	terça-feira, 29 de novembro de 2005	16
Revisão e Testes	15 days	quarta-feira, 30 de novembro de 2005	terça-feira, 20 de dezembro de 2005	17

* A.P.: Atividade Predecessora.

6.2 Apêndice 2

Modelagem do Sistema

6.2.0 MODULO ESTÁGIO

6.2.0.1 Descrição dos Casos de Uso – Módulo Estágio

6.2.0.1.1 Visão Coordenação de Estágios Universidade Federal do Paraná

6.2.0.1.1.1 Módulo Estágio - Inserir Usuário

Caso de uso: Inserir Usuário.

Ator: Coordenação de Estágios.

Descrição Resumida:

O funcionário da Coordenação de Estágios poderá realizar inclusão dos usuários do sistema, sendo eles membros da própria coordenação, membros das empresas ou alunos.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu usuário e senha. O sistema de Estágios informará caso seja digitado usuário ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Inserir Usuário e seleciona o Perfil do usuário que deseja cadastrar. Logo após isso, preenche o formulário que será carregado na tela.

Para os campos de preenchimento obrigatório o sistema realiza validação e retorna mensagem de erro caso algum campo fique em branco. (E2)

Fluxos Alternativos:

A1: Cadastrar Membro do Grupo de Administradores (Coordenação de Estágios):

Este fluxo será executado quando esta opção for selecionada ao executar-se o fluxo Inserir Usuário. Será carregado na tela o formulário específico para cadastro desse membro.

A2: Cadastrar Membro da Empresa:

Este fluxo será executado quando esta opção for selecionada ao executar-se o fluxo Inserir Usuário. Também possui formulário específico para o perfil.

A3: Cadastrar Aluno:

Este fluxo será executado quando esta opção for selecionada ao executar-se o fluxo Inserir Usuário. Possui formulário específico para o perfil.

A4: Salvar:

Se os dados forem inseridos corretamente e validados pelo sistema, as informações serão salvas no banco de dados e o sistema retornará a mensagem "OPERAÇÃO REALIZADA COM SUCESSO!".

A5: Reset:

Limpa os campos do formulário.

A6: Voltar:

Retorna à tela anterior.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

E2: Se não houver nenhum campo preenchido o sistema retorna quais os campos que devem ser preenchidos.

Cenários

Cenário Principal

Carlos, funcionário da coordenação de Estágios, informa seu usuário e senha e acessa o Módulo de Egressos na Intranet Acadêmica. Em seguida acessa a opção Inserir Usuário. Ele precisa inserir um aluno, então seleciona a opção Aluno e preenche os campos do formulário com os dados pessoais do aluno, e insere também as habilidades do mesmo, bem como horários disponíveis para realização do estágio.

Após preencher os campos, clica no botão salvar e os dados são inseridos com sucesso.

Cenários Secundários

- Após preencher o formulário e clicar no botão Salvar, o sistema retorna a mensagem que o CPF informado já está cadastrado.
- Carlos preenche o formulário deixando o campo Data de Nascimento em branco. O sistema retorna a mensagem indicando a necessidade de preenchimento deste campo. (Obs.: esta validação é realizada em todos os campos para preenchimento).

6.2.0.1.1.2 Módulo Estágio - Convênios

Caso de uso: Convênios.

Ator: Coordenação de Estágios.

Descrição Resumida:

O funcionário da Coordenação de Estágios poderá gerenciar a parte relacionada aos convênios e ao cadastramento de novas empresas na base de dados.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu usuário e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Convênios, podendo realizar uma busca pelas empresas conveniadas, utilizando como parâmetro o CNPJ ou a Razão Social da mesma. No retorno da pesquisa o usuário poderá visualizar o código da empresa, o nome, o início e a validade do convênio firmado, campo de descrição para observações e campo situação (válido ou expirado). Poderá também Excluir o Convênio das empresas cadastradas.

Fluxos Alternativos:

A1: Novo Convênio:

Este fluxo possibilita ao usuário inserir um novo convênio para uma nova empresa cadastrada e que ainda não tenha seu convenio no sistema. Ou ainda inserir um novo convênio para uma empresa anteriormente cadastrada e que teve seu convênio vencido. O sistema fará validação dos campos a serem preenchidos. (E2).

A2: Nova Empresa:

Possibilita ao usuário inserir uma nova empresa. Ao clicar nesta opção, carrega-se um formulário para preenchimento dos dados da empresa. Caso a empresa seja um Agente de

Integração o usuário deverá selecionar a check box "Agente Integrador" antes de concluir o cadastro. O sistema fará validação dos campos de preenchimento obrigatório. (E2).

A3: Salvar:

Em ambos os fluxos anteriores, o botão Salvar irá inserir as informações no banco de dados se estiverem com o preenchimento correto.

A4: Reset:

Limpa todos os campos do formulário, está presente nos fluxos Nova Empresa e Novo Convênio.

A5: Página Inicial:

É um link que retorna para a página inicial do sistema, mostrando ao usuário as opções do menu.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

E2: Caso algum campo de preenchimento obrigatório não tenha sido preenchido, ou tenha ocorrido erro de preenchimento de algum campo, de acordo com a validação, o sistema retorna o ocorrido alertando o usuário para o preenchimento.

Cenários

Cenário Principal

Carlos, funcionário da coordenação de Estágios da UFPR, informa seu usuário e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Convênios e a opção Nova Empresa.

Após preencher os campos para cadastrar uma nova empresa, clica no botão salvar. Retornando ao ao fluxo Convênios, o fluxo a ser executado é Novo Convênio, onde ele estabelecerá um convênio para a nova empresa que acabou de cadastrar.

Feito isso clica em Salvar e mais um convênio está efetuado.

Cenários Secundários

- Carlos verifica, ao executar o fluxo Convênios, que a empresa Faz Tudo LTDA está com o convênio inativo, ou seja, venceu a validade. Ele então, conforme anterior contato feito com a empresa e medidas burocráticas seguidas corretamente constata que a mesma deseja cancelar seu convênio com a UFPR. Para manter a base atualizada, ele exclui esse convênio, mas a empresa ainda está na base de dados, caso deseje firmar novo convênio posteriormente.
- Carlos verifica, ao executar o fluxo Convênios, que a empresa Faz Tudo LTDA está com o convênio Inativo, ou seja, venceu a validade. Ele então, conforme anterior contato feito com a empresa e medidas burocráticas seguidas corretamente constata que a mesma renovou seu Convênio. Ele então clica em Novo Convênio, e atualiza as informações referentes ao Convênio com essa empresa, deixando seu status como Ativo novamente.

6.2.0.1.1.3 Módulo Estágio - Agentes de Integração

Caso de uso: Agentes de Integração

Ator: Coordenação de Estágios

Descrição Resumida:

O funcionário da Empresa visualiza os agentes de integração no sistema, tais agentes são empresas externas que assim como a Coordenação de Estágios gerenciam todo o processo de estágio dos estudantes.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário da Coordenação de Estágios acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Agentes de Integração. É carregada uma tela com esses agentes já cadastrados e a opção Excluir (E2). Clicando em Página Inicial retorna à home do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Exclui o Agente de Integração respectivo.

Cenários

Cenário Principal

Carlos, funcionário da Coordenação de Estágios, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Agentes de Integração e visualiza todos os agentes de integração cadastrados no sistema.

Cenários Secundários

- Carlos clica na opção Excluir do Agente CIEE, e o sistema retorna mensagem: Impossível excluir esse Agente de Integração. Tente excluir os termos de compromisso relacionados.

6.2.0.1.1.4 Módulo Estágio - Cursos

Caso de uso: Cursos

Ator: Coordenação de Estágios

Descrição Resumida:

Permite ao usuário inserir ou excluir os cursos que estão disponíveis na Universidade Federal do Paraná.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário da Coordenação de Estágios acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Cursos, onde é carregada uma tela com uma lista de todos os cursos já cadastrados. O usuário poderá inserir novo curso ou excluir um já existente.

O usuário também tem a opção "Página Inicial" que retorna à home do sistema.

Fluxos Alternativos:

A1: Inserir Curso

Este fluxo será executado quando o usuário desejar inserir um novo curso na base. É carregado um formulário para preenchimento do nome do curso e da duração do mesmo. O usuário poderá Salvar (A.1.1), Reset (A.1.2).

A.1.1: Salvar

Insere os dados na base e retorna mostrando a listagem total dos cursos cadastrados.

A.1.2: Reset:

Limpa os campos do formulário.

A.2: Excluir:

Exclui o curso da base. Caso o curso possua vagas vinculadas à ele no sistema significa que é um curso ativo, então o sistema retorna mensagem: "Impossível excluir esse curso. Tente excluir os Alunos e Vagas relacionados."

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Carlos, funcionário da Coordenação de Estágios, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Cursos visualizando uma lista com os cursos e seus respectivos tempos de duração. Carlos decide então inserir um novo curso, e executando esse fluxo preenche os campos com o nome e duração do curso. Ele clica em Salvar, e pode novamente verificar a listagem dos cursos, incluindo o que acabou de ser inserido.

Cenários Secundários

- Carlos decide agora excluir um dos cursos cadastrados e clica em Excluir, porém recebe a seguinte mensagem: Impossível excluir esse curso. Tente excluir os Alunos e Vagas relacionados." Isso significa que este curso não poderá ser excluído pois ainda é ativo, possui alunos matriculados e vinculados à vagas de estágios.
- Carlos deseja inserir um novo curso. Ele insere o nome Tecnologia em Informática e a duração 3 anos. O sistema retorna mensagem alertando que esse curso já se encontra cadastrado na base.

6.2.0.1.1.5 Módulo Estágio - Empresas

Caso de uso: Empresas

Ator: Coordenação de Estágios

Descrição Resumida:

Permite ao usuário visualizar o cadastro das empresas conveniadas com a Universidade Federal do Paraná.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário da Coordenação de Estágios acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Empresas, onde é carregada uma tela com um campo para pesquisa por Razão Social da empresa.

O usuário também tem a opção "Página Inicial" que retorna à home do sistema.

Fluxos Alternativos:

A1: Pesquisar

Este fluxo será executado quando o usuário digitar a Razão Social ou parte da Razão Social de uma empresa, e o sistema retornará na tela os resultados.

A.1.1: Excluir

Após o retorno da pesquisa, o usuário tem a opção de excluir essa empresa da base, mas deve tomar cuidado, pois ao excluir uma empresa serão também excluídos os usuários, termos de compromisso, vagas e convênios a ela associados.

A2: Editar

O usuário poderá alterar informações no cadastro da empresa em questão.

A.2.1: Salvar
As informações alteradas pelo usuário serão salvas no sistema.

A.2: Página Inicial:

Retorna à página inicial do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Carlos, funcionário da Coordenação de Estágios, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Empresas e no campo para pesquisar digita o termo "Mer". A pesquisa retorna o cadastro da empresa Mercadorama. Carlos visualiza e retorna à página principal.

Cenários Secundários

Carlos realiza uma outra pesquisa, agora buscando o termo "Bosch". A pesquisa retorna a empresa "Bosch do Brasil", e Carlos exclui o cadastro da mesma, já que ela não tem mais vínculos com a UFPR.

6.2.0.1.1.6 Módulo Estágio - Vagas

Caso de uso: Vagas

Ator: Coordenação de Estágios

Descrição Resumida:

Permite ao usuário visualizar as vagas disponíveis no sistema, separadas por curso.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário da Coordenação de Estágios acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1).

Uma vez logado no sistema, escolhe a opção Vagas, então é carregada uma tela com uma combo com os cursos cadastrados, escolhe-se uma opção (um curso) e o sistema carrega uma listagem com as vagas disponíveis para aquele curso com as informações (empresa/setor/carga horária/bolsa/sexo/desde/requisitos/horário e situação).

O usuário pode consultar vagas de outros cursos ou ainda tem a opção "Página Inicial" que retorna à Home do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Se o curso escolhido não tiver nenhuma vaga relacionada retorna a mensagem: NÃO HÁ VAGAS NO MOMENTO.

Cenários

Cenário Principal

Carlos, funcionário da Coordenação de Estágios, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Vagas e escolhe na comboBox o curso Tecnologia em Informática. Ele visualiza todas as vagas disponíveis para este curso e retorna à página inicial.

Cenários Secundários

- Carlos escolhe agora outro curso, Agronomia, e recebe a mensagem: NÃO HÁ VAGAS NO MOMENTO. Isso significa que não há nenhuma vaga disponível para este curso. Ele retorna então à pagina inicial.

6.2.0.1.1.7 Módulo Estágio - Usuários

Caso de uso: Usuários

Ator: Coordenação de Estágios

Descrição Resumida:

Permite a busca e edição (exclusão também) dos usuários cadastrados no sistema.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário da Coordenação de Estágios acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Usuários, onde é carregada uma tela de busca, solicitando o nome ou o username do usuário (ou parte dele), para listar todos os usuários basta não preencher nada no campo de busca e clicar em Pesquisar. O sistema retorna uma listagem dos usuários encontrados com o username, nome, e-mail e a opção Excluir (A2). O funcionário pode editar o cadastro do usuário (A1).

Fluxos Alternativos:

A1: Editar Cadastro

Ao clicar no nome do usuário na listagem que retornou o sistema carrega a tela de cadastro do usuário o que permite sua edição, nesta tela são carregados todos os dados do usuário e possui as opções Salvar (A.1.1), Limpar (A.1.2) e Voltar (A.1.3).

A.1.1: Salvar

Salvas as alterações realizadas no sistema.

A.1.2: Reset

Limpa a tela de cadastro.

A.1.3: Voltar

Retorna para a listagem com o resultado da pesquisa.

A2: Excluir

Permite a exclusão de usuários.

A3: Buscar Novamente

Retorna para a tela inicial de busca.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Carlos, funcionário da Coordenação de Estágios, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Usuários e digita no campo "Santos", clicando em pesquisar, retornarão 20 nomes de usuários, que contém a palavra Santos. Ele então clica no nome da usuária Maria dos Santos, e altera seu endereço. Ele salva a alteração e retorna para a pesquisa.

Cenários Secundários

- Carlos decide excluir a usuária Maria dos Santos, então ao retorno da pesquisa clica em Excluir no respectivo nome da usuária.

6.2.0.1.1.8 Módulo Estágio – Termos de Compromisso

Caso de uso: Termos de Compromisso.

Ator: Coordenação de Estágios.

Descrição Resumida:

O funcionário da Coordenação de Estágios poderá gerenciar a parte relacionada aos termos de compromisso.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu usuário e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Termos de Compromisso, podendo realizar uma busca pelos termos cadastrados, utilizando como parâmetro o CNPJ da empresa ou o nome do aluno. No retorno da pesquisa o usuário poderá visualizar os dados referentes ao termo de compromisso, tendo a opção de excluir esse termo (A2) ou fazer nova busca (A3).

Fluxos Alternativos:

A1: Pesquisar:

O usuário escolhe o parâmetro a ser utilizado e digita os dados no campo. Caso ele opte pelo parâmetro "Nome do Aluno", deixe o campo em branco e clique em Pesquisar o sistema retornará todos os termos de compromisso cadastrados. (E2)

A2: Excluir:

Este fluxo possibilita ao usuário excluir um termo de compromisso da base de dados.

A3: Buscar Novamente:

Possibilita ao usuário retornar à tela de pesquisa para realizar nova busca com novos parâmetros.

A4: Página Inicial:

É um link que retorna para a página inicial do sistema, mostrando ao usuário as opções do menu.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

E2: Caso algum o usuário opte pelo parâmetro "CNPJ" para realizar a pesquisa e deixe o campo em branco, o sistema retornará erro, sendo assim o usuário deverá preencher corretamente o campo com o CNPJ referido.

Cenários

Cenário Principal

Carlos, funcionário da coordenação de Estágios da UFPR, informa seu usuário e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Termos de Compromisso e escolhe pesquisar por CNPJ. Ele então digita um CNPJ, que retorna a empresa "C&A Ltda". Ele visualiza as informações de seu interesse, e clica em "Buscar Novamente" para realizar nova busca.

Cenários Secundários

- Carlos deseja pesquisar todos os termos de compromisso existentes na base, então ele seleciona o parâmetro "Nome do Aluno" e não preenche nada no campo. Ao clicar em pesquisar todos os termos são exibidos, e ele exclui os termos de alguns alunos que já deixaram de realizar estágio.

BIBLIOTECA DO SETOR DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA DA UFPR

6.2.0.1.1.9 Módulo Estágio – Estágio Interno

Caso de uso: Estágio Interno.

Ator: Coordenação de Estágios.

Descrição Resumida:

O funcionário da Coordenação de Estágios poderá gerenciar a parte relacionada às vagas ofertadas pela UFPR, no caráter de empresa/Unidade Concedente de Estágio, caracterizando-se assim como Estágio Interno.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o ser funcionário da coordenação de Estágios da UFPR acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado, escolhe a opção Estágio Interno. O sistema traz uma tela com todas as vagas que a UFPR como empresa disponibilizou e o usuário tem as opções de Inserir Vaga (A1), Inserir Requisitos (A2), Inserir Horário (A3), Editar (A4), Excluir (A5) e Preencher (A6). Ele poderá também retornar à página inicial do sistema ao clicar em "Página Inicial".

Fluxos Alternativos:

A.1: Inserir Vaga

Ao executar esse fluxo é carregada uma tela com um formulário para que o usuário da empresa preencha de acordo com as especificações para a vaga. Após terminar o preenchimento clica em Salvar (A.1.1).

A.1.1: Salvar

Salva os novos dados inseridos no formulário e redireciona para a página onde estão listadas todas as vagas, inclusive a nova vaga inserida.

A. 2: Inserir Requisitos

Este fluxo é executado quando é inserida uma nova vaga, clica-se sobre o campo Requisitos dessa vaga e a seguir em Inserir Requisitos. O usuário poderá inserir Novo Requisito (A.2.1). O usuário também tem a opção de voltar para a tela de vagas clicando em "Voltar para Vagas".

A.2.1: Novo Requisito:

Ao executar este fluxo aparecerá o código da vaga que foi inserida, e um campo para ser incluído o nome do requisito.

A.2.1.1: Salvar:

Salva o novo requisito inserido.

A.2.1.2: Reset:

Limpa o campo Requisito.

A.2.13: Vagas

Retorna para a página onde são listadas as vagas.

A.3: Inserir Horário

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Horário da respectiva vaga, aparece a mensagem: "NÃO HÁ HORÁRIOS CADASTRADOS PARA ESSA VAGA", já que ainda não tinha sido definido um horário. O usuário clica então em Inserir Horário e preenche os campos de horário de início e horário de término relativos a carga horária diária no estágio. Executa o fluxo Salvar (A.3.1) e insere o novo horário. Poderá assim inserir mais horários disponíveis para a mesma vaga, e até mesmo excluí-los (A.3.2). Tem a opção de voltar para a tela das vagas listadas ("Vagas") (A.3.3).

A.3.1: Salvar

Salva o novo horário inserido.

A.3. 2: Excluir

Exclui o respectivo horário.

A.3.3: Vagas

Retorna para a página onde são listadas as vagas.

A.4: Editar

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Editar da respectiva vaga, aparece o nome da empresa, cujo funcionário é o usuário do sistema, e o cadastro da vaga para que possa ser alterado. O usuário poderá limpar todos os campos (A.4.1) E após inserir as novas informações clicar em salvar (A.4.2).

A.4.1: Reset

Retorna os campos à condição anterior.

A.4.2: Salvar

Salva as alterações e retorna para a tela principal da lista de vagas.

A.3.3: Vagas

Retorna para a página onde são listadas as vagas.

A5: Excluir

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Excluir da respectiva vaga, a mesma é excluída do sistema.

A6: Preencher

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Preencher da respectiva vaga, o usuário é redirecionado para uma página de Busca de Estagiário, onde ele irá digitar todo ou parte do nome do estagiário que preencherá essa vaga e clicará em Pesquisar (A.6.1).

A.6.1: Pesquisar

Este fluxo retornará uma lista com o(s) nome(s) do(s) estagiário(s) pesquisado(s). Então o usuário selecionará o estagiário que preencherá a vaga e clicará em Salvar (A.6.1.1).

A.6.1.1: Salvar

Redireciona para a criação de um novo termo de compromisso. É carregada uma tela onde serão inseridos dados do contrato, como a vigência do mesmo e o agente de

integração, que nesse caso poderá ser a própria Coordenação de Estágios da UFPR ou um agente externo. Caso o estágio seja obrigatório, o usuário deverá selecionar a opção "Estágio Obrigatório" ao término do preenchimento do formulário. Após inserir esses dados, clica em Salvar e será gerado um Termo de Compromisso em um arquivo .pdf (Adobe Acrobat Reader), que poderá se impresso através do menu do próprio navegador que o usuário estiver utilizando.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Walter, funcionário da coordenação de Estágios da UFPR, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Vagas Estágio Interno, pois deseja inserir uma nova vaga que está disponível na UFPR no setor de Ciências Biológicas.

Ele então preenche os dados relativos à vaga, e também os horários e requisitos necessários.

Dias depois, a vaga já foi preenchida, depois de todo o processo de entrevista com o aluno, Walter agora irá atualizar no sistema o preenchimento desta vaga. Ele então acessa novamente o sistema e na opção Vagas Estágio Interno, clica em "Preencher" na referida vaga, localizando o aluno Manoel Ribas na base de estagiários e inserindo os dados do termo de compromisso. Ele clica em Salvar e a operação é realizada com sucesso, o termo de compromisso é gerado em .pdf, e Walter o imprime.

Cenários Secundários

- Walter está para atualizar o sistema com o preenchimento de uma vaga, porém houve um problema em relação aos horários do aluno a ser contratado, pois ele estuda no período vespertino e não poderá cumprir a carga horária de 6h diárias. Então antes de executar o fluxo Preencher ele irá clicar no campo Horário e excluir o horário atual, inserindo assim o horário correto, e também irá executar o fluxo Editar, para editar o campo Horas/Semana da vaga conforme necessário. Aí então poderá executar o fluxo

Preencher, inserindo os dados corretos e disponíveis do aluno no termo de compromisso.

- Walter acessa o sistema para verificar o quadro de vagas. Percebe que existem duas vagas já defasadas, que não foram preenchidas e que também não estão mais abertas. Ele então executa o fluxo Excluir e exclui essas vagas do sistema.

6.2.0.1.1.10 Módulo Estágio – Relatório de Seguro

Caso de uso: Relatório de Seguro.

Ator: Coordenação de Estágios.

Descrição Resumida:

O funcionário da Coordenação de Estágios poderá gerar os Relatórios de Seguros, referentes aos benefícios de seguros dos estagiários.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu usuário e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Relatório de Seguro, e poderá gerar os relatórios mensais de estagiários para benefício de Seguro.

Os relatórios são gerados de acordo com mês e ano especificados, que poderão ser selecionados nas combo box presentes na página. Ao final da geração, o usuário precisa envia-los às respectivas seguradoras, para que as mesmas tomem as devidas providências. O documento é gerado em um arquivo *.pdf* (Adobe Acrobat Reader), através do botão "Enviar", e o usuário poderá imprimi-lo utilizando o menu do próprio navegador.

Fluxos Alternativos:

O usuário poderá selecionar as seguintes opções para gerar um relatório:

A1: Cancelar o Seguro:

Gerará um relatório pedindo o cancelamento do seguro para os estagiários que tenham deixado de realizar estágio em determinado mês e ano.

A2: Manter o Seguro

Gerará um relatório pedindo que o seguro seja mantido para os estagiários que em determinado mês e ano manteram-se permanentes em seus estágios.

A3: Incluir no Seguro:

Gerará um relatório pedindo a inclusão dos estagiários que em determinado mês e ano iniciaram um estágio.

A4: Relatório Geral:

Gerará um relatório com todos os campos anteriores para determinado mês e ano selecionados.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Carlos, funcionário da coordenação de Estágios da UFPR, informa seu usuário e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Relatório de Seguro e escolhe o item Relatório Geral para gerar um relatório a respeito dos itens: Cancelar Seguro, Manter Seguro e Incluir no Seguro. Então clica no botão enviar e o relatório é gerado, ele o imprimi através de seu navegador.

Cenários Secundários

- Carlos deseja gerar um relatório sobre o item Cancelar Seguro para o mês de abril do ano de 2005, mas não houve cancelamentos para este mês neste ano, então o relatório é gerado em branco, e Carlos sai da aplicação.

6.2.0.1.1.11 Módulo Estágio – Email do Sistema

Caso de uso: Email do Sistema.

Ator: Coordenação de Estágios.

Descrição Resumida:

O funcionário da Coordenação de Estágios poderá configurar um endereço de email, que enviará a senha dos usuários quando estes a solicitarem em caso de esquecimento.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação de Estágios da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu usuário e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Email do Sistema, e poderá configurar uma conta de email, juntamente com seu servidor smtp, username, porta e senha, para que o sistema envie automaticamente a senha de um usuário quando este a solicitar.

Fluxos de Exceção:

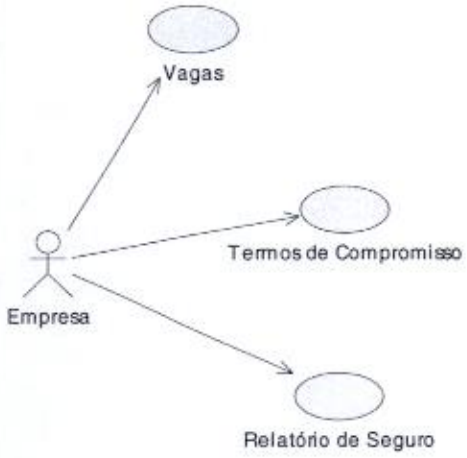
E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

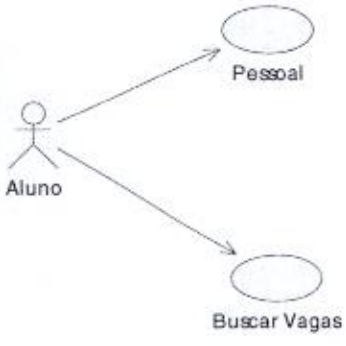
Cenários

Cenário Principal

Carlos, funcionário da coordenação de Estágios da UFPR, informa seu usuário e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Email do Sistema e insere os dados de uma conta de email, criada especialmente para esta funcionalidade no Módulo Estágio. Ele também insere a configuração do servidor smtp, username, porta e senha. Ele marca a check box para autenticar o servidor e clica em Enviar. Está finalizada a configuração do email do sistema.

6.2.0.1.2 Visão Empresas /Visão Aluno





6.2.0.1.2.1 Módulo Estágio - Vagas

Caso de uso: Vagas

Ator: Usuário da empresa

Descrição Resumida:

Permite ao usuário da empresa o gerenciamento das vagas disponíveis para estágio.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário de uma empresa conveniada com a UFPR.

Fluxo Principal: Este caso de uso começa quando o usuário da empresa conveniada com a UFPR acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado, escolhe a opção Vagas. O sistema traz uma tela com todas as vagas que a empresa disponibilizou e o usuário tem as opções de Inserir Vaga (A1), Inserir Requisitos (A2), Inserir Horário (A3), Editar (A4), Excluir (A5) e Preencher (A6). Ele poderá também retornar à página inicial do sistema ao clicar em "Página Inicial".

Fluxos Alternativos:

A.1: Inserir Vaga

Ao executar esse fluxo é carregada uma tela com um formulário para que o usuário da empresa preencha de acordo com as especificações para a vaga. Após terminar o preenchimento clica em Salvar (A.1.1).

A.1.1: Salvar

Salva os novos dados inseridos no formulário e redireciona para a página onde estão listadas todas as vagas, inclusive a nova vaga inserida.

A. 2: Inserir Requisitos

Este fluxo é executado quando é inserida uma nova vaga, clica-se sobre o campo Requisitos dessa vaga e a seguir em Inserir Requisitos. O usuário poderá inserir Novo Requisito (A.2.1). O usuário também tem a opção de voltar para a tela de vagas clicando em "Voltar para Vagas".

A.2.1: Novo Requisito:

Ao executar este fluxo aparecerá o código da vaga que foi inserida, e um campo para ser incluído o nome do requisito.

A.2.1.1: Salvar:

Salva o novo requisito inserido.

A.2.1.2: Reset:

Limpa o campo Requisito.

A.2.13: Vagas

Retorna para a página onde são listadas as vagas.

A.3: Inserir Horário

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Horário da respectiva vaga, aparece a mensagem: "NÃO HÁ HORÁRIOS CADASTRADOS PARA ESSA VAGA", já que ainda não tinha sido definido um horário. O usuário clica então em Inserir Horário e preenche os campos de horário de início e horário de término relativos a carga horária diária no estágio. Executa o fluxo Salvar (A.3.1) e insere o novo horário. Poderá assim inserir mais horários disponíveis para a mesma vaga, e até mesmo excluí-los (A.3.2). Tem a opção de voltar para a tela das vagas listadas ("Vagas") (A.3.3).

A.3.1: Salvar

Salva o novo horário inserido.

A.3. 2: Excluir

Exclui o respectivo horário.

A.3.3: Vagas

Retorna para a página onde são listadas as vagas.

A.4: Editar

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Editar da respectiva vaga, aparece o nome da empresa, cujo funcionário é o usuário do sistema, e o cadastro da vaga para que possa ser alterado. O usuário poderá limpar todos os campos (A.4.1) E após inserir as novas informações clicar em salvar (A.4.2).

A.4.1: Reset

Retorna os campos à condição anterior.

A.4.2: Salvar

Salva as alterações e retorna para a tela principal da lista de vagas.

A.3.3: Vagas

Retorna para a página onde são listadas as vagas.

A5: Excluir

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Excluir da respectiva vaga, a mesma é excluída do sistema.

A6: Preencher

Este fluxo é executado na página que está listando todas as vagas que a empresa disponibilizou. Ao clicar sobre o campo Preencher da respectiva vaga, o usuário é redirecionado para uma página de Busca de Estagiário, onde ele irá digitar todo ou parte do nome do estagiário que preencherá essa vaga e clicará em Pesquisar (A.6.1).

A.6.1: Pesquisar

Este fluxo retornará uma lista com o(s) nome(s) do(s) estagiário(s) pesquisado(s). Então o usuário selecionará o estagiário que preencherá a vaga e clicará em Salvar (A.6.1.1).

A.6.1.1: Salvar

Redireciona para a criação de um novo termo de compromisso. É carregada uma tela onde serão inseridos dados do contrato, como a vigência do mesmo e o agente de integração, que nesse caso poderá ser a própria Coordenação de Estágios da UFPR ou um agente externo. Caso o estágio seja obrigatório, o usuário deverá selecionar a opção "Estágio Obrigatório" ao término do preenchimento do formulário. Após inserir esses dados, clica em Salvar e será gerado um Termo de Compromisso em um arquivo .pdf (Adobe Acrobat Reader), que poderá ser impresso através do menu do próprio navegador que o usuário estiver utilizando.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Joselito, funcionário da empresa Faz Tudo Ltda, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Vagas, pois deseja inserir uma nova vaga que está disponível na empresa.

Ele então preenche os dados relativos à vaga, e também os horários e requisitos necessários.

Dias depois, a vaga já foi preenchida, depois de todo o processo de entrevista com o aluno, Joselito agora irá atualizar no sistema o preenchimento desta vaga. Ele então acessa novamente o sistema e na opção Vagas, clica em Preencher na referida vaga, localizando o aluno Manoel Ribas na base de estagiários e inserindo os dados do termo de compromisso. Ele clica em Salvar e é gerado o termo de compromisso em .pdf, e Joselito o imprime.

Cenários Secundários

- Joselito está para atualizar o sistema com o preenchimento de uma vaga, porém houve um problema em relação aos horários do aluno a ser contratado, pois ele estuda no período vespertino e não poderá cumprir a carga horária de 6h diárias. Então antes de executar o fluxo Preencher ele irá clicar no campo Horário e excluir o horário atual, inserindo assim o horário correto, e também irá executar o fluxo Editar, para editar o campo Horas/Semana da vaga conforme necessário. Aí então poderá executar o fluxo

Preencher, inserindo os dados corretos e disponíveis do aluno no termo de compromisso.

- Joselito acessa o sistema para verificar o quadro de vagas. Percebe que existem duas vagas já defasadas, que não foram preenchidas e que também não estão mais abertas. Ele então executa o fluxo Excluir e exclui essas vagas do sistema.

6.2.0.1.2.2 Módulo Estágio - Termos de Compromisso

Caso de uso: Termos de Compromisso

Ator: Usuário da empresa

Descrição Resumida:

Permite ao usuário visualizar os termos de compromisso cadastrados no sistema.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário de uma empresa conveniada com a UFPR.

Fluxo Principal: Este caso de uso começa quando o usuário da empresa conveniada com a UFPR acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Termos de Compromisso, onde é carregada uma listagem com os termos cadastrados no sistema, nesta lista constam as informações: Empresa, Estagiário, Inicio, Validade, Bolsa auxílio, horário, curso, setor, requisitos e agente de integração. O usuário poderá Editar (A1) os termos, Excluir (A2) ou ainda acessar a opção Página Inicial (A3), que retorna para a tela inicial do sistema.

Fluxos Alternativos:

A.1: Editar

O usuário poderá editar informações dos termos de compromisso conforme necessário.

A.1.1: Salvar

Gera um novo termo de compromisso, com as alterações realizadas no fluxo anterior. O usuário poderá imprimi-lo através do próprio navegador.

A.1.2: Voltar para Termos de Compromisso:

Redireciona o usuário para listagem de termos de compromisso que sua respectiva empresa possui com os estagiários alunos da UFPR.

A.2: Excluir

O usuário poderá excluir o termo de compromisso da base. Fazendo isso, a vaga volta a ficar com status Aberta e poderá ser preenchida por outro estagiário, ou ainda, poderá ser renovado o contrato deste estagiário, sendo necessário registrar um novo termo de compromisso com novas datas, excluindo-se então o anterior. Se a vaga deixar de existir na empresa, poderá ser excluída da base.

A.3: Pagina Inicial

Retorna para a página inicial do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Joselito, funcionário da empresa Faz Tudo Ltda, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Termos de Compromisso, pois deseja visualiza-los para verificar a validade de um deles.

Cenários Secundários

- Joselito verifica a validade de um dos termos de compromisso, da aluna Maria da Graça, pois seu contrato será renovado. Ele então excluir o termo de compromisso, e no Fluxo vagas prenche novamente com Maria das Graças e renova seu termo de compromisso.
- Joselito verifica a validade de um dos termos de compromisso e verifica que está expirado. Ele tem a informação que o contrato não será renovado, e a vaga não mais existirá na empresa. Ele exclui o termo de compromisso e em seguida, no fluxo Vagas, exclui a vaga em questão referida.
- Joselito verifica a validade de um dos termos de compromisso e verifica que está expirado. Ele tem a informação que o contrato não será renovado, mas a vaga continuára a existir na empresa e poderá ser preenchida por outro estagiário. A vaga então voltar a ficar com status Aberta no sistema.

6.2.0.1.2.3 Módulo Estágio – Relatório de Seguro

Caso de uso: Relatório de Seguro.

Ator: Usuário da empresa

Descrição Resumida:

O funcionário da empresa poderá gerar os Relatórios de Seguros, referentes aos benefícios de seguros dos estagiários.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser funcionário de uma empresa conveniada com a UFPR.

Fluxo Principal: Este caso de uso começa quando o usuário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu usuário e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Relatório de Seguro, e poderá gerar os relatórios mensais de estagiários para benefício de Seguro.

Os relatórios são gerados de acordo com mês e ano especificados, que poderão ser selecionados nas combo box presentes na página. Ao final da geração, o usuário precisa envia-los às respectivas seguradoras, para que as mesmas tomem as devidas providências. O documento é gerado em um arquivo *.pdf* (Adobe Acrobat Reader), através do botão "Enviar", e o usuário poderá imprimi-lo utilizando o menu do próprio navegador.

Fluxos Alternativos:

O usuário poderá selecionar as seguintes opções para gerar um relatório:

A1: Cancelar o Seguro:

Gerará um relatório pedindo o cancelamento do seguro para os estagiários que tenham deixado de realizar estágio em determinado mês e ano.

A2: Manter o Seguro

Gerará um relatório pedindo que o seguro seja mantido para os estagiários que em determinado mês e ano manteram-se permanentes em seus estágios.

A3: Incluir no Seguro:

Gerará um relatório pedindo a inclusão dos estagiários que em determinado mês e ano iniciaram um estágio.

A4: Relatório Geral:

Gerará um relatório com todos os campos anteriores para determinado mês e ano selecionados.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Antônia, funcionária da empresa Faz Tudo Ltda, informa seu usuário e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Relatório de Seguro e escolhe o item Relatório Geral para gerar um relatório a respeito dos itens: Cancelar Seguro, Manter Seguro e Incluir no Seguro. Então clica no botão enviar e o relatório é gerado, ela o imprimi através de seu navegador.

Cenários Secundários

- Antônia deseja gerar um relatório sobre o item Cancelar Seguro para o mês de abril do ano de 2005, mas não houve cancelamentos para este mês neste ano, então o relatório é gerado em branco, e Antônia sai da aplicação.

6.2.0.1.3 Visão Aluno

6.2.0.1.3.1 Módulo Estágio - Buscar Vagas

Caso de uso: Buscar Vagas

Ator: Aluno.

Descrição Resumida:

Aluno poderá visualizar as vagas disponíveis no sistema especificamente para seu perfil.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser aluno da UFPR, devidamente matriculado.

Fluxo Principal: Este caso de uso começa quando o aluno da UFPR acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Estágios identificando-se com seu login e senha. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado, escolhe a opção Buscar Vagas (A1).

Fluxos Alternativos:

A.1: Buscar Vagas

Ao clicar nesta opção, será carregada uma tela para o aluno com todas as vagas disponíveis que estejam adequadas ao seu perfil. A tela possui as informações: Inclusão da Vaga (data), Setor, Carga Horária, Bolsa, Requisitos, Horário, Empresa, Endereço e Contato. Caso não haja vagas disponíveis para esse aluno no momento o sistema informará: "NÃO HÁ VAGAS NO MOMENTO.".

A.1.1: Contato

Caso o aluno tenha interesse em alguma vaga, ele deve entrar em contato com a empresa através do telefone disponibilizado no sistema para agendar uma entrevista. Após isso, caso o mesmo seja aprovado, transcorrerão os processos estabelecidos pela UFPR para a efetivação do estágio.

A.2: Página Inicial

Retorna para a página inicial do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Jaqueline da Silva, aluna da UFPR informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Buscar Vagas e visualiza as vagas disponíveis para o seu perfil. Para Jaqueline, aluna do curso de Tecnologia em Informática aparecem 5 vagas. Ela opta por uma das vagas e anota o telefone da empresa, para agendar uma entrevista.

Dias depois Jaqueline foi aprovada na entrevista, então dirige-se à Coordenação de Estágios para efetivar seu contrato de estágio e termo de compromisso. O sistema é atualizado na visão Empresa, por parte do usuário da empresa, para que esta vaga não apareça mais como disponível.

Cenários Secundários

- Jaqueline acessa o sistema afim de consultar vagas para estágio, mas não há nenhuma vaga disponível para o perfil dela no momento, ela então sai da aplicação.

6.2.0.1.3.2 Módulo Estágio - Pessoal

Caso de uso: Pessoal

Ator: Aluno.

Descrição Resumida:

Permite ao aluno alterar seus dados de cadastro no sistema.

- Fluxo de Eventos:

Pré-Condições: Usuário deve ser aluno da UFPR, devidamente matriculado.

Fluxo Principal: Este caso de uso começa quando o aluno da UFPR acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa no Módulo de Estágios, identificando-se com seu login e senha, a opção Pessoal. O sistema de Estágios informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema. O aluno então altera sua informações conforme necessário.

Fluxos Alternativos:

A.1: Habilidades

Este fluxo é executado quando em Pessoal o aluno clica em Habilidades, então o sistema carrega uma tela com o login do aluno e, caso o aluno já tenha algumas habilidades cadastradas, as mesmas aparecerão listadas. Neste fluxo existem os subfluxos:

A.1.1: Inserir Habilidade

O aluno pode clicar em "Procurar Habilidades", assim serão listados todos os requisitos necessários para que se preencha uma vaga em seu respectivo curso. Ele poderá então seleciona-lo(s) e passar a concorrer à vaga em questão. Ou, caso ele já possua tais requisitos o sistema informará: "Não foram encontradas habilidades. Talvez todas já estejam incluídas em seu perfil."

Poderá também inserir uma "Nova Habilidade" (A.1.1.1).

A.1.1.1: Nova Habilidade

Sistema carrega uma tela para que seja incluído o nome da habilidade e possui as opções "Salvar" (A.1.1.2) e "Reset" (A.1.1.3), após salvar o sistema retorna para a tela de lista de habilidades, nesta tela ainda existe o botão "Voltar para Habilidades" A.1.1.4.

A.1.1.2: Salvar

Salvas as alterações realizadas no sistema.

A.1.1.3: Reset

Limpa os campos.

A.1.1.4: Voltar para Habilidades:

Retorna à página da lista de habilidades.

A.1.2: Excluir

Exclui a habilidade cadastrada para o perfil do aluno.

A.2: Horários Disponíveis

Sistema carrega uma tela com os horários disponíveis já cadastrados pelo aluno (se houver), onde constam o código e o horário.

A.2.1: Inserir Horário

Sistema carrega uma tela com o código do aluno, opção para inclusão de inicio e fim do horário do estágio (das - às). Nesta tela temos as opções "Salvar" (A.2.1.1), "Reset" (A.2.1.2) e ainda "Listar Horários disponíveis" (A.2.1.3).

A.2.1.1: Salvar

Salva as alterações e retorna para a tela de horários já cadastrados pelo usuário.

A.2.1.2: Reset

Limpa os campos da tela em questão.

A.2.1.3: Listar Horários Disponíveis:

Retorna para a tela de horários já cadastrados pelo usuário.

A.2.2: Excluir

Exclui um horário cadastrado.

A.3 Salvar

Salva as alterações realizadas no sistema.

A.4 Reset:

Retorna os campos à posição inicial.

A.5: Voltar

Retorna para a tela inicial do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

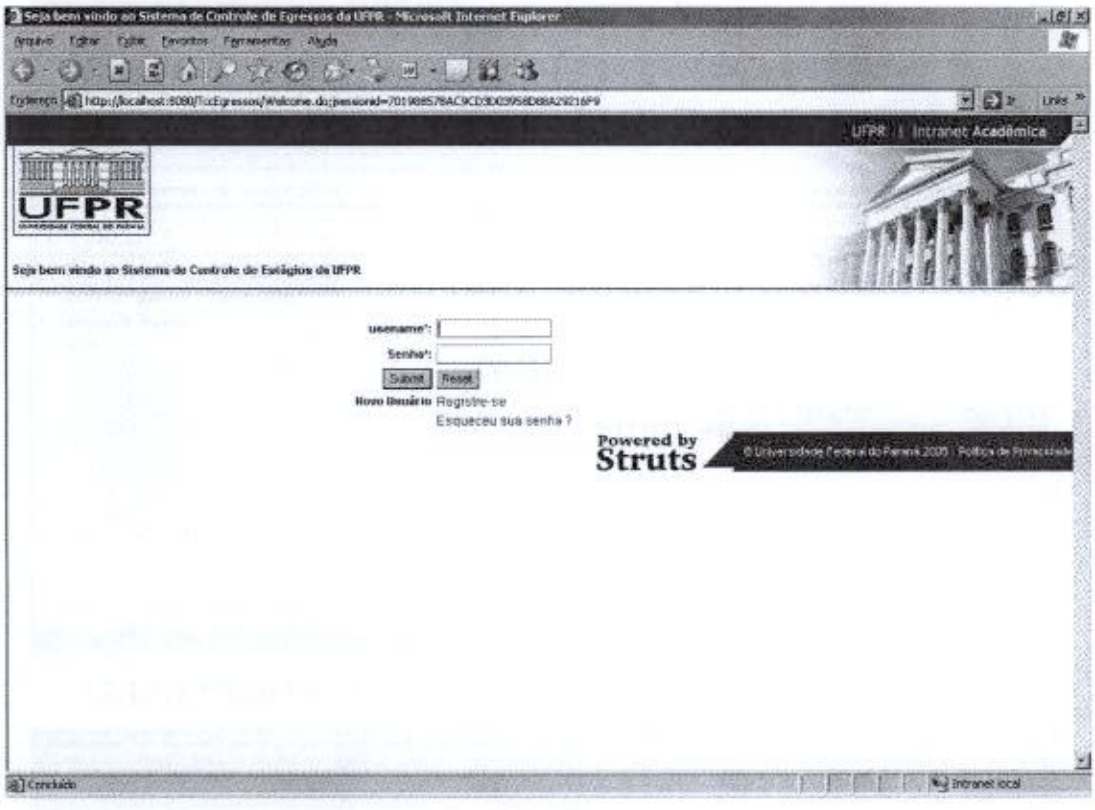
Cenário Principal

Jaqueline da Silva, aluna da UFPR, informa seu login e senha e acessa o Módulo de Estágios na Intranet Acadêmica. Em seguida acessa a opção Pessoal e atualiza seu telefone, que foi modificado recentemente. Clica em salvar e a base está atualizada.

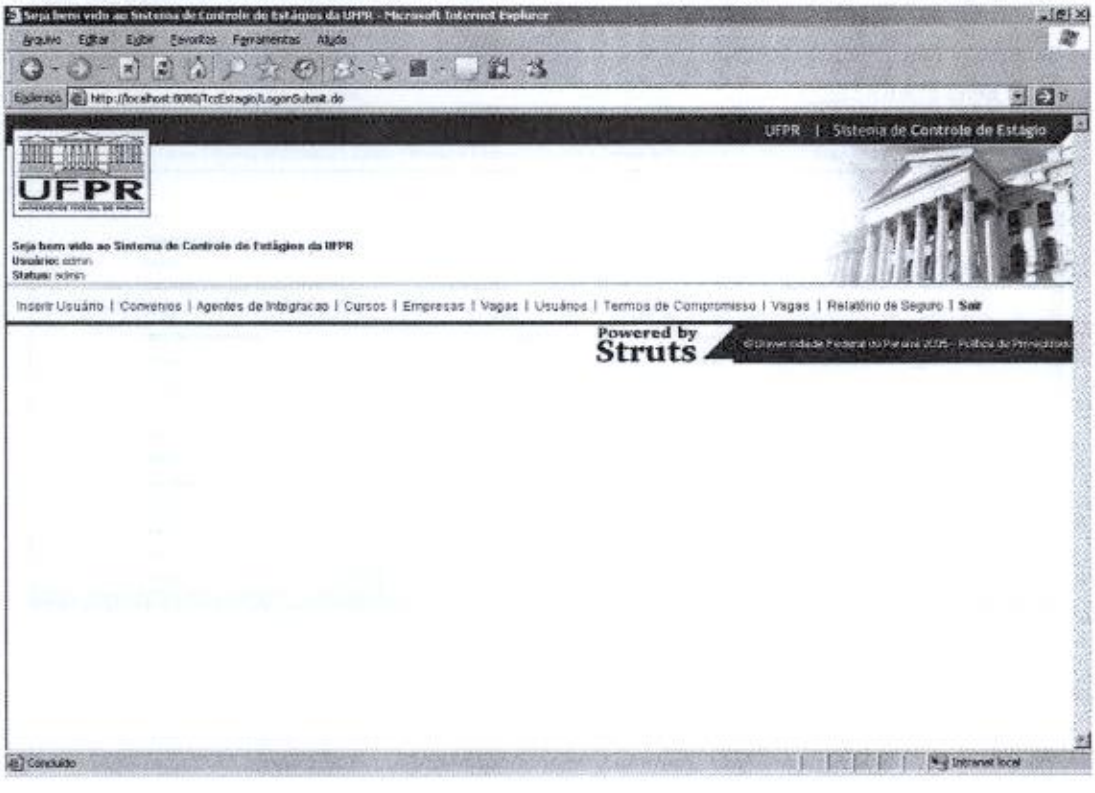
6.2.1 Diagrama de Telas

6.2.1.1 Visão Coordenação de Estágios UFPR

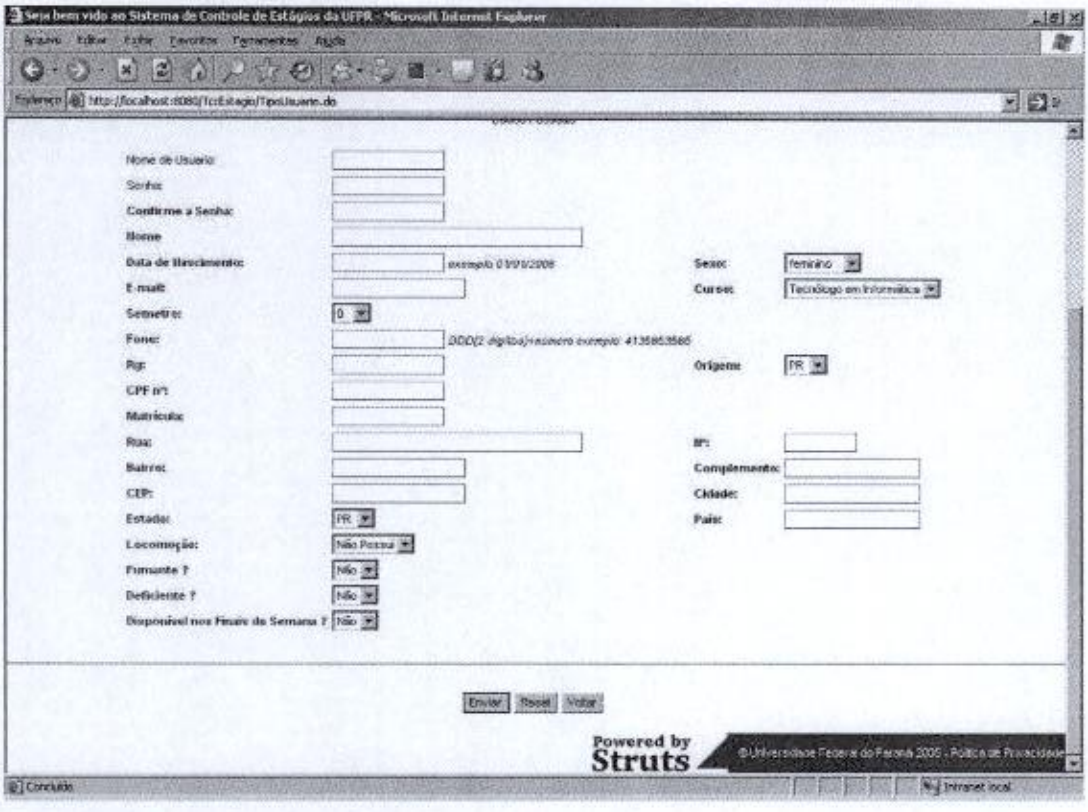
6.2.1.1.1 Login:



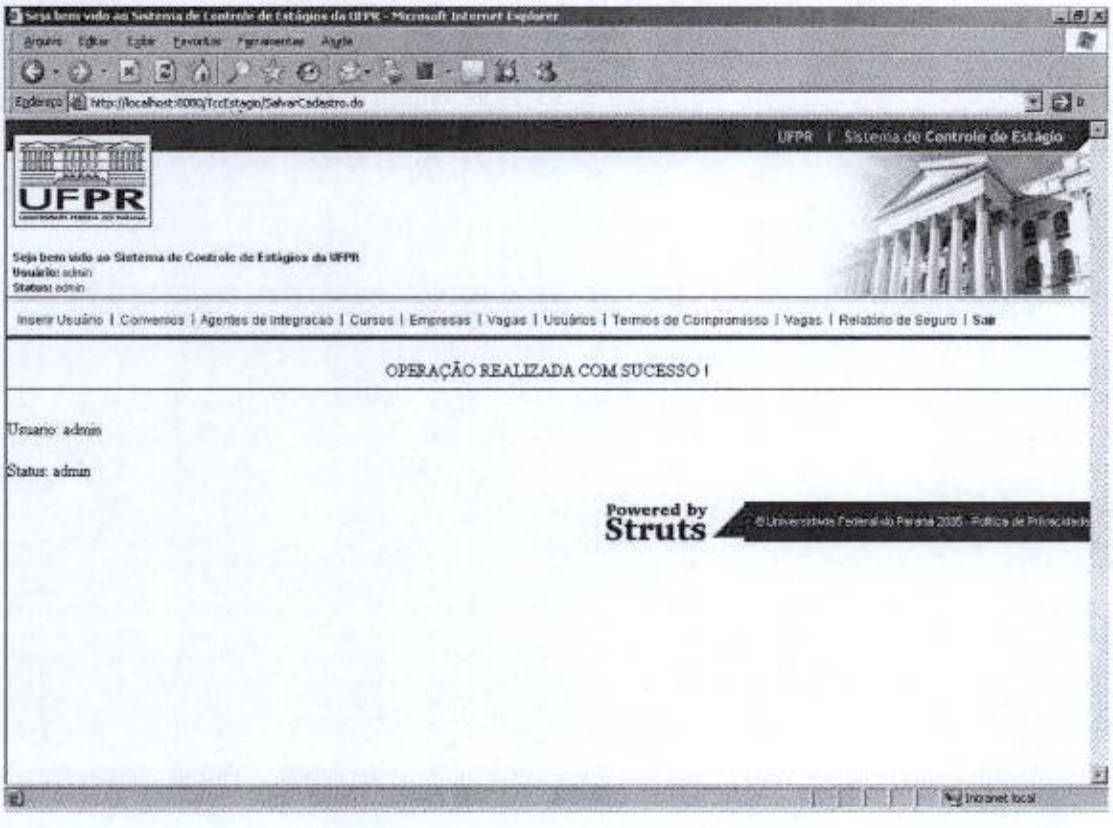
6.2.1.1.2 Tela Inicial:



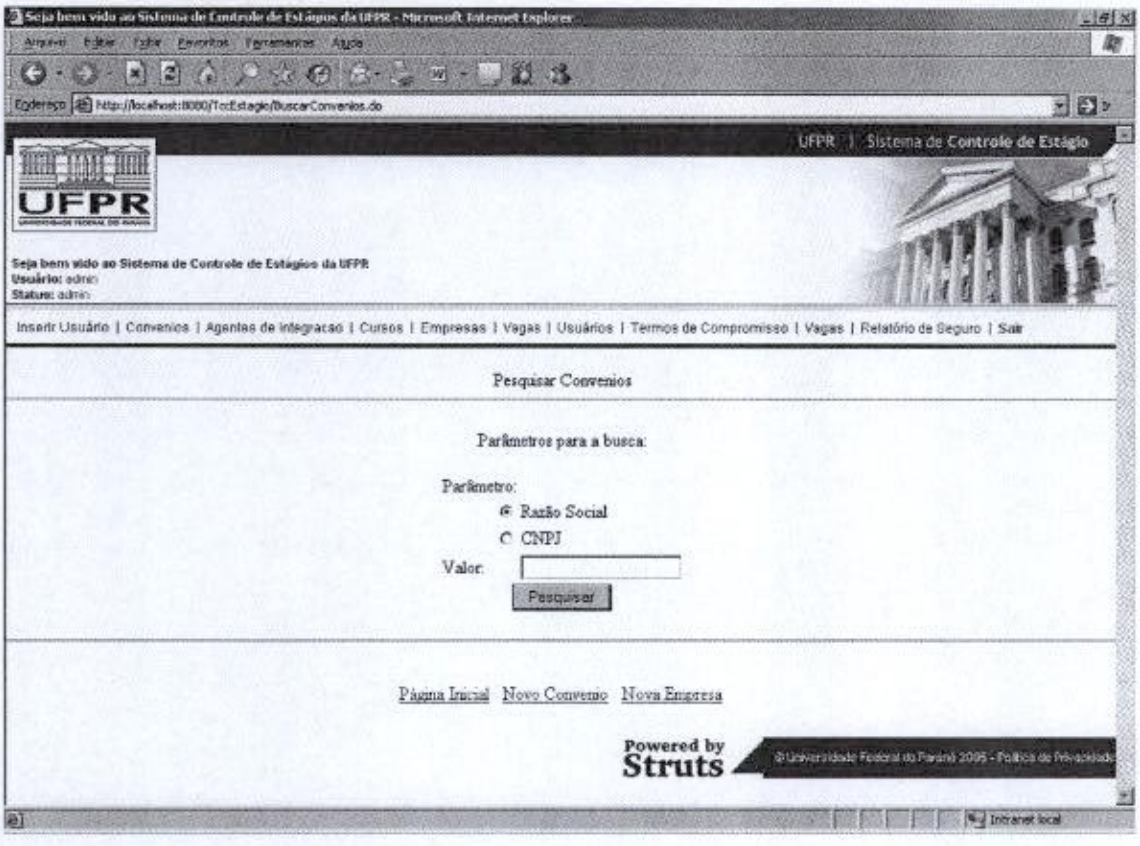
6.2.1.1.5 Fluxo Inserir Usuário – Formulário de Cadastro Parte 2



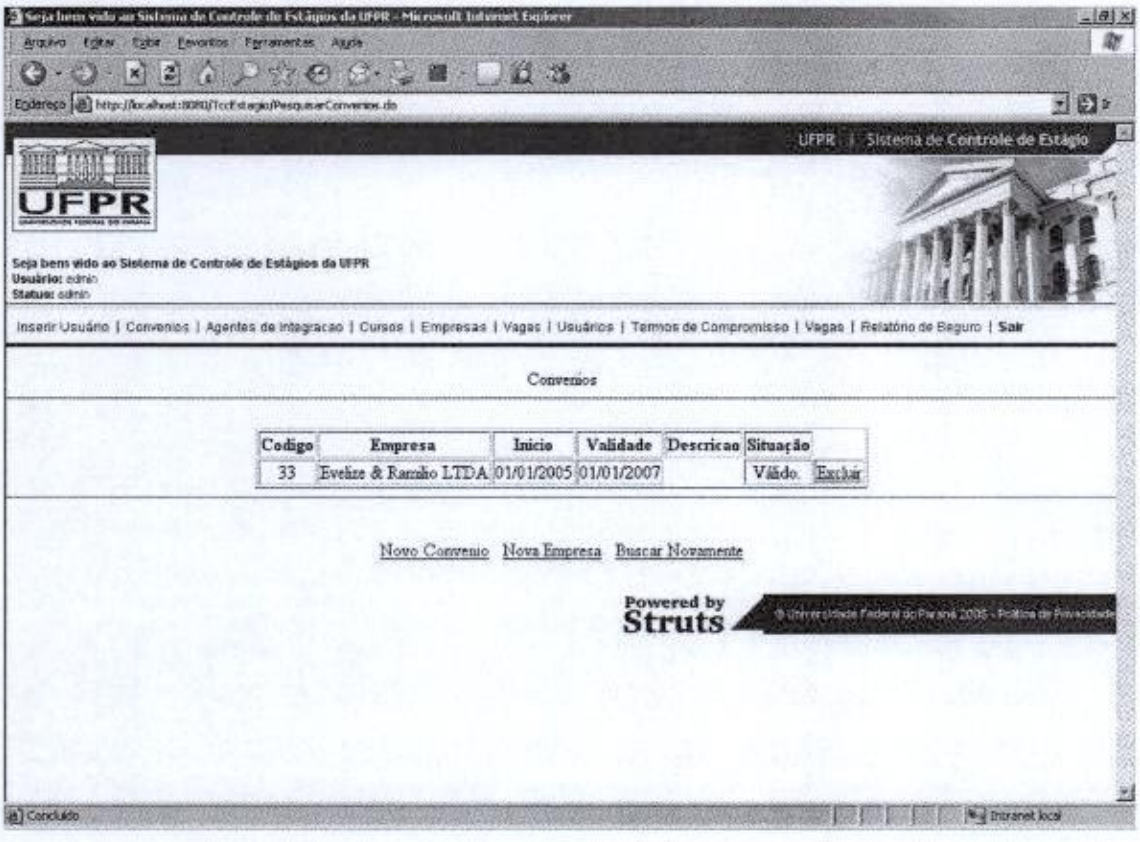
6.2.1.1.6 Fluxo Inserir Usuário – Operação realizada com sucesso



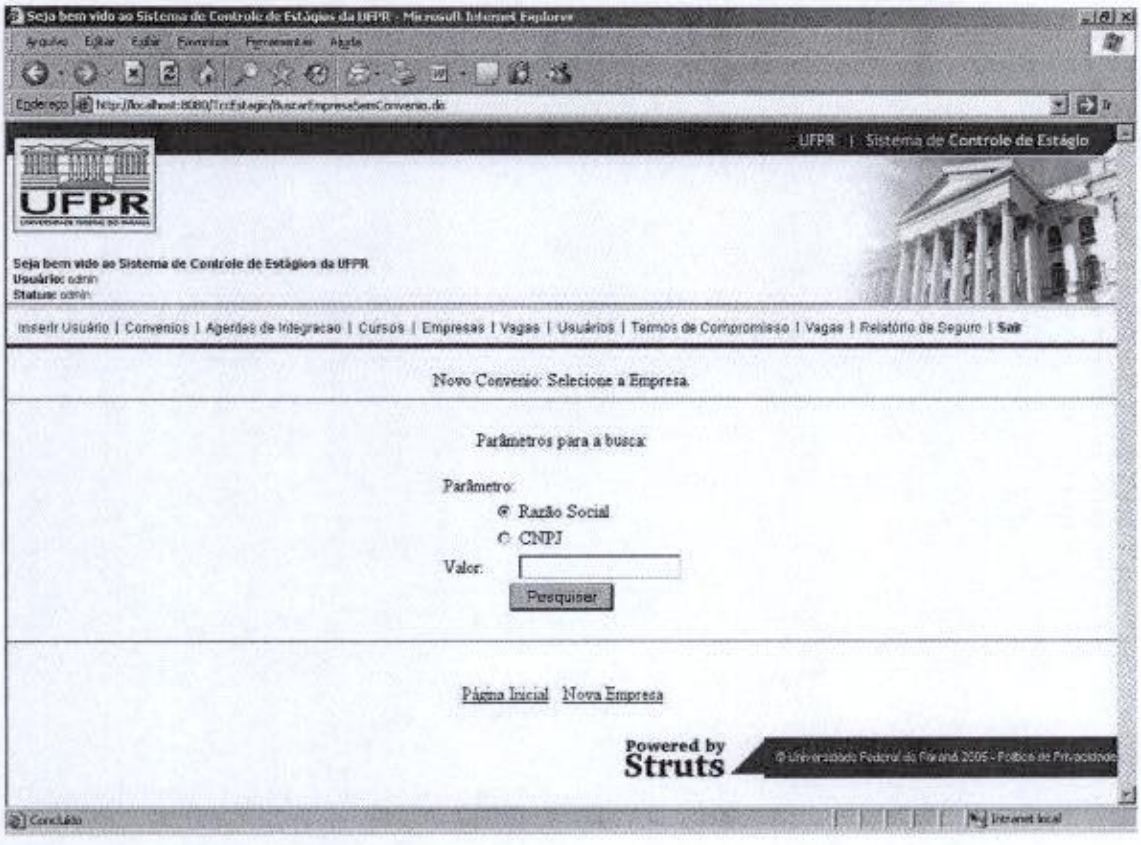
6.2.1.1.7 Fluxo Convênios - Pesquisar



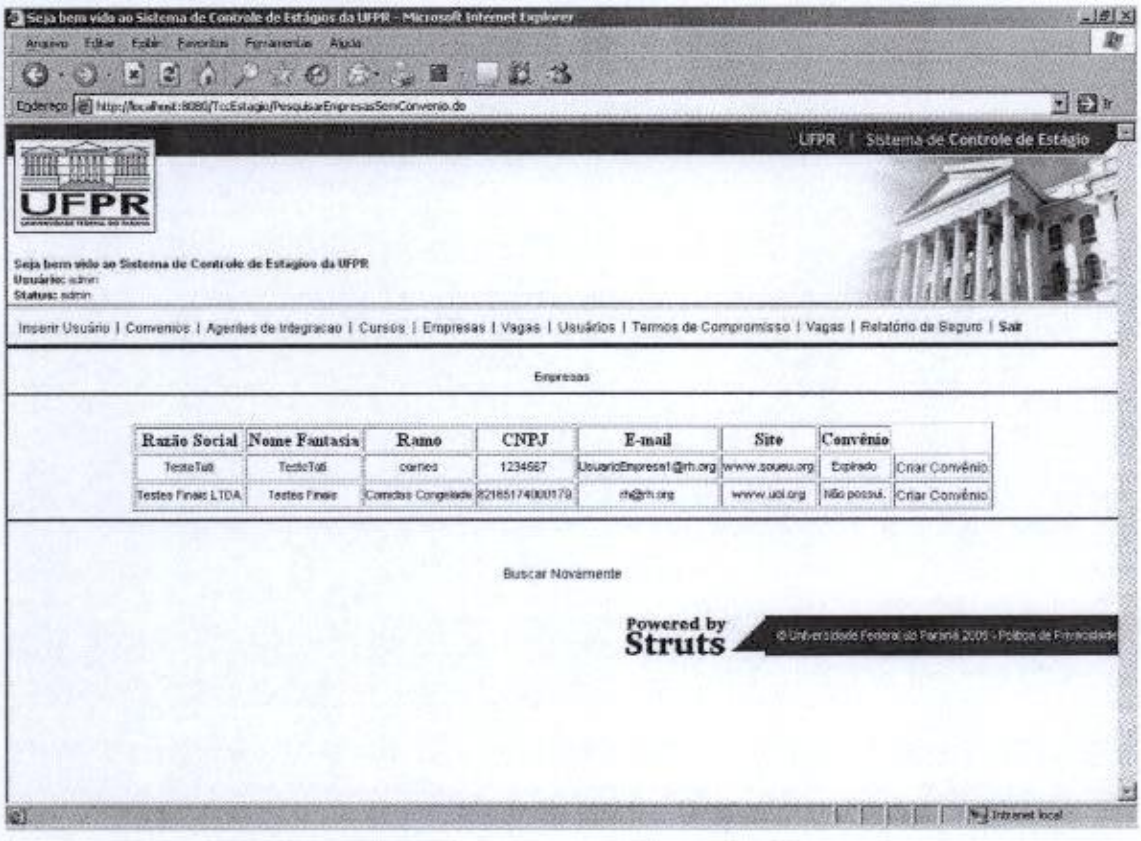
6.2.1.1.8 Fluxo Convênios – Visualizando Convênio Pesquisado



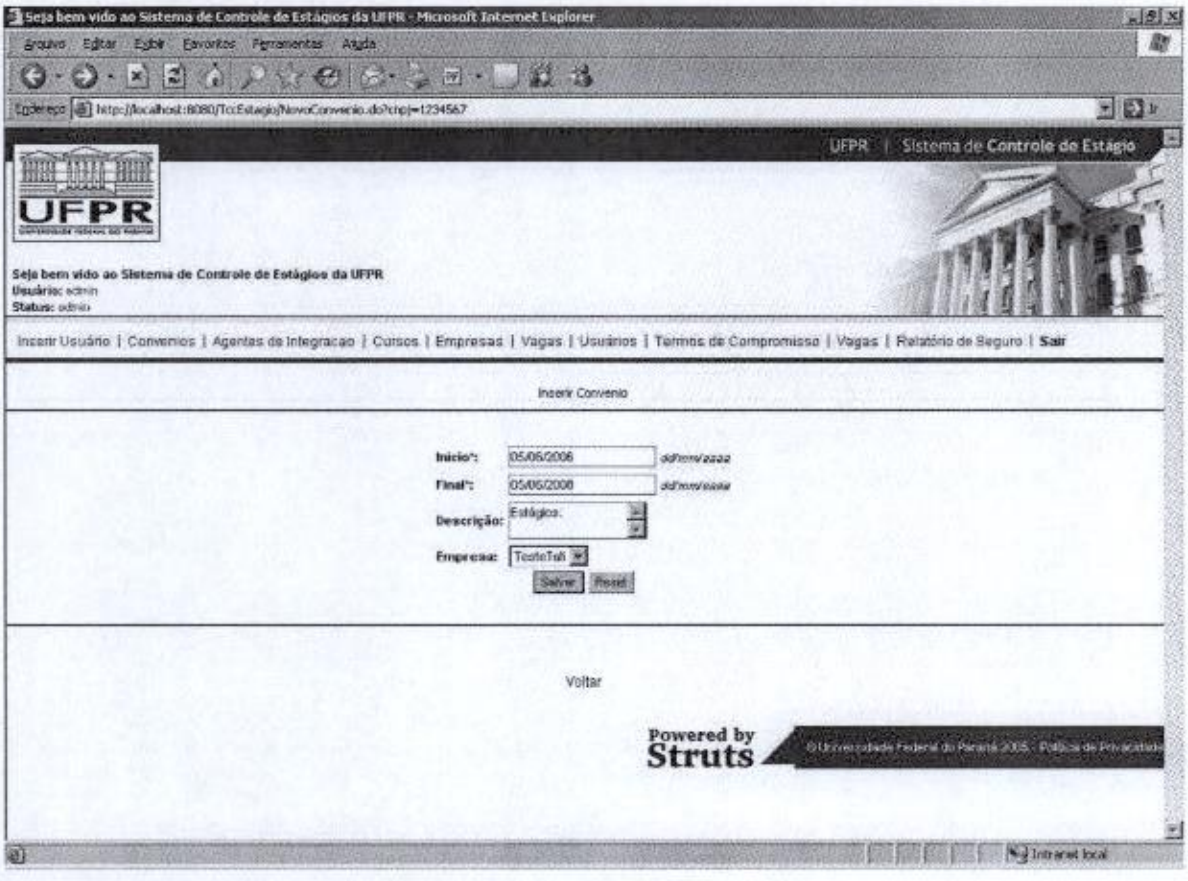
6.2.1.1.9 Fluxo Convênios – Pesquisa por empresas não conveniadas



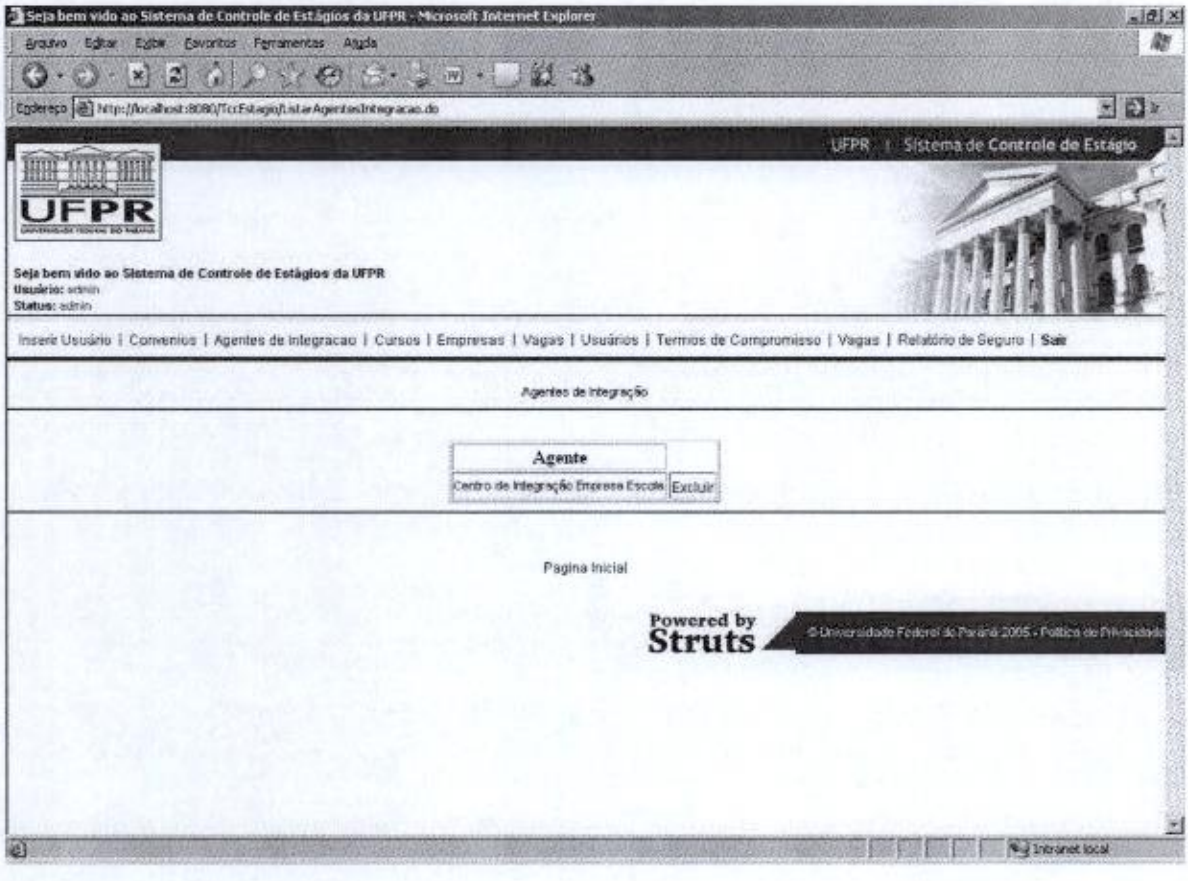
6.2.1.1.10 Fluxo Convênios – Visualização de empresas não conveniadas



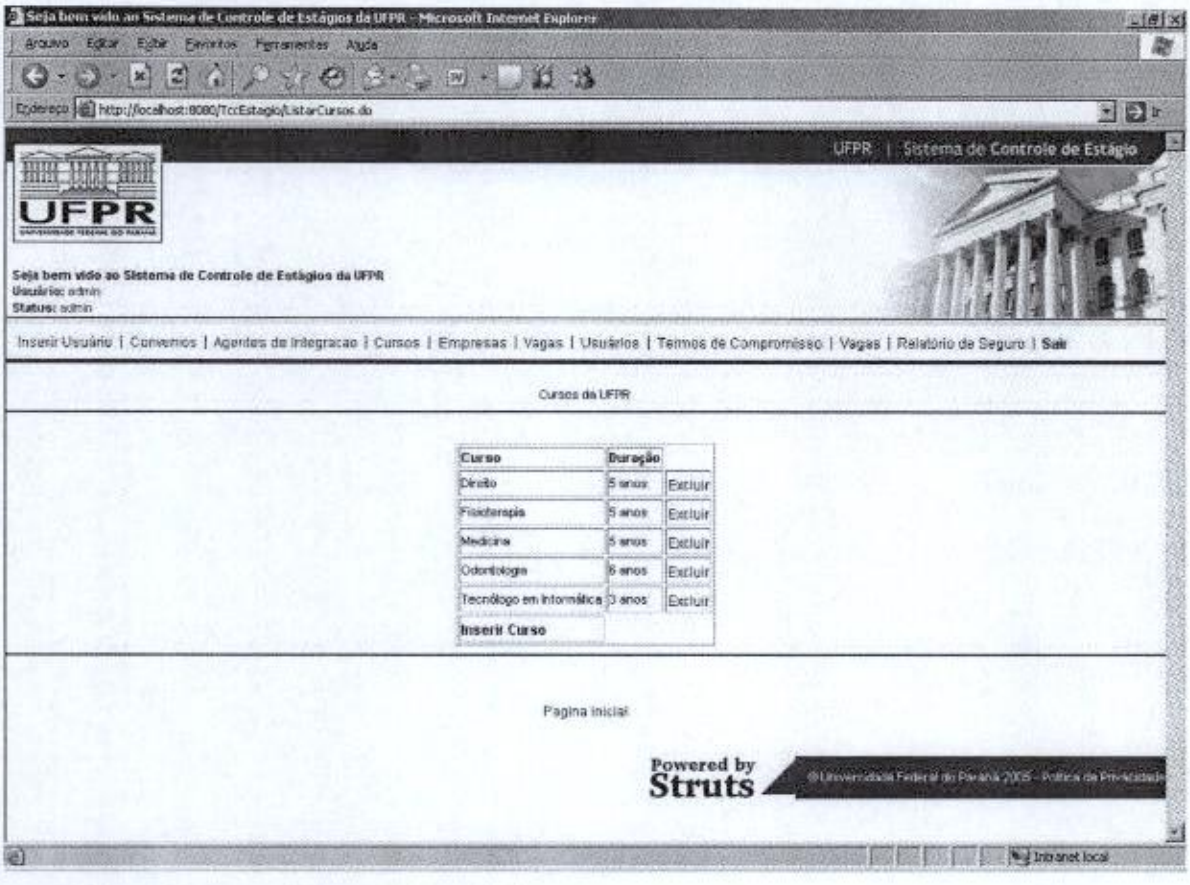
6.2.1.1.11 Fluxo Convênios - Criar Convênios



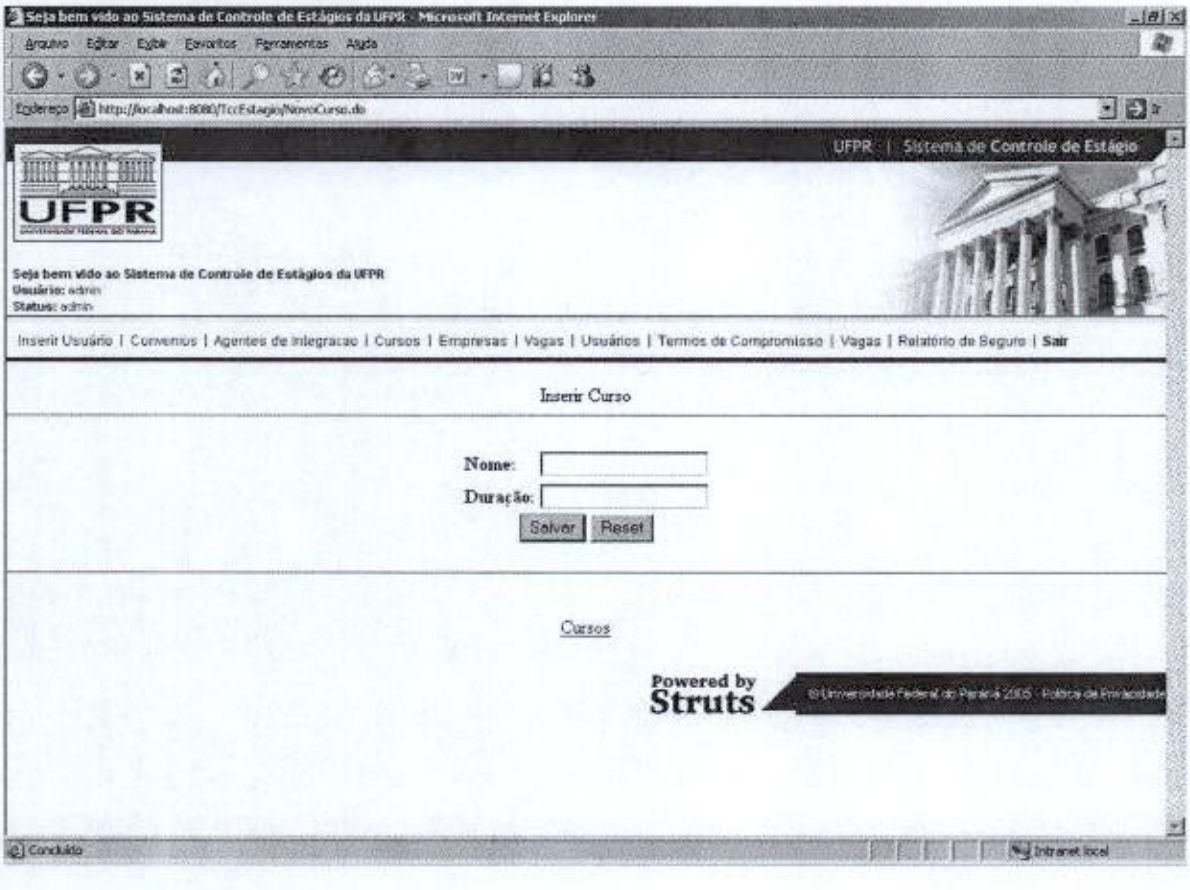
6.2.1.1.12 Fluxo Agentes de Integração – Visualizar



6.2.1.1.13 Fluxo Cursos – Visualizar Cursos Cadastrados

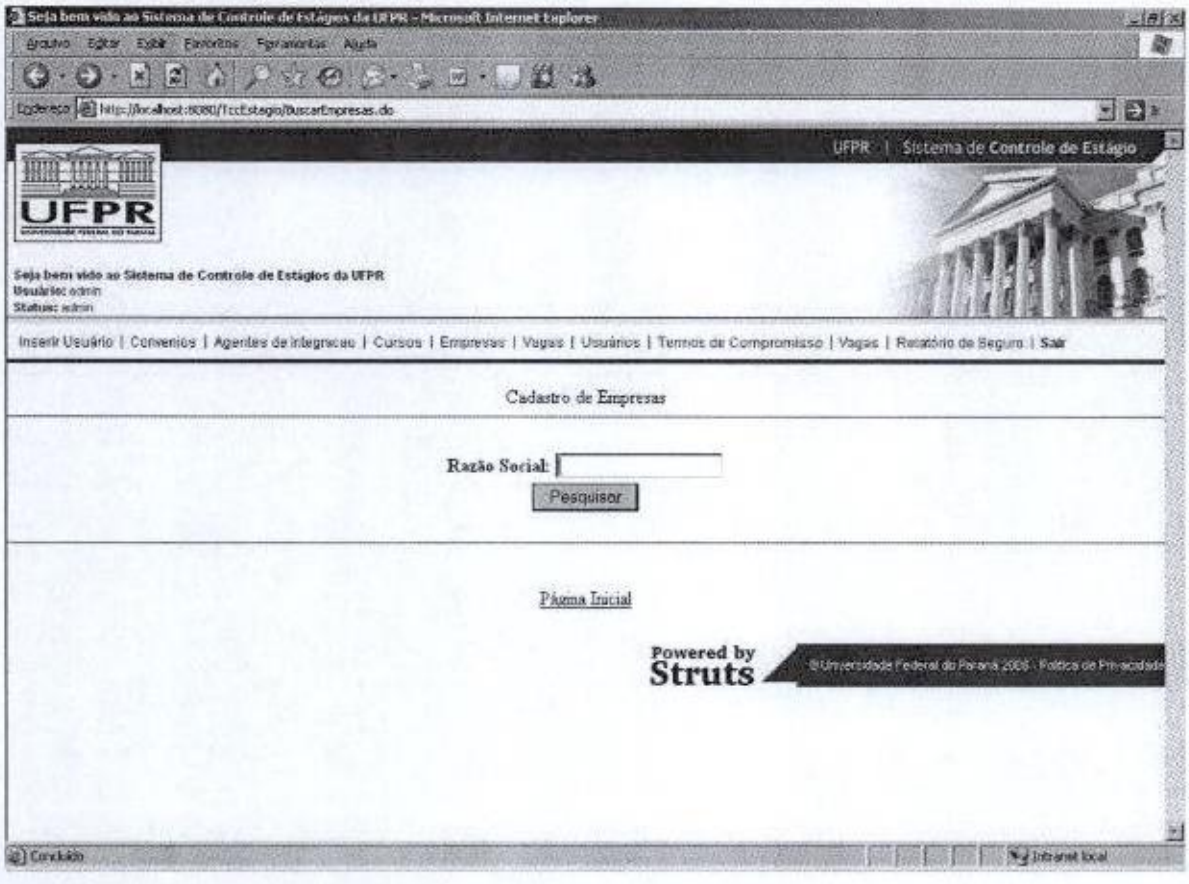


6.2.1.1.14 Fluxo Cursos – Inserir Curso

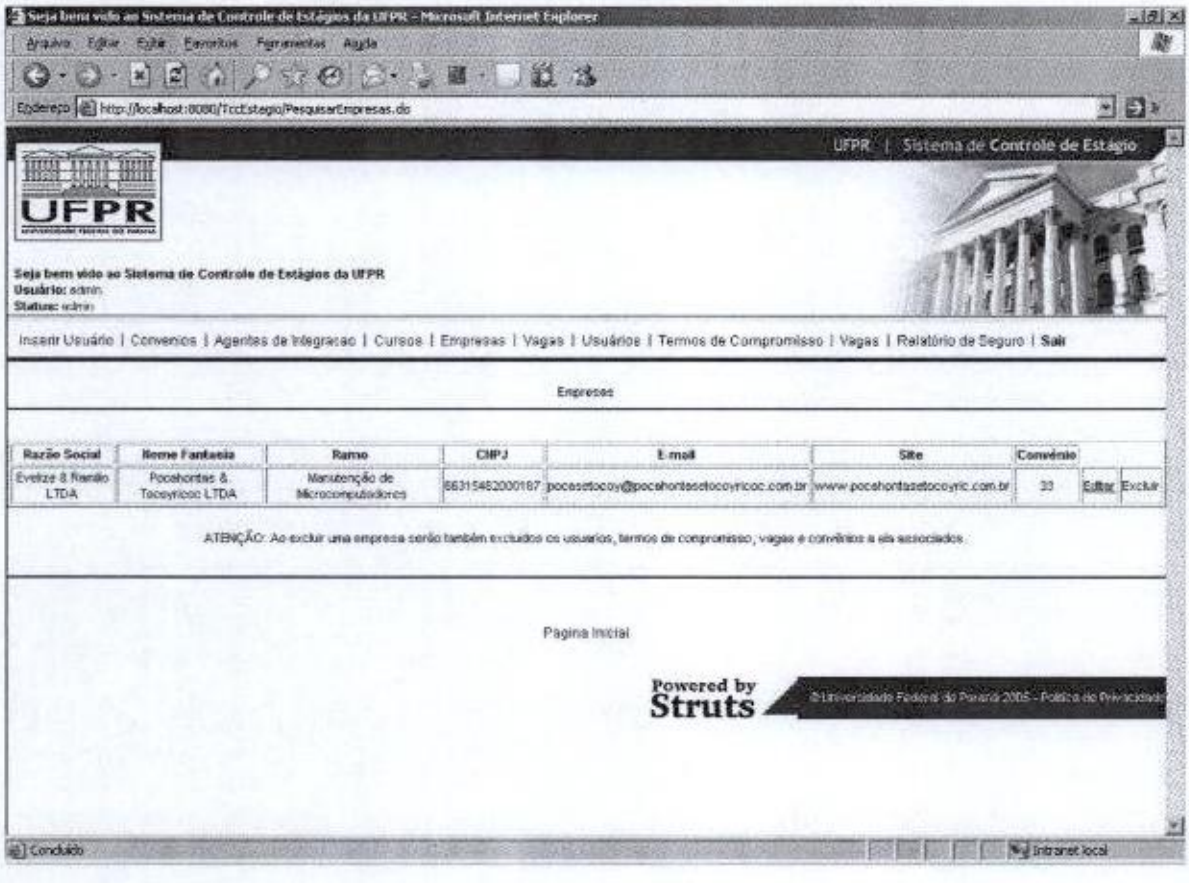


BIBLIOTECA DO SETOR DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA DA UFPR

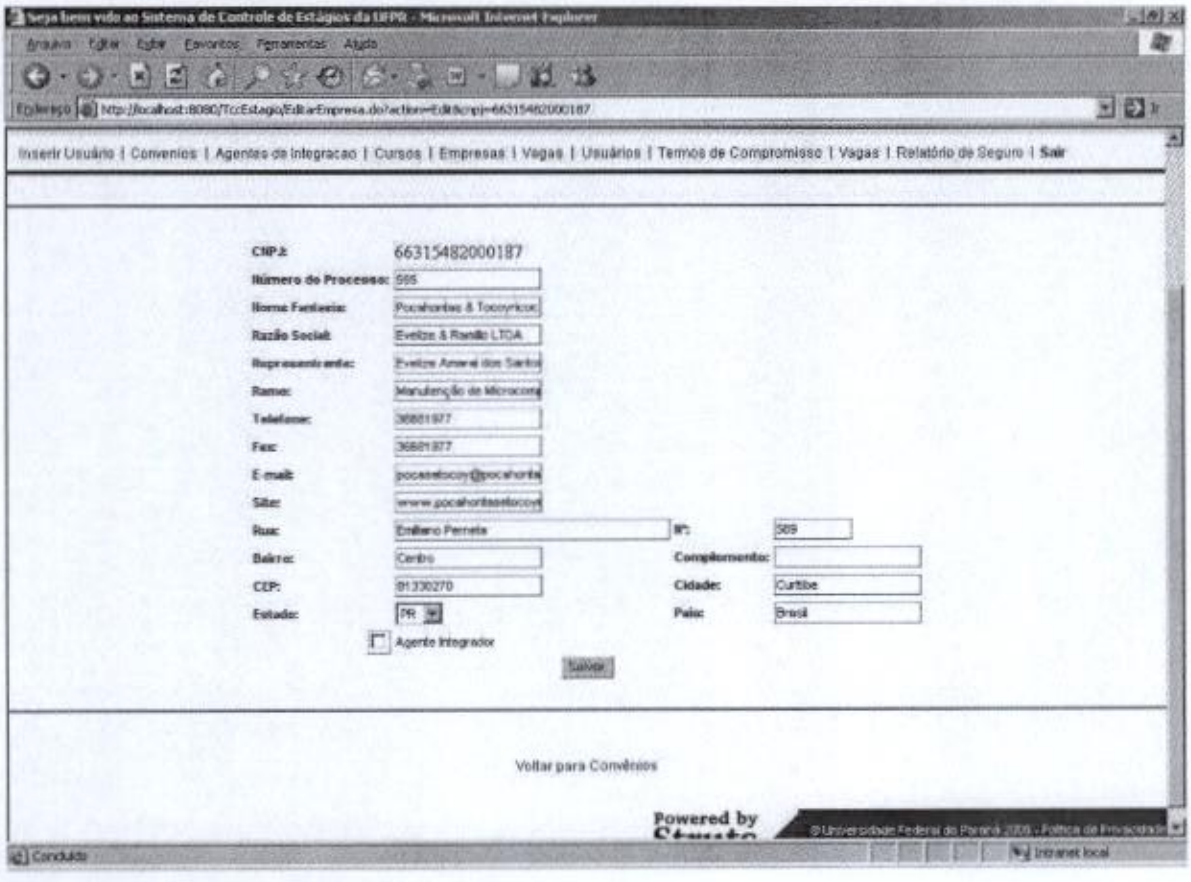
6.2.1.1.15 Fluxo Empresas – Pesquisar Razão Social



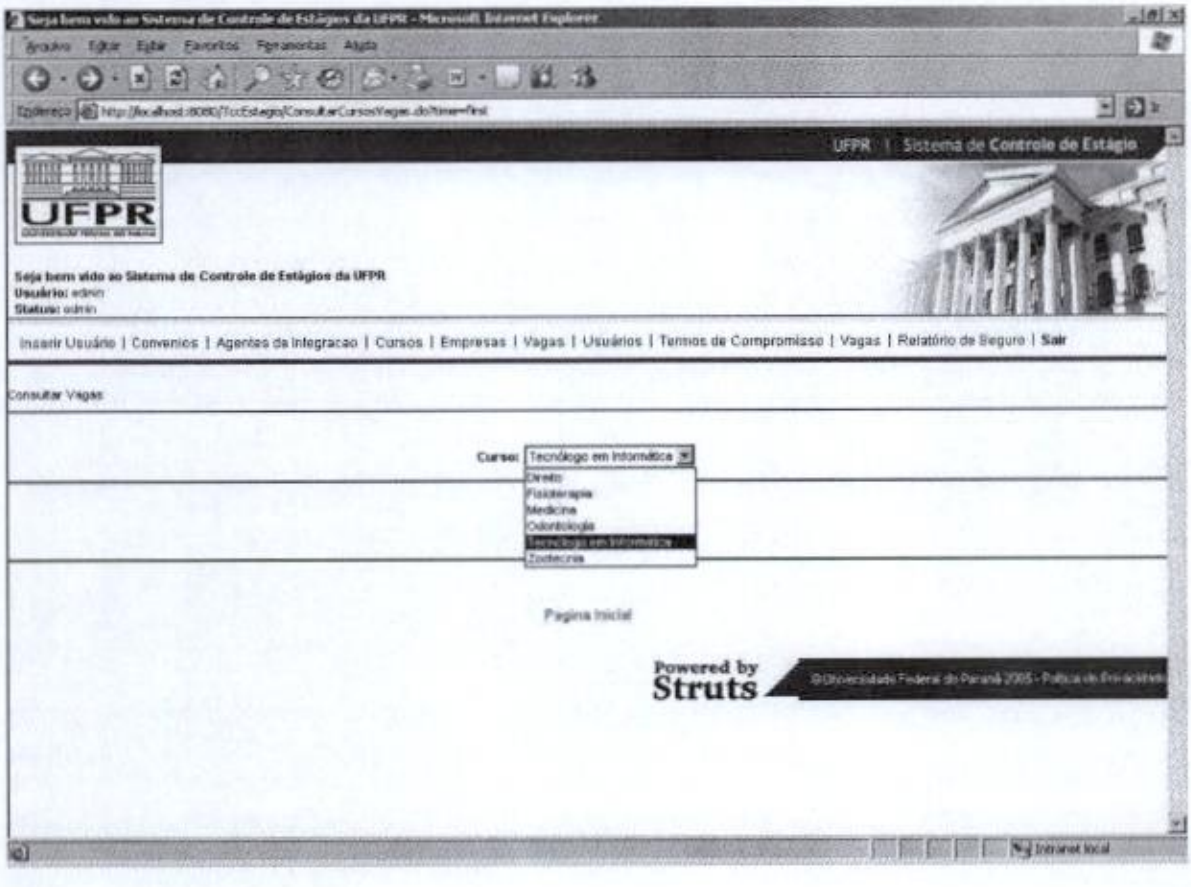
6.2.1.1.16 Fluxo Empresas – Visualizando resultados da pesquisa



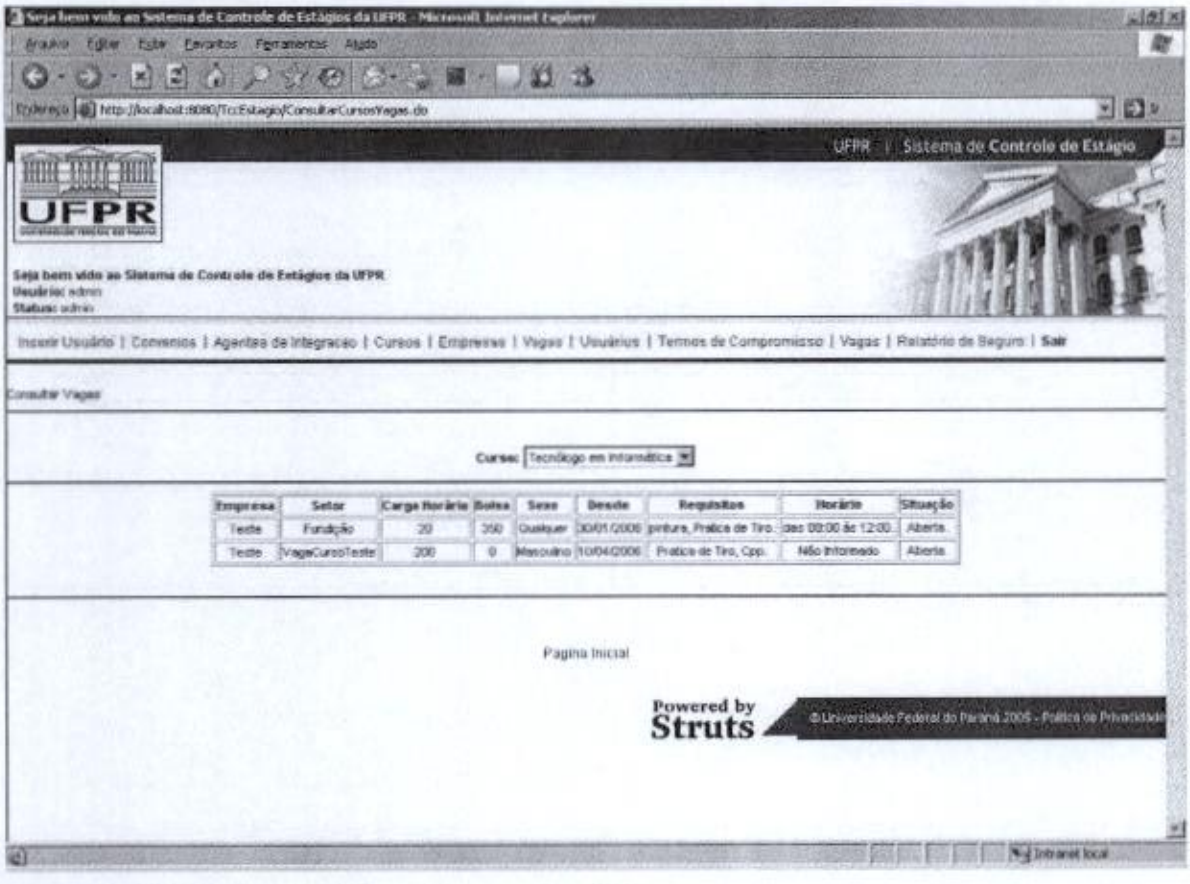
6.2.1.1.17 Fluxo Empresas – Editar Cadastro



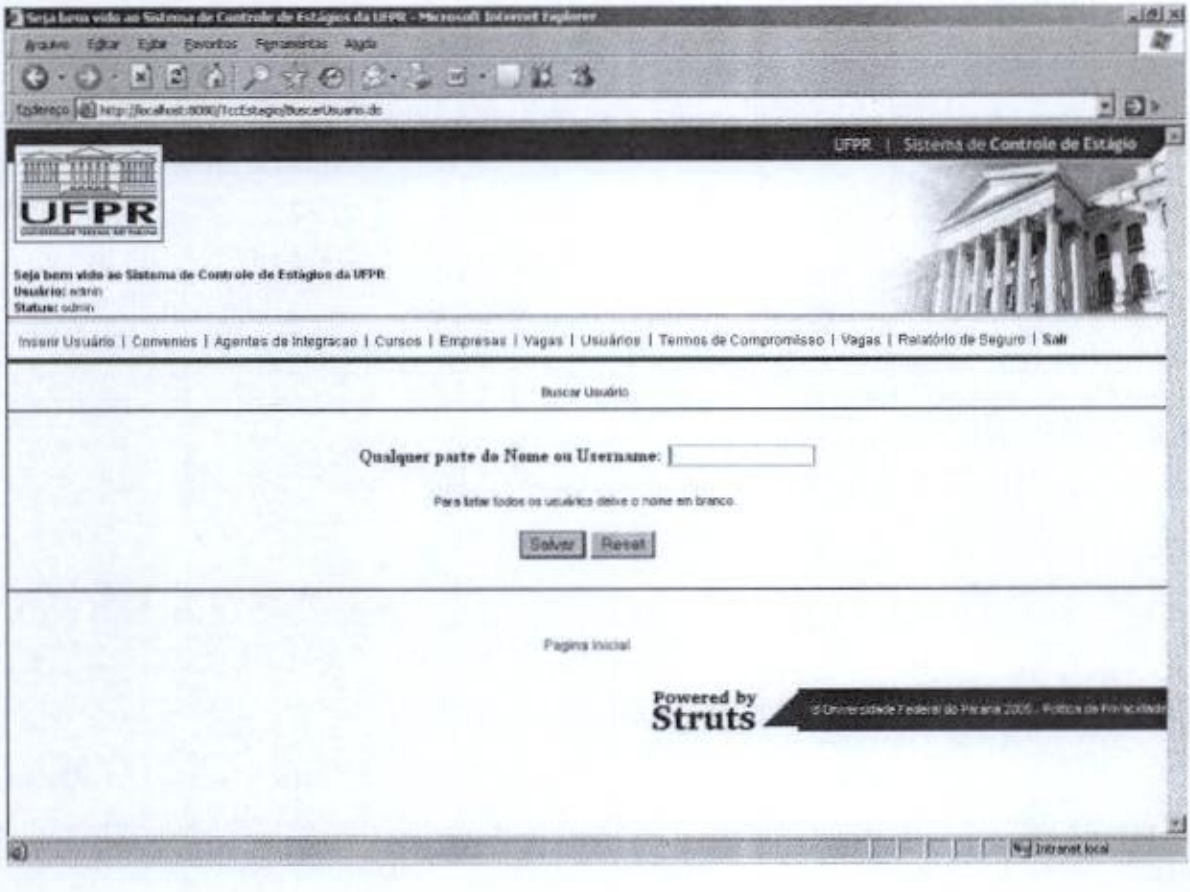
6.2.1.1.18 Fluxo Vagas – Selecionando Curso



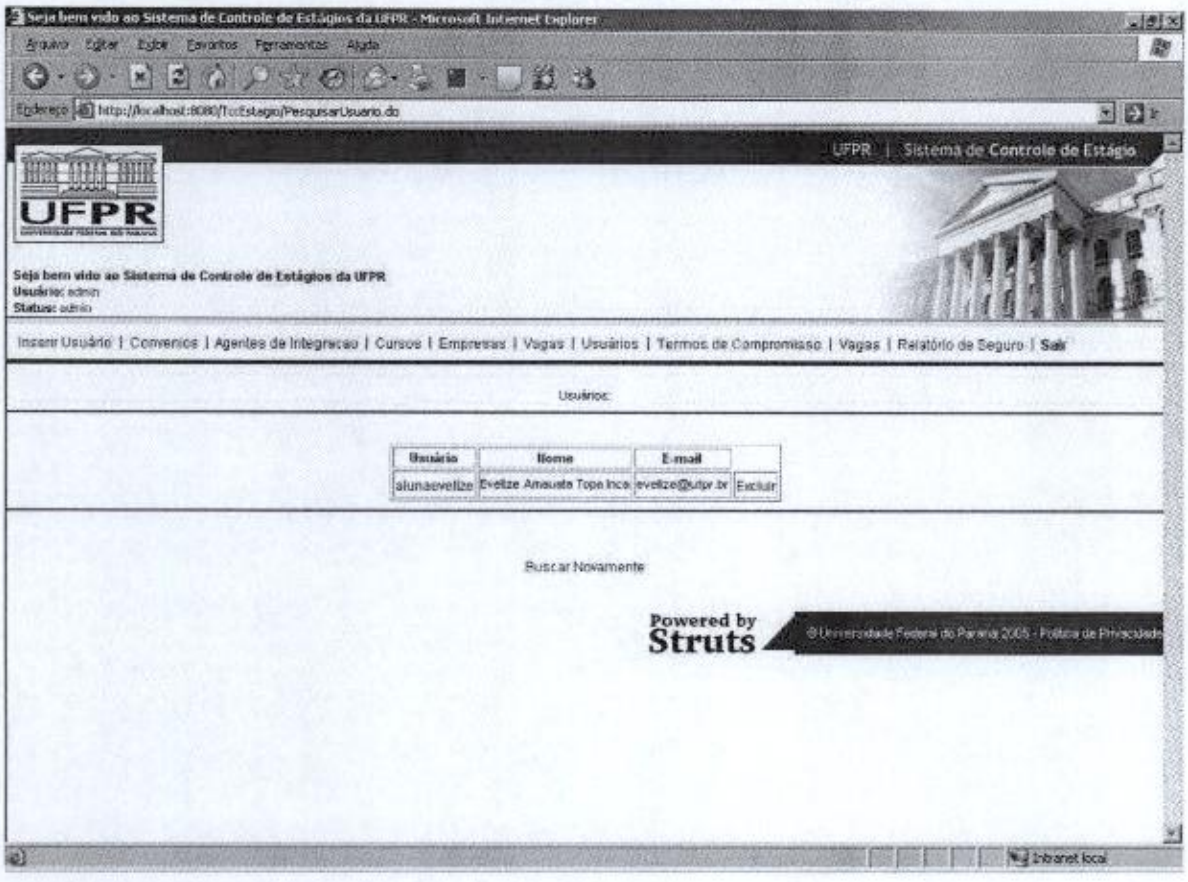
6.2.1.1.19 Fluxo Vagas – Visualizando Vagas do Curso Selecionado



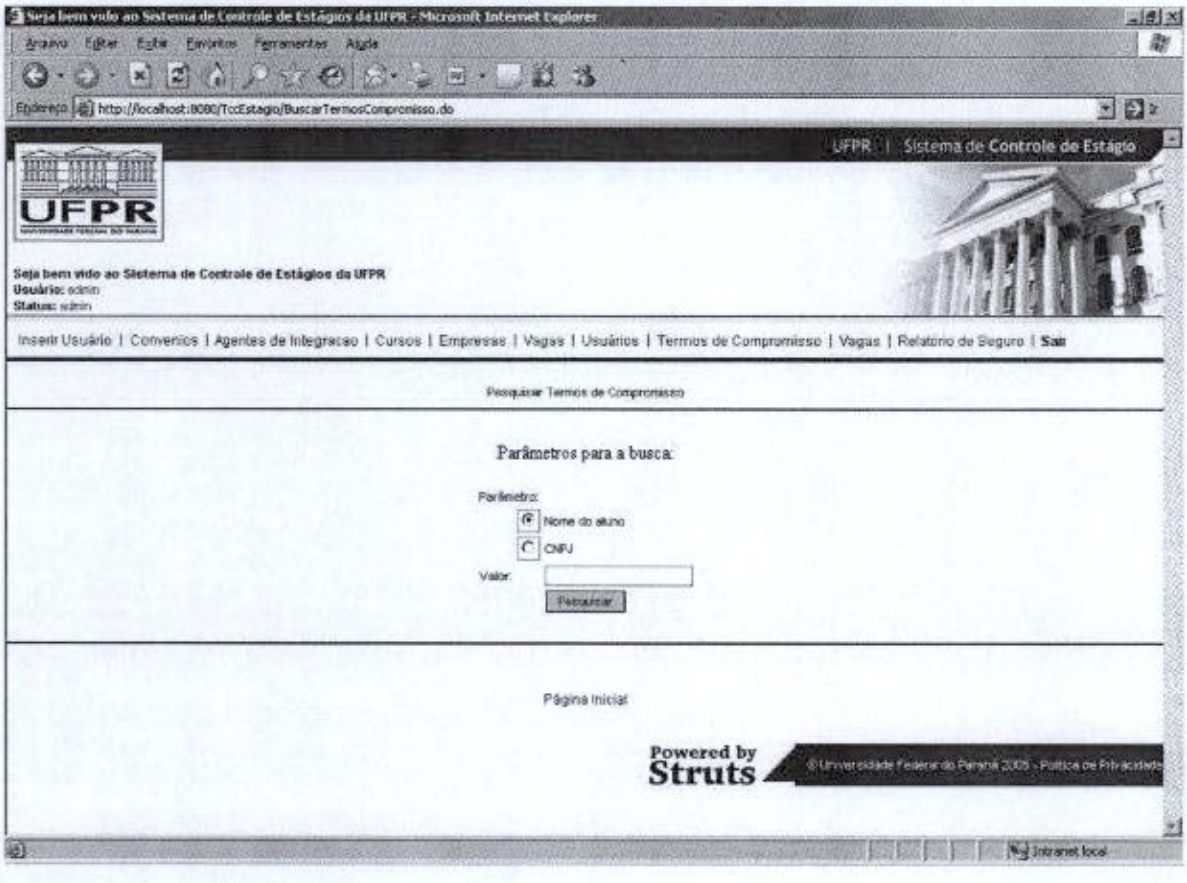
6.2.1.1.20 Fluxo Usuários – Buscar Usuário



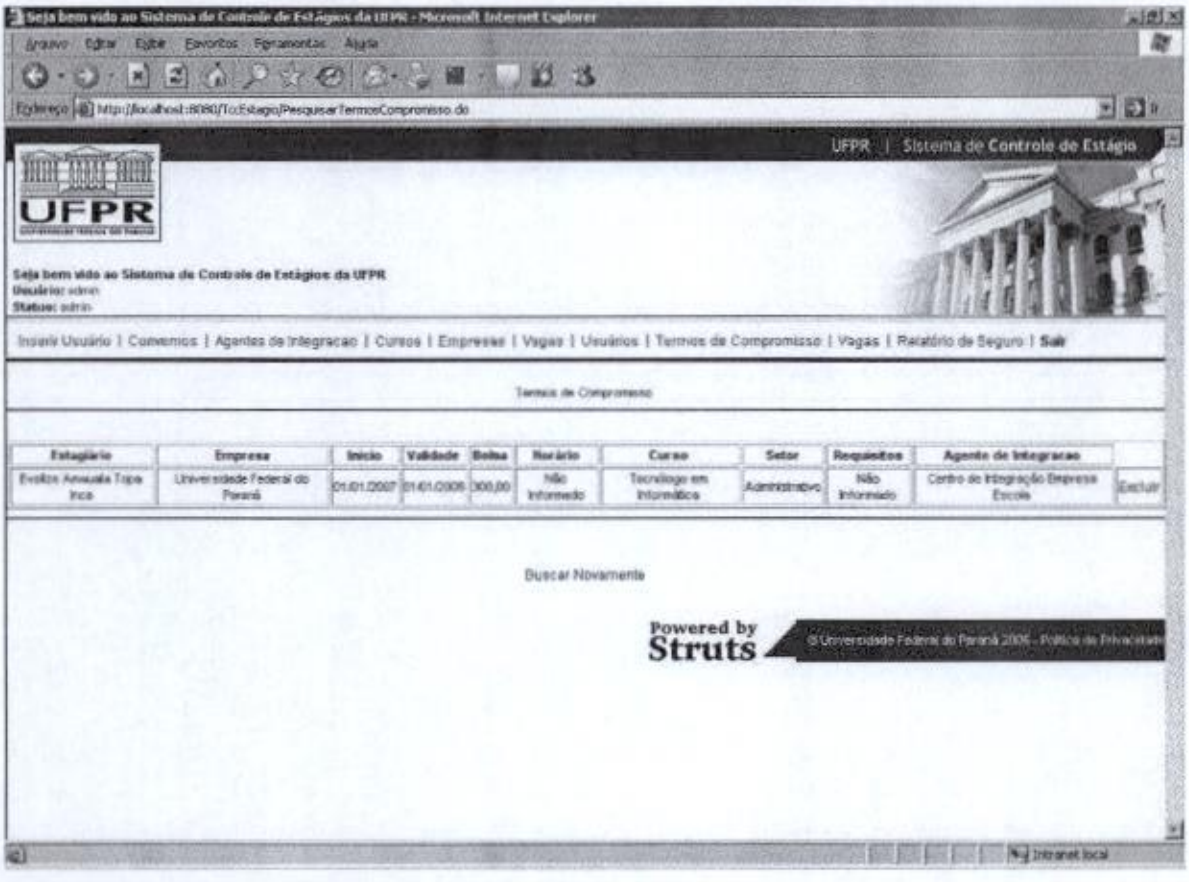
6.2.1.1.21 Fluxo Usuários – Visualizando Resultado da Pesquisa



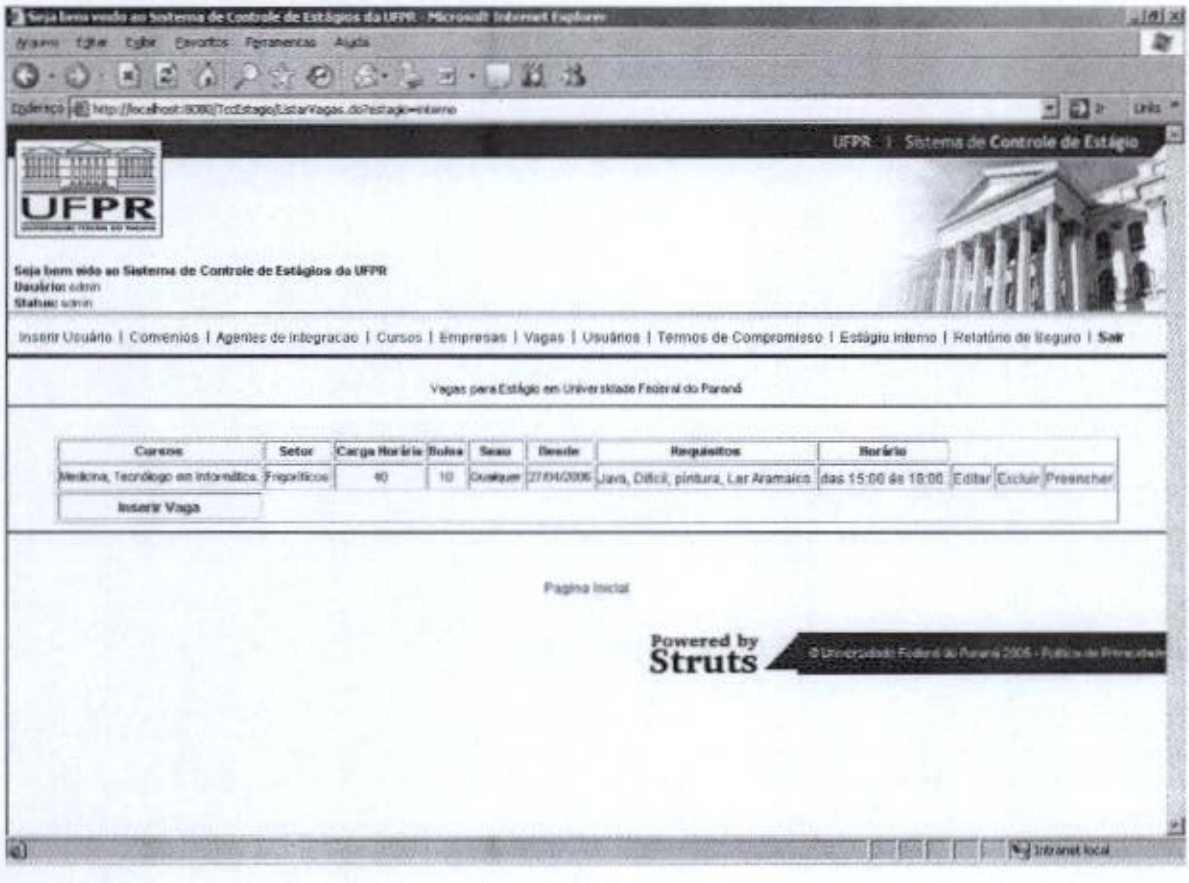
6.2.1.1.22 Fluxo Termos de Compromisso – Busca



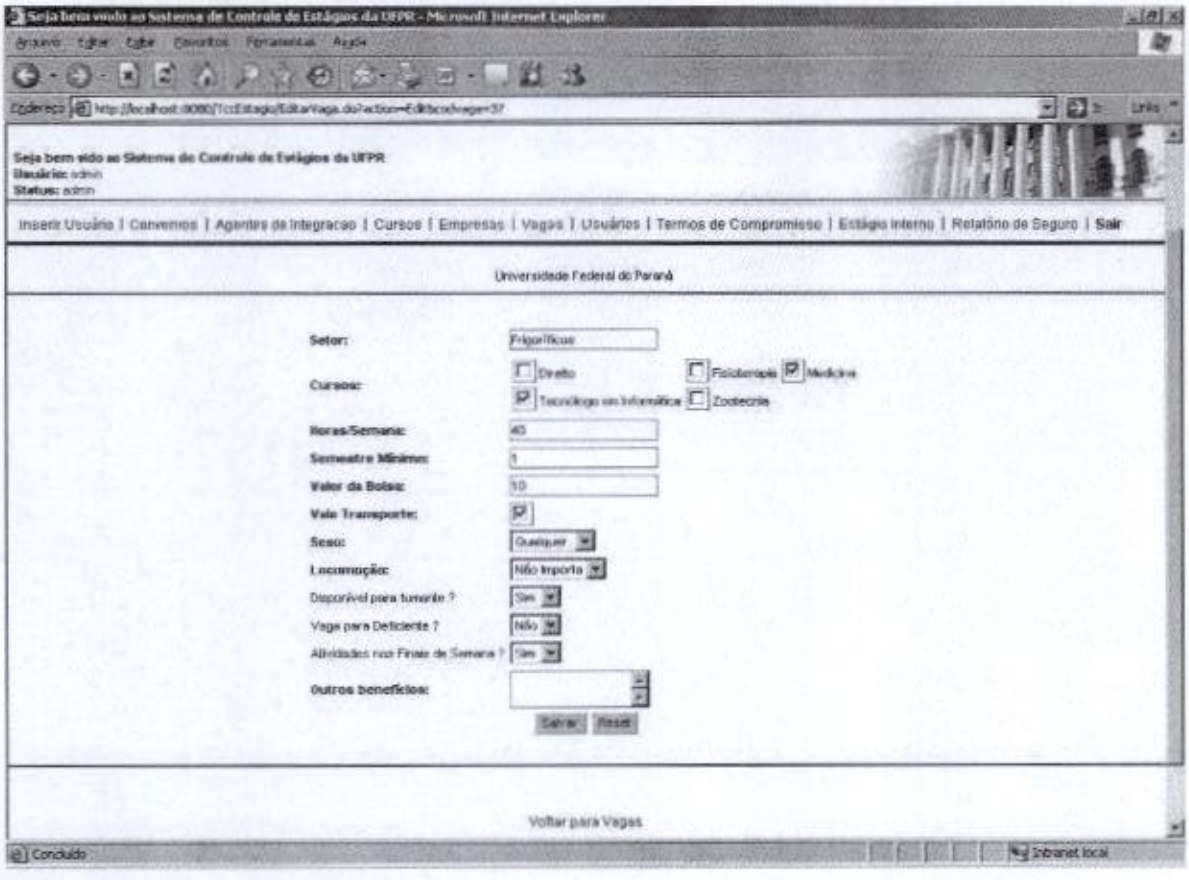
6.2.1.1.23 Fluxo Termos de Compromisso – Visualizando Resultado da Pesquisa



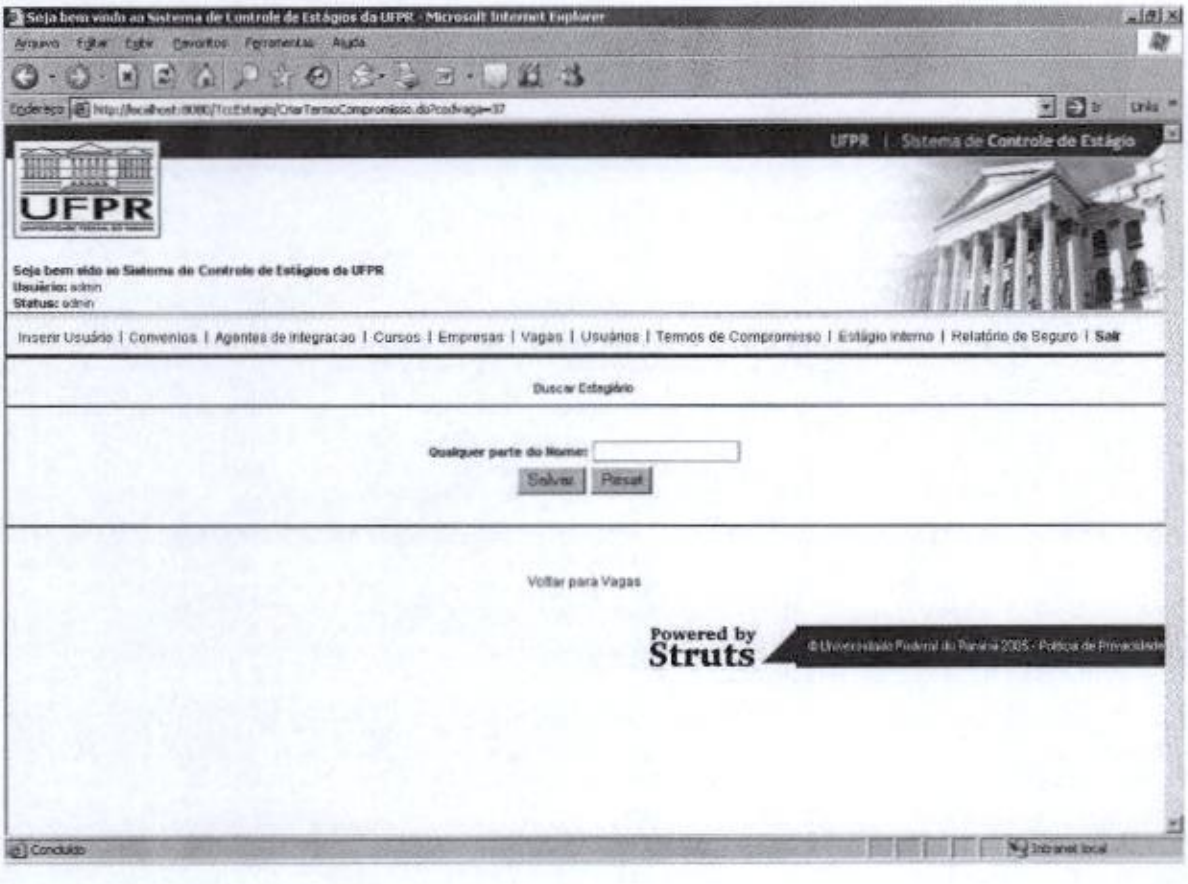
6.2.1.1.24 Fluxo Estágio Interno – Visualizando Vagas Disponíveis



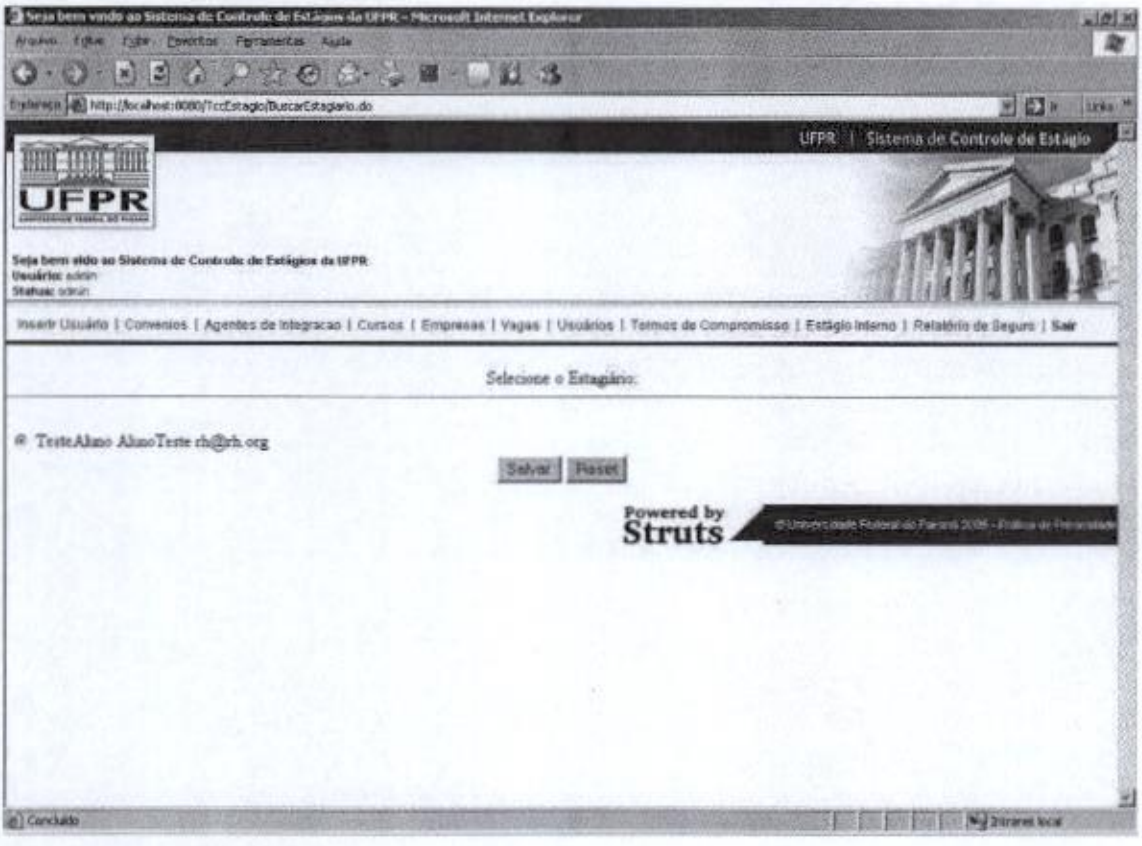
6.2.1.1.25 Fluxo Estágio Interno – Editar Vaga



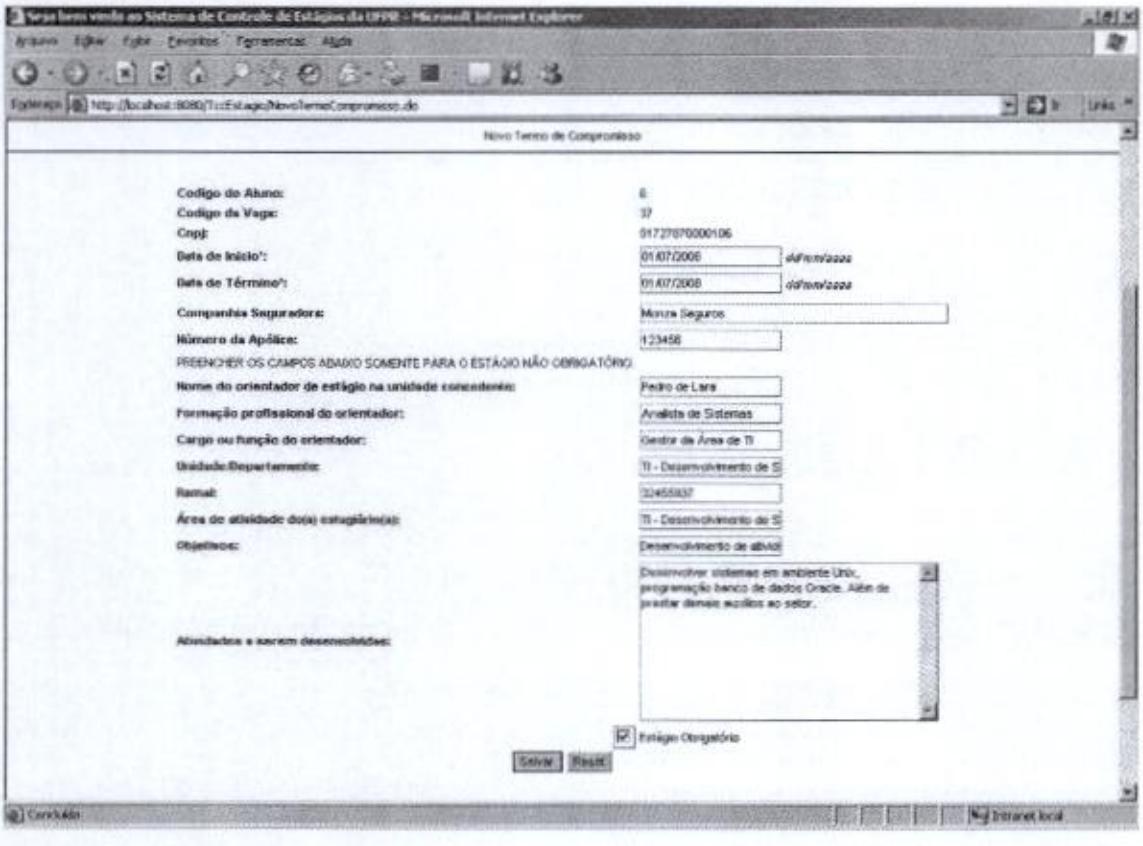
6.2.1.1.26 Fluxo Estágio Interno – Preencher Vaga (Buscando Estagiário)



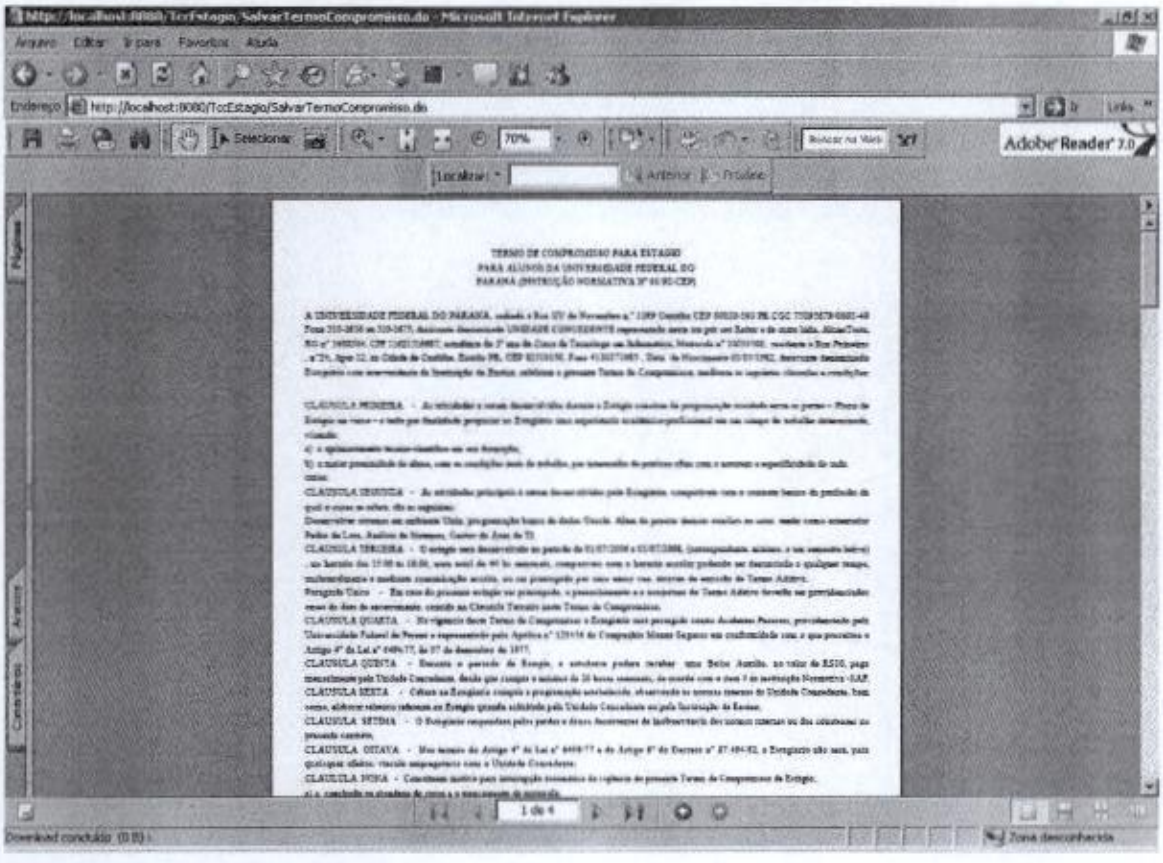
6.2.1.1.27 Fluxo Estágio Interno – Selecionando Estagiário



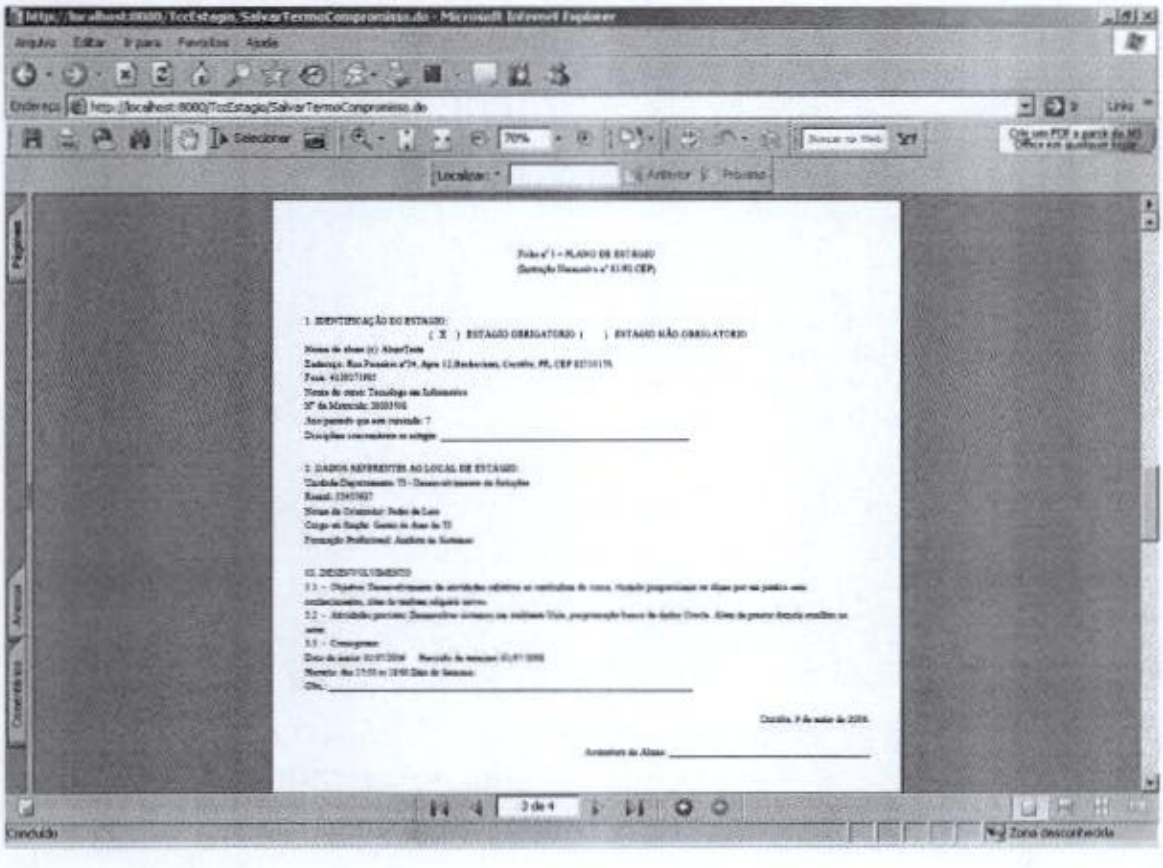
6.2.1.1.28 Fluxo Estágio Interno – Formulário de Preenchimento do Termo de Compromisso



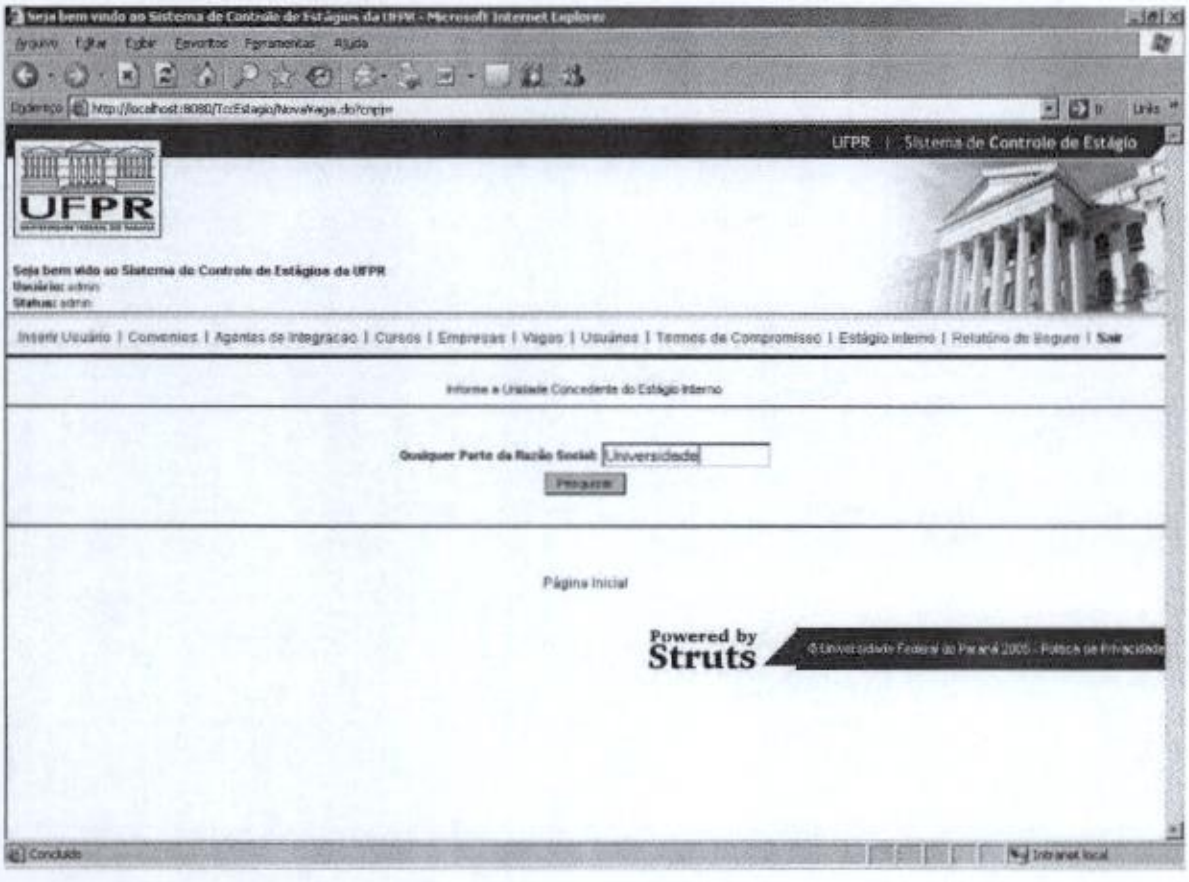
6.2.1.1.29 Fluxo Estágio Interno – Termo de Compromisso



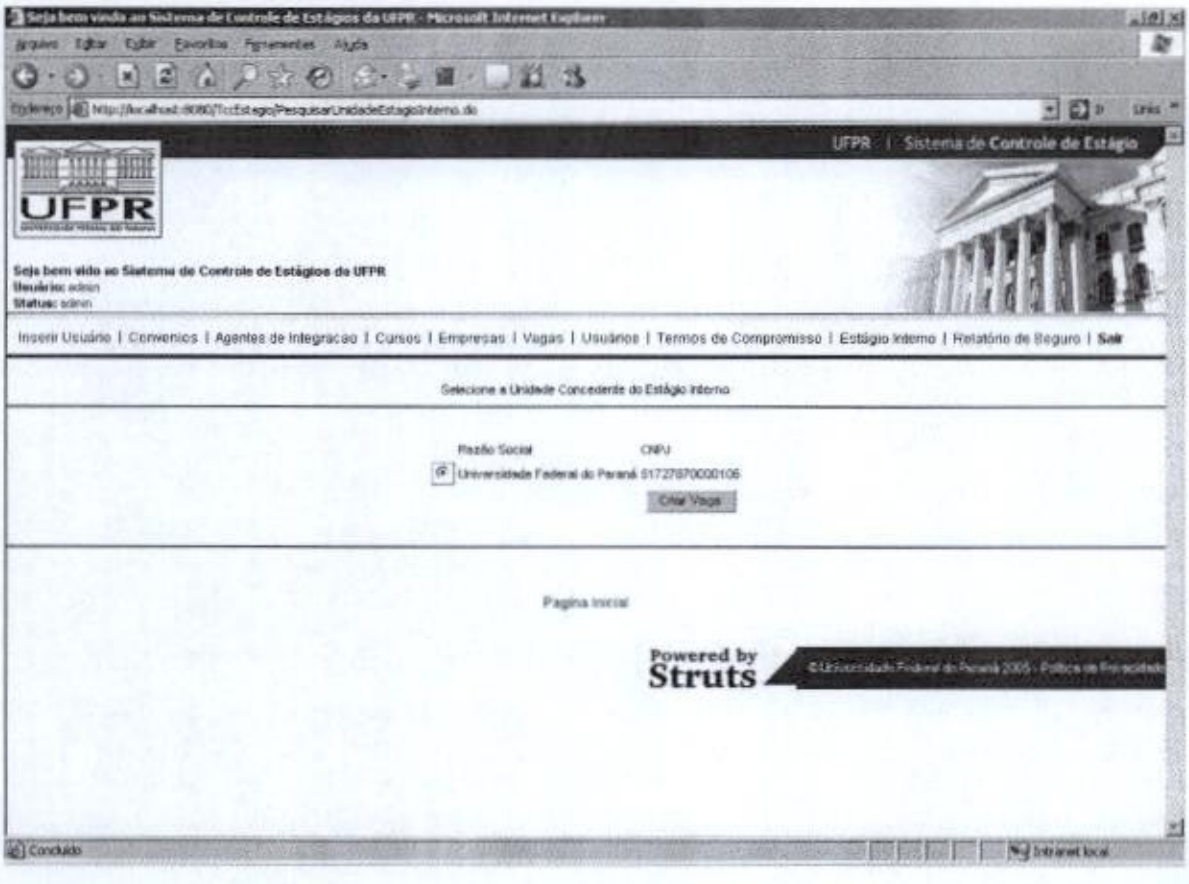
6.2.1.1.30 Fluxo Estágio Interno – Plano de Estágio



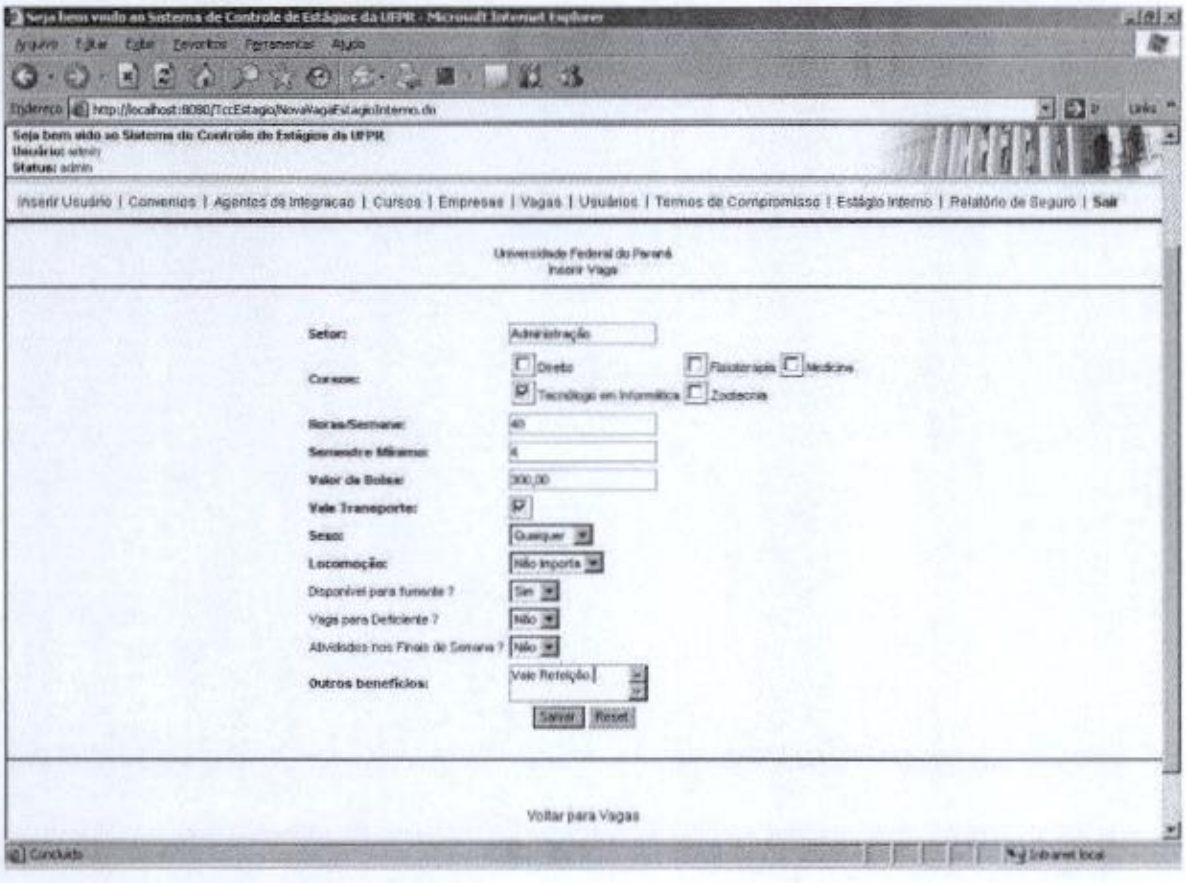
6.2.1.1.31 Fluxo Estágio Interno – Busca Unidade Concedente



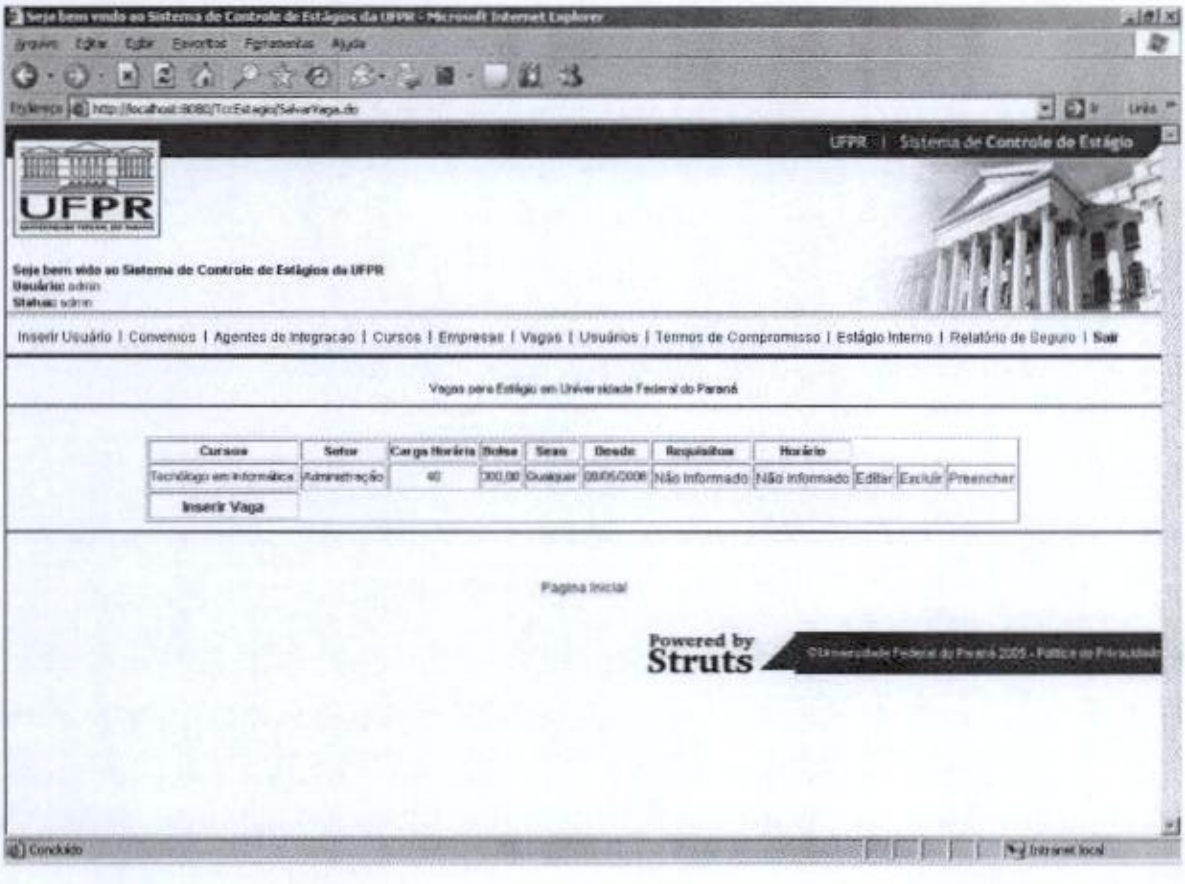
6.2.1.1.32 Fluxo Estágio Interno – Criar Vaga



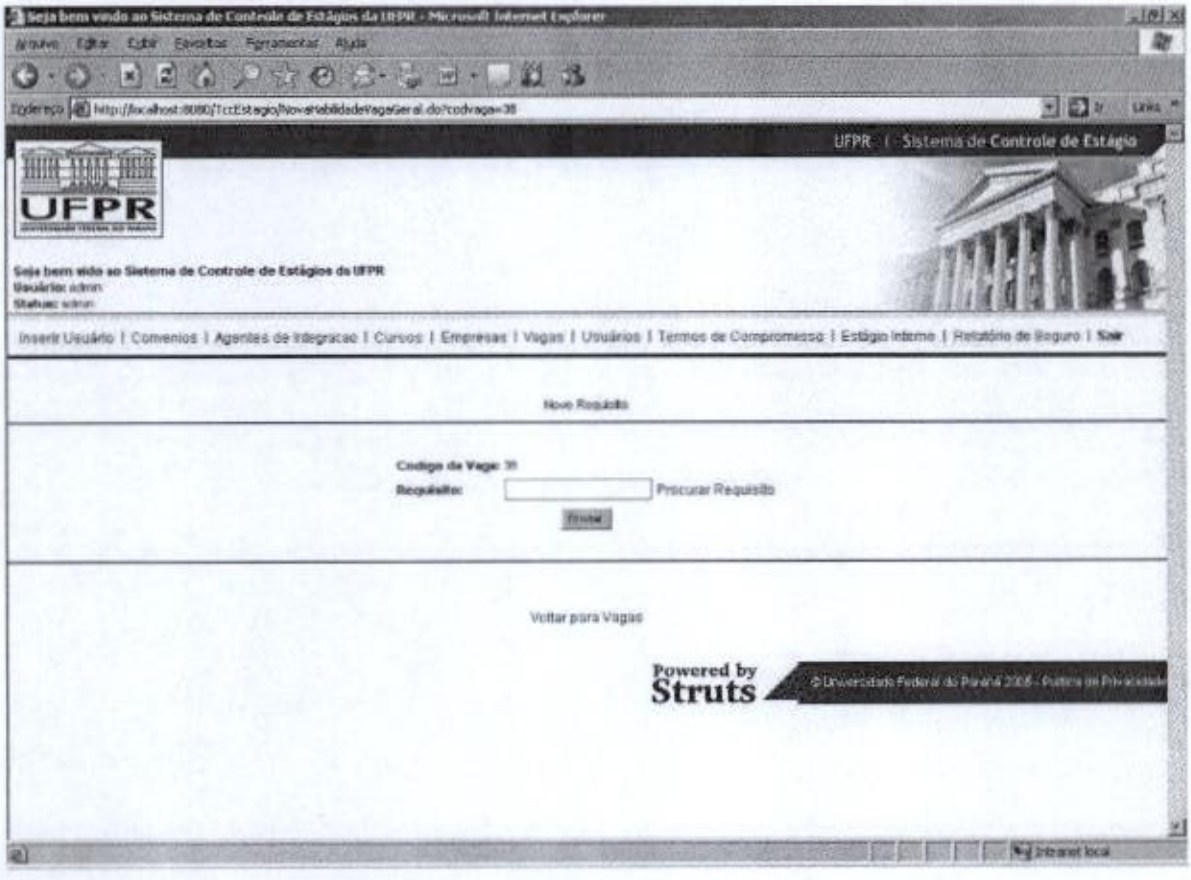
6.2.1.1.33 Fluxo Estágio Interno – Preenchendo Formulário para Inserir Vaga



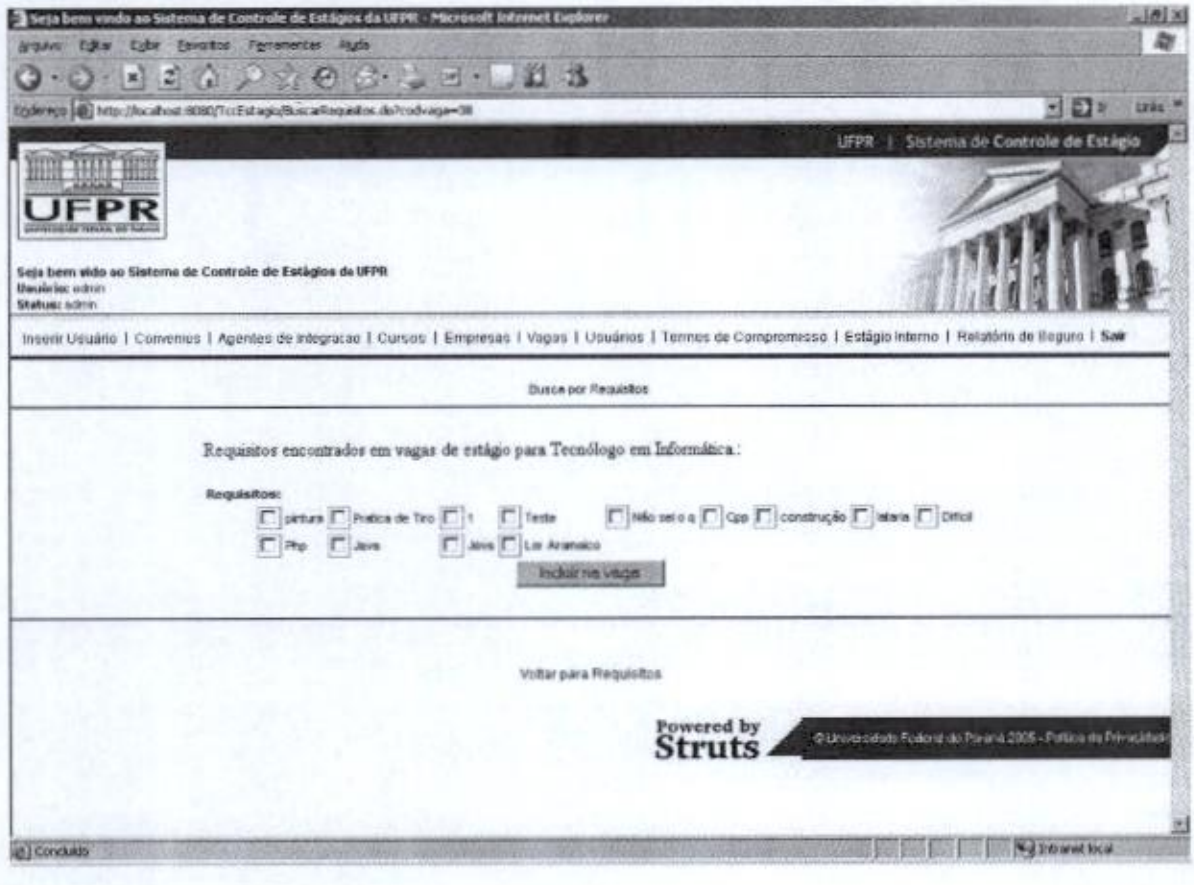
6.2.1.1.34 Fluxo Estágio Interno – Visualizando Vaga Inserida



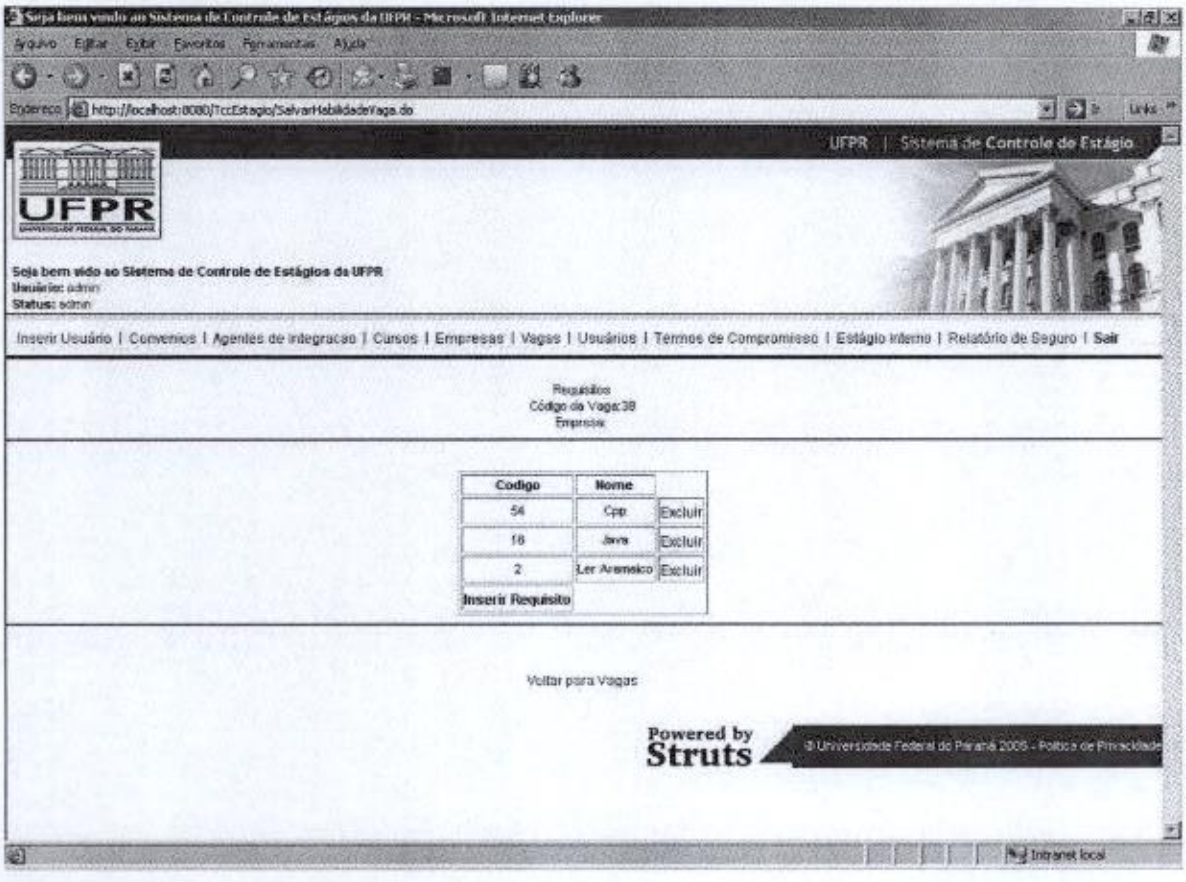
6.2.1.1.35 F luxo Estágio Interno – Inserir Requisito



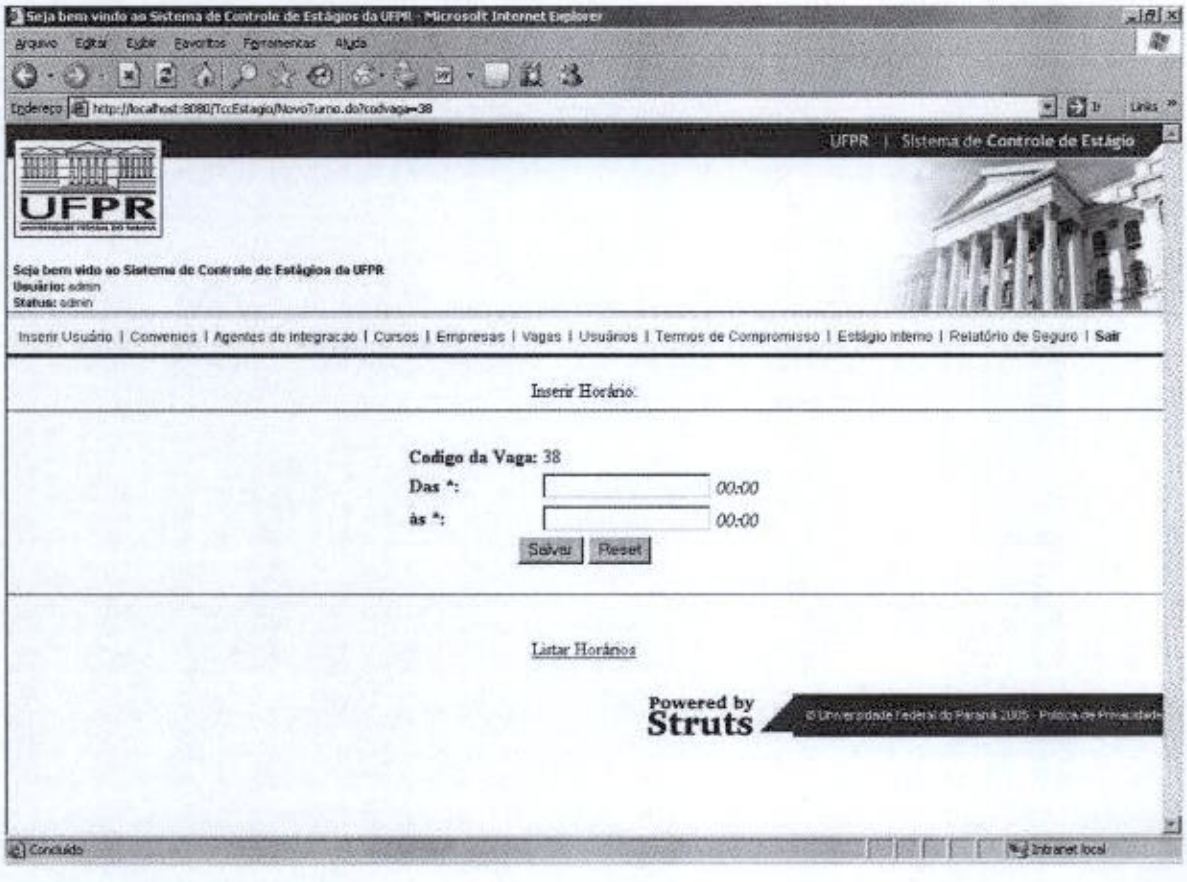
6.2.1.1.36 Fluxo Estágio Interno – Procurar Requisitos já cadastrados



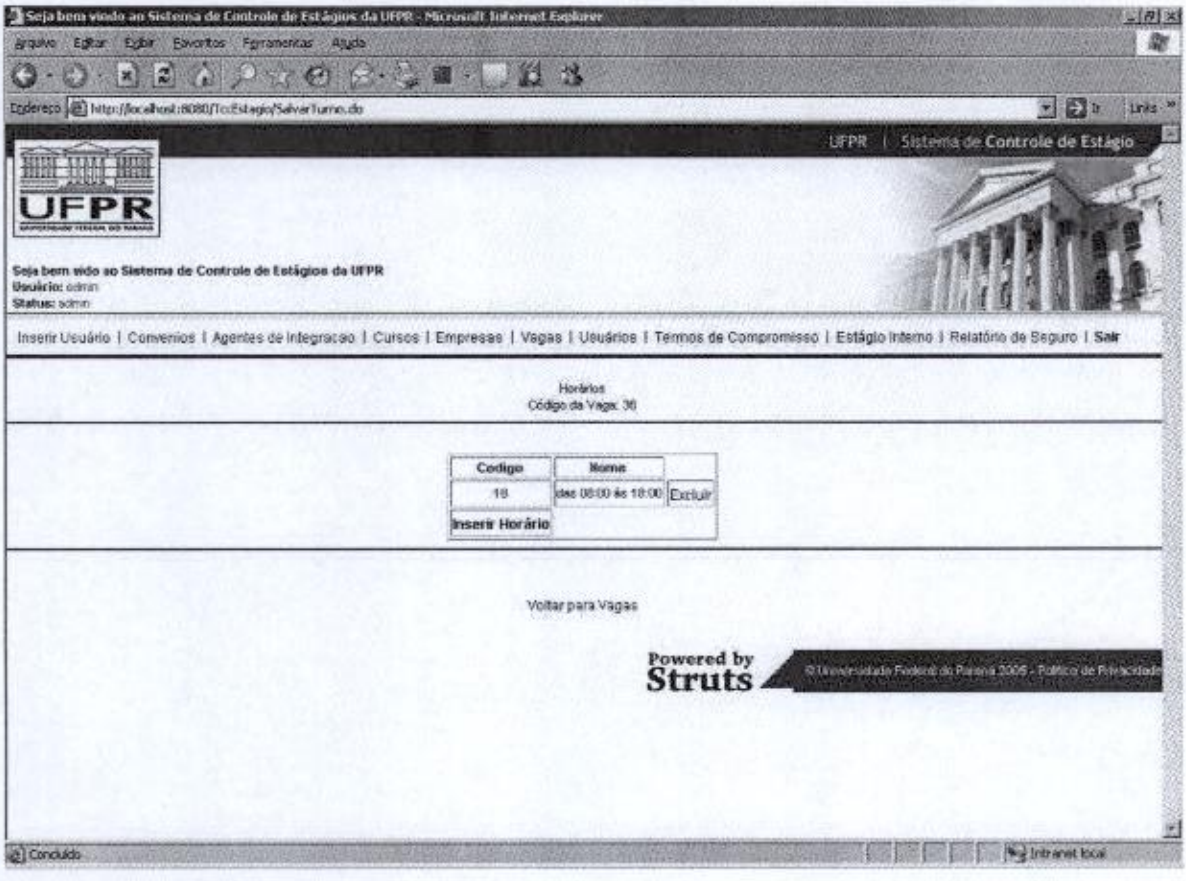
6.2.1.1.37 Fluxo Estágio Interno – Visualizando Requisitos



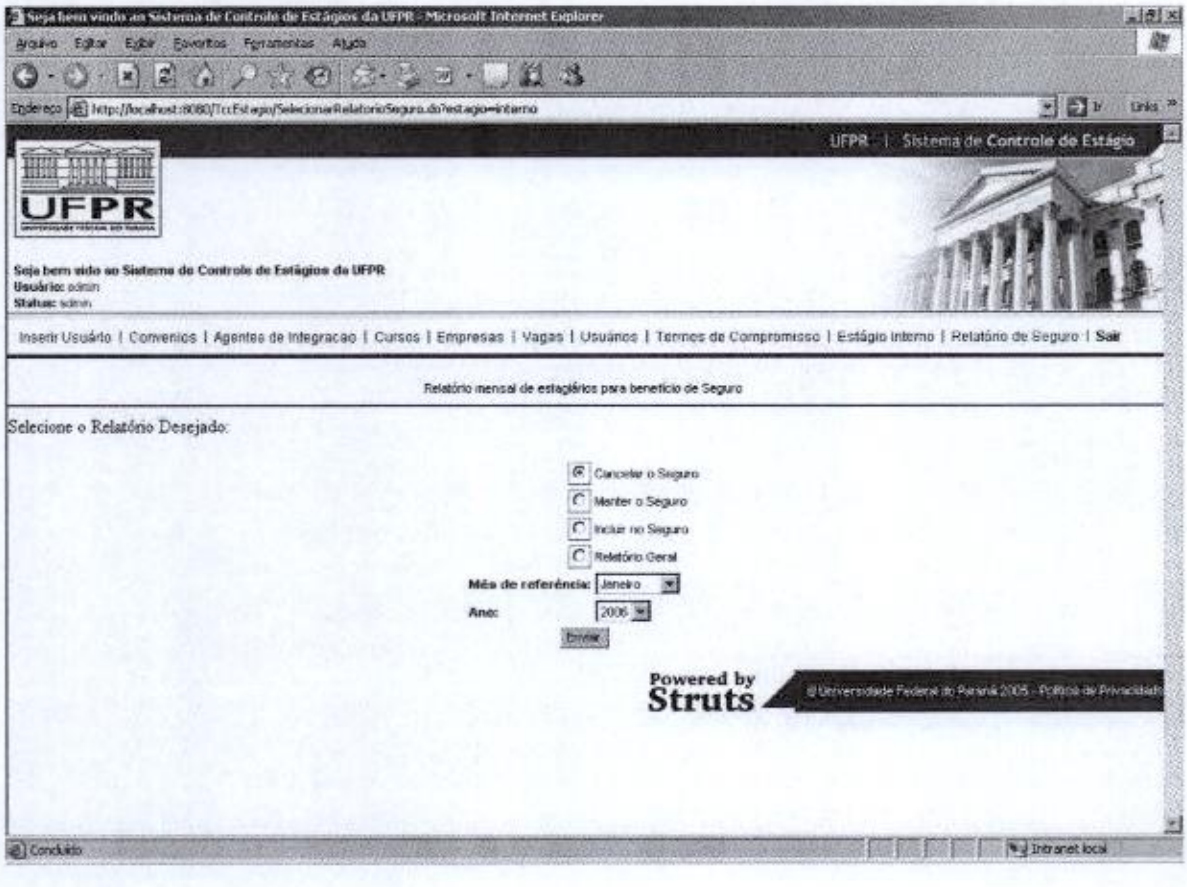
6.2.1.1.38 Fluxo Estágio Interno – Inserir Horários



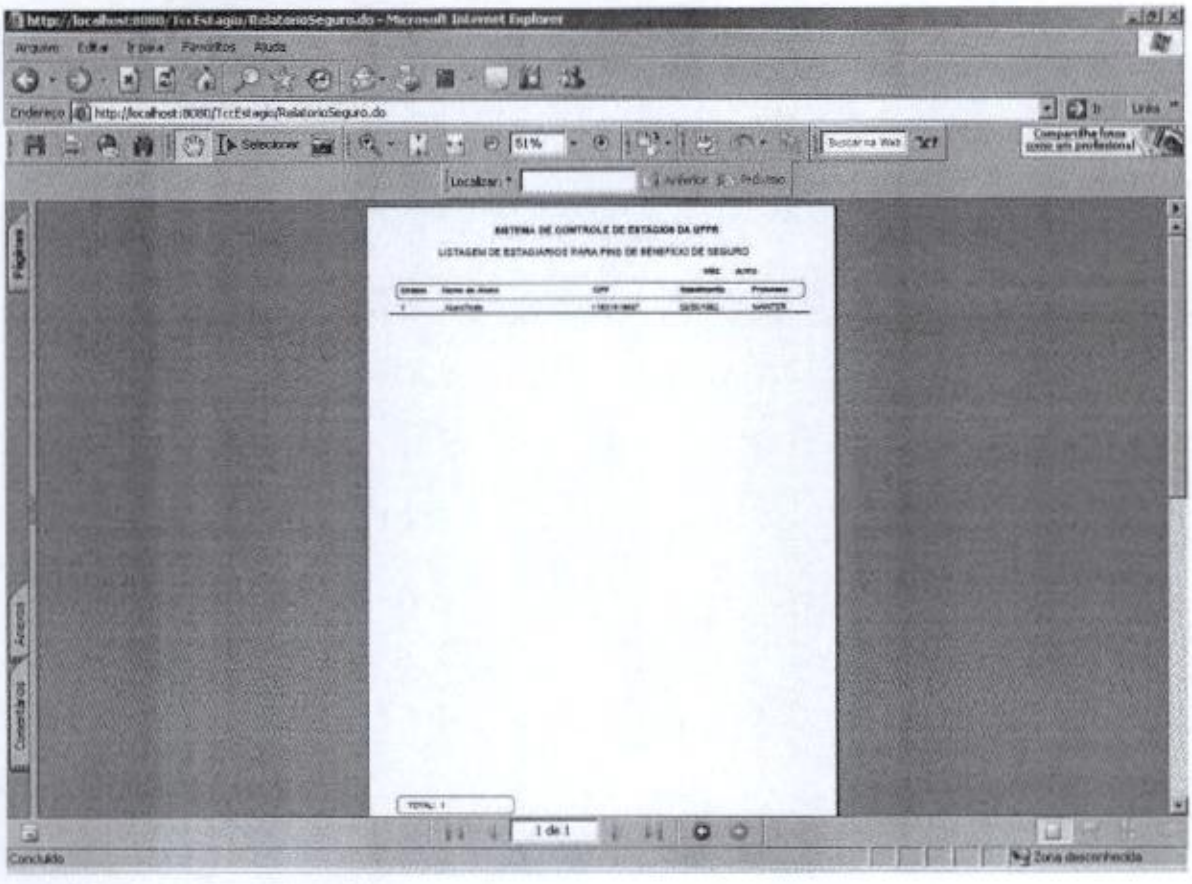
6.2.1.1.39 Fluxo Estágio Interno – Visualizando Horários



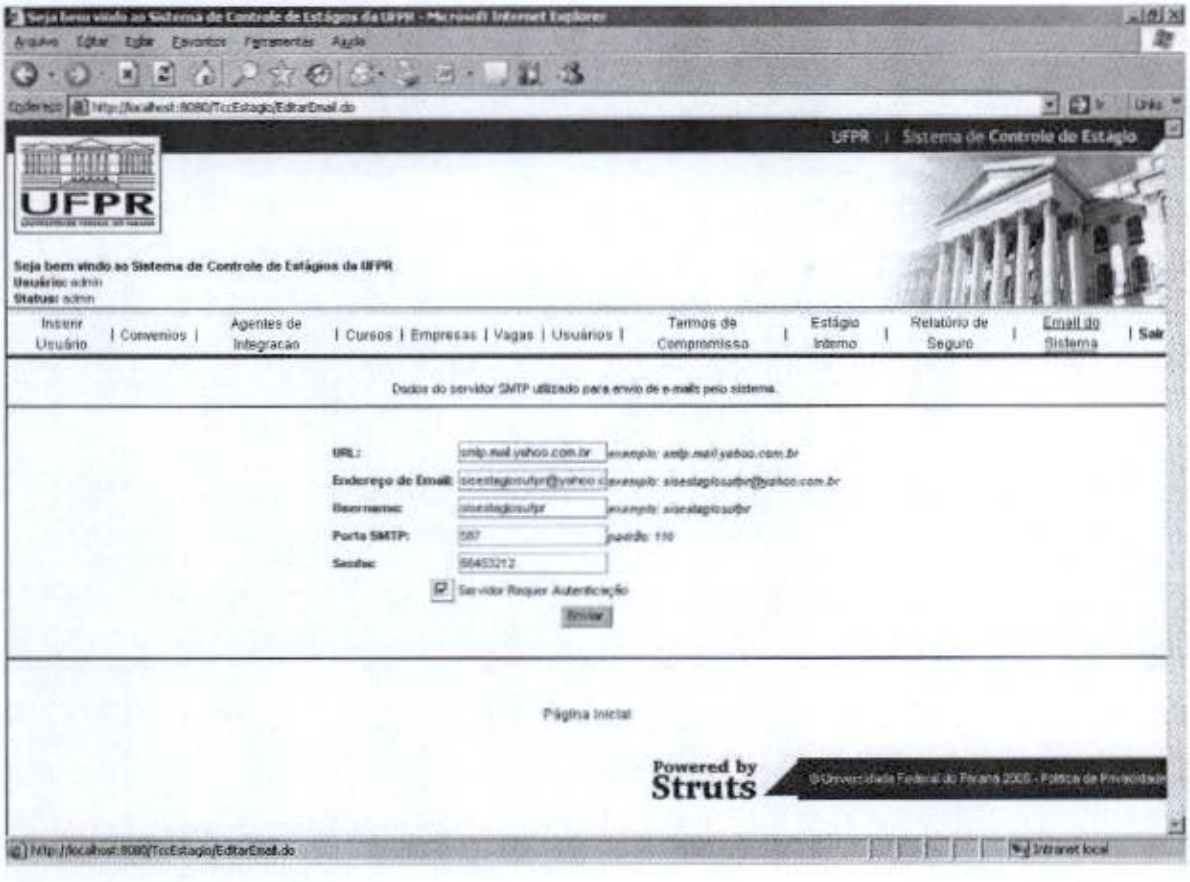
6.2.1.1.40 Fluxo Relatório de Seguro – Selecionando Relatório



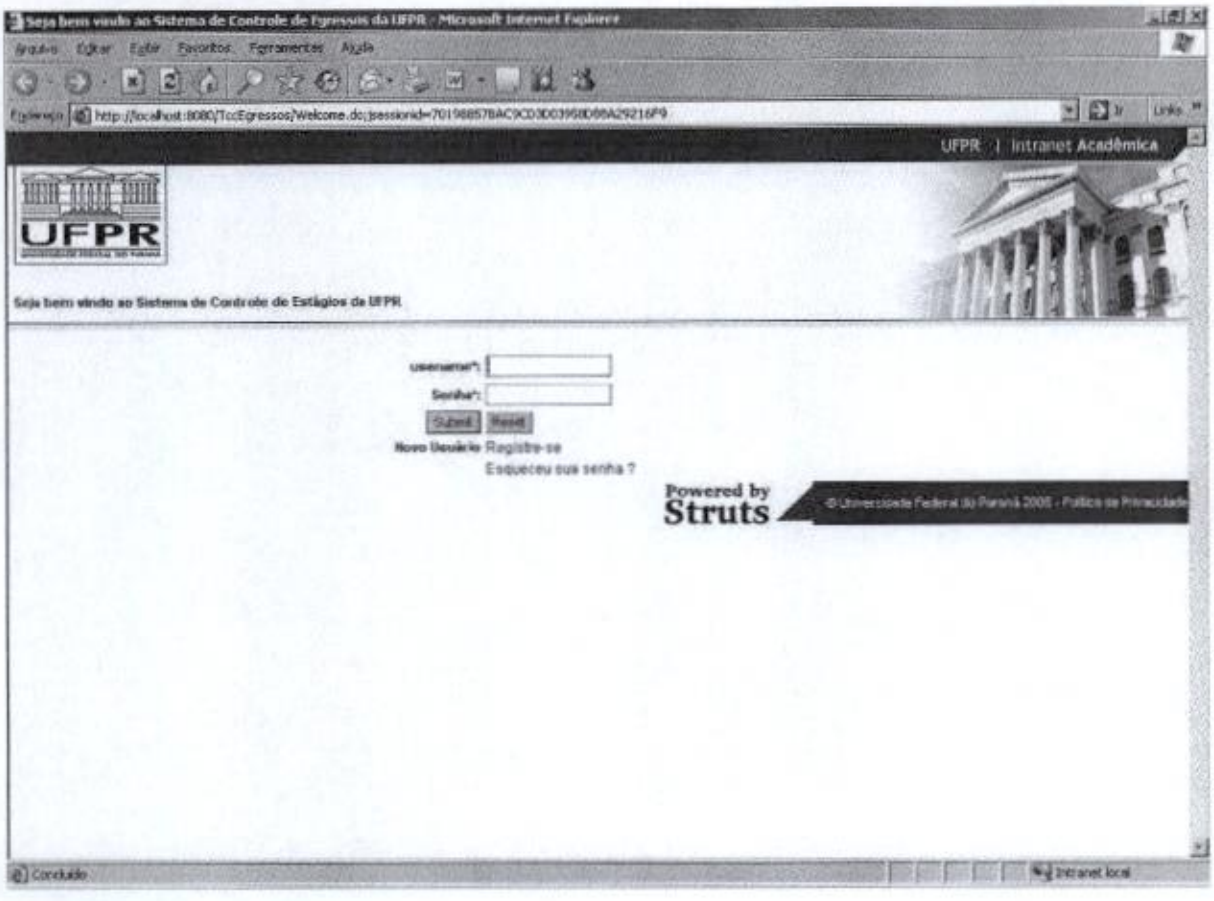
6.2.1.1.41 Fluxo Relatório de Seguro – Relatório de Seguro



6.2.1.1.42 Fluxo Email do Sistema

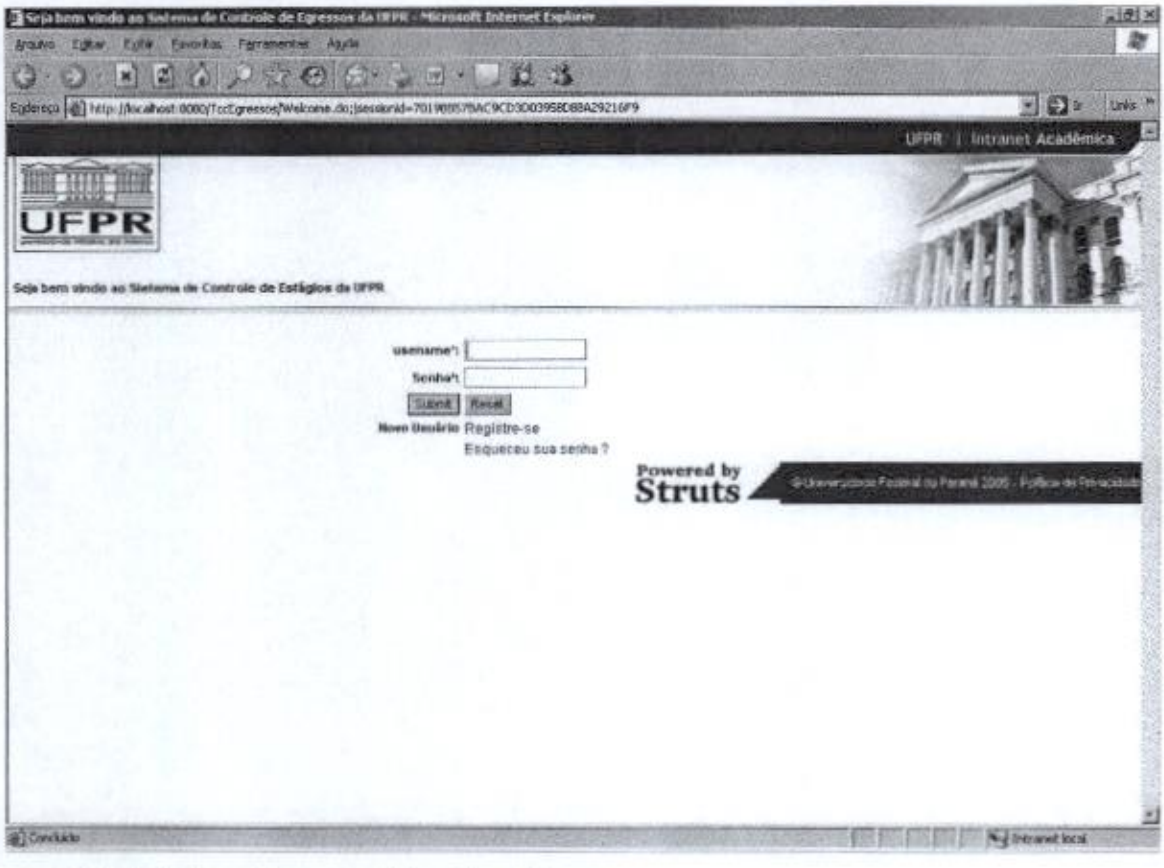


6.2.1.1.43 Sair

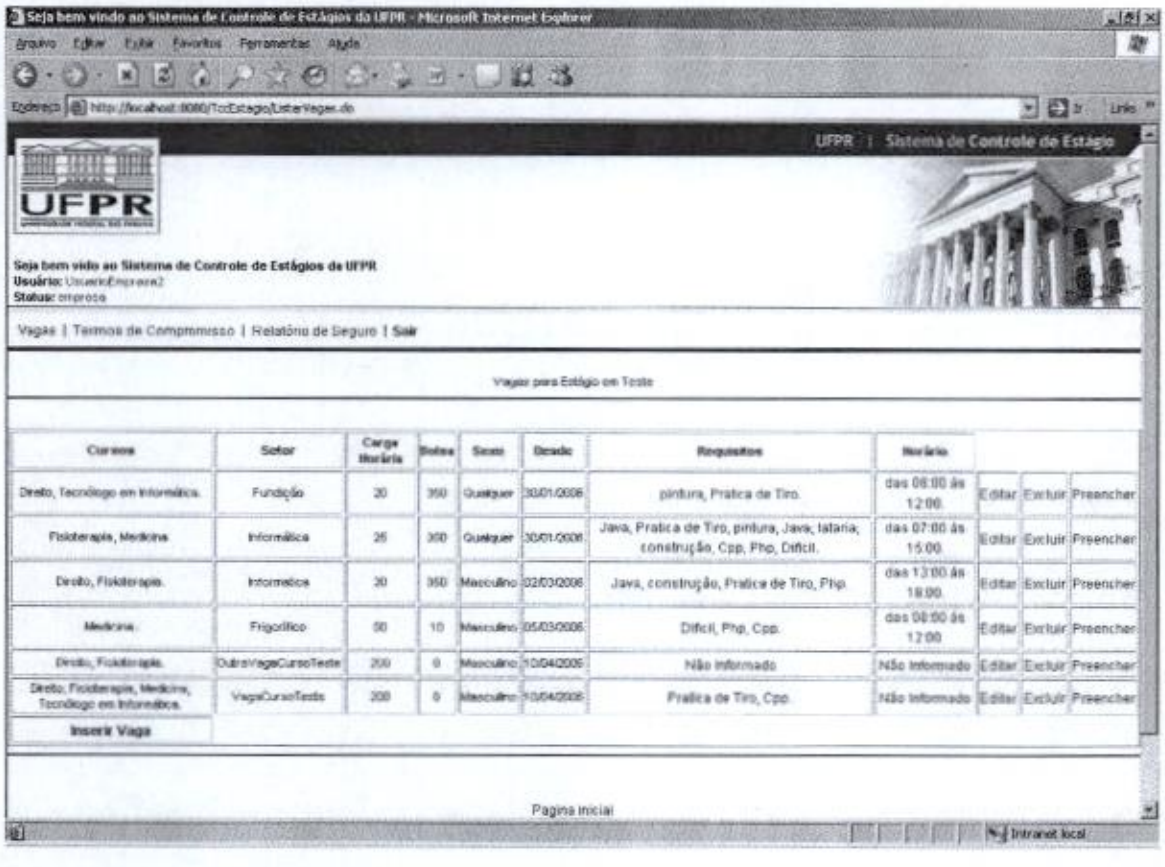

UFPR | Intranet Acadêmica
UFPR
Seja bem vindo ao Sistema de Controle de Estágios da UFPR
Powered by
Struts

6.2.1.2 Visão Empresas

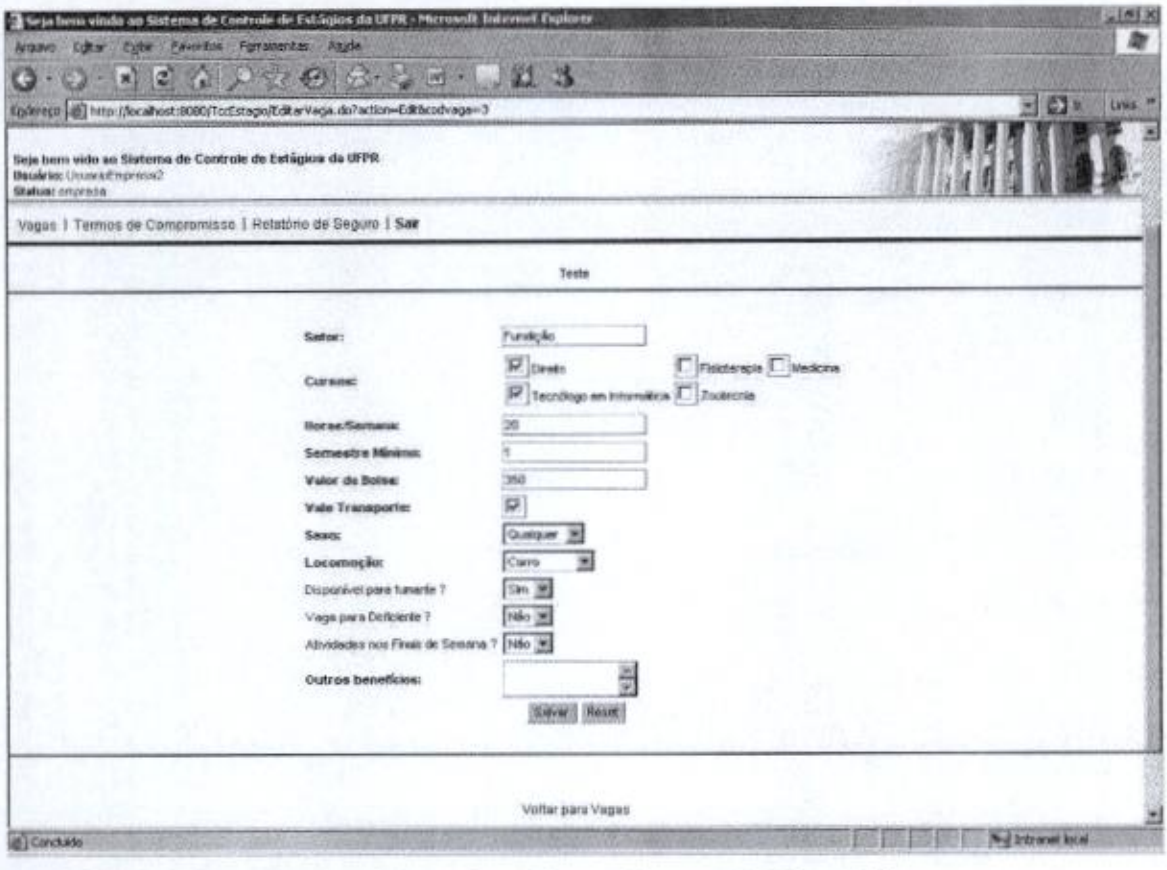
6.2.1.2.1 Login



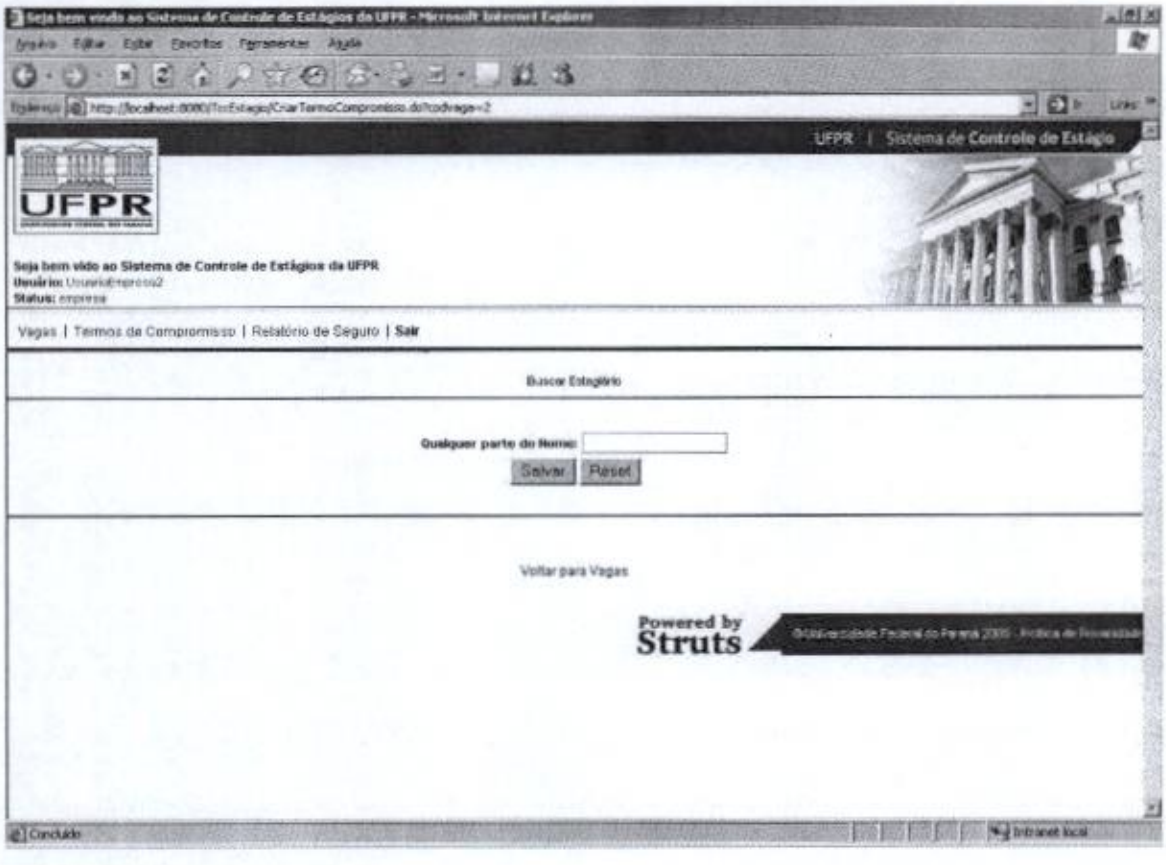
6.2.1.2.2 Fluxo Vagas – Visualizar Vagas



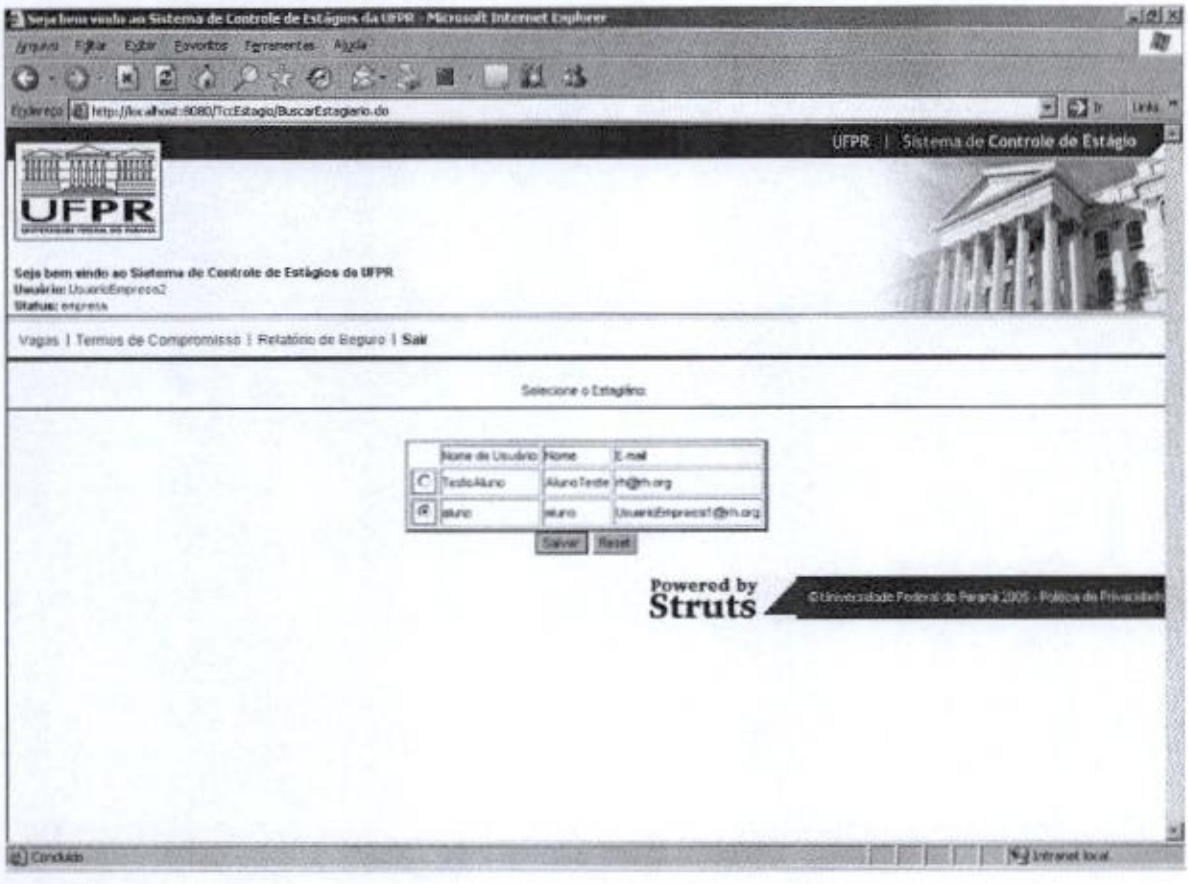
6.2.1.2.3 Fluxo Vagas – Editar Vaga



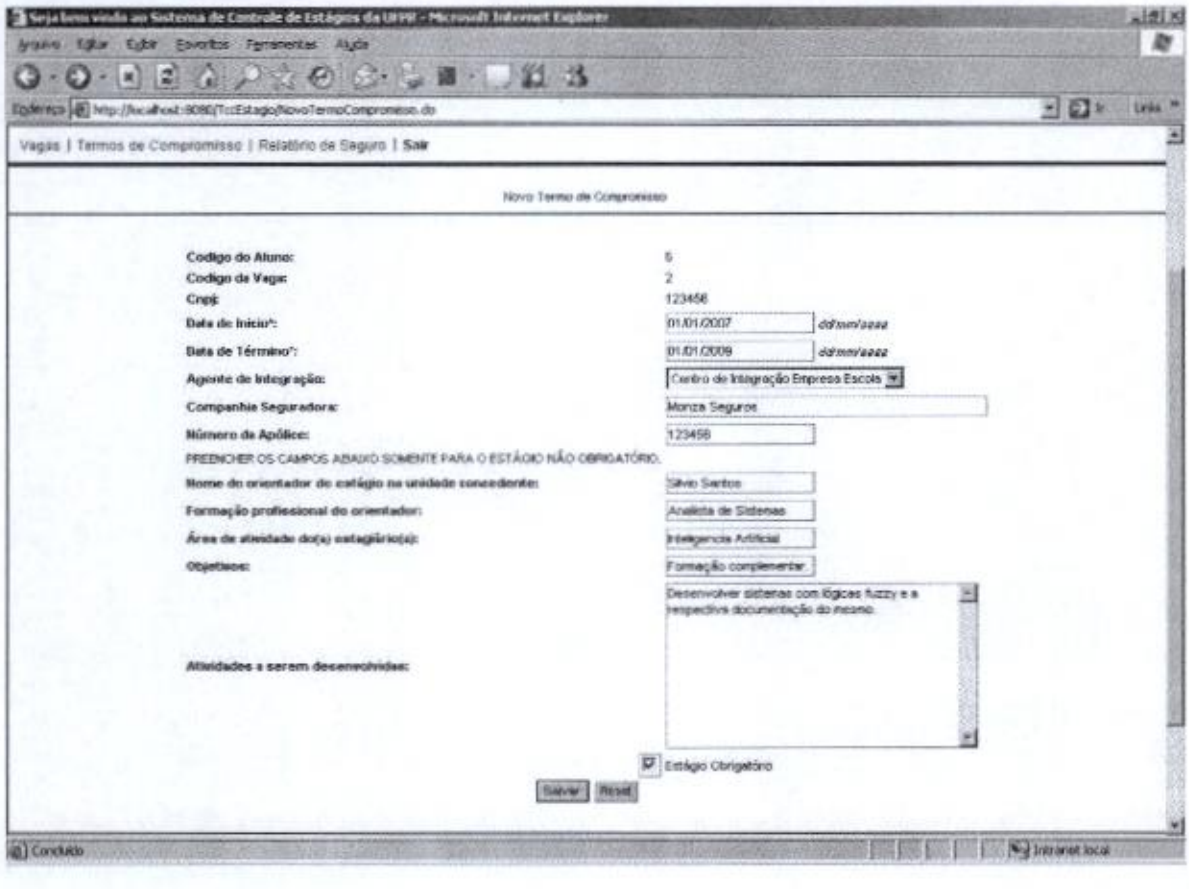
6.2.1.2.4 Fluxo Vagas – Preencher Vaga – Buscando Estagiário



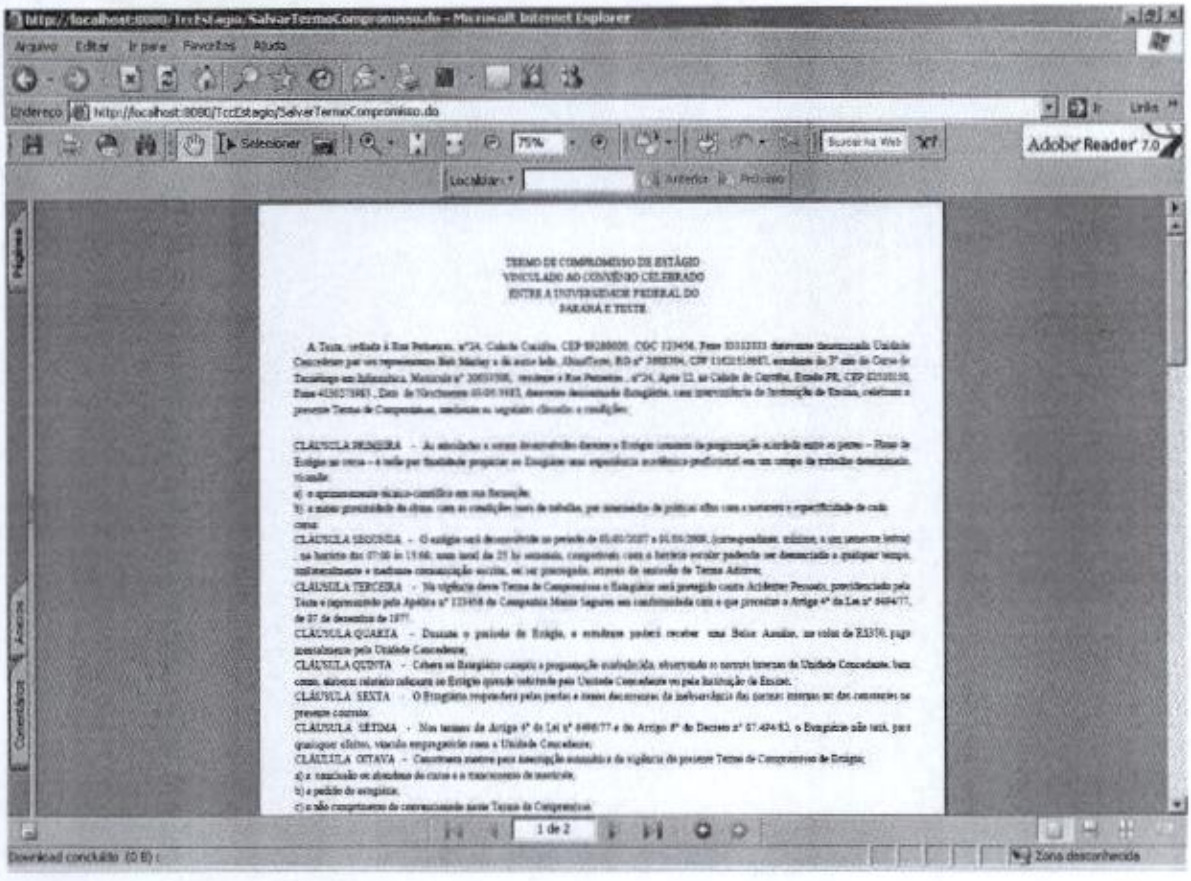
6.2.1.2.5 Fluxo Vagas – Selecionando Estagiário



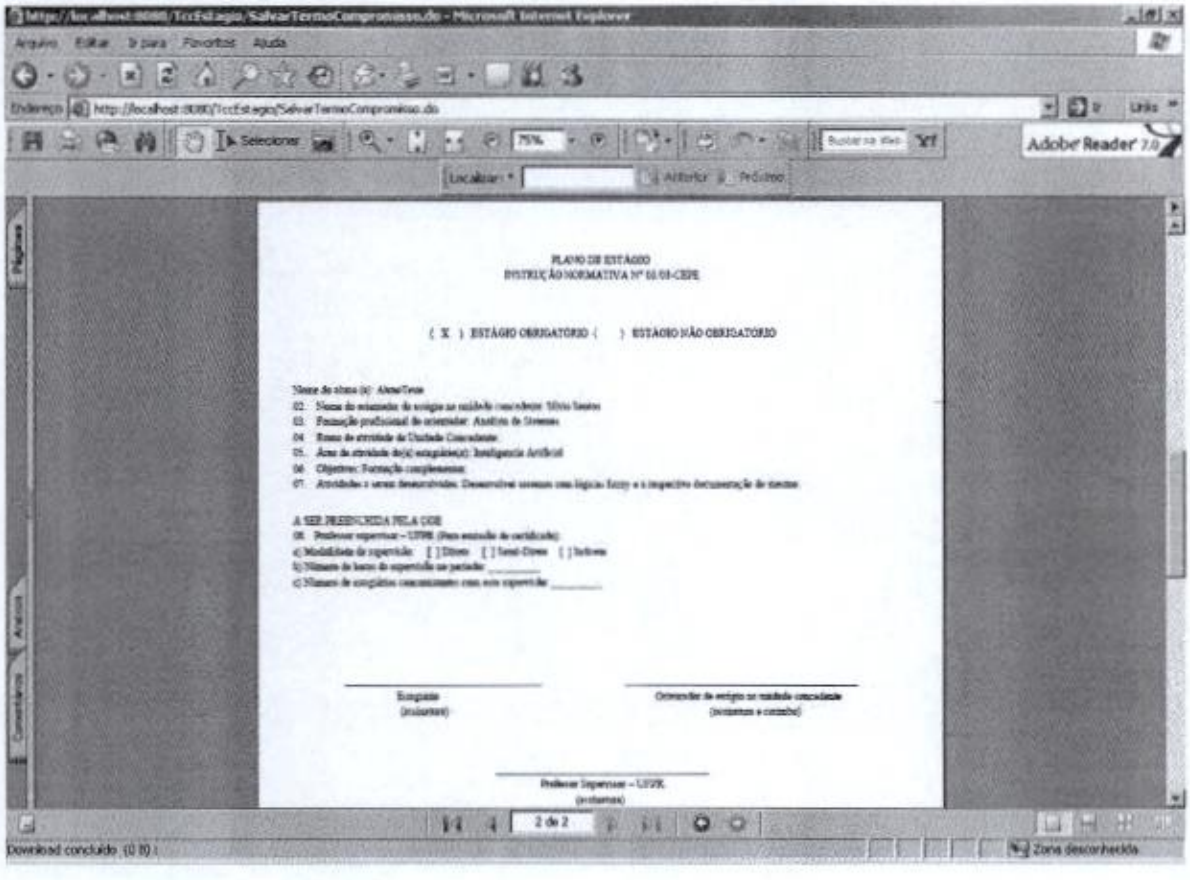
6.2.1.2.6 Fluxo Vagas – Formulário para Novo Termo de Compromisso



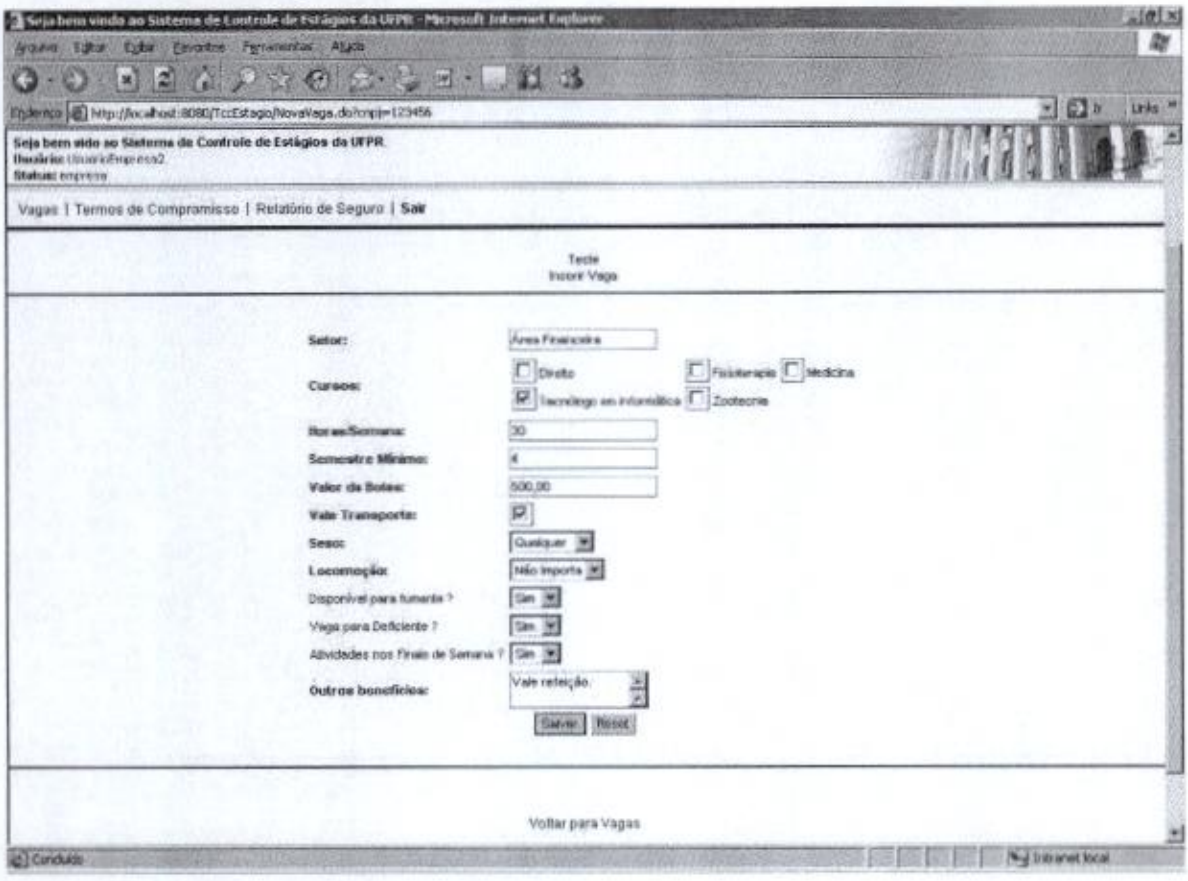
6.2.1.2.7 Fluxo Vagas – Termo de Compromisso



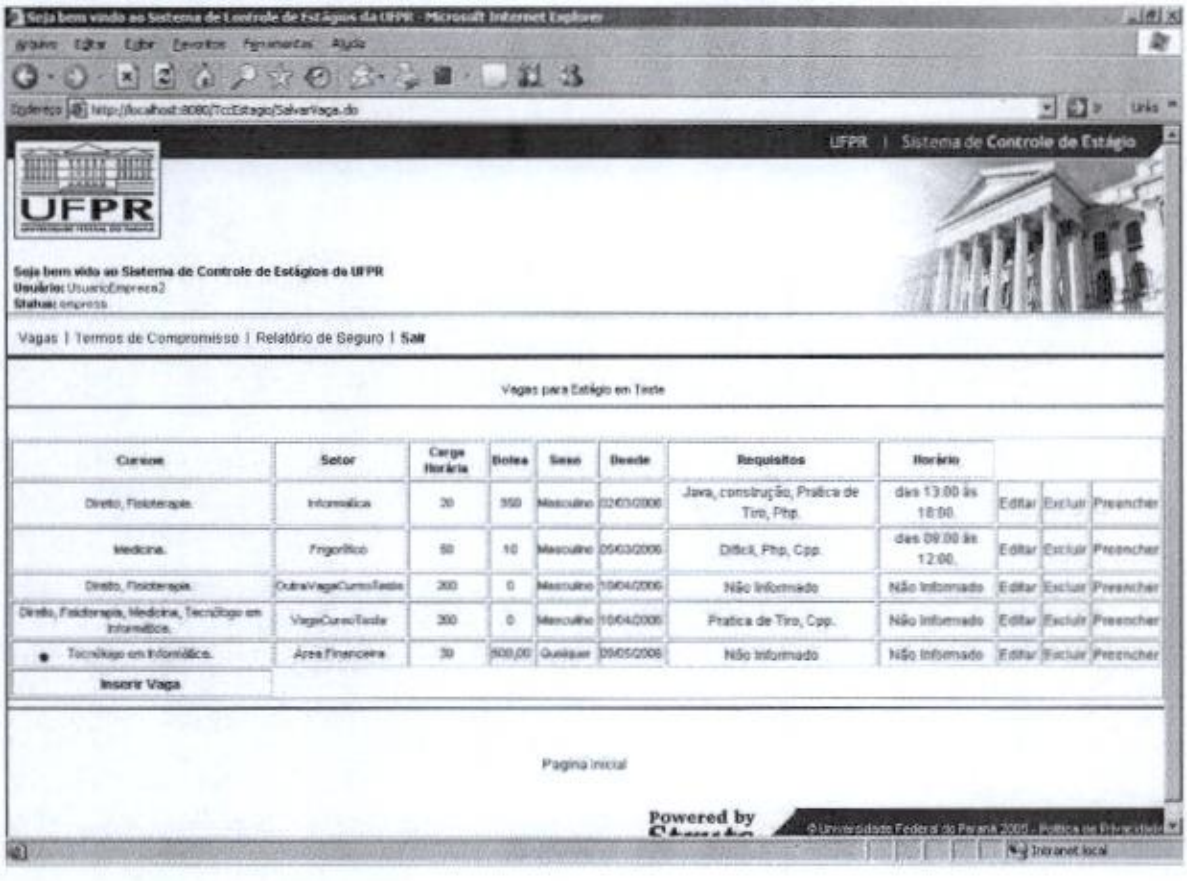
6.2.1.2.8 Fluxo Vagas – Plano de Estágio



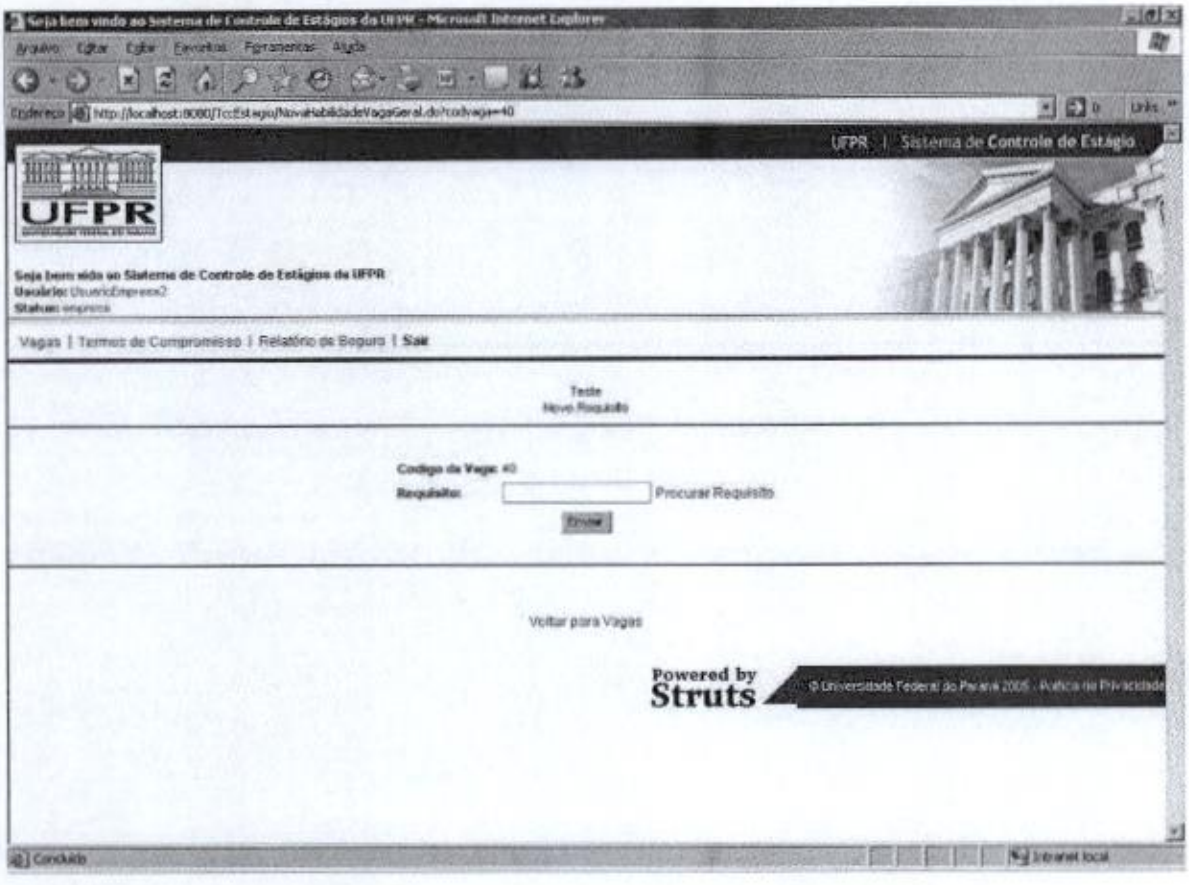
6.2.1.2.9 Fluxo Vagas – Inserir Vaga



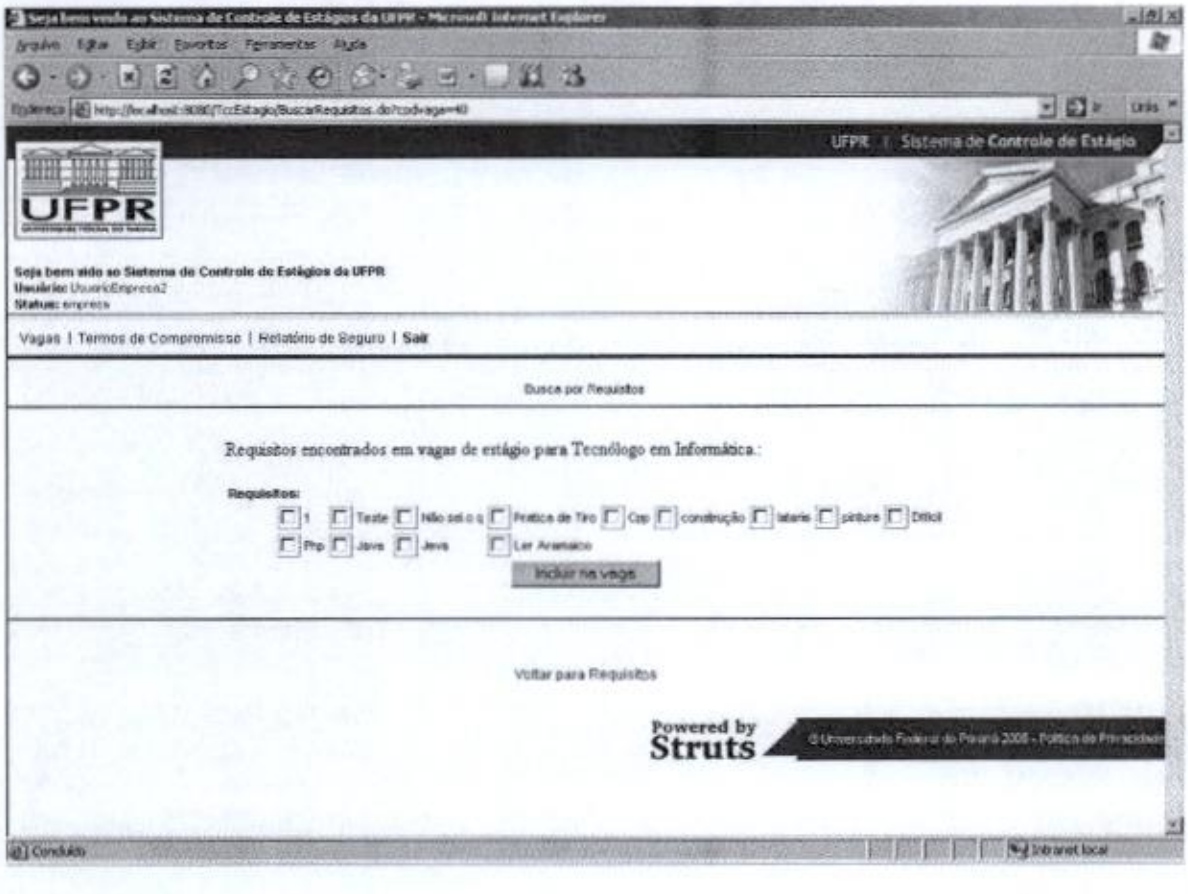
6.2.1.2.10 Fluxo Vagas – Visualizando Vaga Inserida



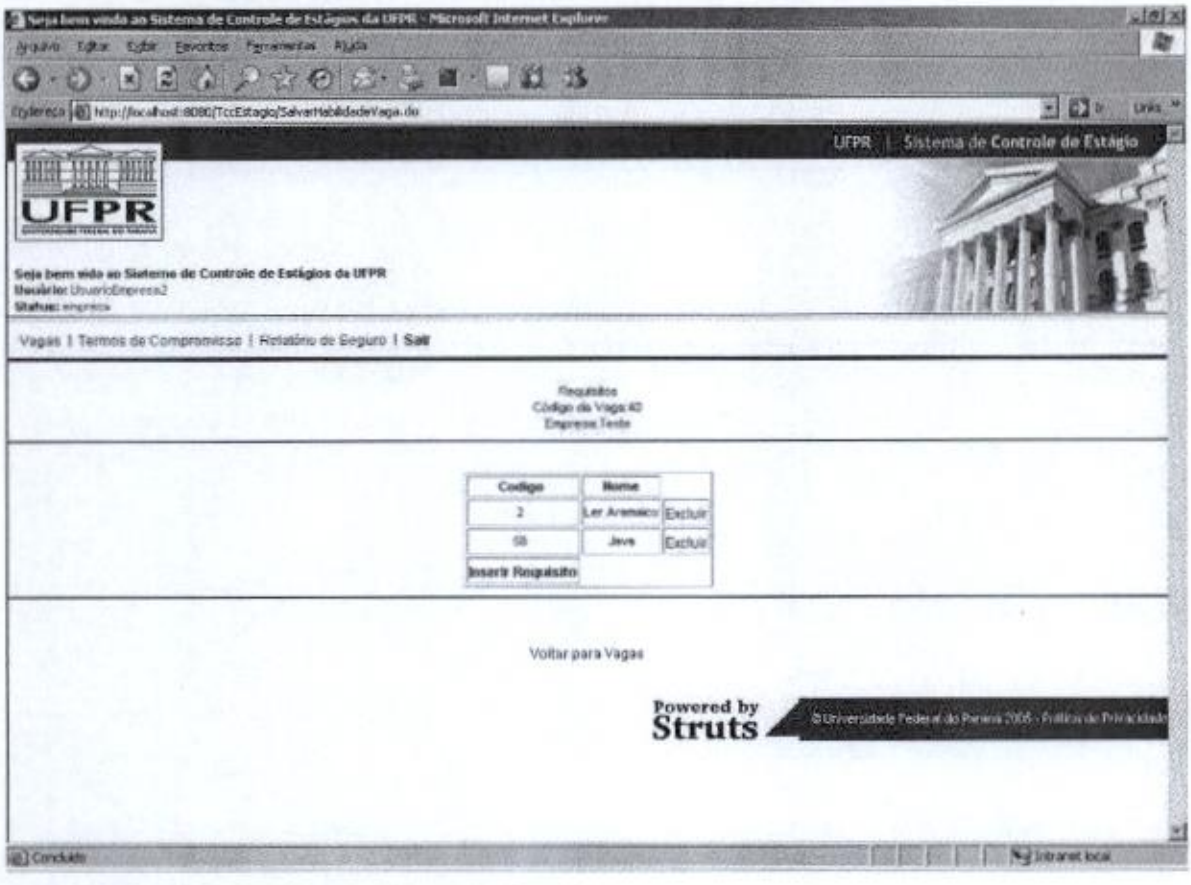
6.2.1.2.11 Fluxo Vagas – Inserir Requisito



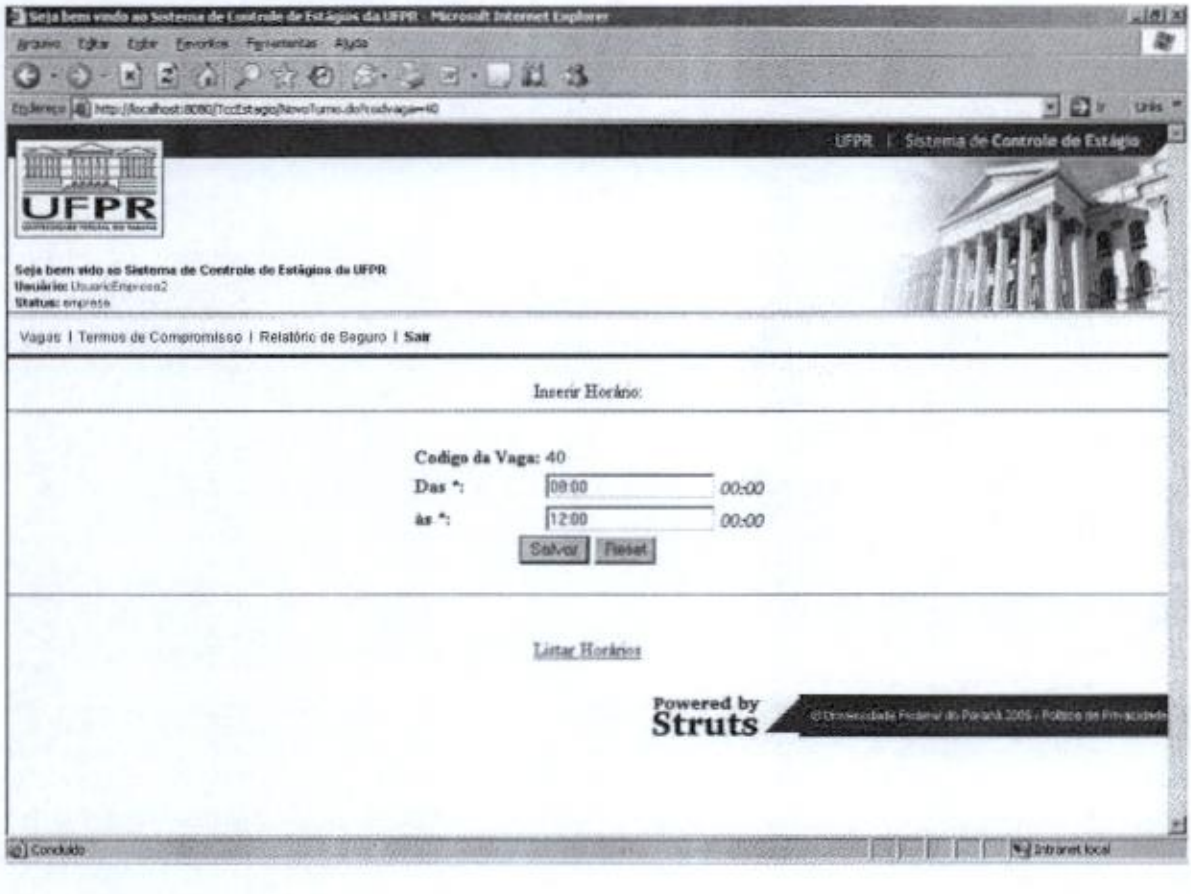
6.2.1.2.12 Fluxo Vagas – Buscando por requisitos já cadastrados



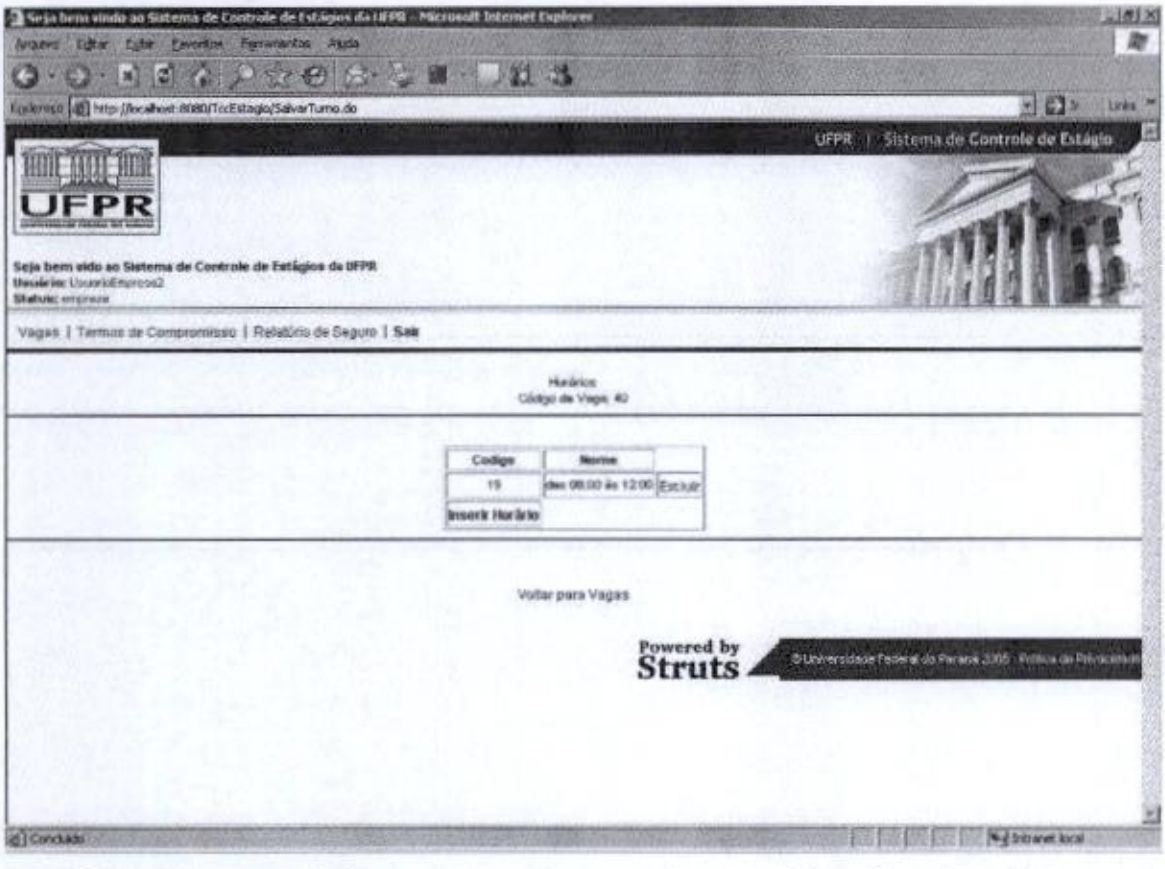
6.2.1.2.13 Fluxo Vagas – Visualizando Requisitos Inseridos



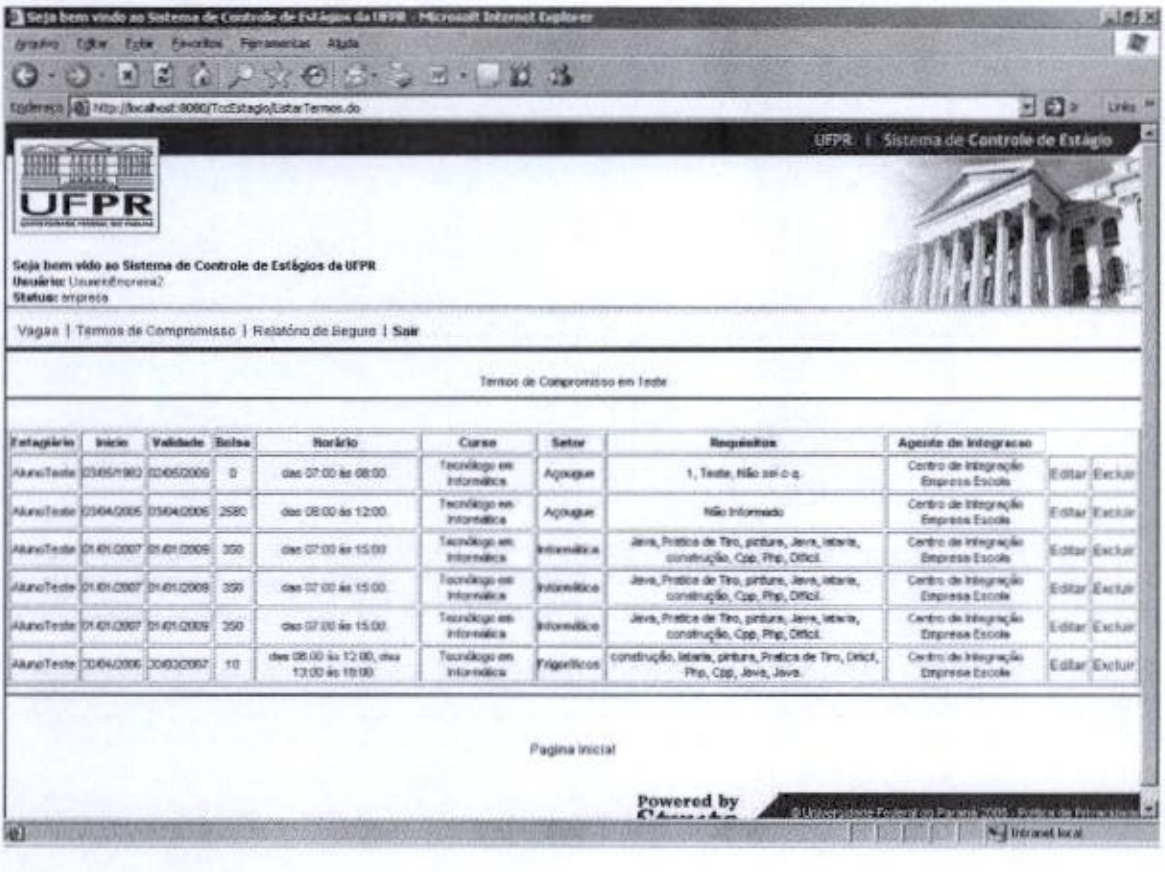
6.2.1.2.14 Fluxo Vagas – Inserir Horários



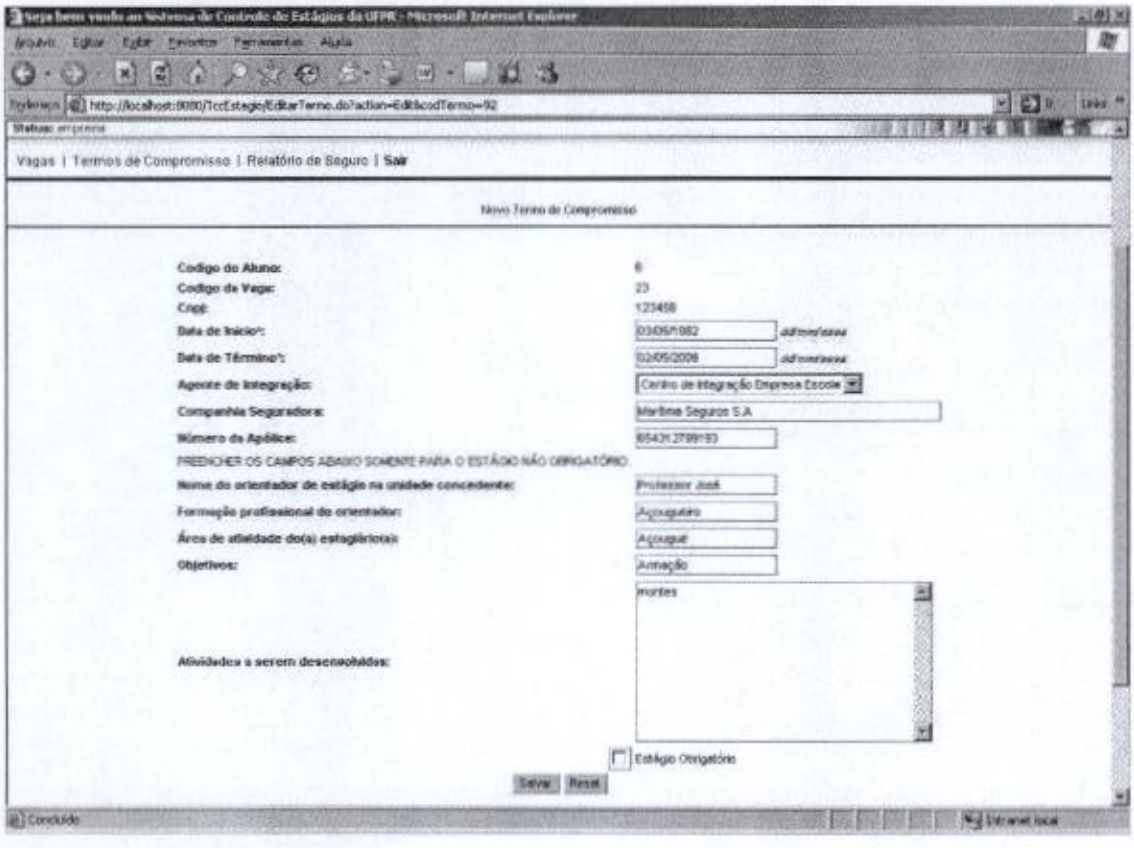
6.2.1.2.15 Fluxo Vagas – Visualizando Horários Inseridos



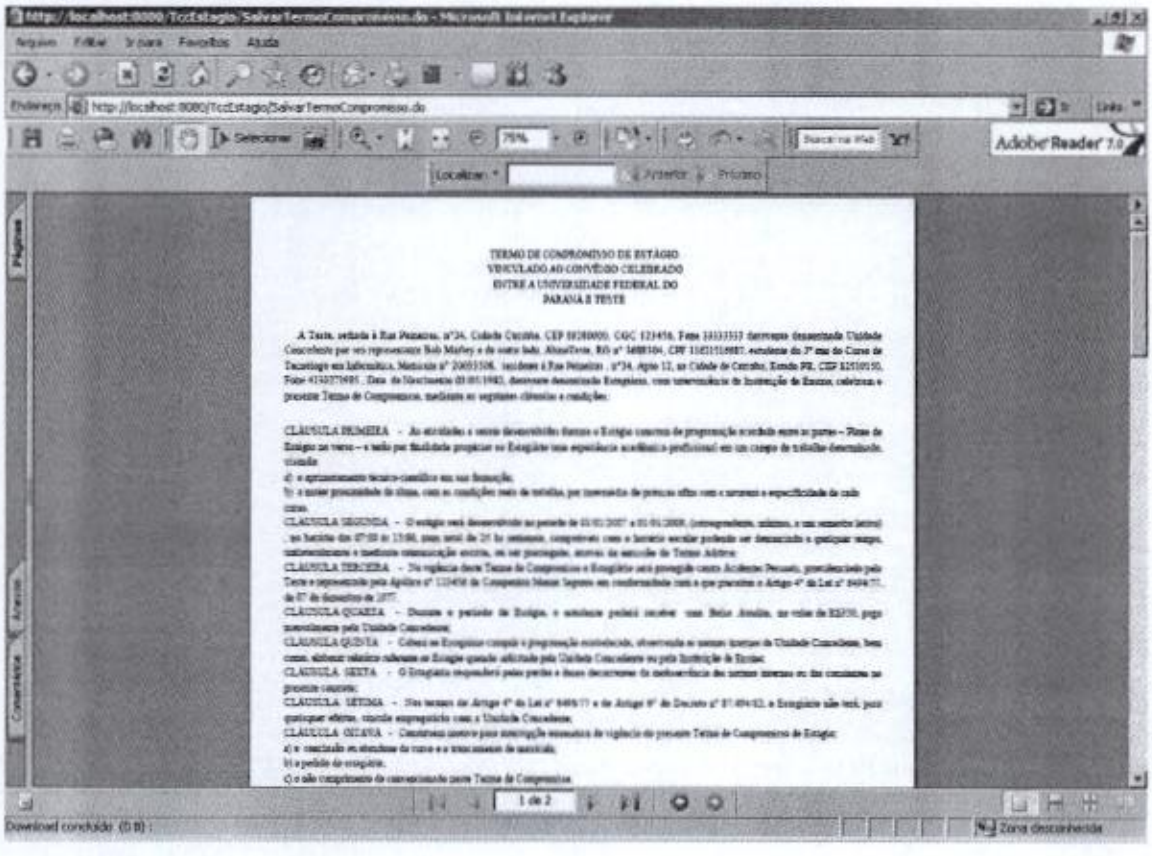
6.2.1.2.16 Fluxo Termos de Compromisso – Visualizando Termos de Compromisso inseridos



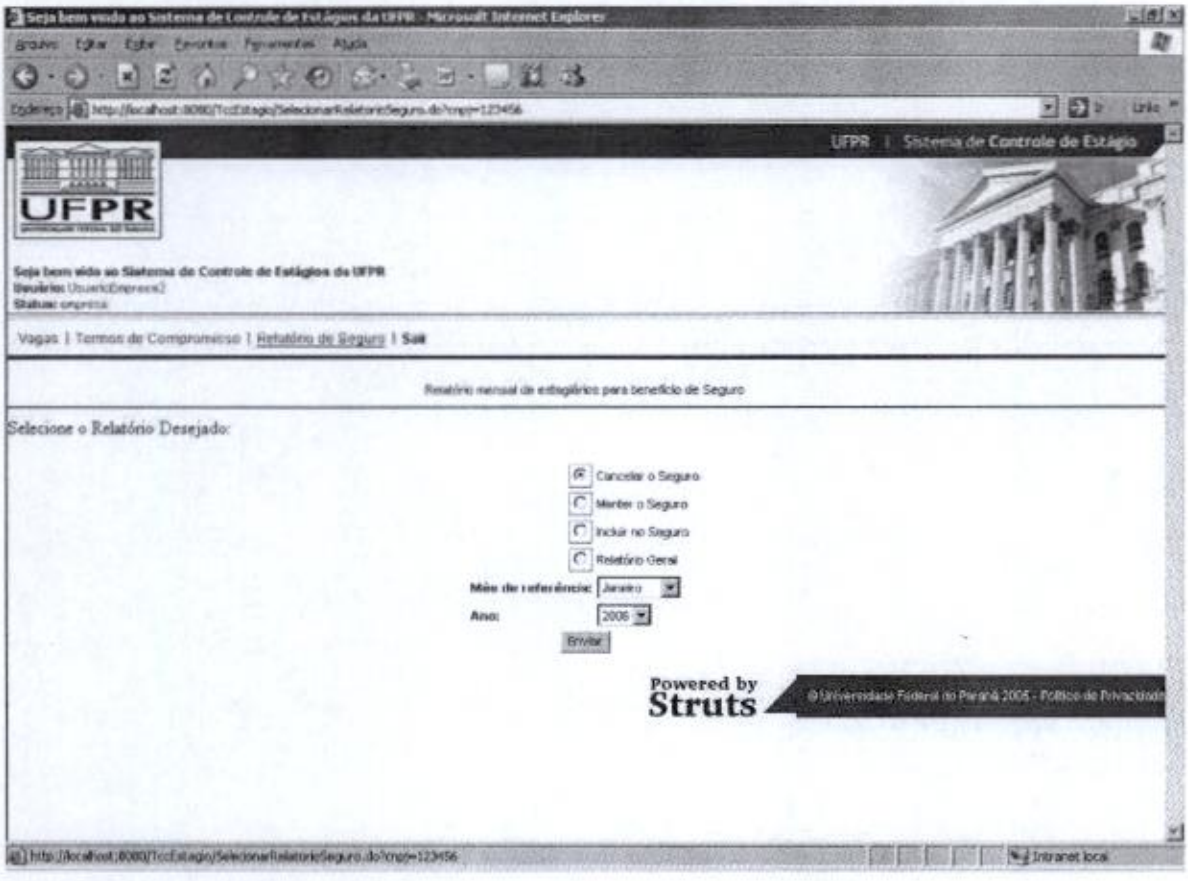
6.2.1.2.17 Fluxo Termos de Compromisso – Editar Termo de Compromisso



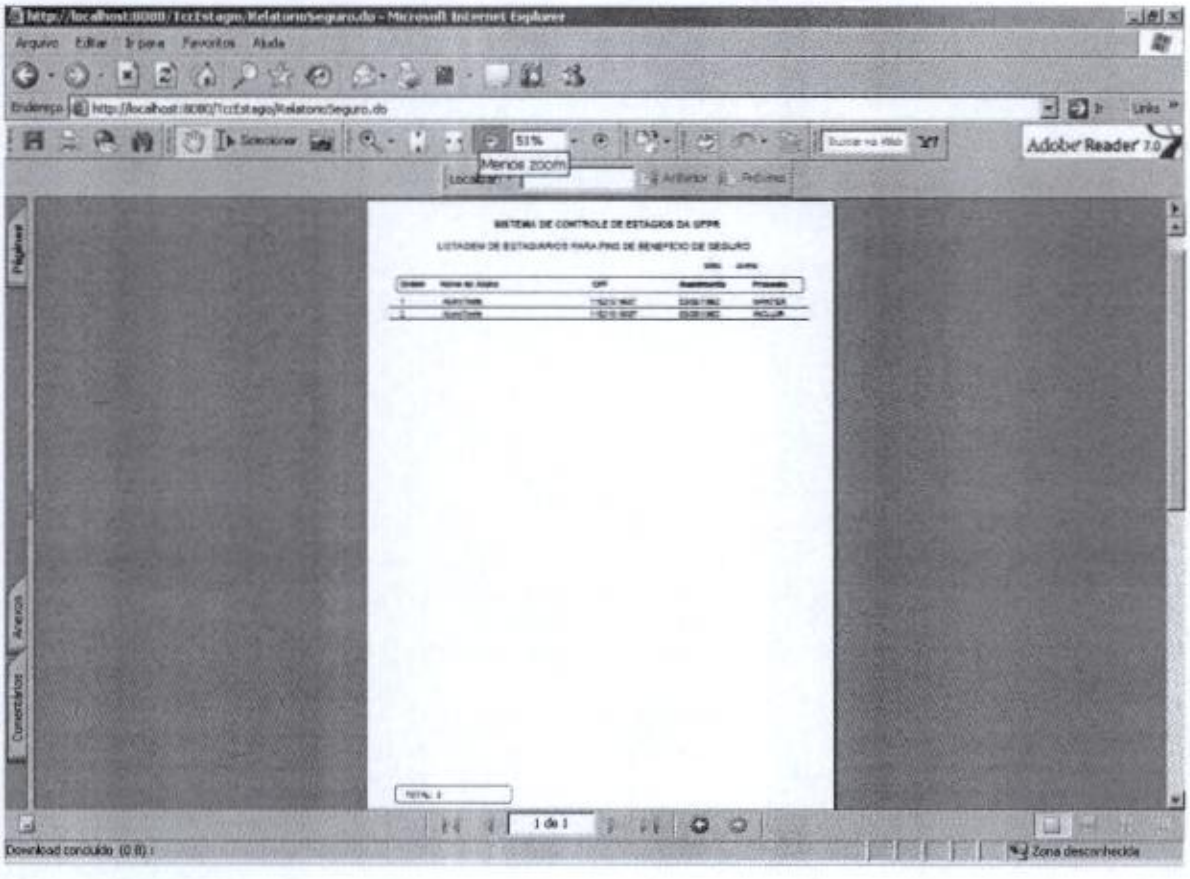
6.2.1.2.18 Fluxo Termos de Compromisso – Visualizando Termo de Compromisso alterado



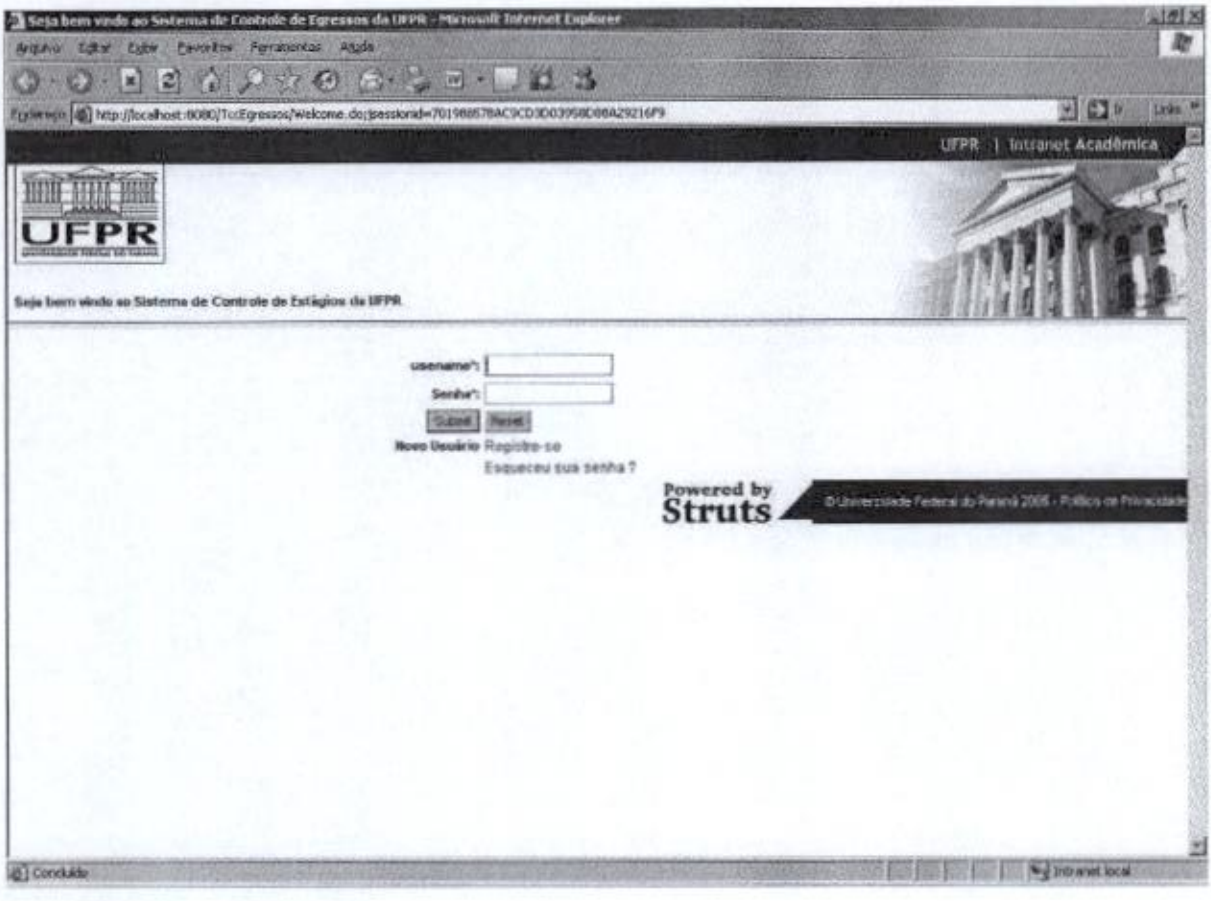
6.2.1.2.19 Fluxo Relatório de Seguro – Selecionando relatório



6.2.1.2.20 Fluxo Relatório de Seguro – Visualizando relatório

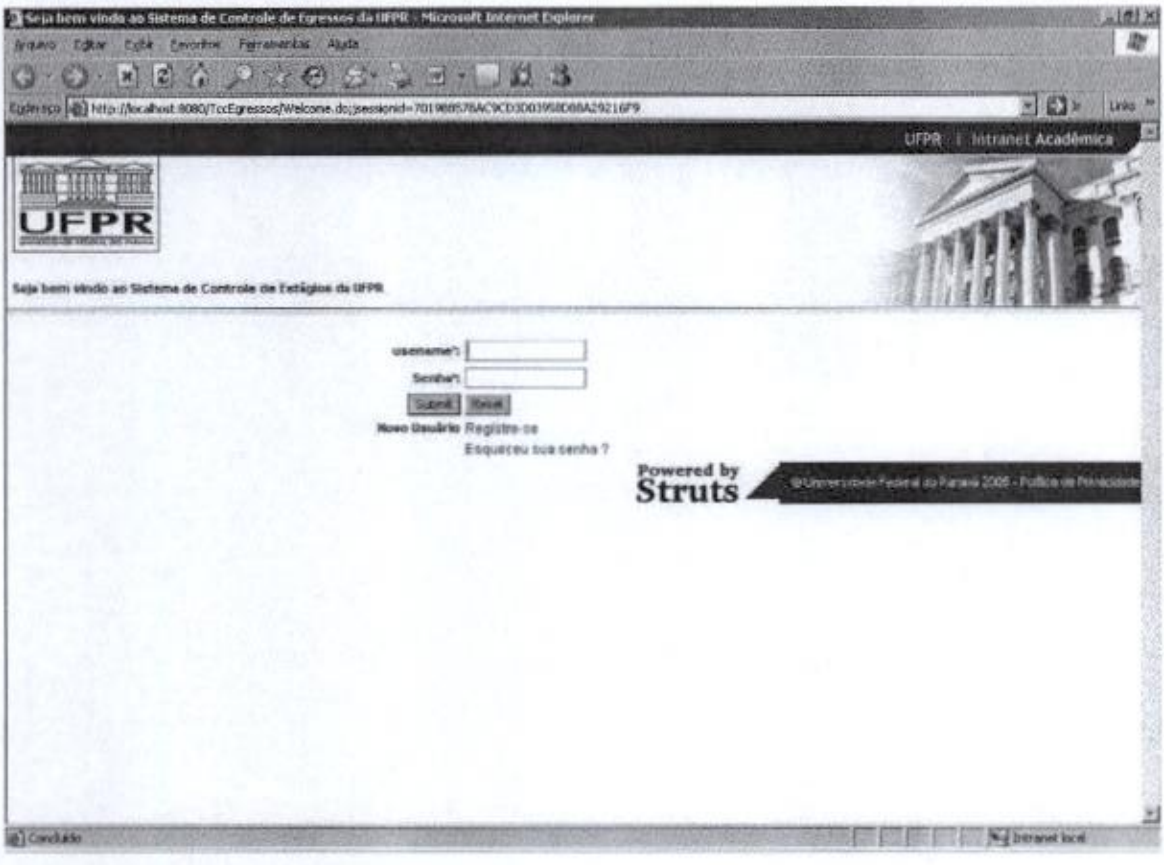


6.2.1.2.21 Sair

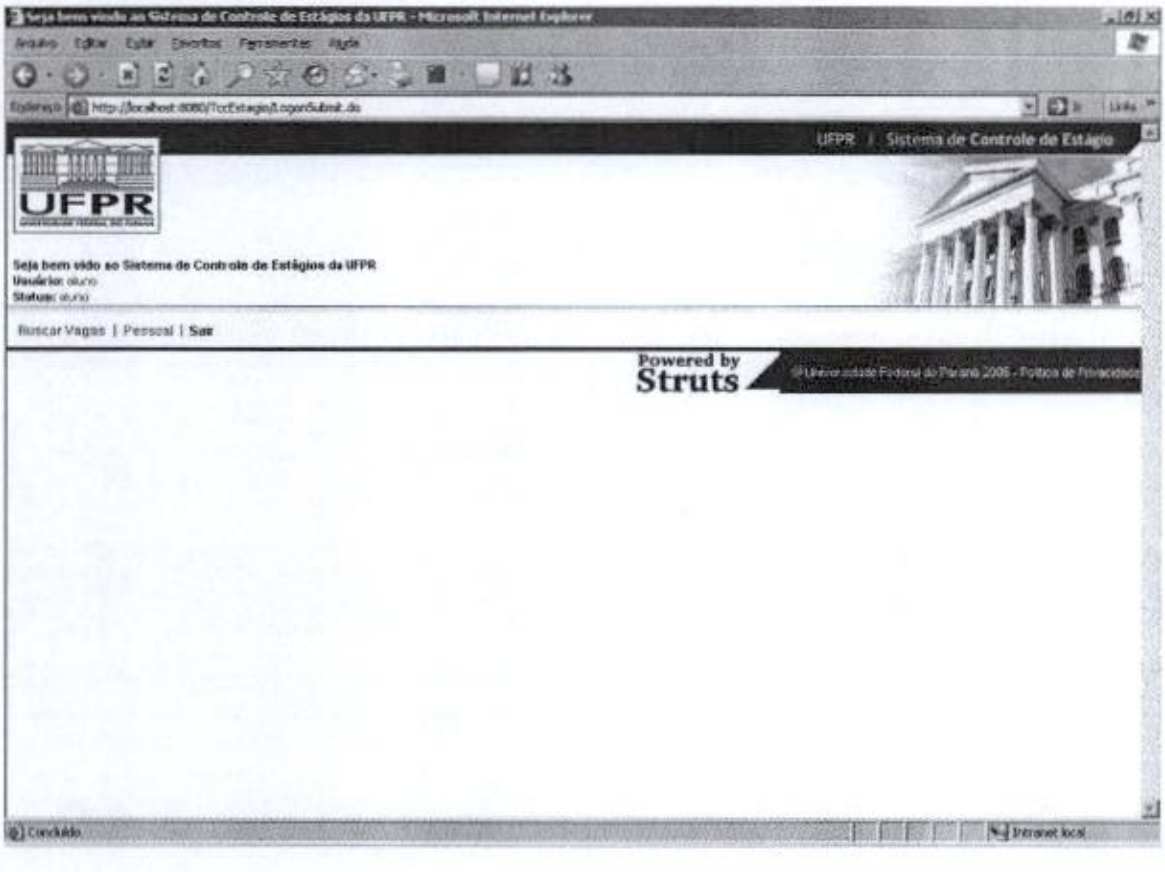

Seja bem vindo ao Sistema de Controle de Egressos da UFPR - Microsoft Internet Explorer
UFPR | Intranet Acadêmica
UFPR
Seja bem vindo ao Sistema de Controle de Estágios da UFPR
usename*:
Senha*:
Submit
Reset
Novo Usuário Registre-se
Esqueceu sua senha ?
Powered by
Struts
Concluído
Intranet local

6. 2.1.3 Visão Aluno

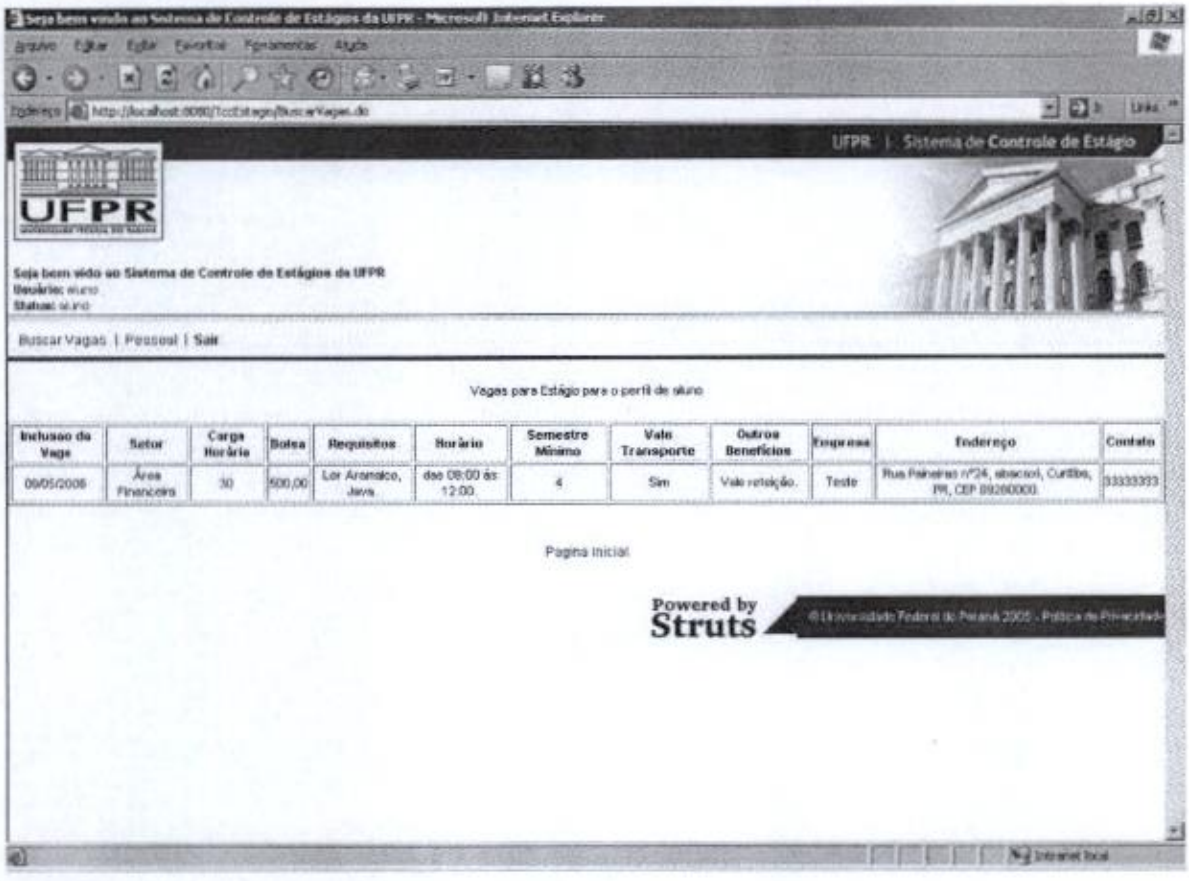
6. 2.1.3.1 Login



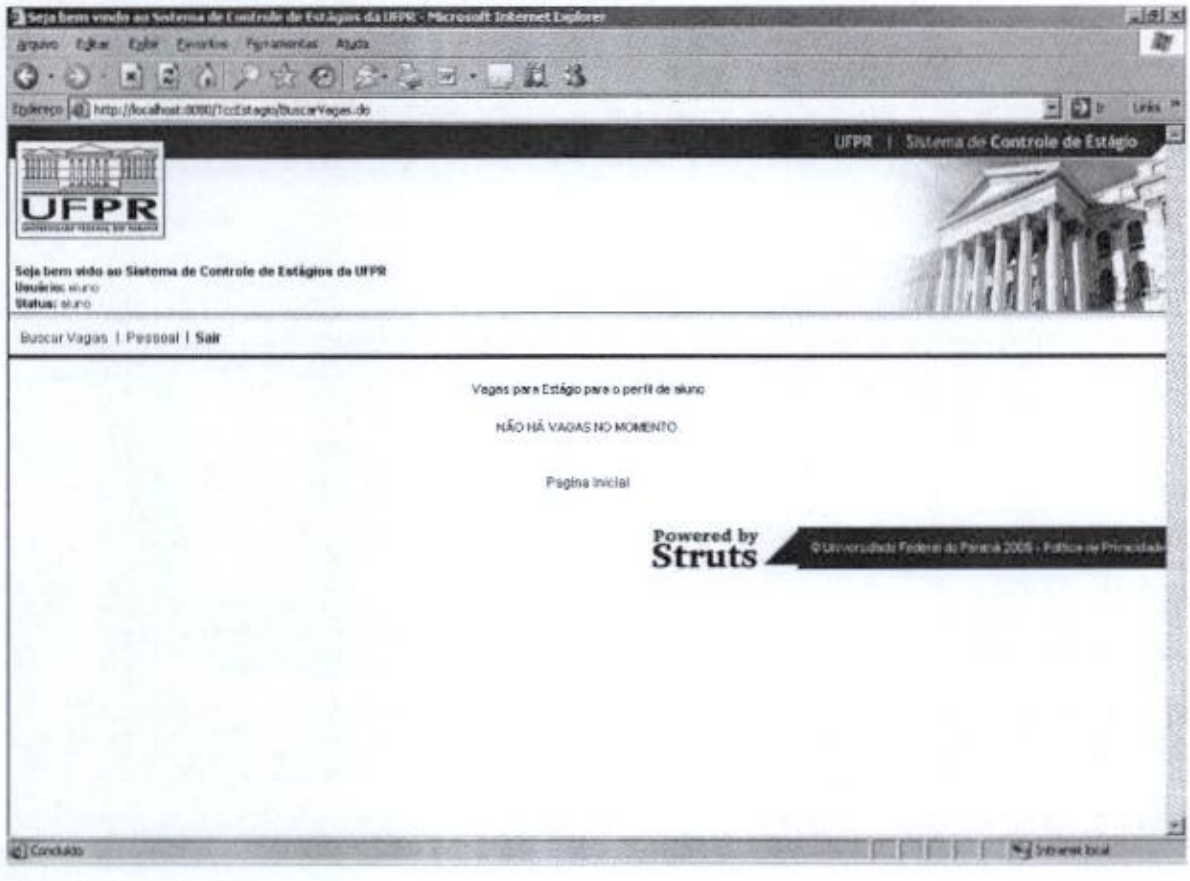
6. 2.1.3.2 Tela Inicial



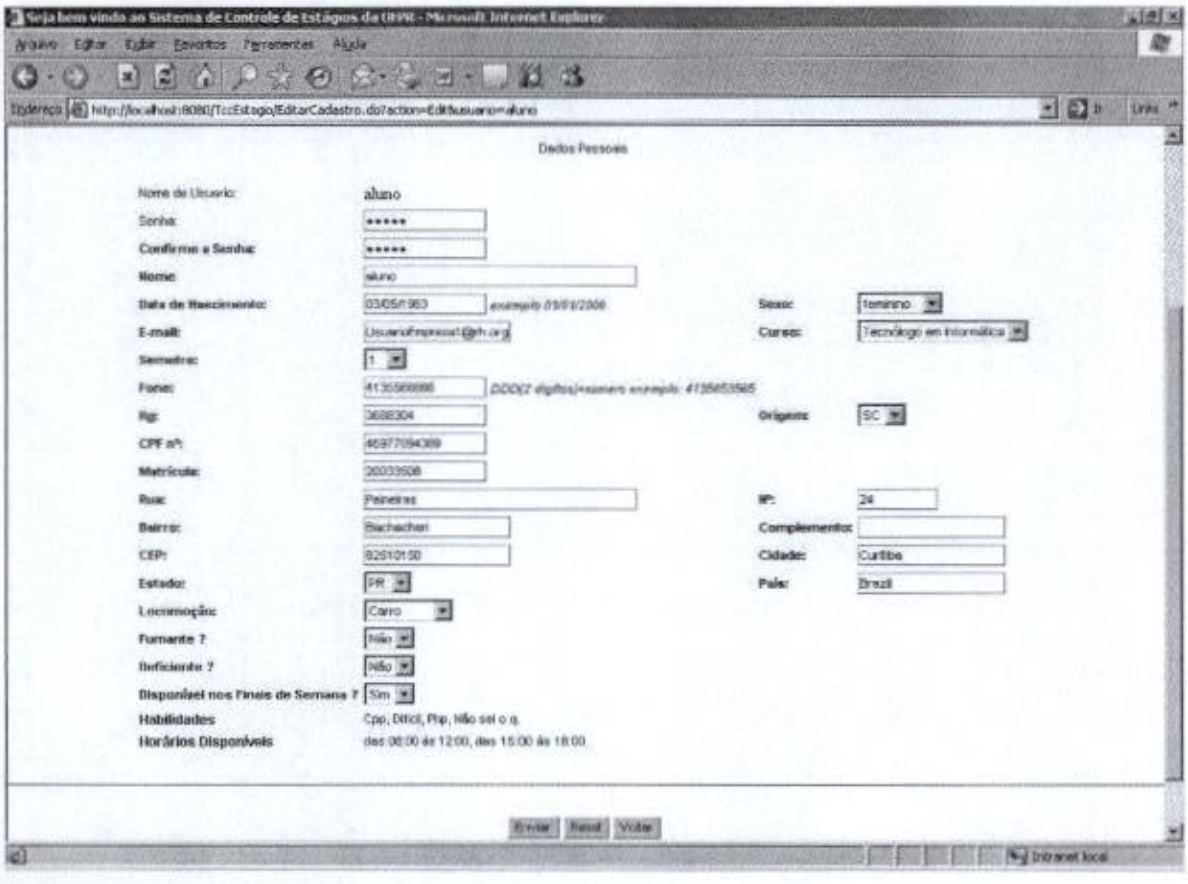
6. 2.1.3.3 Fluxo Buscar Vagas – Visualizando Vagas Disponíveis



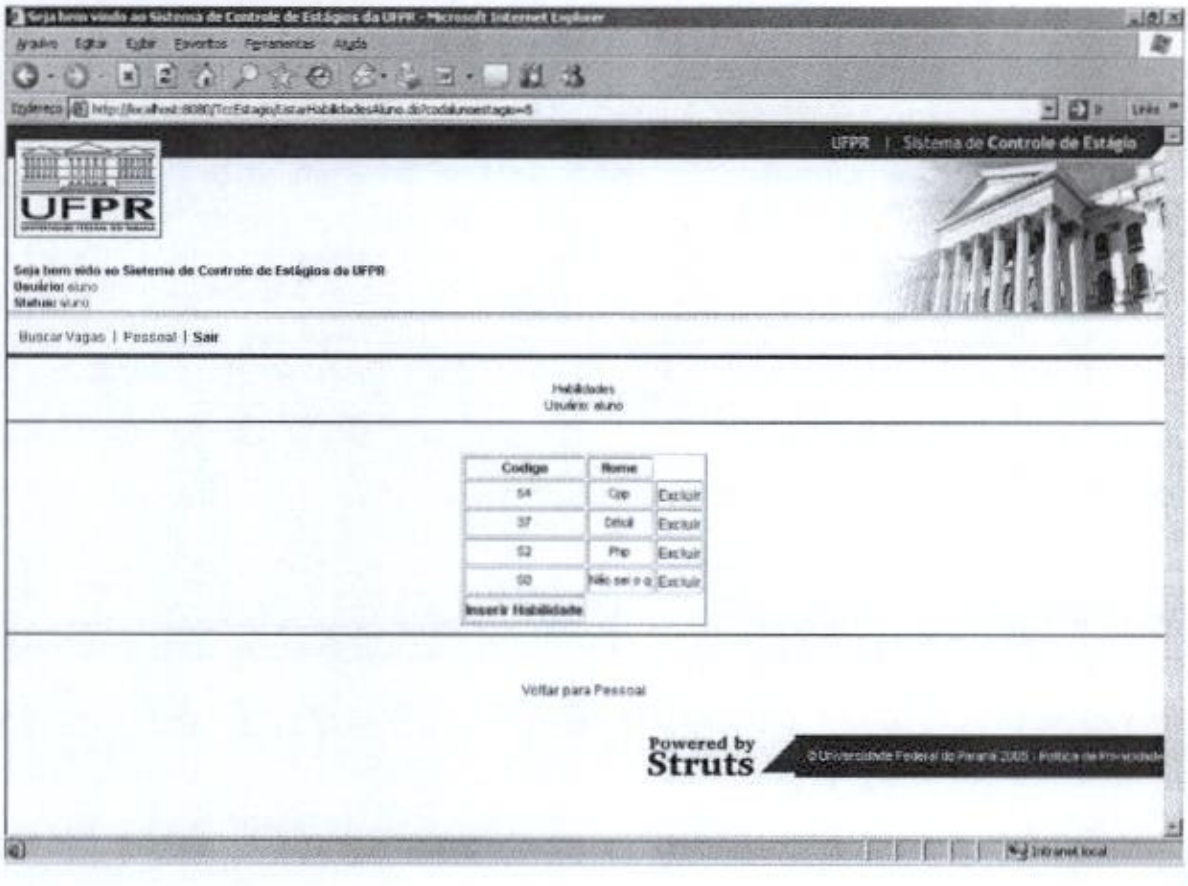
6. 2.1.3.4 Fluxo Buscar Vagas – Não há Vagas



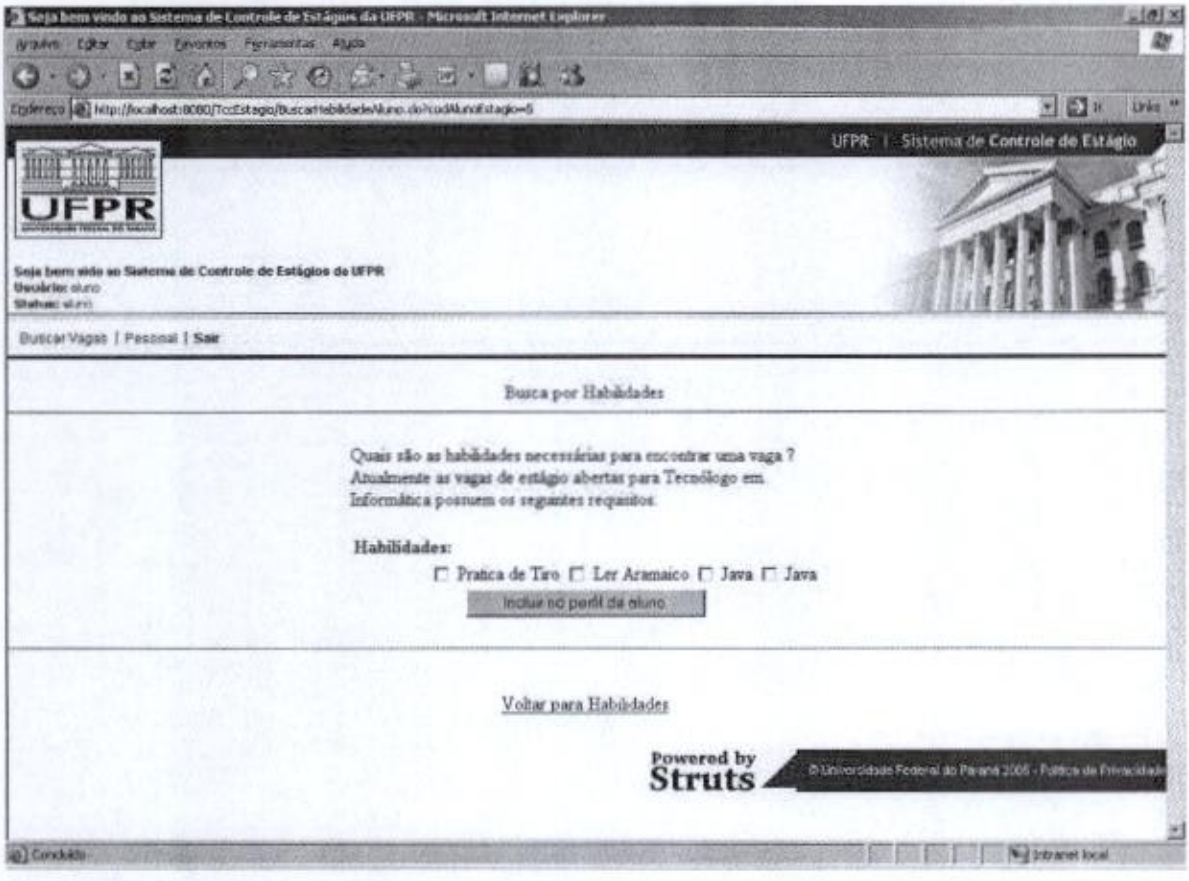
6. 2.1.3.5 Pessoal – Editar



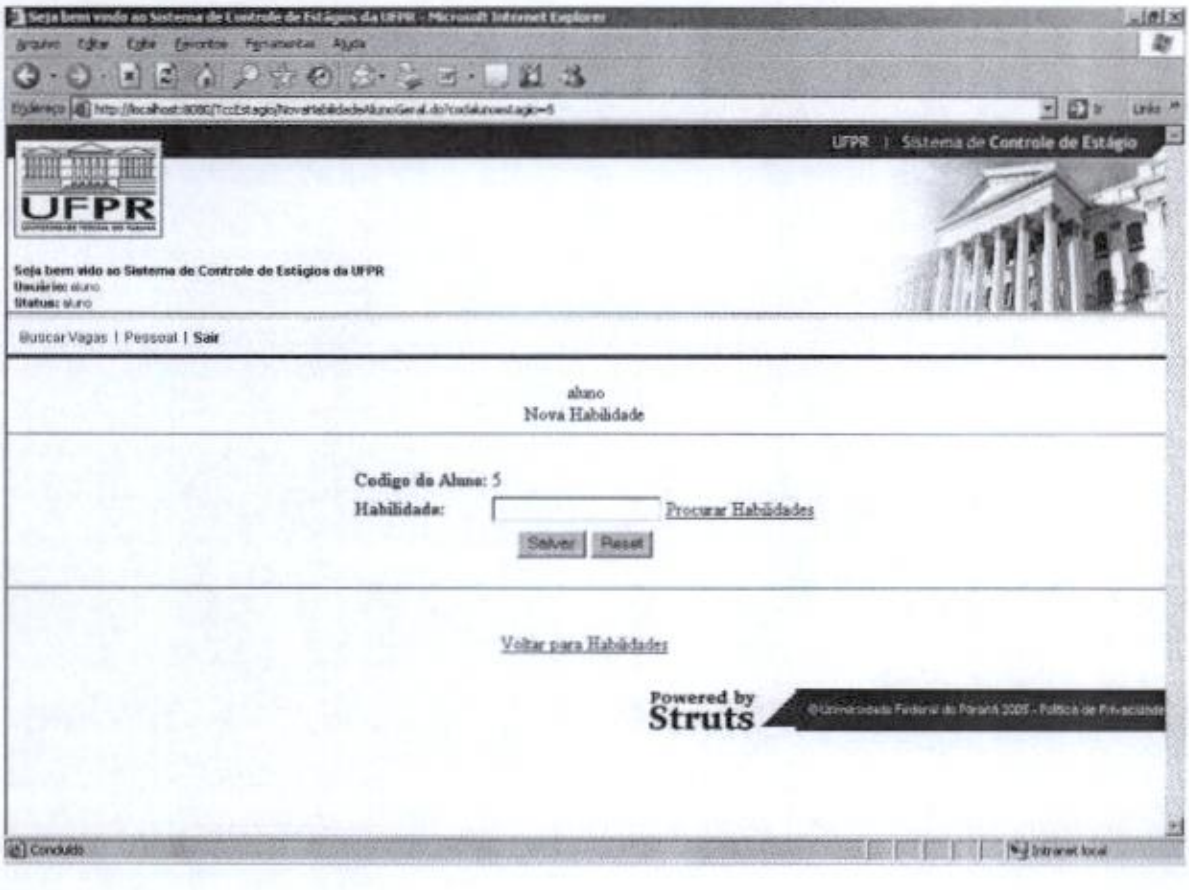
6. 2.1.3.6 Fluxo Pessoal – Visualizar Habilidades



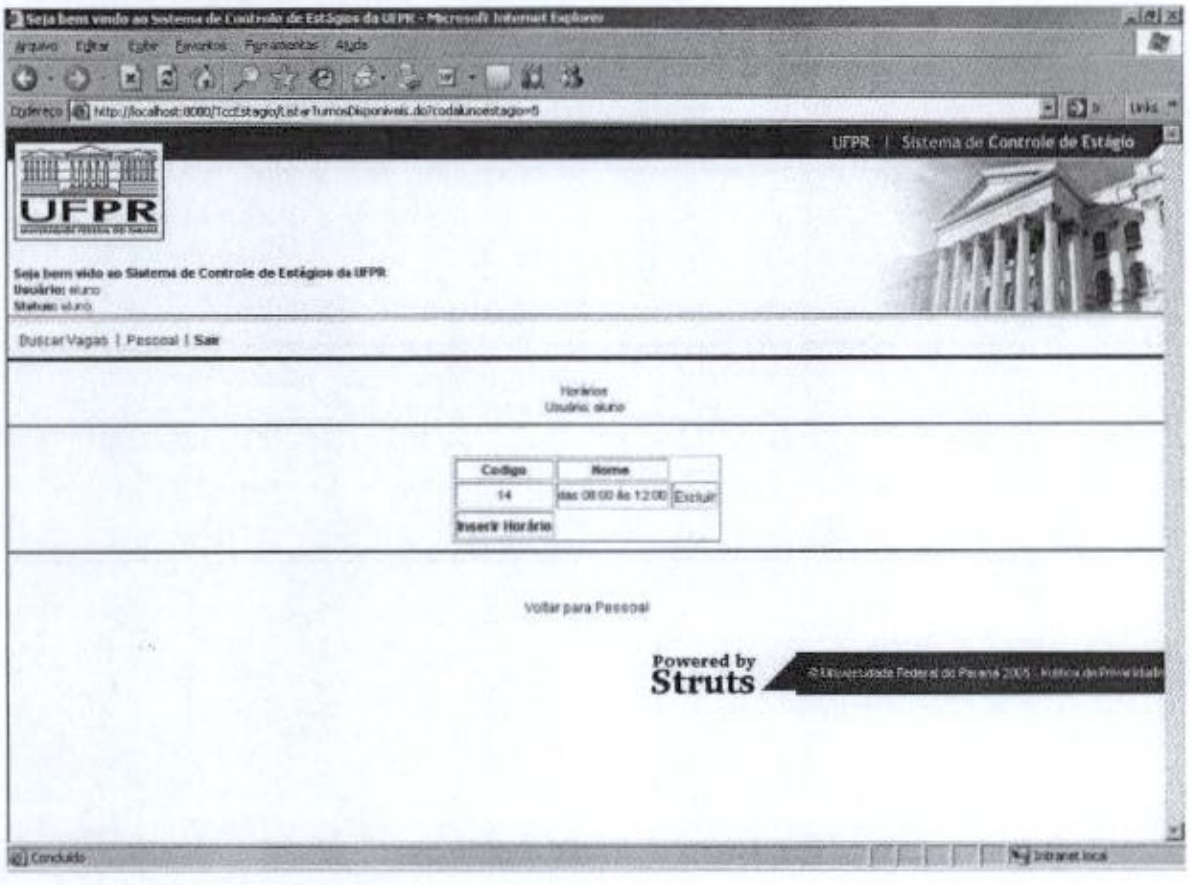
6. 2.1.3.7 Fluxo Pessoal – Procurar Habilidades



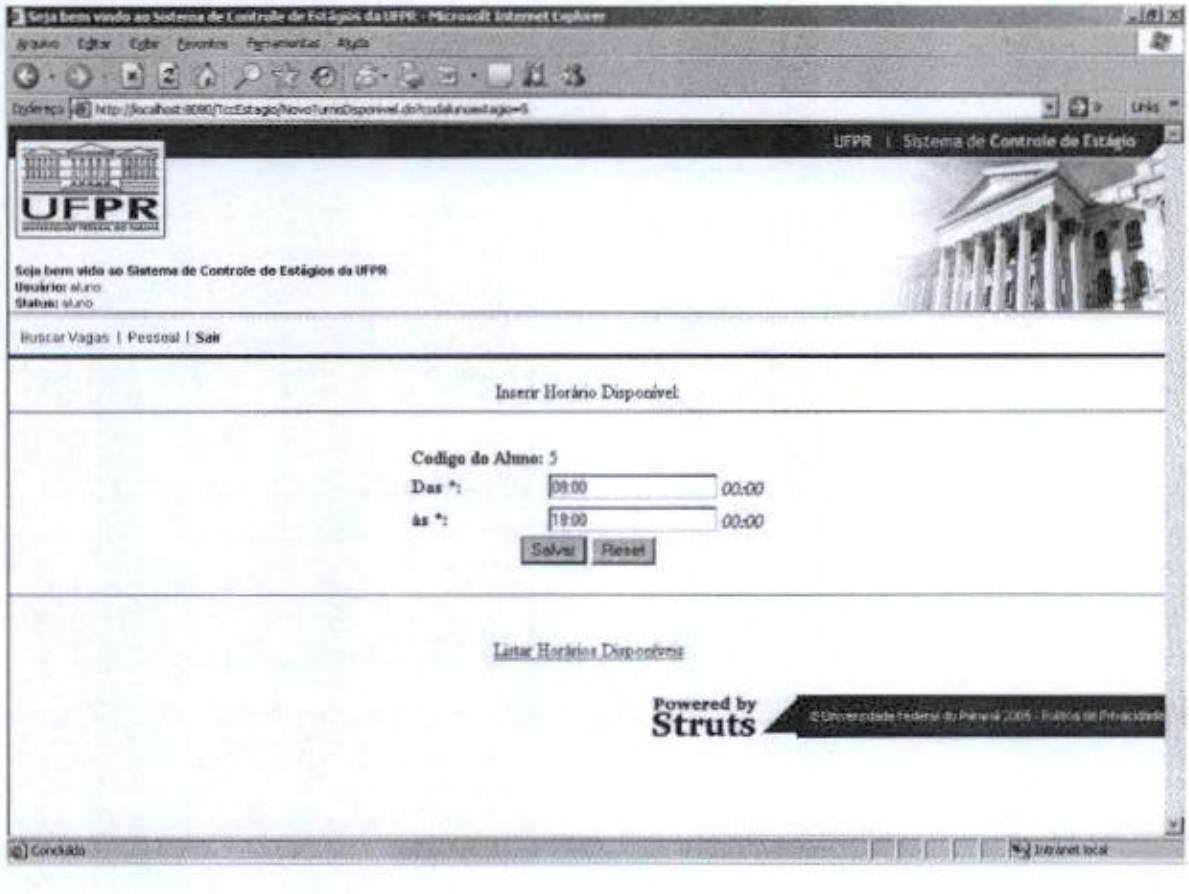
6. 2.1.3.8 Fluxo Pessoal – Inserir Habilidades



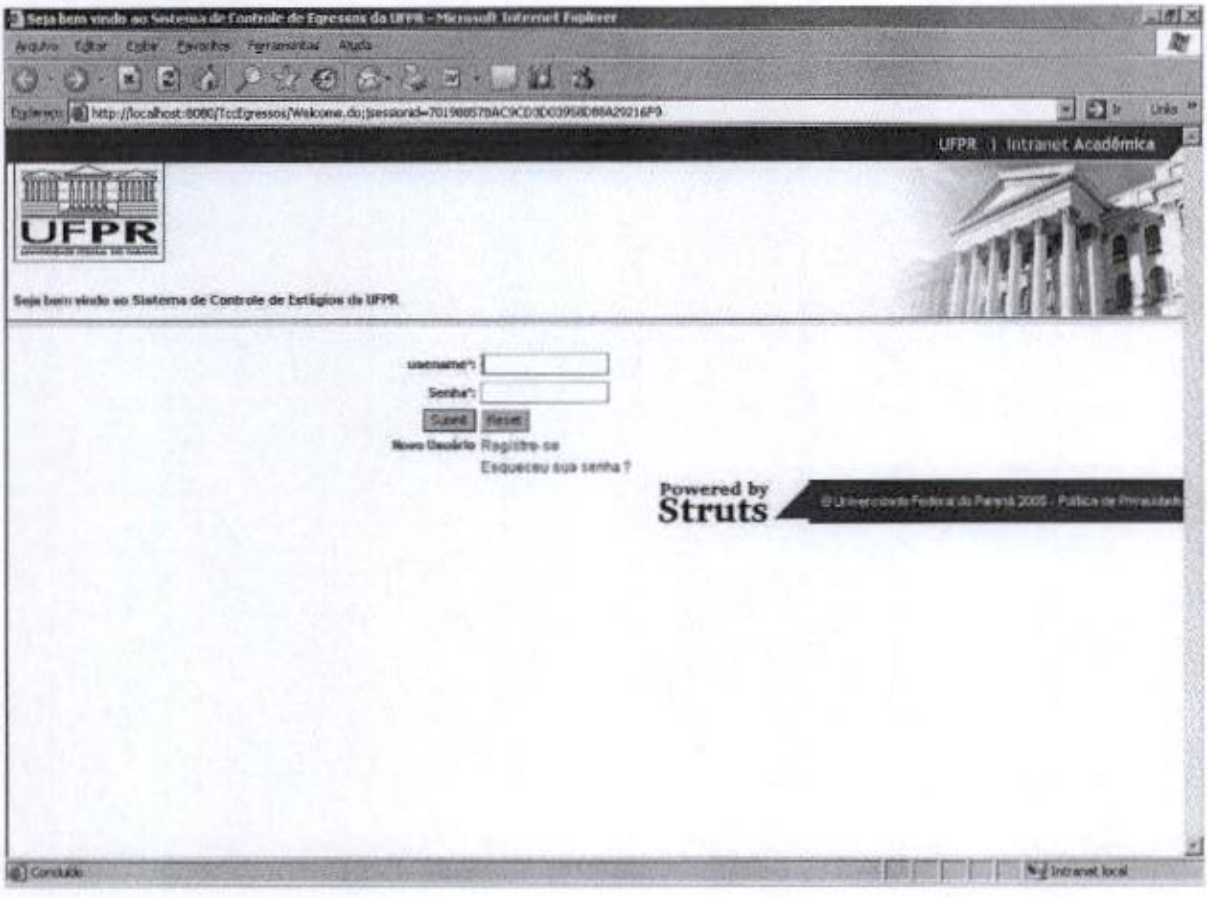
6. 2.1.3.9 Fluxo Pessoal – Visualizar Horários



6. 2.1.3.10 Fluxo Pessoal – Inserir Horários

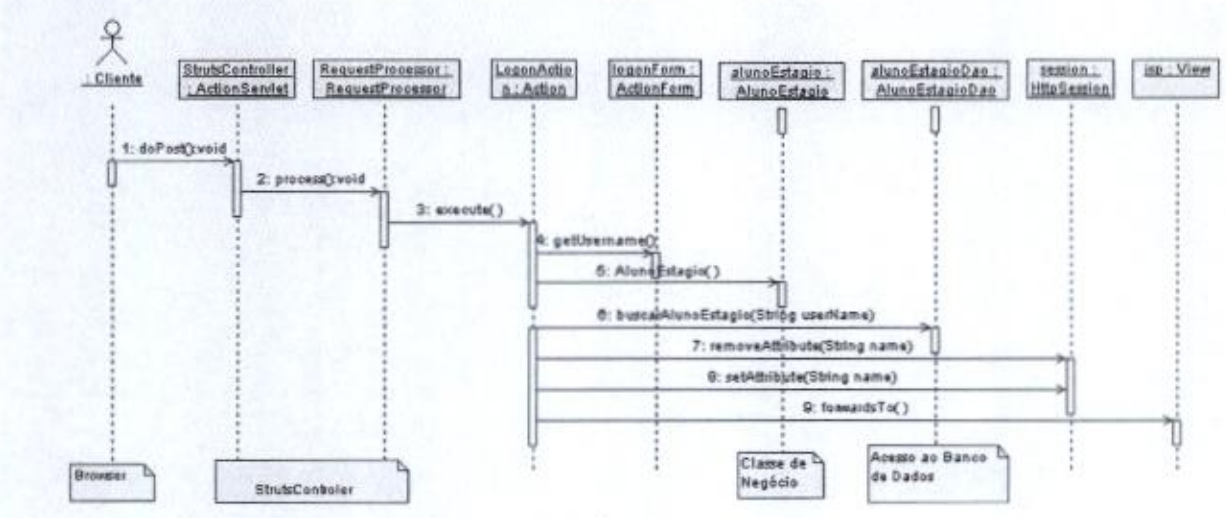


6. 2.1.3.11 Sair

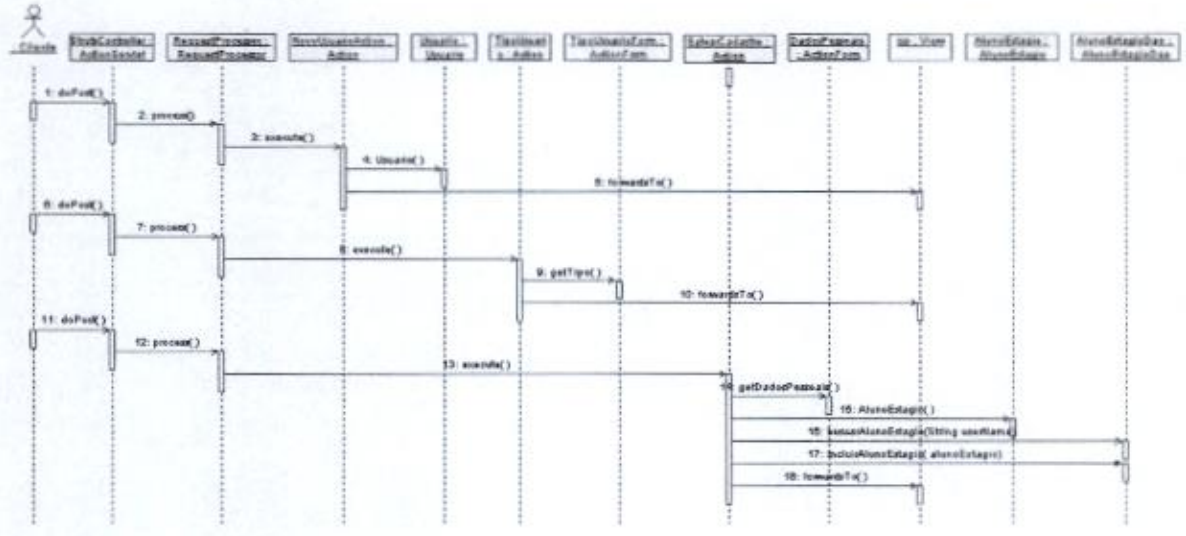

UFPR | Intranet Acadêmica
UFPR
Seja bem vindo ao Sistema de Controle de Estágios da UFPR
Submit
Reset
Registre-se
Esqueceu sua senha ?
Powered by
Struts

6.2.2 Diagramas de Seqüência

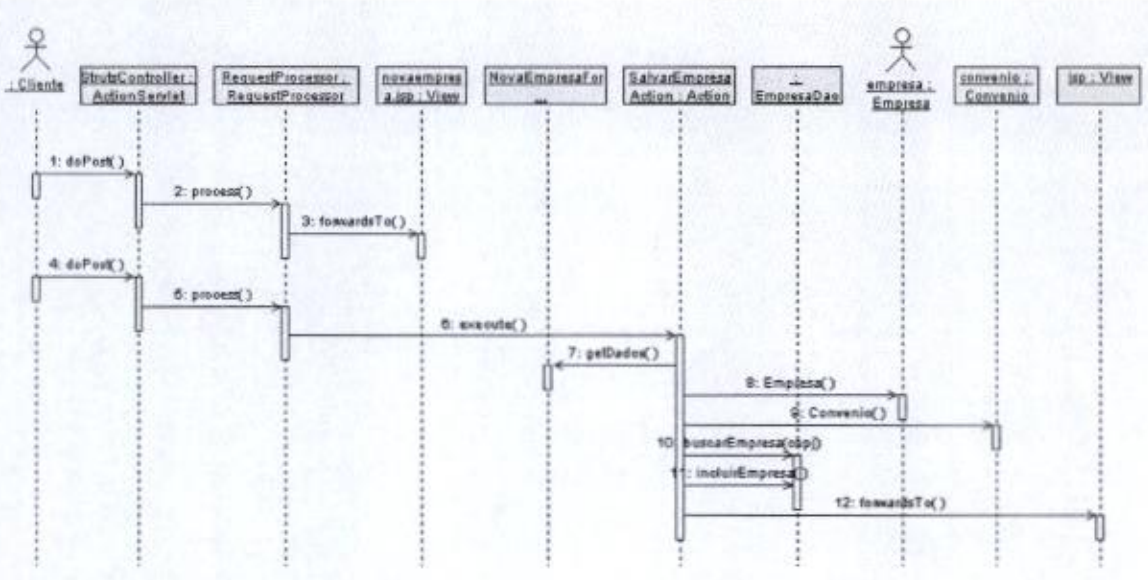
6.2.2.1Login



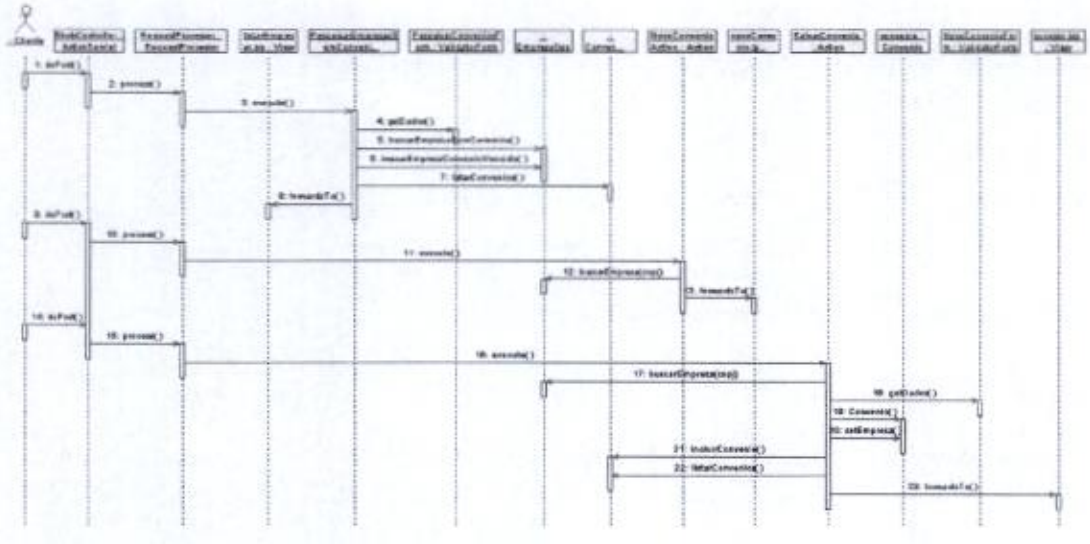
6.2.2.2 Inserir Usuário



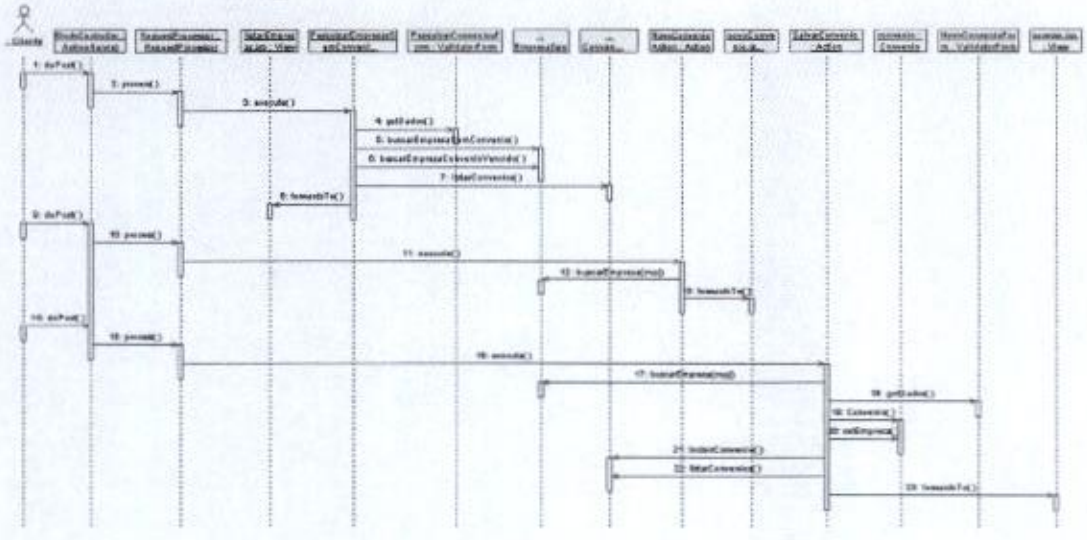
6.2.2.3 Nova Empresa



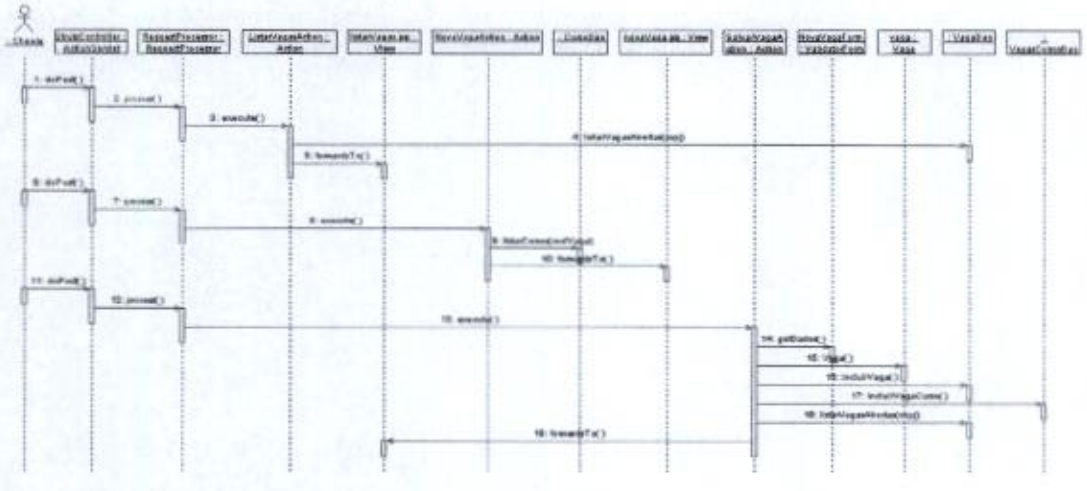
6.2.2.4 Novo Convênio



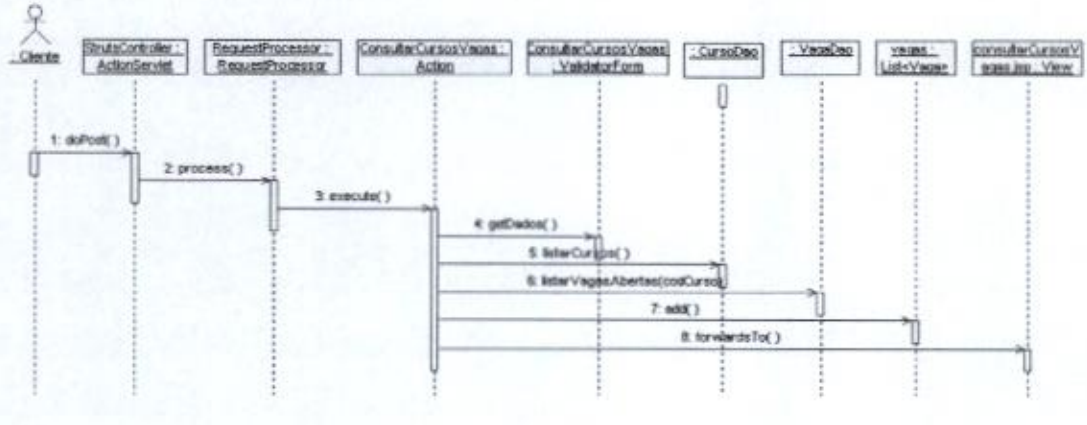
6.2.2.5 Criar termo de Compromisso



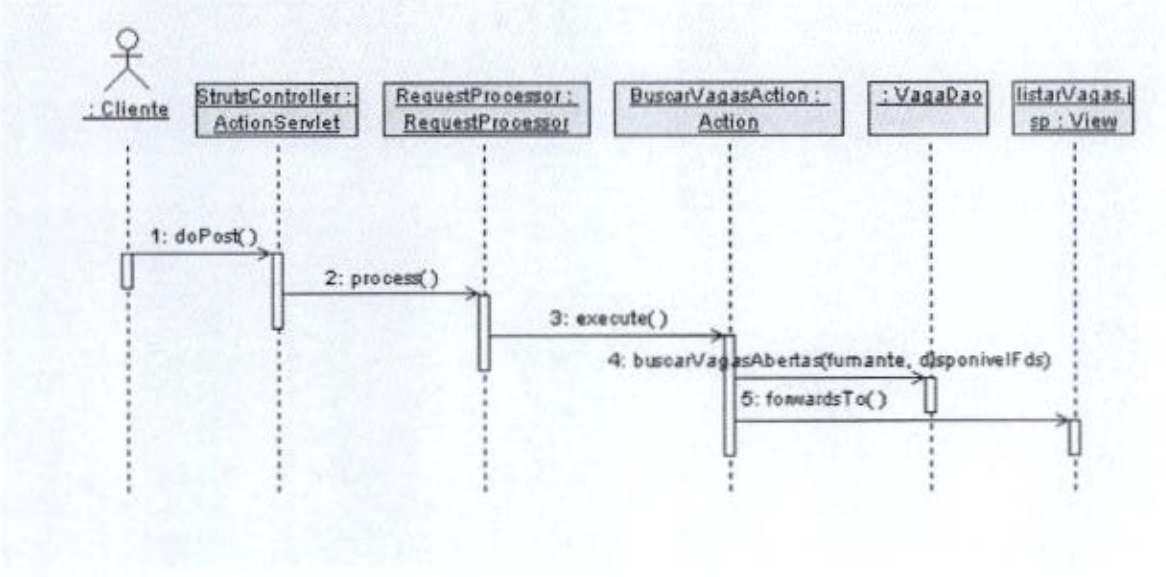
6.2.2.6 Ofertar Vaga



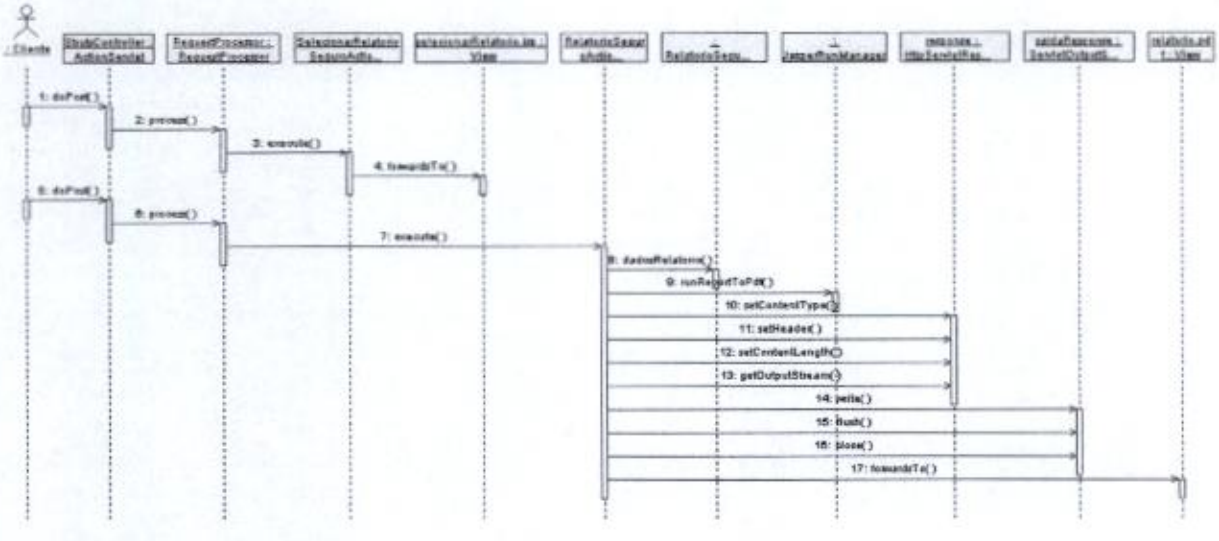
6.2.2.7 Consultar Cursos Vagas



6.2.2.8 Buscar Vagas

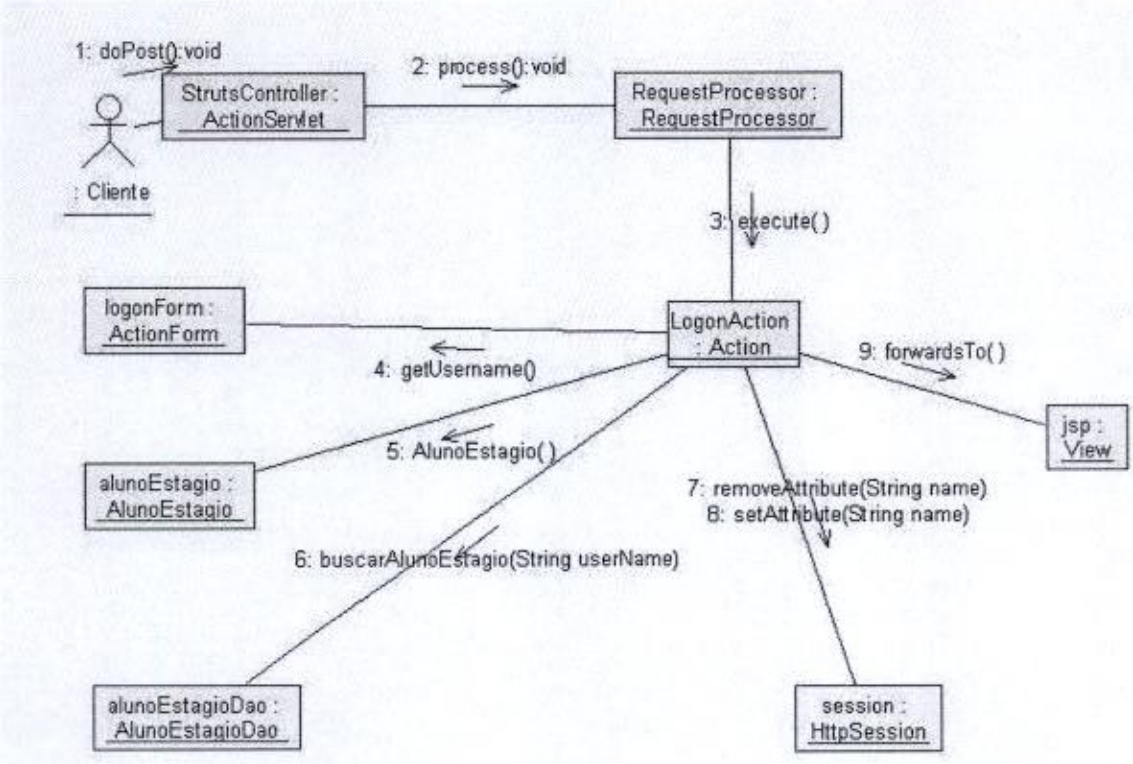


6.2.2.9 Relatório de Seguro

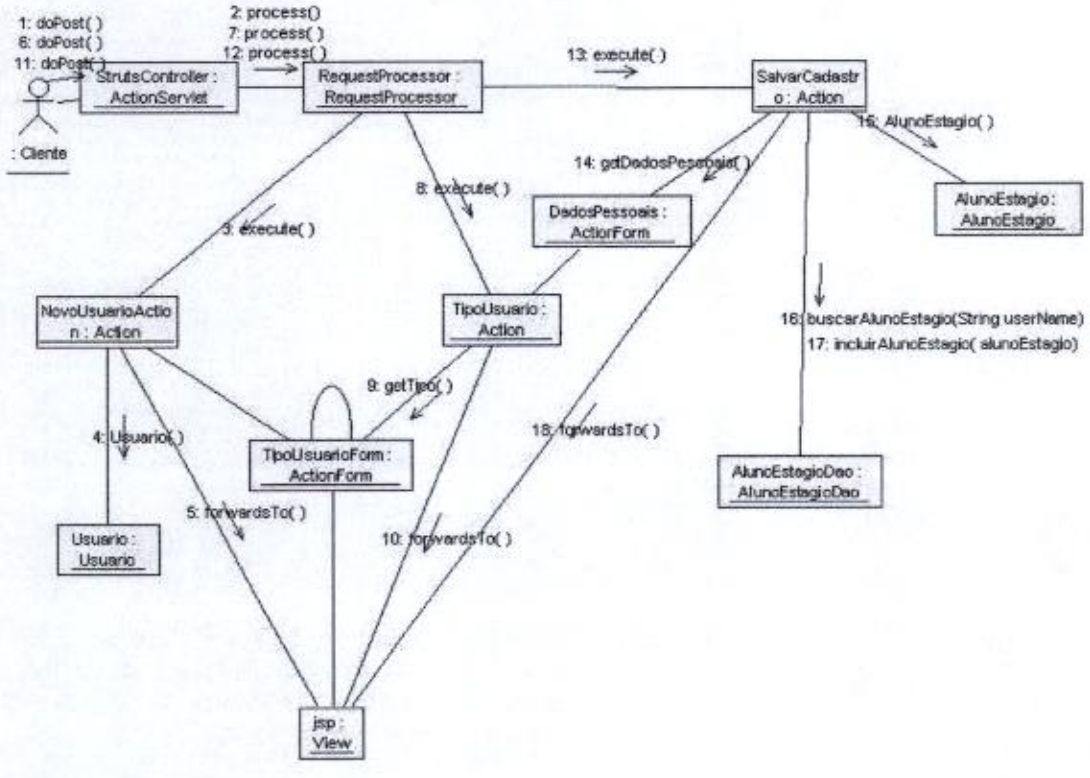


6.2.3 Diagramas de Colaboração

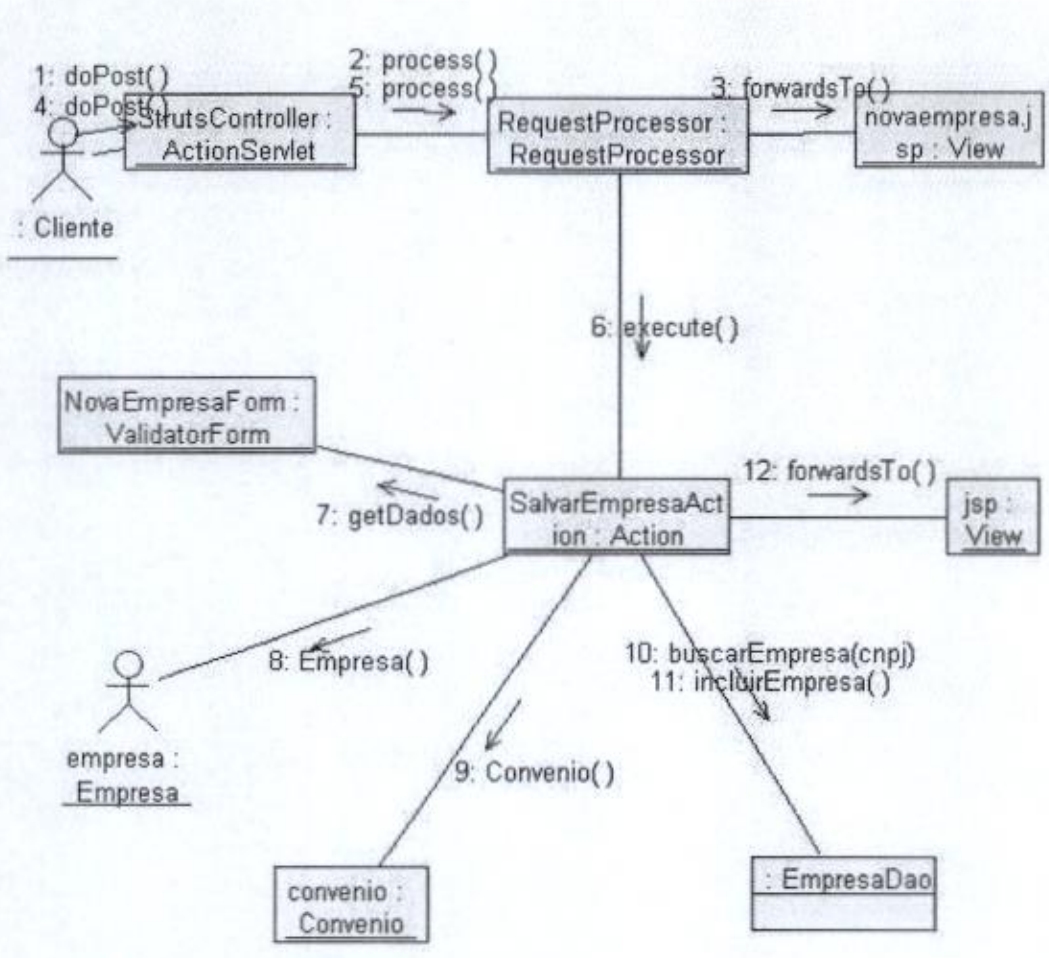
6.2.3.1 Login



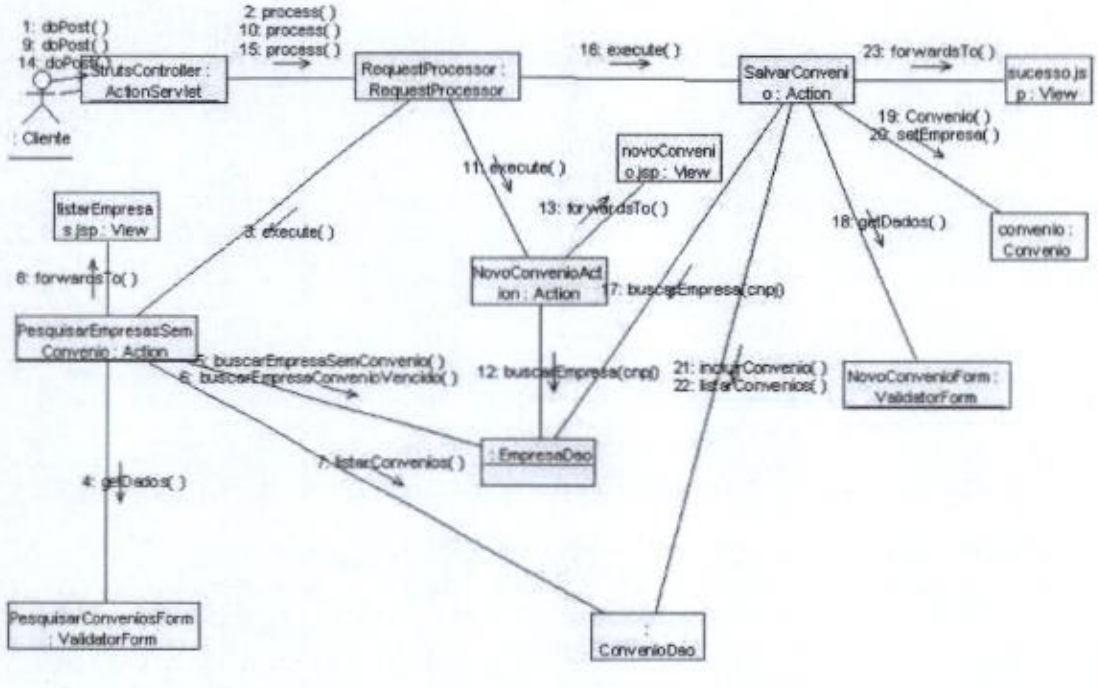
6.2.3.2 Inserir Usuário



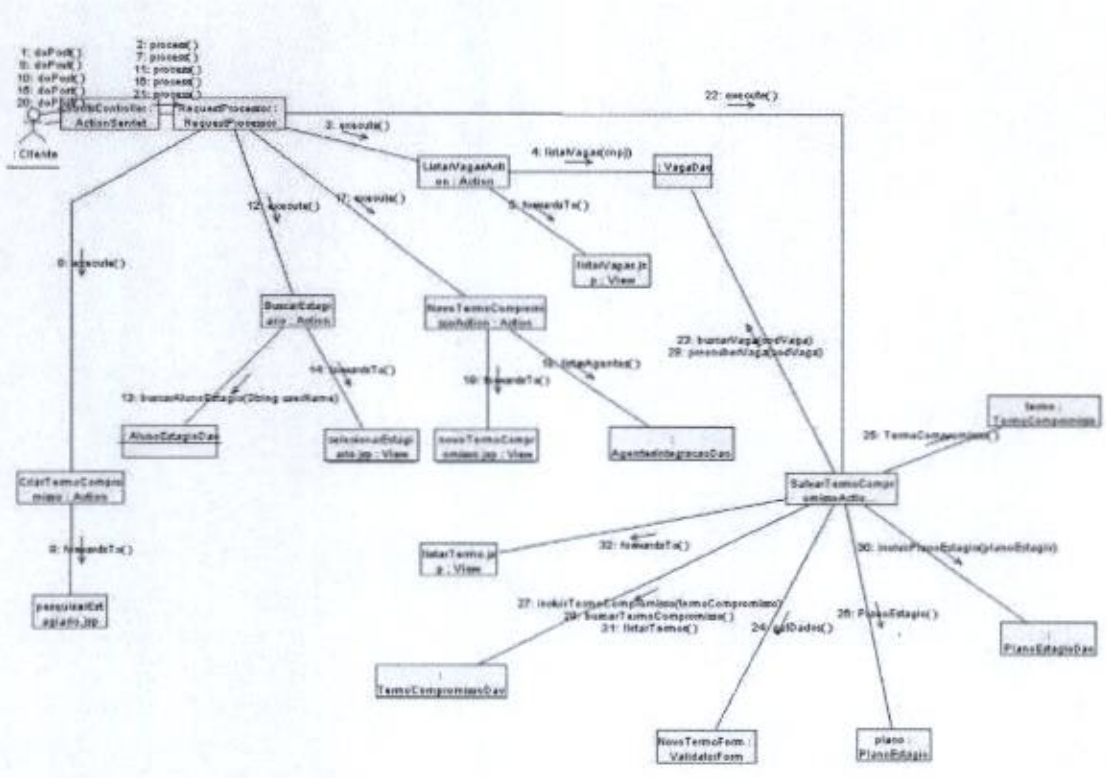
6.2.3.3 Nova Empresa



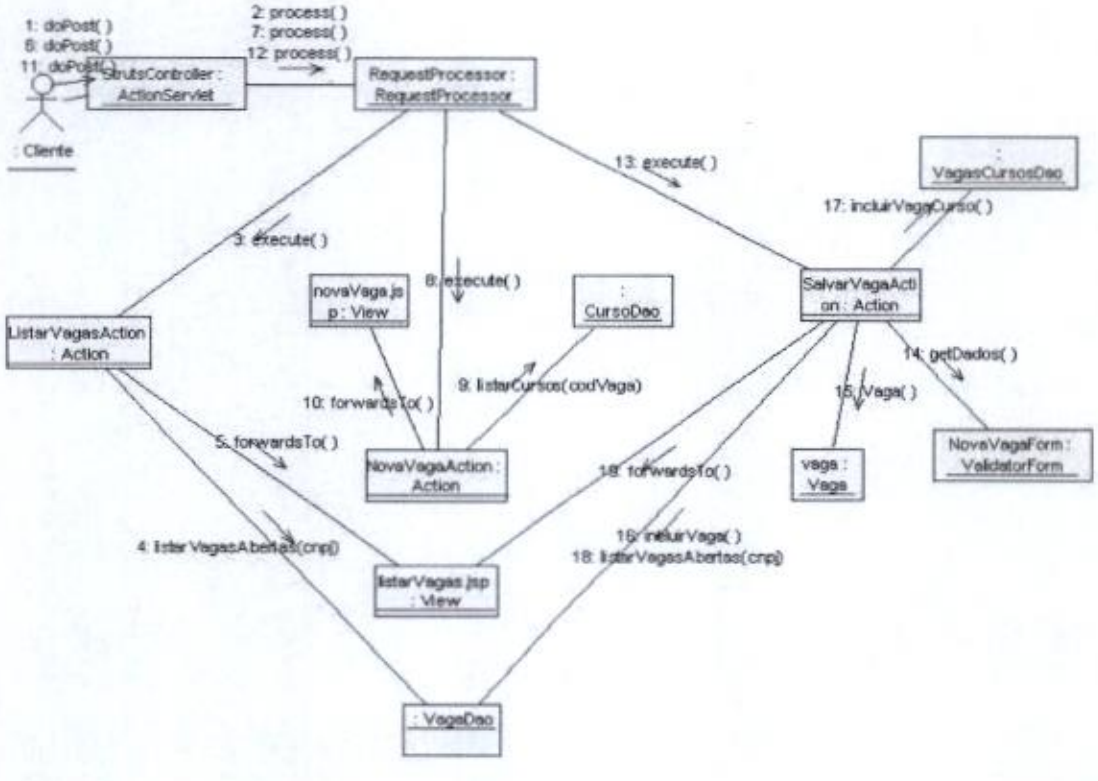
6.2.3.4 Novo Convênio



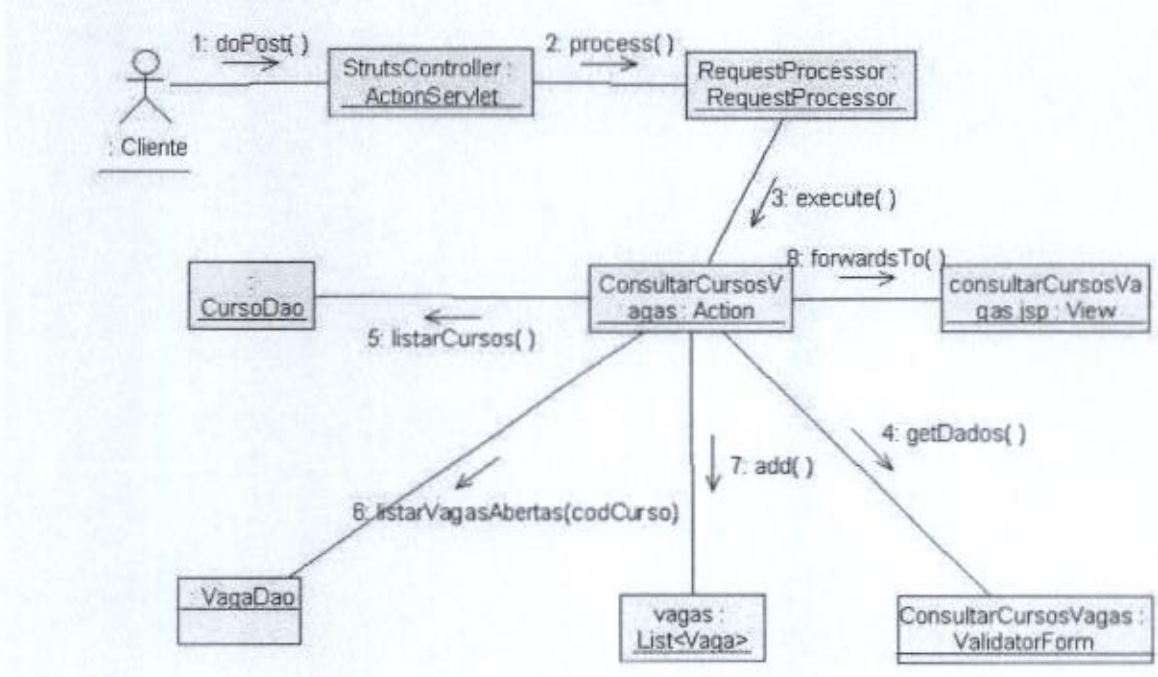
6.2.3.5 Criar Termo de Compromisso



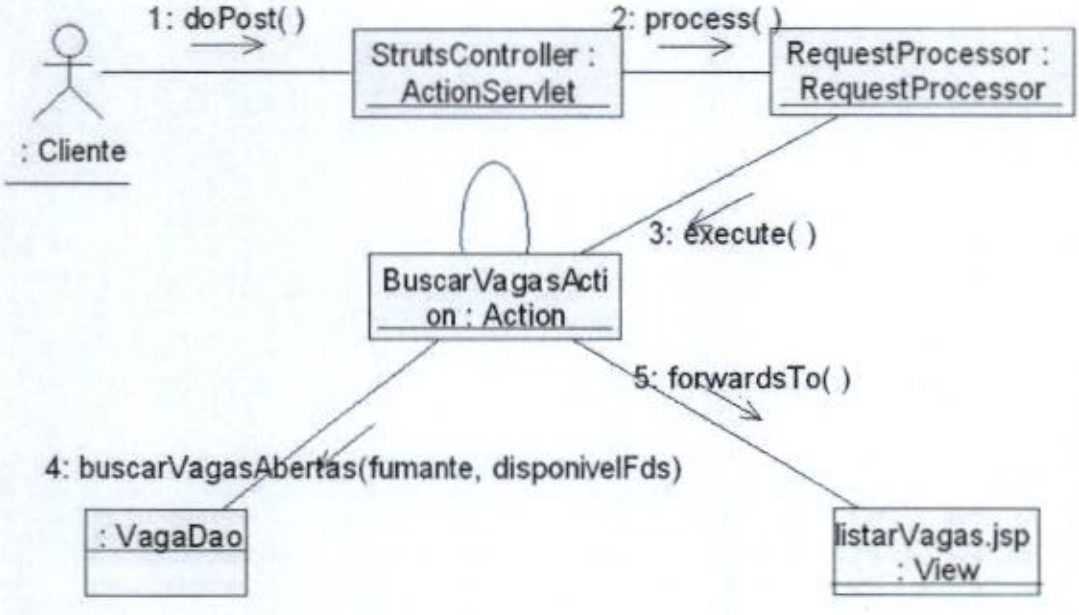
6.2.3.6 Ofertar Vaga



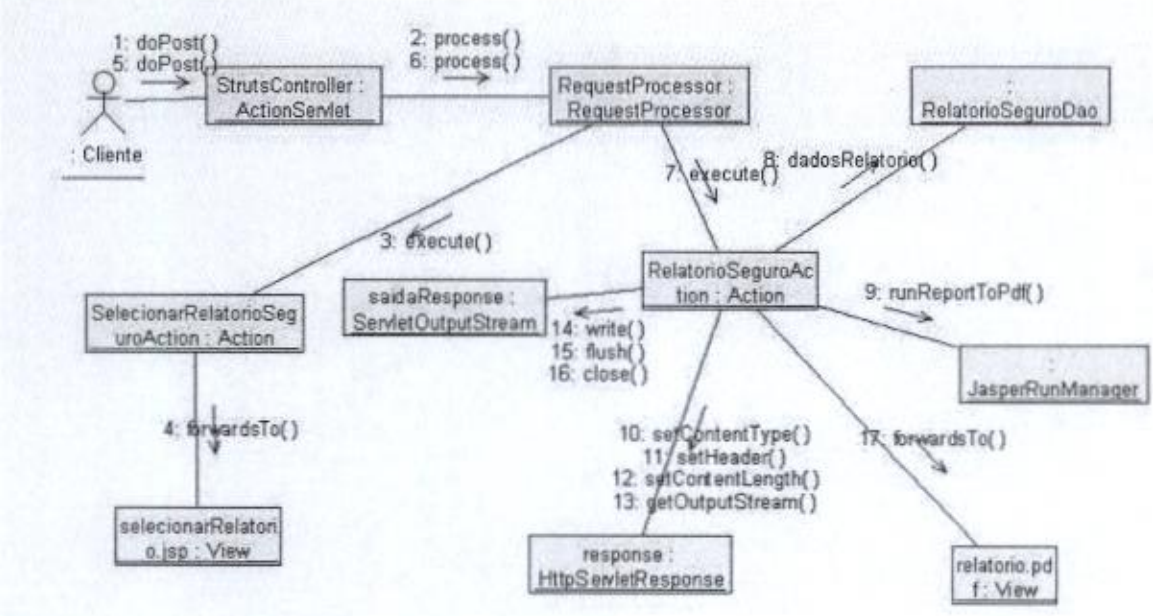
6.2.3.7 Consultar Cursos Vagas



6.2.3.8 Buscar Vagas



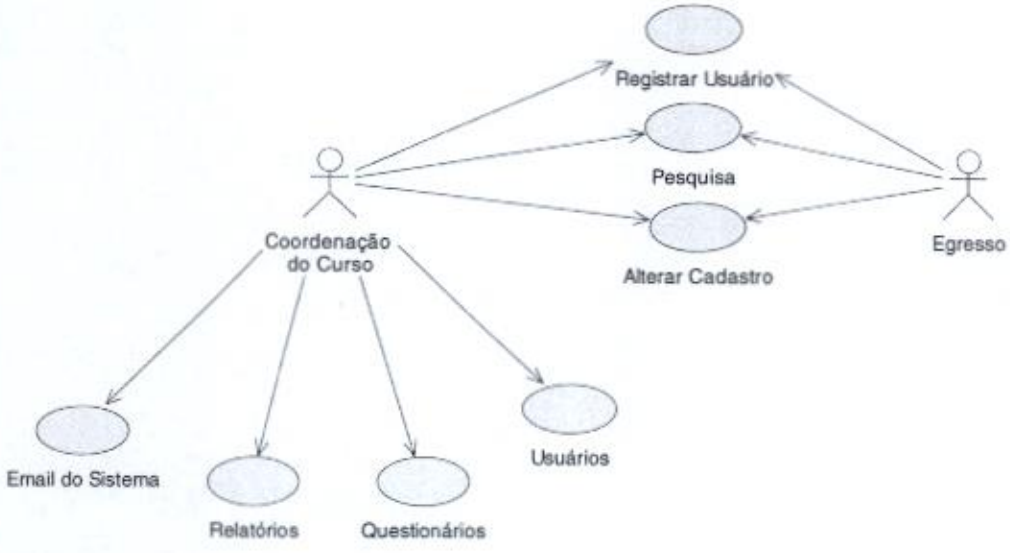
6.2.3.9 Relatório de Seguro



6.2.4 MODULO EGRESSOS

6.2.4.1 Descrição dos Casos de Uso – Módulo Egressos

6.2.4.1.1 Visão Coordenação do Curso



6.2.4.1.1.1 Módulo Egressos – Registrar Usuário

Caso de uso: Registrar Usuário.

Ator: Coordenação do Curso.

Descrição Resumida:

O funcionário da Coordenação do Curso poderá realizar inclusão de usuários, tanto egressos como funcionários da Coordenação.

- Fluxo de Eventos:

Caso de Uso: Registrar Usuário.

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do curso de Gestão da Informação da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite

o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Registrar Usuário e preenche o formulário de cadastro. Caso o aluno já seja cadastrado, o sistema retorna a mensagem "CPF já cadastrado".

Para os campos de preenchimento obrigatório o sistema realiza validação e retorna mensagem de erro caso algum campo fique em branco. (E2)

Fluxos Alternativos:

A1: Cadastrar Funcionário da Coordenação:

Este fluxo será executado quando ao preencher o formulário seja marcada a opção "Membro do Grupo da Coordenação". Caso esta opção não seja marcada, o sistema cadastrará um egresso.

A2: Salvar:

Se os dados forem inseridos corretamente e validados pelo sistema, as informações serão salvas no banco de dados e o sistema retornará a mensagem "OPERAÇÃO REALIZADA COM SUCESSO!".

A3: Limpar:

Limpa os campos do formulário.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Se não houver nenhum campo preenchido o sistema retorna quais os campos que devem ser preenchidos.

Cenários

Cenário Principal

Carlos, funcionário da coordenação do curso de Gestão da Informação, informa seu login e senha e acessa o Módulo de Egressos na Intranet Acadêmica. Em seguida acessa a opção Registrar Usuário e preenche os campos do formulário.

Após preencher os campos, clica no botão salvar e os dados são inseridos com sucesso.

Cenários Secundários

- Após preencher o formulário e clicar no botão Salvar, o sistema retorna a mensagem que o CPF informado já está cadastrado.
- Carlos preenche o formulário deixando o campo Cidade em branco. O sistema retorna a mensagem indicando a necessidade de preenchimento deste campo. (Obs.: esta validação é realizada em todos os campos, exceto o complemento).

6.2.4.1.1.2 Módulo Egressos – Usuários

Caso de uso: Usuários.

Ator: Coordenação do Curso.

Descrição Resumida:

Este caso de uso permite à coordenação do curso buscar informações referentes aos usuários do sistema egressos.

Fluxo de Eventos:

Caso de Uso: Usuários

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do curso de Gestão da Informação da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção "Usuários", o sistema carrega uma tela com os campos: CPF e Nome, o funcionário poderá digitar o CPF ou parte do nome (E2), após informar os dados o funcionário deve clicar na opção pesquisar e o sistema carrega uma tela com os resultados da pesquisa (Nome, CPF) , ao clicar no CPF o funcionário poderá visualizar o cadastro e se necessário altera-lo(A1), pode ainda Excluir o cadastro (A2) ou ainda realizar nova busca (A3).

Fluxos Alternativos:

A1:Visualizar/Alterar Cadastro

Após a execução da pesquisa, o funcionário pode clicar no CPF e o sistema carrega a tela de cadastro, nesta tela o funcionário poderá voltar para o resultado ou alterar os dados no cadastro do usuário.

A2: Excluir

Após a execução da pesquisa o funcionário poderá excluir um cadastro existente no sistema.

A3: Realizar nova Busca

Na tela de resultado há a opção de Realizar nova busca, onde, ao clicar nesta opção o sistema retorna para a tela inicial de busca.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Caso o funcionário digite um CPF ou nome inválido, o sistema retorna mensagem de erro, informando que o campo esta preenchido incorretamente.

Cenários

Cenário Principal

Carlos, funcionário da coordenação do curso de Gestão da Informação, informa seu login e senha e acessa o Módulo de Egressos na Intranet Acadêmica. Em seguida acessa a opção Usuários, preenche o campo CPF e clica em pesquisar, o sistema retorna o resultado da pesquisa, neste caso um (1) usuário cadastrado.

Cenários Secundários

- Carlos digita no campo CPF parte de um nome, o sistema o informa que o valor informado no campo é inválido.
- Carlos digita um número no campo nome, o sistema o informa que o valor informado no campo é inválido.

6.2.4.1.1.3 Módulo Egressos – Questionários

Caso de uso: Questionários.

Ator: Coordenação do Curso.

Descrição Resumida:

Este caso de uso permite à coordenação do curso de Gestão da informação elaborar questionários referentes ao curso de acordo com seus interesses, afim de buscar informações com os egressos a respeito de como foi o curso por exemplo, etc.

- Fluxo de Eventos:

Caso de Uso: Questionários

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do curso.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Questionários, onde ele poderá elaborar questões pertinentes aos interesses da Coordenação a fim de realizar avaliação da parte do egresso para o curso concluído.

Fluxos Alternativos:

A1: Novo

Esta opção possibilita ao usuário a criação de um novo questionário no sistema. Após clicar nesta opção, o usuário poderá inserir as questões no questionário criado.

A.1.1: Inserir Questão

Possibilita ao usuário a criação de perguntas com suas respectivas alternativas de respostas. A questão também poderá ser inclusa no formato de questão discursiva, para isso é preciso que o usuário marque a *check box* Questão Aberta.

Após isso clica em salvar e insere a questão, que será salva com um código automático seqüencial e inserida no questionário criado, que também recebe um código automático. (E3)

Para incluir mais questões no mesmo questionário basta clicar em Inserir Questão.

A.1.2: Editar Questão

Possibilita ao usuário modificar uma questão posteriormente elaborada, bastando clicar em cima do número da questão que ela abrirá para edição.

A.1.3 Aumentar Ordem

Com esta opção o usuário pode mudar a ordem de prioridade que ele deseja que as questões apareçam no questionário. Por exemplo, se ele quiser que a questão 2 apareça antes da questão 1.

A.1.4 Excluir Questão

Exclui a questão do questionário corrente.

A2: Questionários

Ao clicar nesta opção o usuário retorna à tela onde estão listados todos os questionários e pode edita-los clicando em seu código.

A.2.1: Ativar Questionário

Refere-se à opção "Ativo" na tela "Elaborar Questionários". Ao clicar nesta opção, o questionário referente ficará ativo para que o egresso responda.

A.2.2 Excluir Questionário

Exclui o questionário escolhido e consequentemente todas as suas questões.

A3: Estatísticas

Esta opção retorna as estatísticas e estimativas a respeito das questões do questionário corrente, mostrando a porcentagem para cada tipo de respostas obtidas. Possibilita visualizar cada questão com suas respectivas porcentagens separadamente, caso o usuário clique no nome da questão. (E3)

A4: Questões

Ao clicar nesta opção, no rodapé da página, o usuário é redirecionado para a tela do questionário corrente, podendo assim visualizar todas as questões.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Caso o usuário não digite nada no campo "Pergunta" o sistema retornará erro atentando para o preenchimento deste campo. E caso ele digite uma pergunta e não preencha as alternativas, automaticamente será gerada uma questão aberta.

E3: Caso o questionário não tenha sido respondido o sistema retornará as Estatísticas vazias.

Cenários

Cenário Principal

Carlos, funcionário da coordenação do curso de Gestão da Informação, acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha, e irá elaborar um questionário a ser respondido para os egressos.

Como deseja inserir um questionário novo, utilizará a opção "Novo" e depois em "Inserir Questão" começará a elaborar as perguntas referentes ao questionário.

A primeira pergunta é digitada e em seguida suas alternativas. Ele salva e parte para inserir mais 10 questões.

A questão de número 5 precisa ter um campo aberto para as respostas dos alunos, então ele marca a opção "Questão Aberta" no formulário.

Terminando de elaborar o questionário, ele clica na opção "Questionários" logo abaixo de onde aparecem as questões e visualiza todos os questionários que possui, deixando ativo

este mesmo que acabou de elaborar. Clica em OK e seu questionário está pronto para que os egressos respondam.

Cenários Secundários

- Ao cadastrar as perguntas de seu questionário, Carlos esquece das alternativas e a questão é cadastrada como aberta, porém não poderia ser uma questão aberta. Então, na tela de "Elaborar Questionários" ele clica no código do questionário, e abrirá o formulário com todas as questões, ele então clica no número da questão para poder editá-la e coloca as alternativas, modificando corretamente a questão.

6.2.4.1.1.4 Módulo Egressos – Relatórios

Caso de uso: Relatórios.

Ator: Coordenação do Curso.

Descrição Resumida:

Este caso de uso permite à coordenação do curso obter relatórios sobre os alunos egressos.

Fluxo de Eventos:

Caso de Uso: Relatórios

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do curso.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, o funcionário deverá escolher a opção "Relatórios", que lhe exibirá duas opções "Listar Egressos em ordem de ano de conclusão" ou "Listar Egressos em ordem de nome", se optar pela primeira opção o sistema carrega todos os campos do cadastro ordenados pelo ano de conclusão do curso, caso contrário ele gera o relatório em ordem alfabética, tendo como base o nome do aluno egresso, o sistema permite voltar para tela anterior (A1).

A1: Voltar

Após realização do relatório o sistema permite o retorno para tela inicial da geração de relatórios.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

O funcionário da coordenação do curso Tecnólogo em Informática faz seu login no sistema de egressos, escolhe a opção "Relatórios", e escolhe a opção "Listar por ano de conclusão", o sistema retorna o relatório ordenado pelo ano de conclusão do curso.

6.2.4.1.1.5 Módulo Egressos – Email do Sistema

Caso de uso: Email do Sistema.

Ator: Coordenação do Curso.

Descrição Resumida:

O funcionário da Coordenação do Curso poderá configurar um endereço de email, que enviará a senha dos usuários quando estes a solicitarem em caso de esquecimento.

- Fluxo de Eventos:

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do Curso em questão.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu usuário e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Email do Sistema, e poderá configurar uma conta de email, juntamente com seu servidor smtp, username, porta e senha, para que o sistema envie automaticamente a senha de um usuário quando este a solicitar.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login (nome de usuário) estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

Carlos, funcionário da coordenação do Curso de Gestão da Informação da UFPR, informa seu usuário e senha e acessa o Módulo de Egressos na Intranet Acadêmica. Em seguida acessa a opção Email do Sistema e insere os dados de uma conta de email, criada especialmente para esta funcionalidade no Módulo Egressos. Ele também insere a

configuração do servidor smtp, username, porta e senha. Ele marca a check box para autenticar o servidor e clica em Enviar. Está finalizada a configuração do email do sistema.

6.2.4.1.1.6 Módulo Egressos – Pesquisa

Caso de uso: Pesquisa

Ator: Coordenação do Curso.

Descrição Resumida:

Este caso de uso permite à coordenação do curso visualizar o questionário ativo na pesquisa feita para os egressos.

Fluxo de Eventos:

Caso de Uso: Pesquisa

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do curso.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, o funcionário deverá escolher a opção "Pesquisa", que lhe permitirá visualizar a pesquisa que está ativa para os egressos.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

Cenários

Cenário Principal

O funcionário da coordenação do curso Tecnólogo em Informática faz seu login no sistema de egressos, escolhe a opção "Pesquisa", podendo assim visualizar a pesquisa ativa para os egressos.

6.2.4.1.1.7 Módulo Egressos – Alterar Cadastro

Caso de uso: Alterar Cadastro

Ator: Coordenação do Curso

Descrição Resumida:

O funcionário da Coordenação do Curso poderá realizar atualização em seu cadastro pessoal.

- Fluxo de Eventos:

Caso de Uso: Atualizar Cadastro.

Pré-Condições: Usuário do sistema deve ser funcionário da coordenação do curso de Gestão da Informação da UFPR.

Fluxo Principal: Este caso de uso começa quando o funcionário acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, escolhe a opção Alterar Cadastro, podendo assim editar seu próprio cadastro. O usuário tem a opção de Salvar as alterações (A1), limpar campos (A2) e voltar, onde voltará ao menu anterior.

Caso durante a edição do cadastro, o usuário deixe de preencher algum dos campos o sistema realiza validação e retorna mensagem de erro caso algum campo fique em branco. (E2)

Fluxos Alternativos:

A1: Salvar:

Se os dados forem inseridos corretamente e validados pelo sistema, as informações serão salvas no banco de dados e o sistema retornará a mensagem "OPERAÇÃO REALIZADA COM SUCESSO!".

A2: Limpar:

Limpa os campos do formulário.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Se não houver nenhum campo preenchido o sistema retorna quais os campos que devem ser preenchidos.

Cenários

Cenário Principal

Carlos, funcionário da coordenação do curso de Gestão da Informação, informa seu login e senha e acessa o Módulo de Egressos na Intranet Acadêmica. Em seguida acessa a opção Alterar Cadastro e edita os campos referentes ao seu endereço.

Após preencher os campos, clica no botão salvar e os dados são alterados com sucesso.

Cenários Secundários

- Carlos deseja alterar seu cadastro, realiza algumas alterações, mas esquece de preencher o campo Cidade e clica em salvar. O sistema informa ao usuário que o campo cidade não foi preenchido. Carlos preenche o campo e as alterações são salvas.

6.2.4.1.2 Visão Egresso

6.2.4.1.2.1 Módulo Egressos – Registrar Usuário

Caso de uso: Registrar Usuário

Ator: Egresso.

Descrição Resumida:

Este caso de uso é iniciado por um egresso do Curso de Gestão da Informação da Universidade Federal do Paraná. Ele permite ao egresso se cadastrar no sistema.

- Fluxo de Eventos

Caso de Uso: Registrar usuário

Pré-Condições

Usuário deve ser obrigatoriamente egresso do curso de Gestão da Informação da Universidade Federal do Paraná.

Fluxo Principal: Este caso de uso começa quando o egresso acessa a Intranet Acadêmica informando sua identificação e senha, escolhe a opção "Novo Usuário – Registre-se" no Módulo de Egressos, após clicar na opção um formulário de cadastro é carregado, e o usuário preenche o mesmo, caso o aluno já seja cadastrado, o sistema retorna a mensagem "CPF já cadastrado".

Para os campos de preenchimento obrigatório o sistema realiza validação e retorna mensagem de erro caso algum campo fique em branco. (E1)

Fluxos Alternativos:

A2: Salvar:

Se os dados forem inseridos corretamente e validados pelo sistema, as informações serão salvas no banco de dados e o sistema retornará a mensagem "OPERAÇÃO REALIZADA COM SUCESSO!".

A3: Limpar:

Limpa os campos do formulário.

Fluxos de Exceção:

E1: Se não houver nenhum campo preenchido o sistema retorna quais os campos que devem ser preenchidos.

Cenários

Cenário Principal

Armando, formado no curso de Gestão da Informação da UFPR, acessa o sistema de egressos na intranet acadêmica, na tela de login escolhe a opção para realizar novo cadastro, preenche então o formulário e clica em salvar.

Cenários Secundários

- Após preencher o formulário e clicar no botão Salvar, o sistema retorna a mensagem que o CPF informado já está cadastrado.
- Armando preenche o formulário deixando o campo Cidade em branco. O sistema retorna a mensagem indicando a necessidade de preenchimento deste campo. (Obs.: esta validação é realizada em todos os campos, exceto o complemento).

6.2.4.1.2.2 Módulo Egressos – Pesquisa

Caso de Uso: Pesquisa.

Ator: Egresso.

Descrição Resumida:

Este caso de uso é iniciado por um egresso do Curso de Gestão da Informação da Universidade Federal do Paraná. Ele permite ao egresso responder a uma pesquisa elaborada pela Coordenação do Curso.

- Fluxo de Eventos

Caso de Uso: Responder Pesquisa

Pré-Condições

Usuário deve ser obrigatoriamente egresso do curso de Gestão da Informação da Universidade Federal do Paraná.

Fluxo Principal: Este caso de uso começa quando o egresso acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, o egresso irá escolher a opção Pesquisa e irá responder ao questionário elaborado pela Coordenação do Curso (E2), o egresso poderá responder a pesquisa ativa uma única vez (E3).

Fluxos Alternativos

A1: Salvar

Após responder a pesquisa o egresso clica no botão salvar, a pesquisa é salva.

Fluxos de Exceção

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Caso o egresso deixe de responder a alguma pergunta, o sistema retorna: "Pesquisa incompleta".

E3: O egresso clica em pesquisa e o sistema retorna a informação "Você já respondeu a pesquisa", cada pesquisa poderá ser respondida uma única vez.

Cenários

Cenário Principal

Armando, formado no curso de Gestão da Informação da UFPR, acessa o sistema de egressos na Intranet Acadêmica, informa seu login e senha, acessa a opção Pesquisa e responde as questões, após responder clica em salvar.

Cenários Secundários

- Armando digita usuário ou senha inválidos, o sistema retorna uma mensagem alertando sobre esse erro e disponibiliza novamente a tela de Login.
- Armando tenta responder a pesquisa e o sistema retorna a mensagem de que a pesquisa já foi respondida.

6.2.4.1.2.3 Módulo Egressos – Alterar Cadastro

Caso de uso: Alterar Cadastro.

Ator: Aluno Egresso.

Descrição Resumida:

O usuário, egresso do curso em questão, poderá realizar alterações em seus dados pessoais.

- Fluxo de Eventos:

Caso de Uso: Alterar Cadastro

Usuário deve ser obrigatoriamente egresso do curso de Gestão da Informação da Universidade Federal do Paraná.

Fluxo Principal: Este caso de uso começa quando o egresso acessa a Intranet Acadêmica informando sua identificação e senha. Após isso, acessa o Módulo de Egressos identificando-se com seu login e senha. O sistema de Egressos informará caso o usuário digite o login ou a senha incorretos (E1). Caso o usuário tenha esquecido sua senha, poderá executar o fluxo "Esqueceu sua senha?", e a senha será enviada para o seu email, previamente cadastrado no sistema.

Uma vez logado no sistema, o egresso irá escolher a opção Alterar Cadastro e irá realizar as alterações necessárias, após isso, clica em salvar.

Fluxos Alternativos:

A1: Salvar:

Após realização da alteração dos dados, o usuário clica na opção salvar e os dados são gravados.

A2: Limpar:

Limpa os campos do formulário.

A3: Voltar

Volta para tela inicial do sistema.

Fluxos de Exceção:

E1: Sistema retorna ao usuário que sua senha ou login estão incorretos. O caso de uso termina.

E2: Se não houver nenhum campo preenchido o sistema retorna quais os campos que devem ser preenchidos.

Cenários

Cenário Principal

Armando, formado no curso de Gestão da Informação da UFPR, acessa o sistema de egressos na intranet acadêmica, informa seu login e senha, acessa a opção Alterar Cadastro, altera seu endereço e clica em salvar.

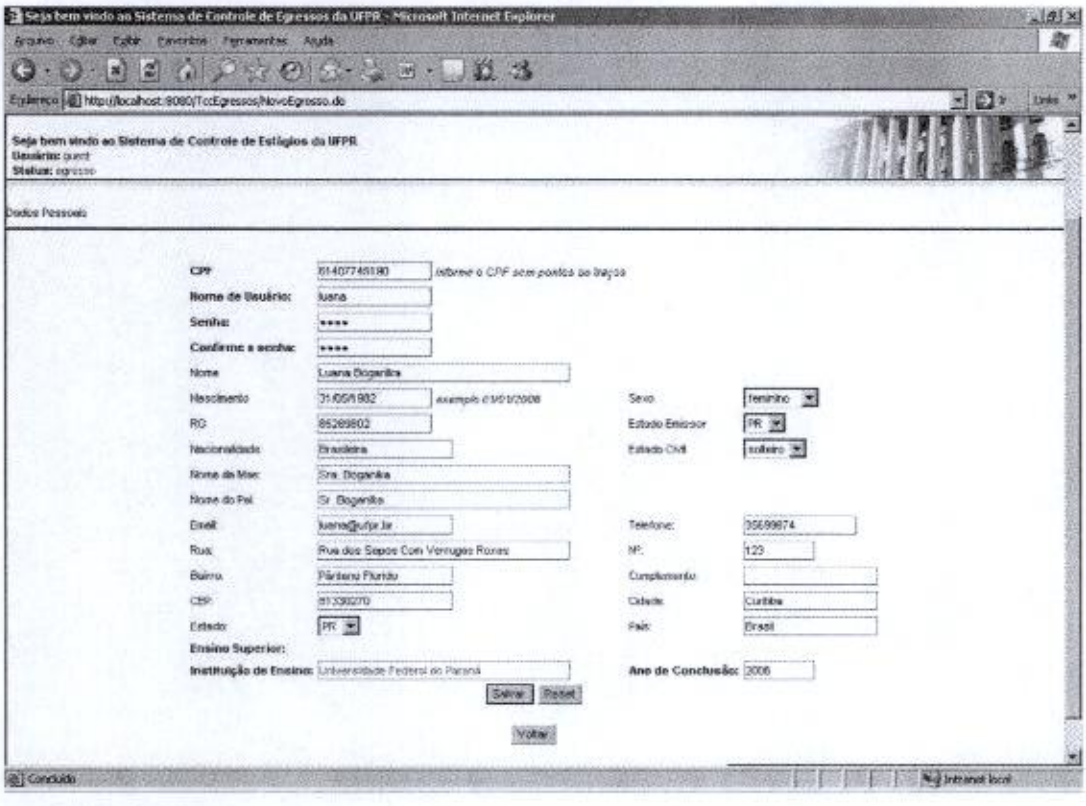
Cenários Secundários

- Armando digita usuário ou senha inválidos, o sistema retorna uma mensagem alertando sobre esse erro e disponibiliza novamente a tela de Login.
- Armando altera os dados, mas deixa o campo cidade em branco, o sistema informa quanto ao erro.

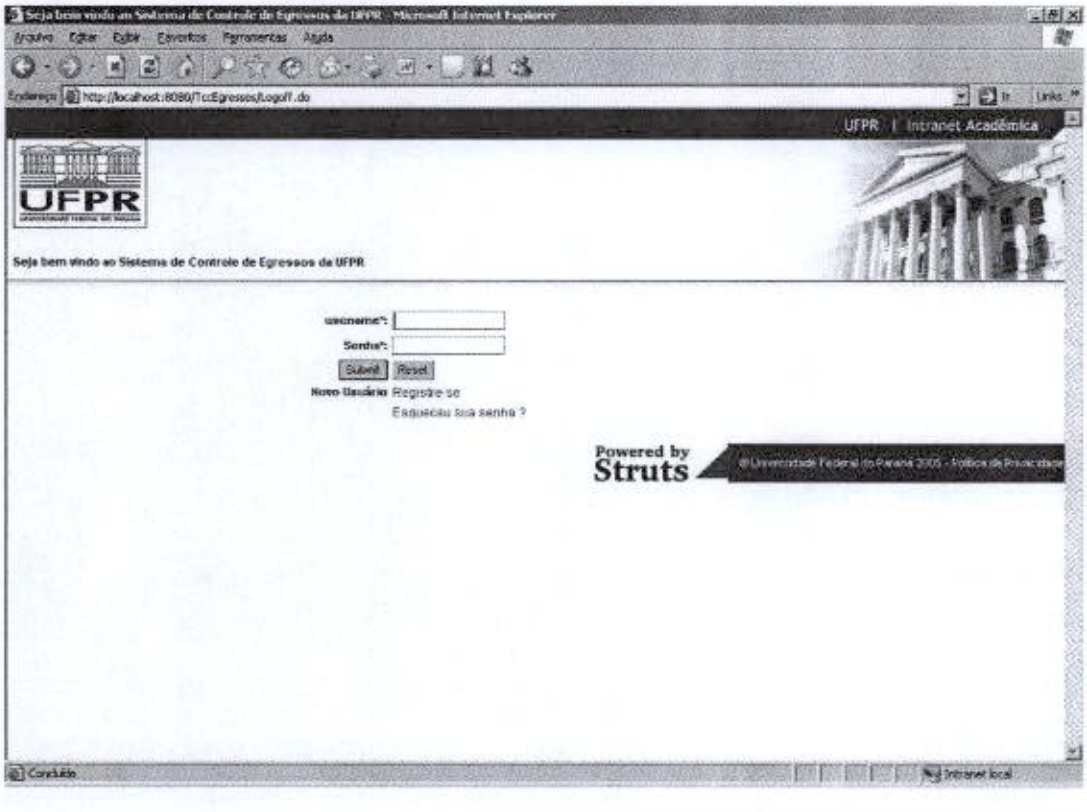
6.2.5 Diagramas de Telas

6.2.5.1 Visão Egresso

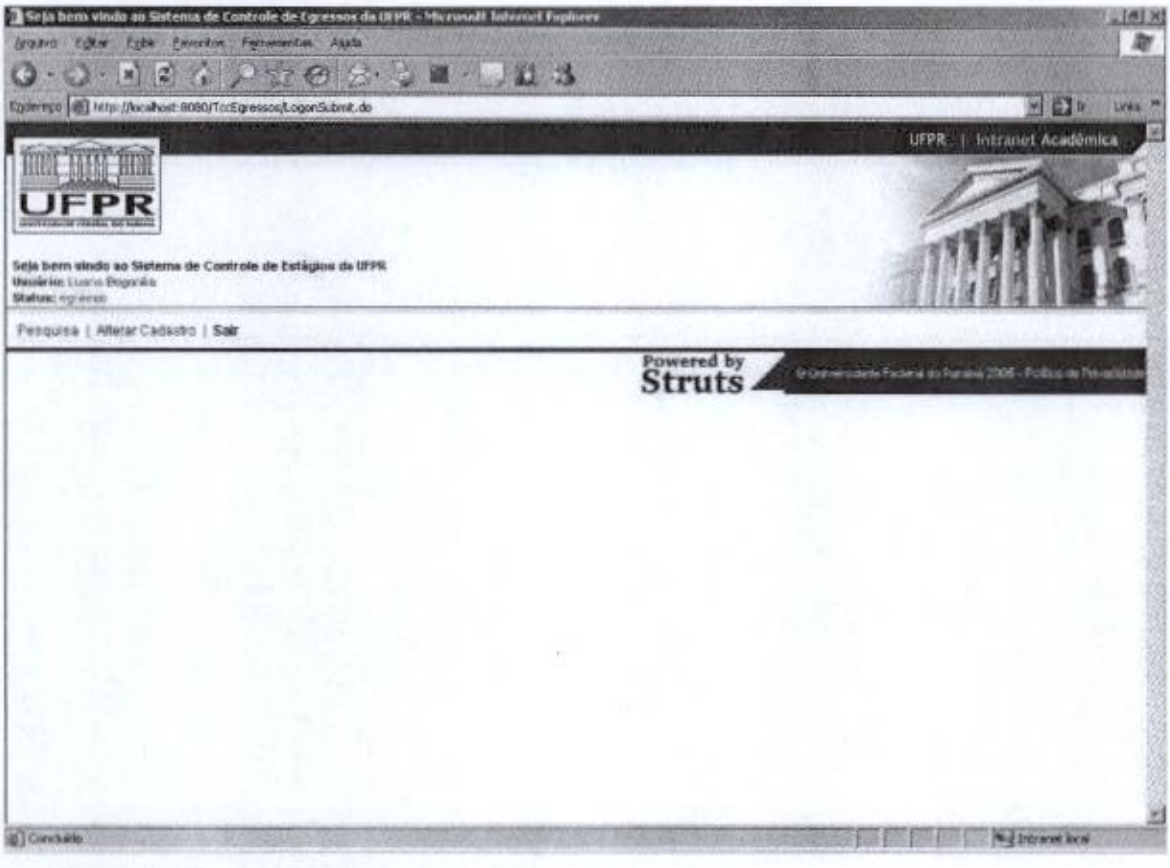
6.2.5.1.1 Fluxo Registre-se – Novo Usuário



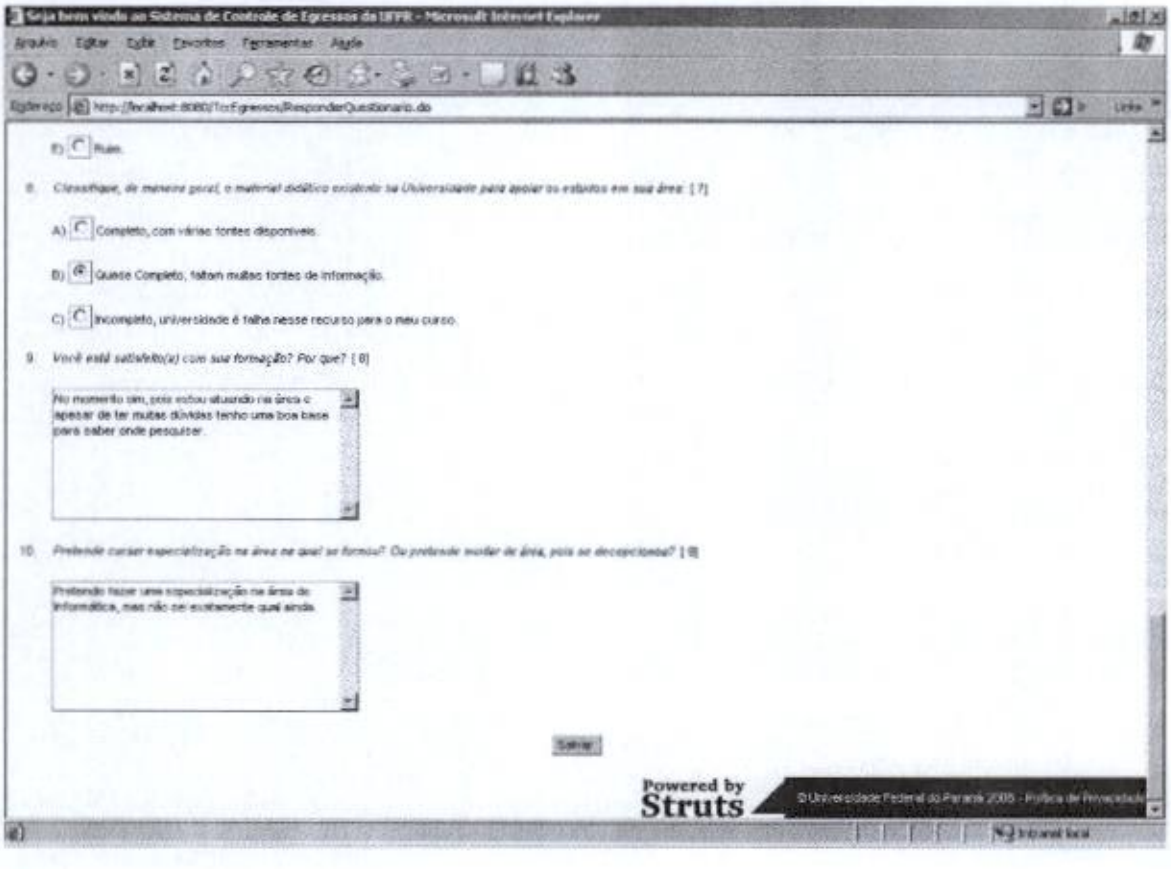
6.2.5.1.2 Login



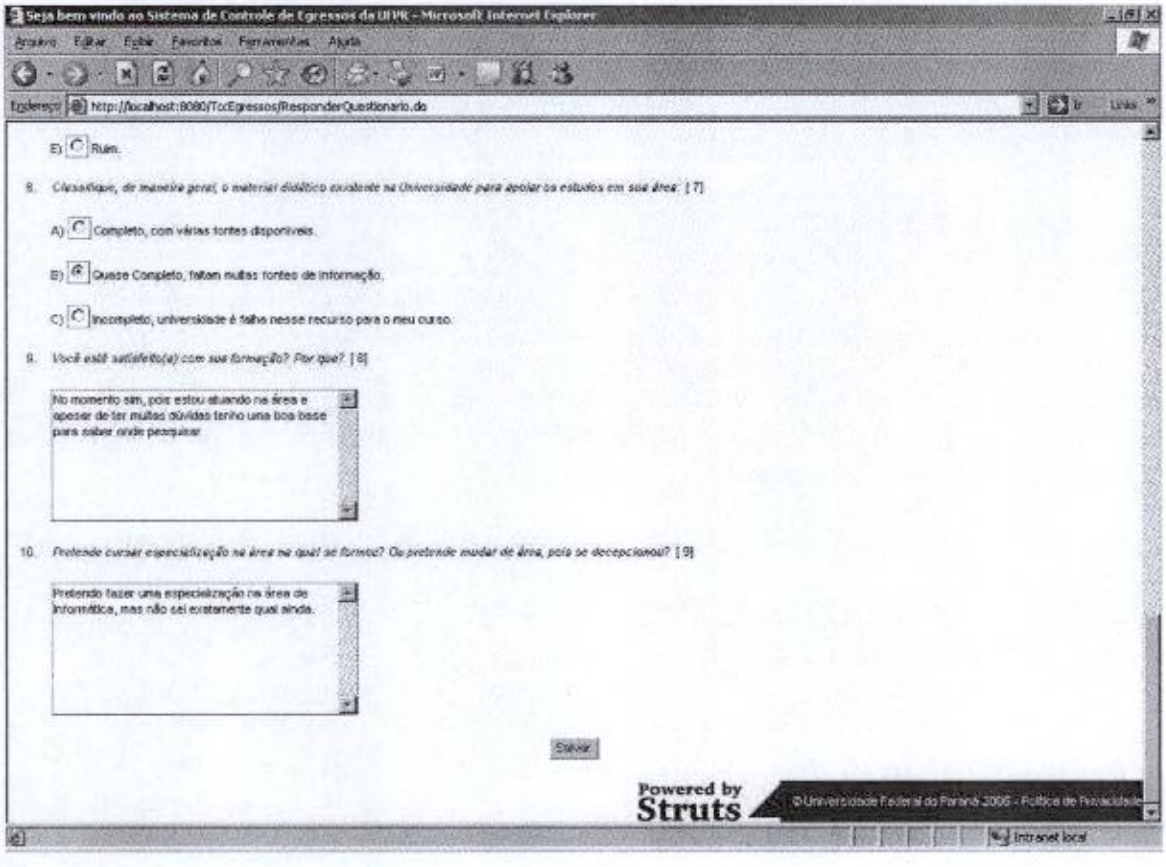
6.2.5.1.3 Tela Inicial



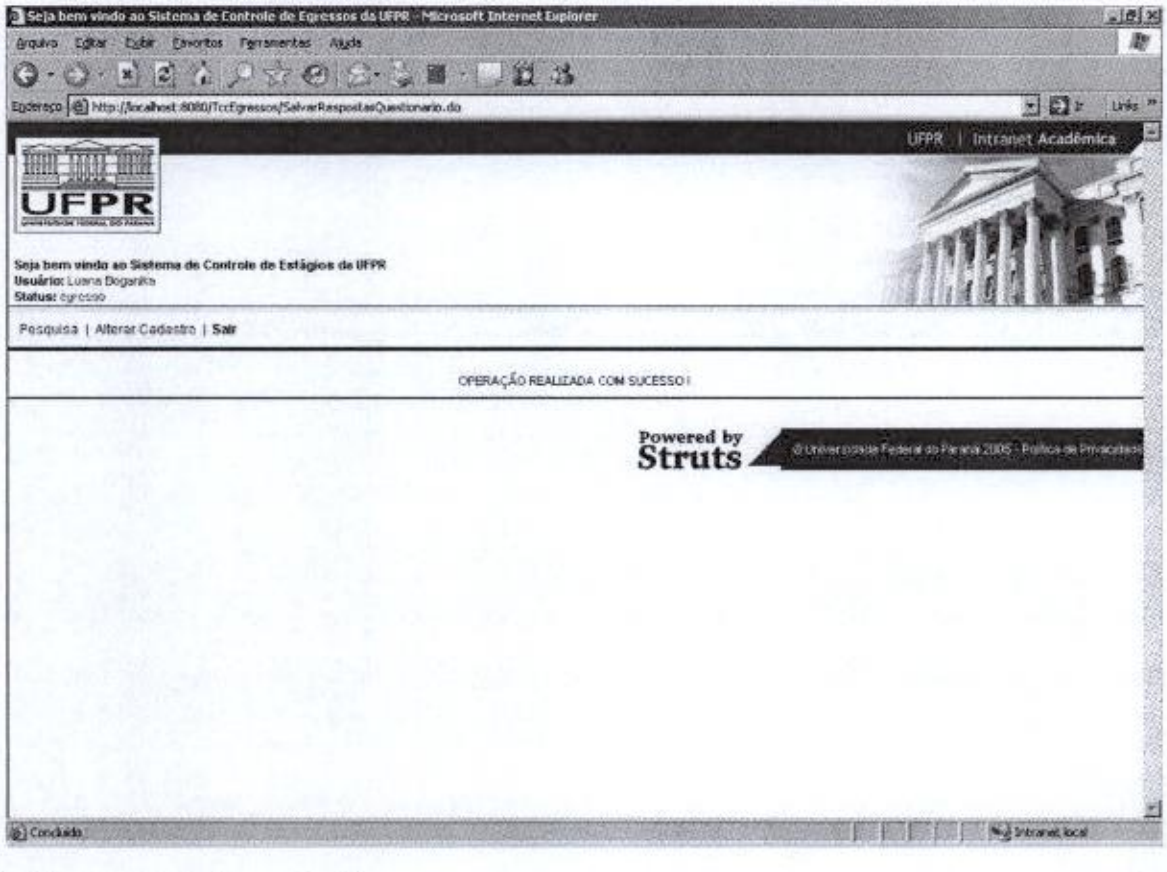
6.2.5.1.4 Fluxo Pesquisa – Usuário Respondendo



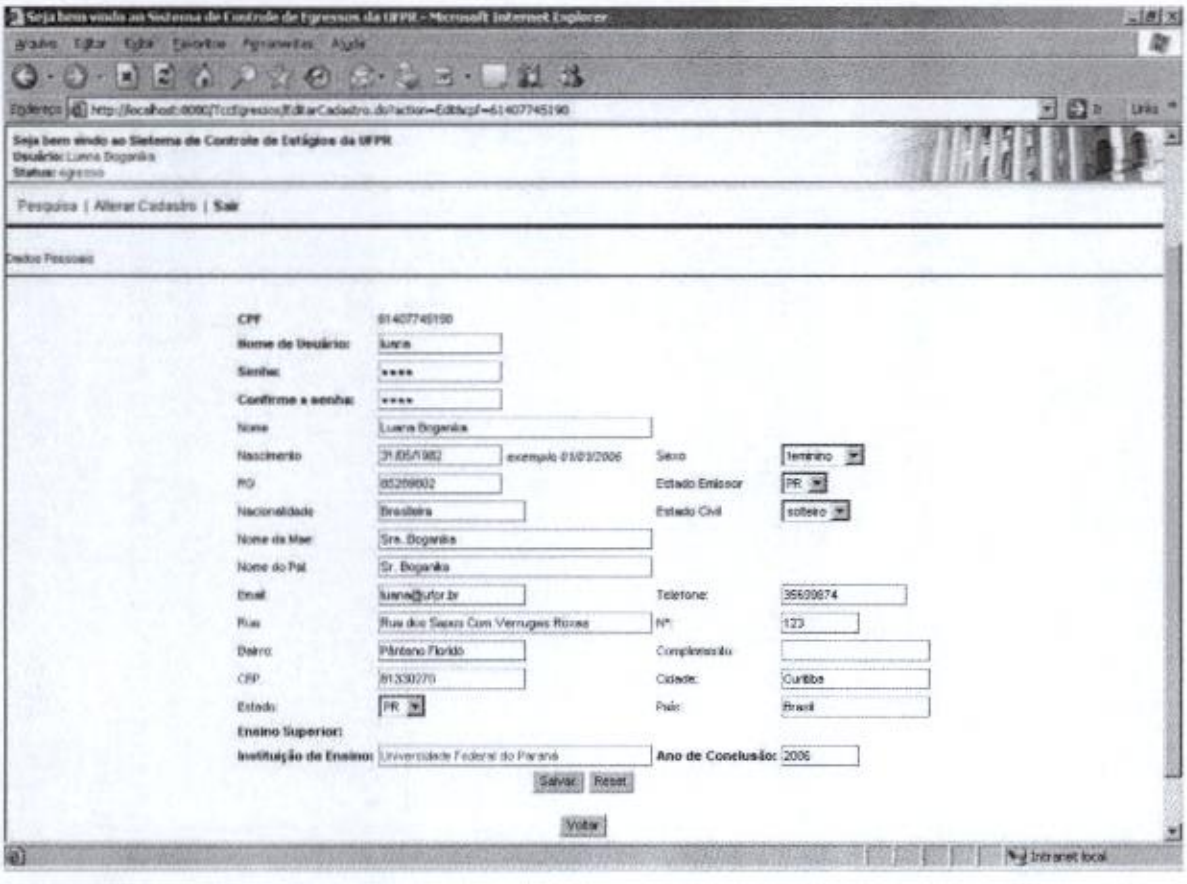
6.2.5.1.5 Fluxo Pesquisa – Usuário Respondendo (Continuação)



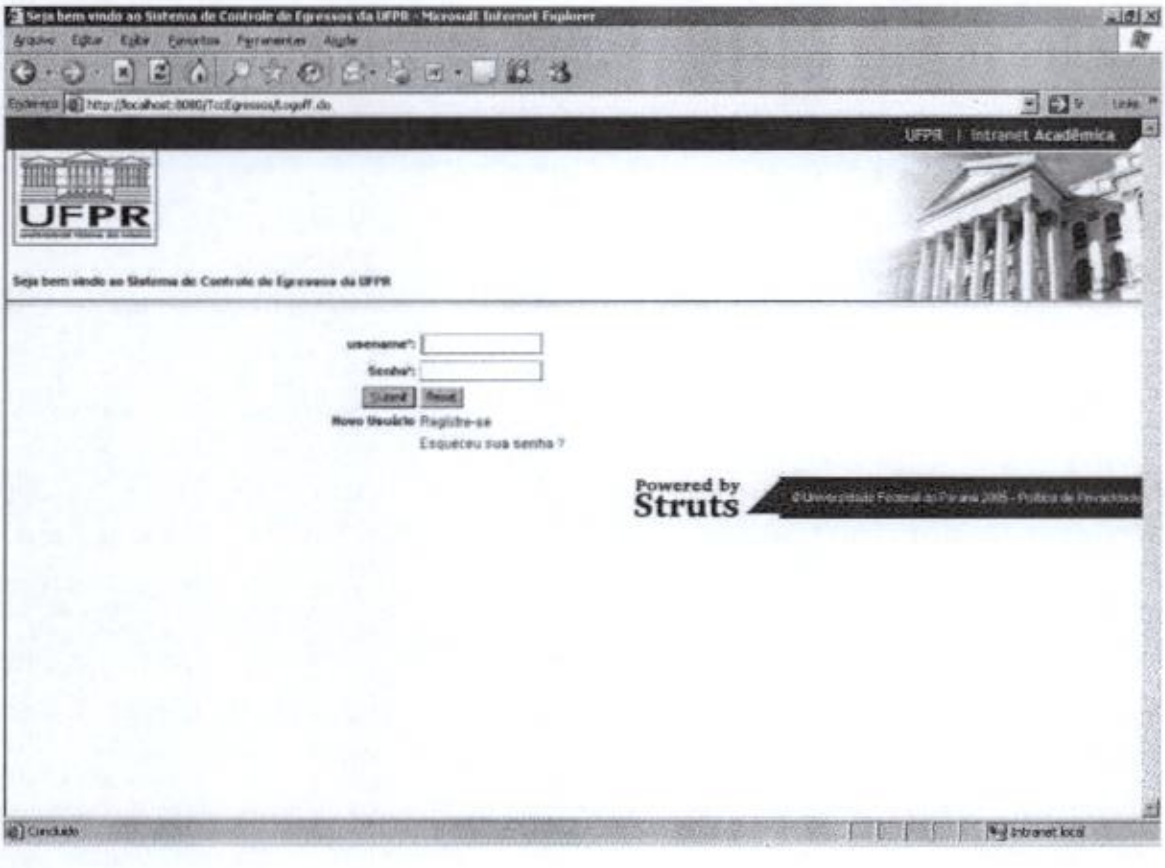
6.2.5.1.6 Fluxo Pesquisa – Pesquisa Respondida



6.2.5.1.7 Fluxo Alterar Cadastro – Usuário Edita Dados Pessoais

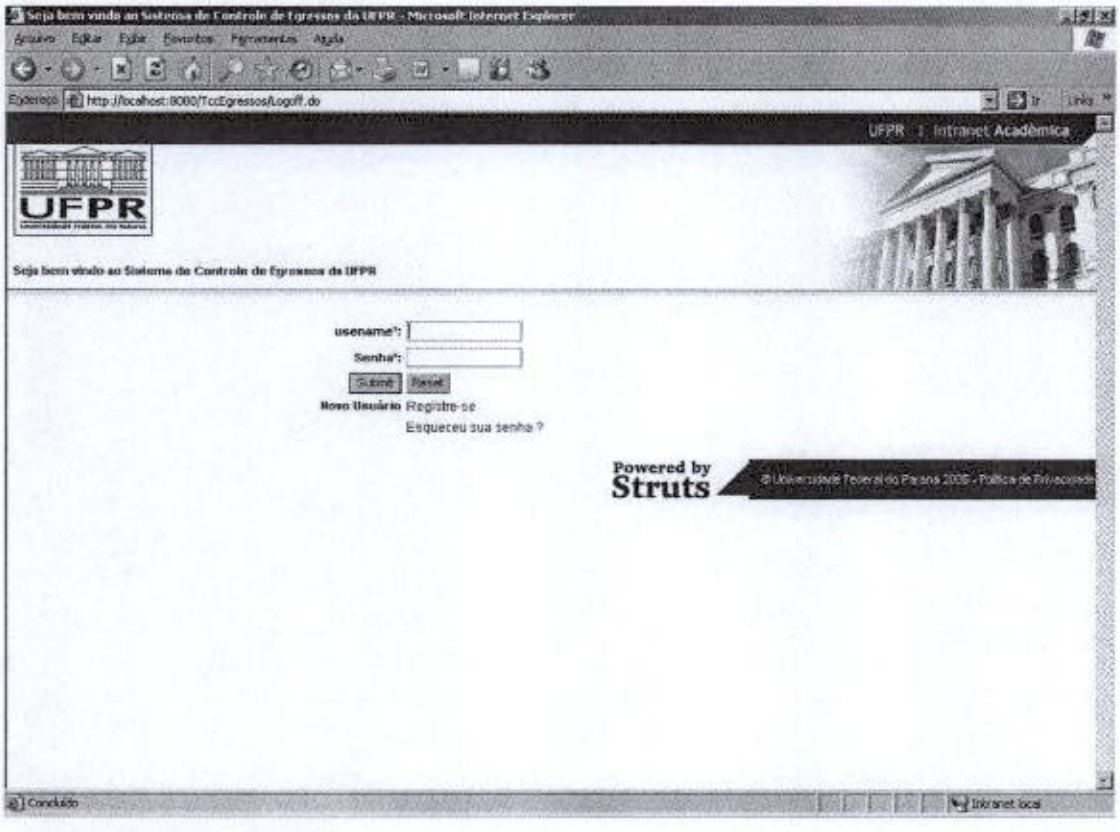


6.2.5.1.8 Sair

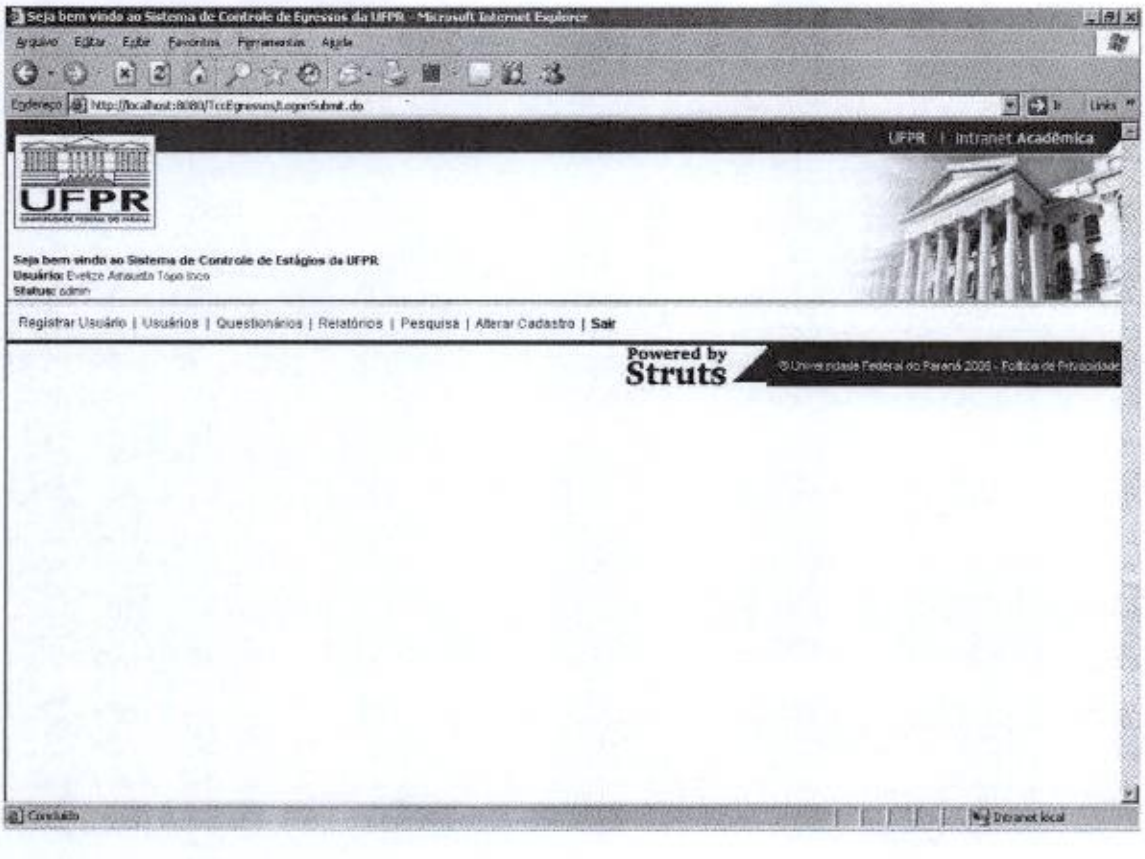


6.2.5.2 Visão Coordenação

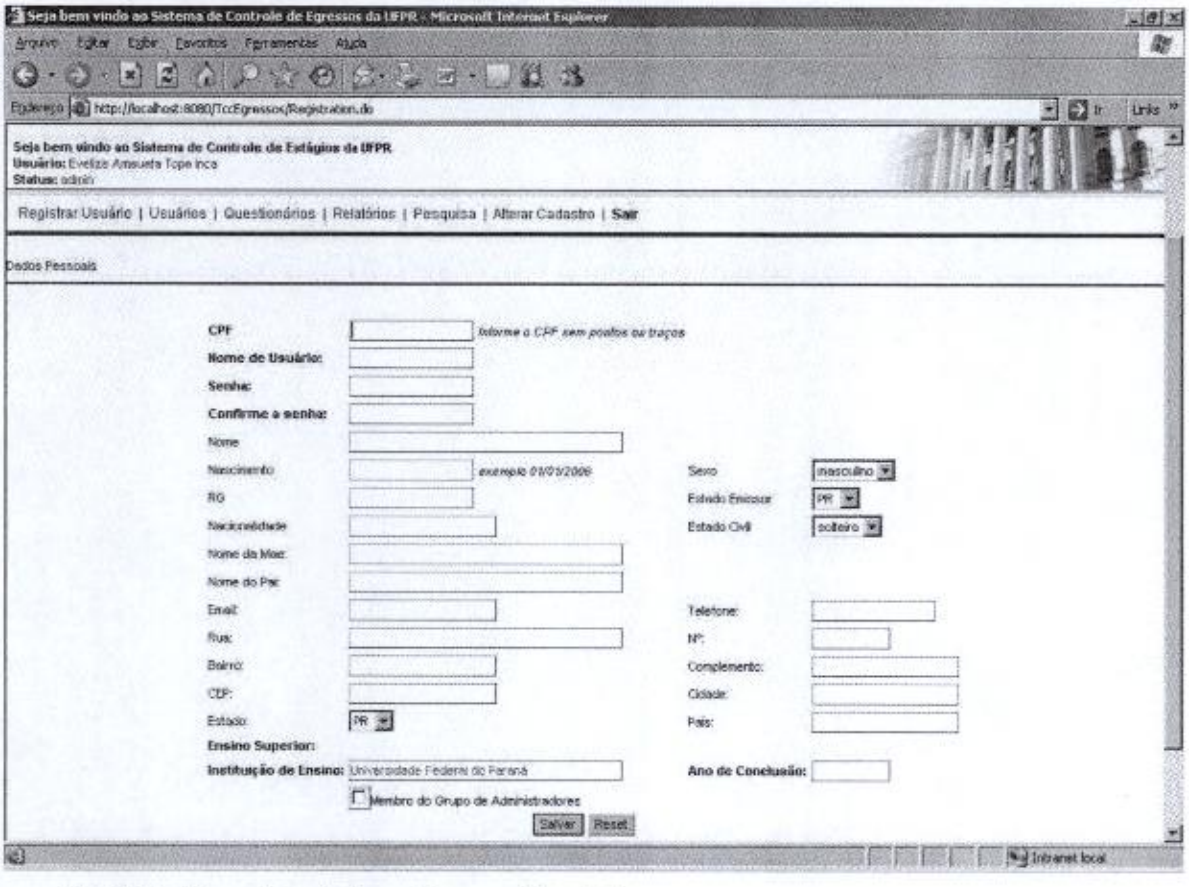
6.2.5.2.1 Login



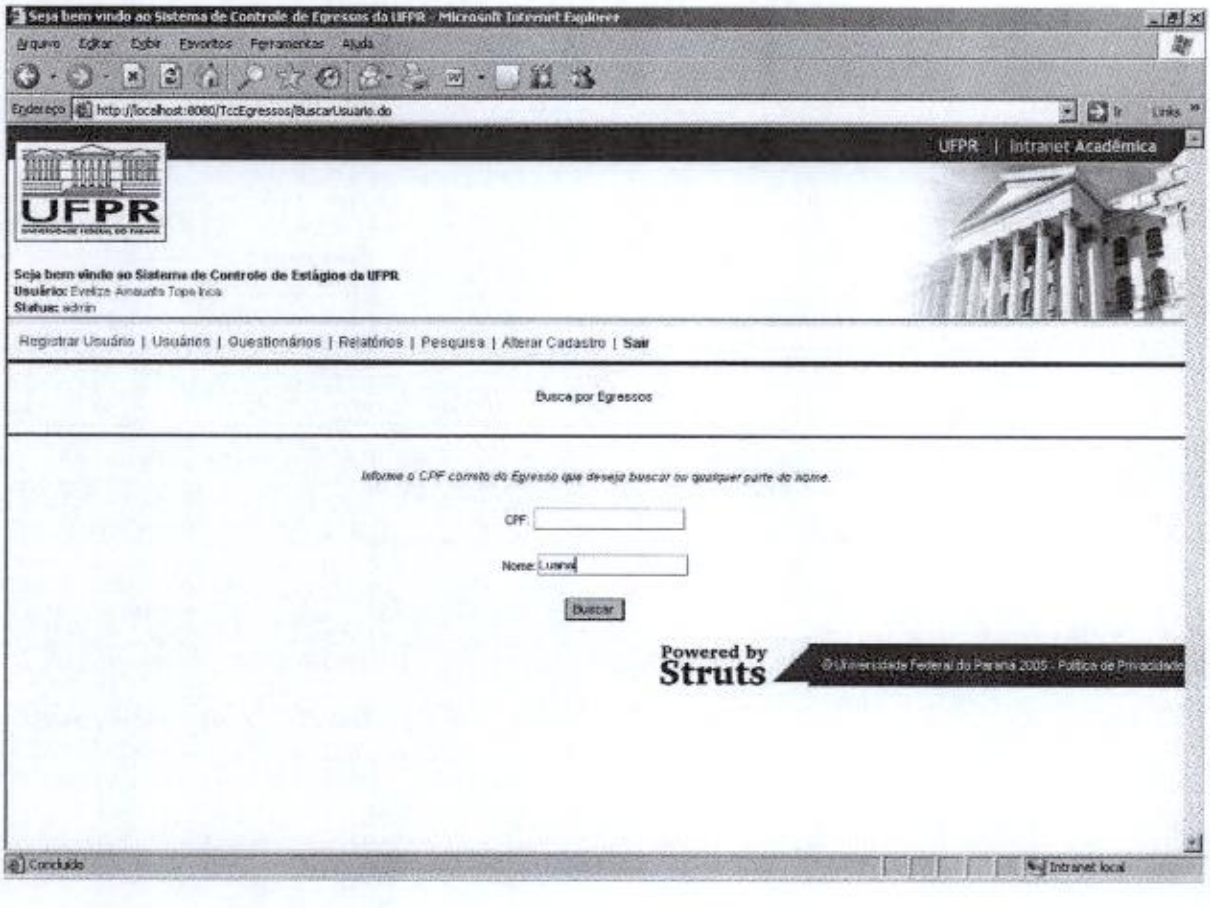
6.2.5.2.2 Tela Inicial



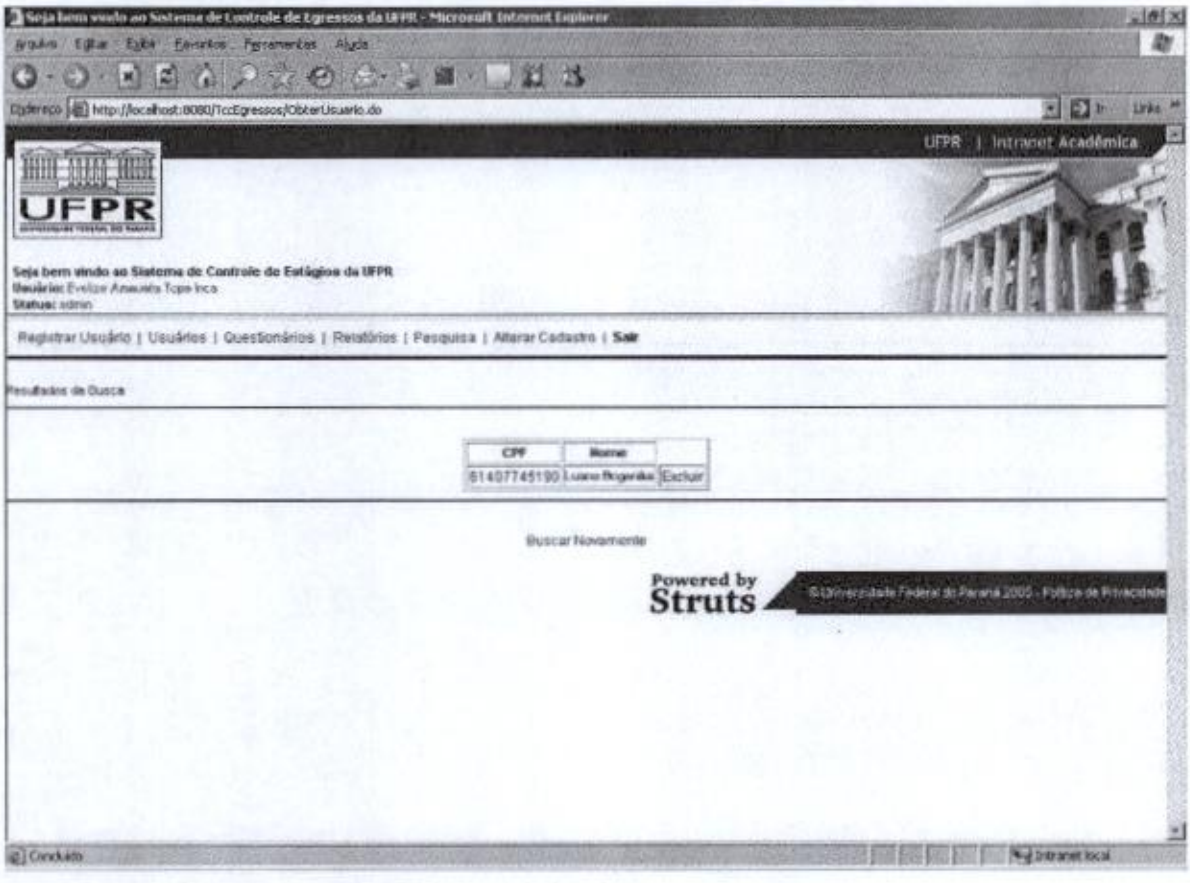
6.2.5.2.3 Fluxo Registrar Usuário – Formulário de Cadastro



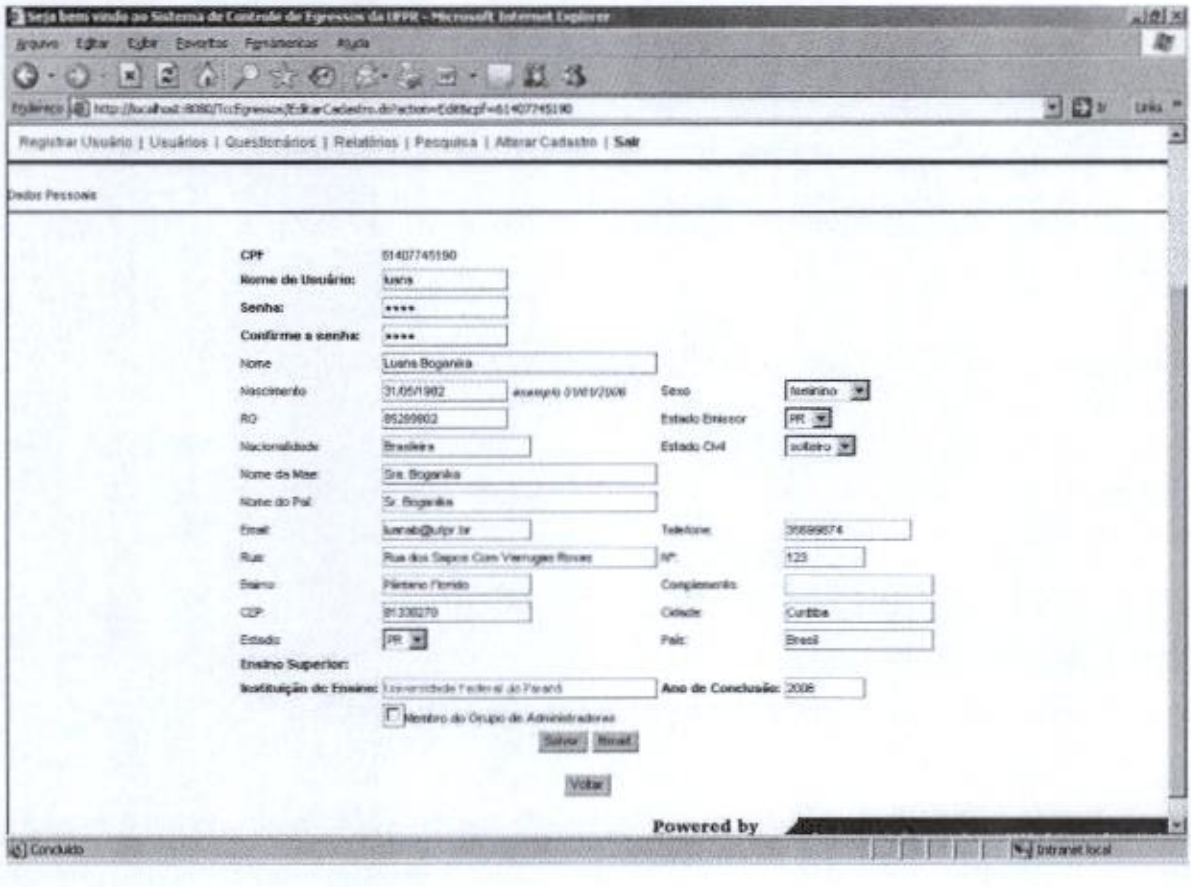
6.2.5.2.4 Fluxo Usuários – Buscar Usuários



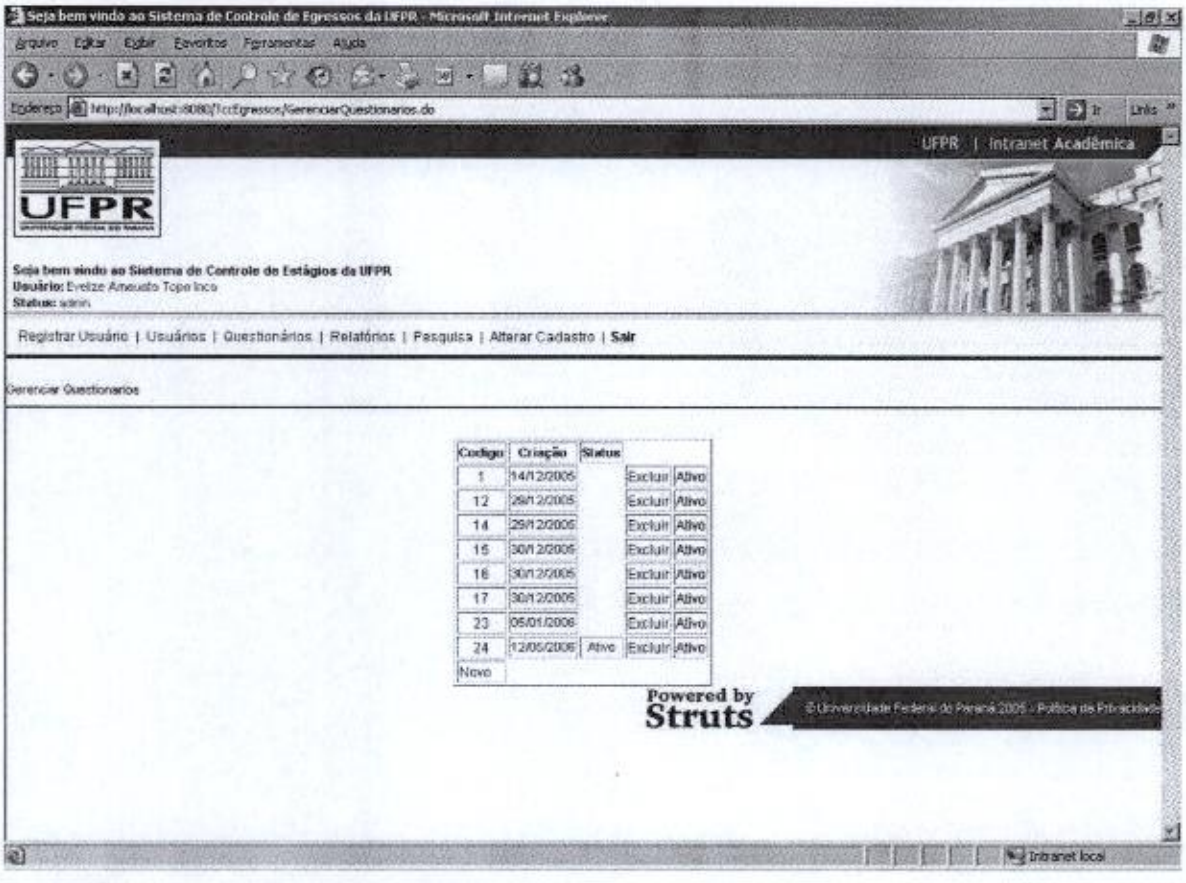
6.2.5.2.5 Fluxo Usuários – Listando Usuário(s) Encontrado(s)



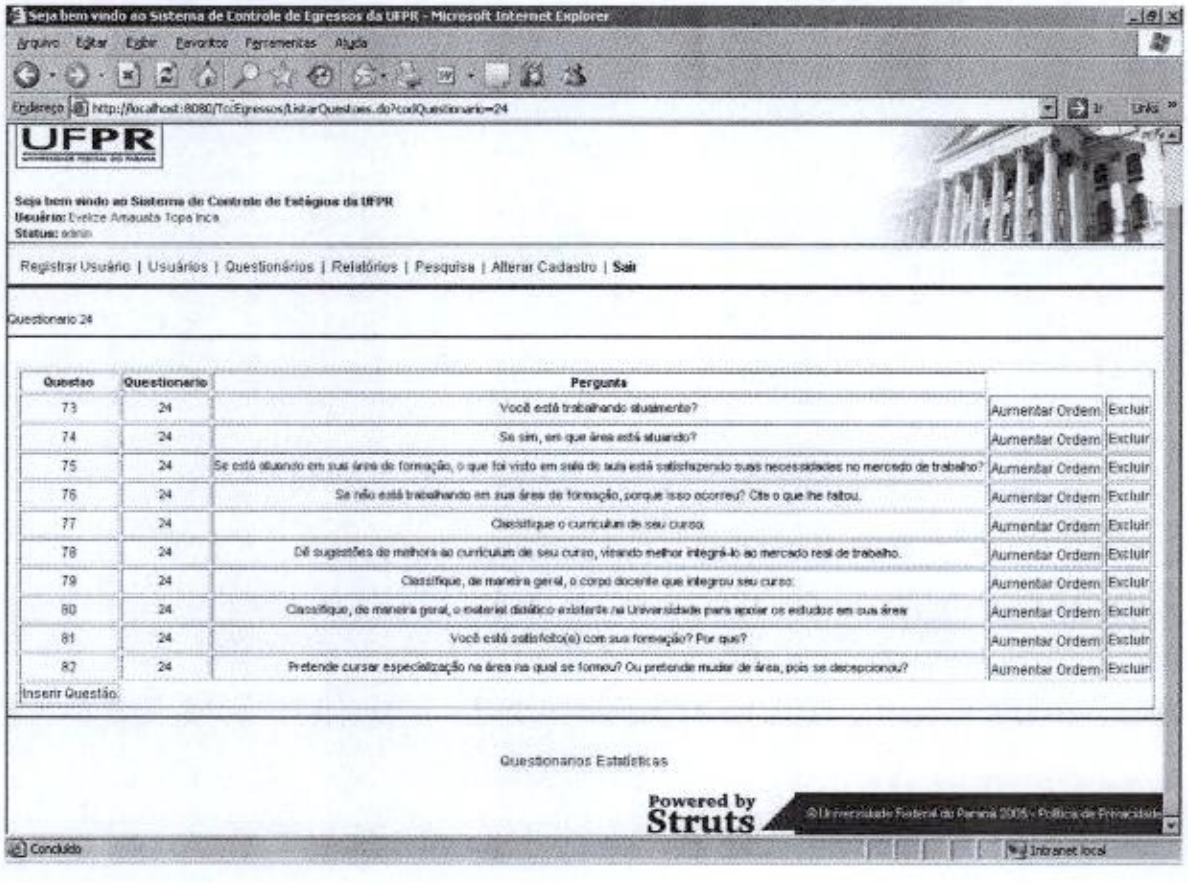
6.2.5.2.6 Fluxo Usuários – Editando Cadastro



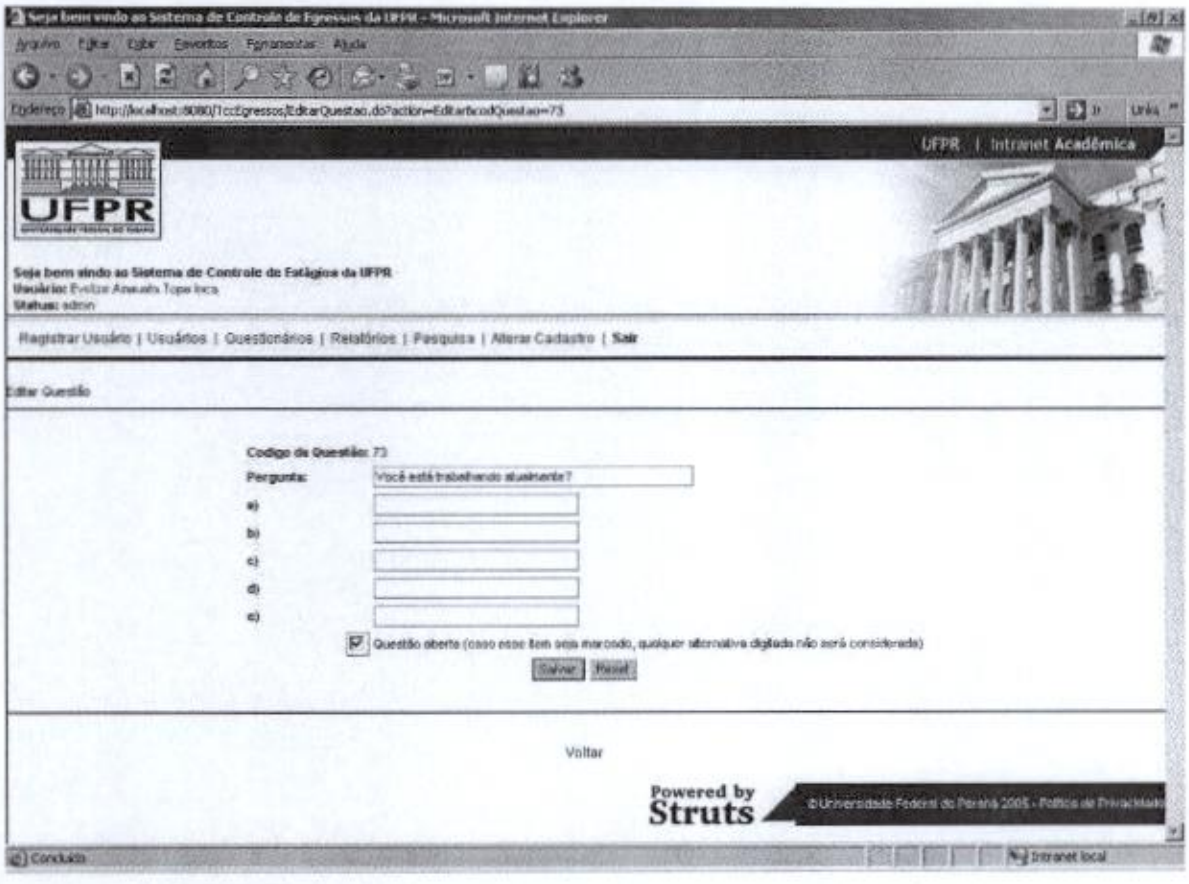
6.2.5.2.7 Fluxo Questionários - Visualizar



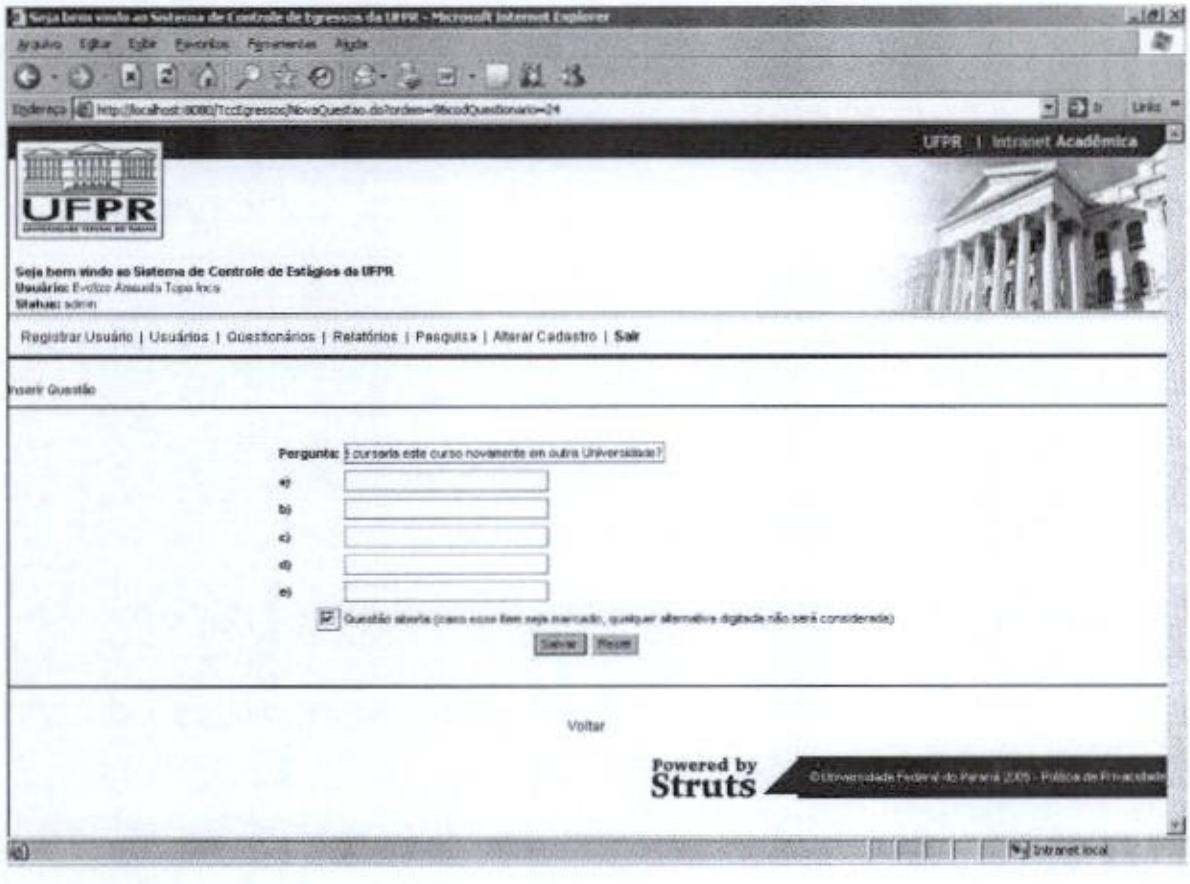
6.2.5.2.8 Fluxo Questionários – Visualizando Questionário já inserido



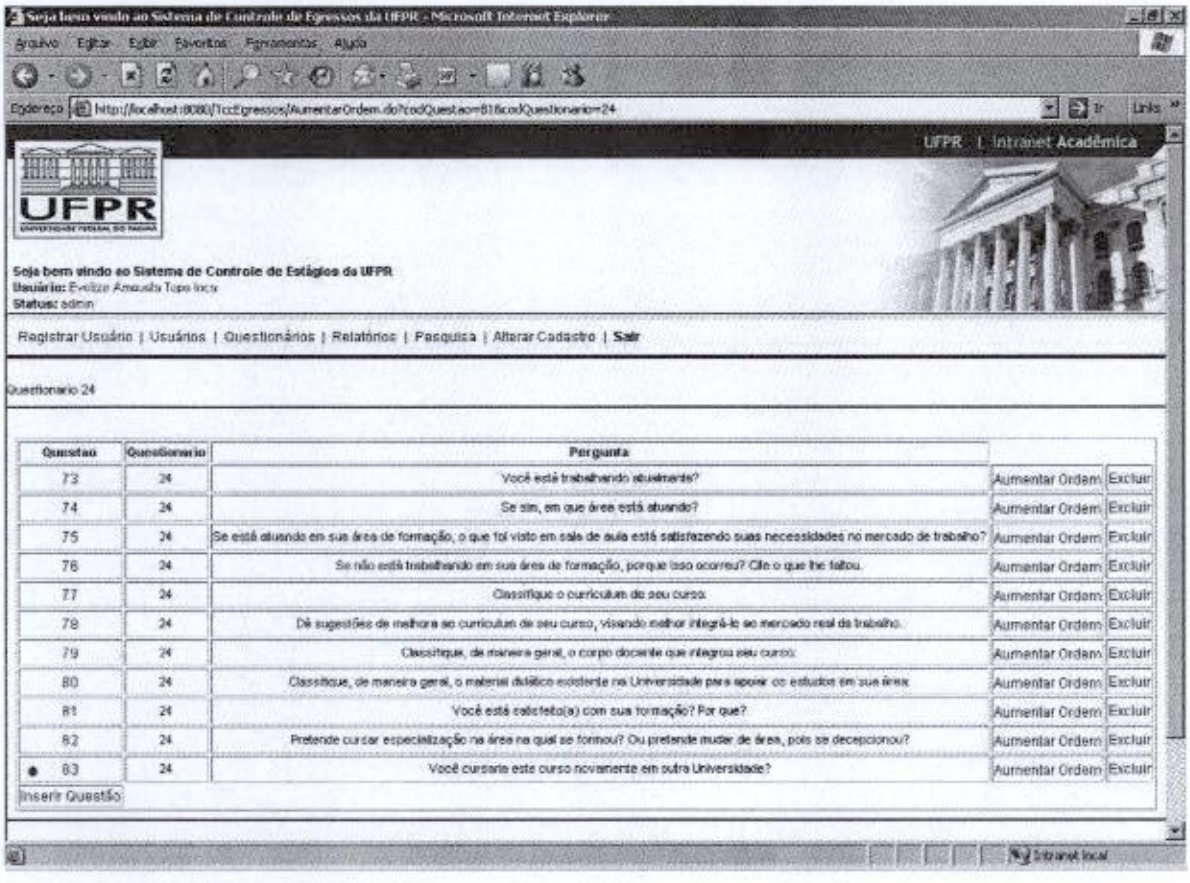
6.2.5.2.9 Fluxo Questionários – Editar Questão



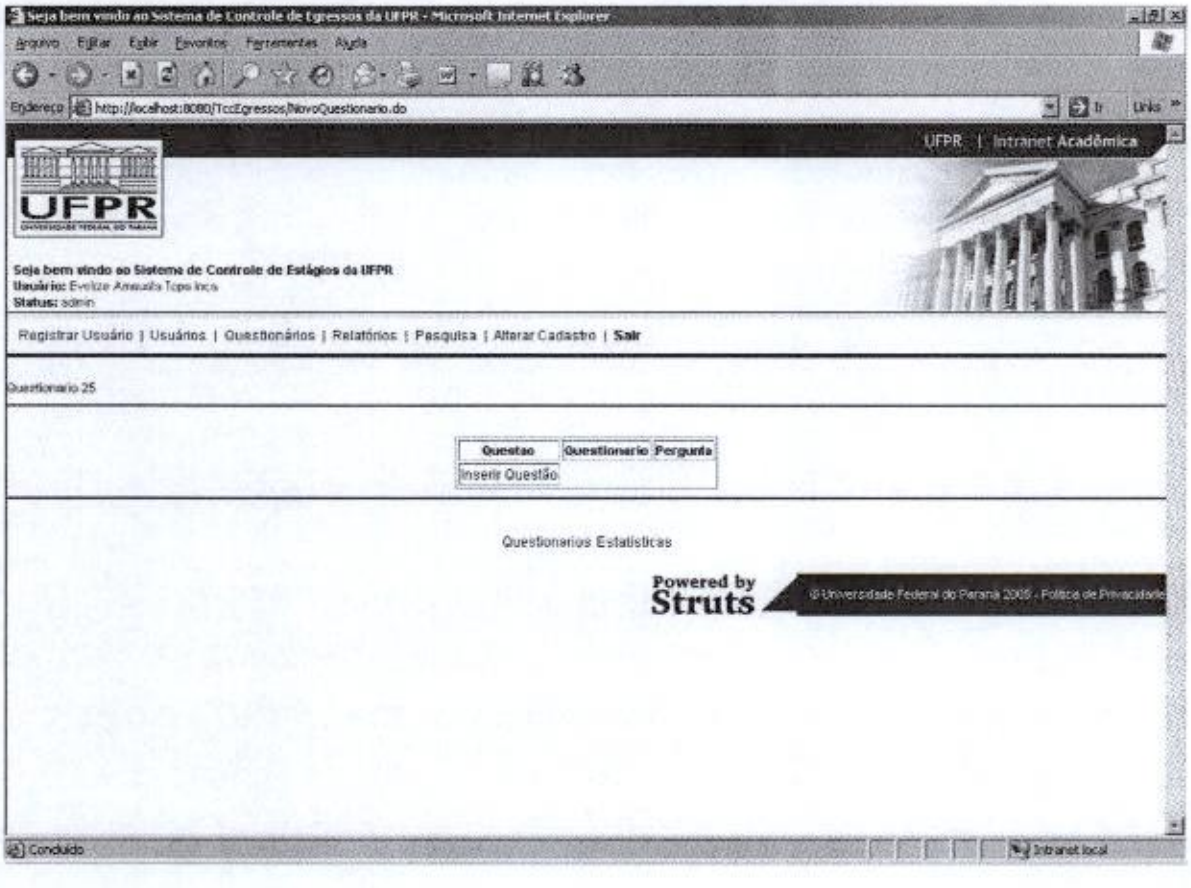
6.2.5.2.10 Fluxo Questionários – Inserir Questão no questionário existente



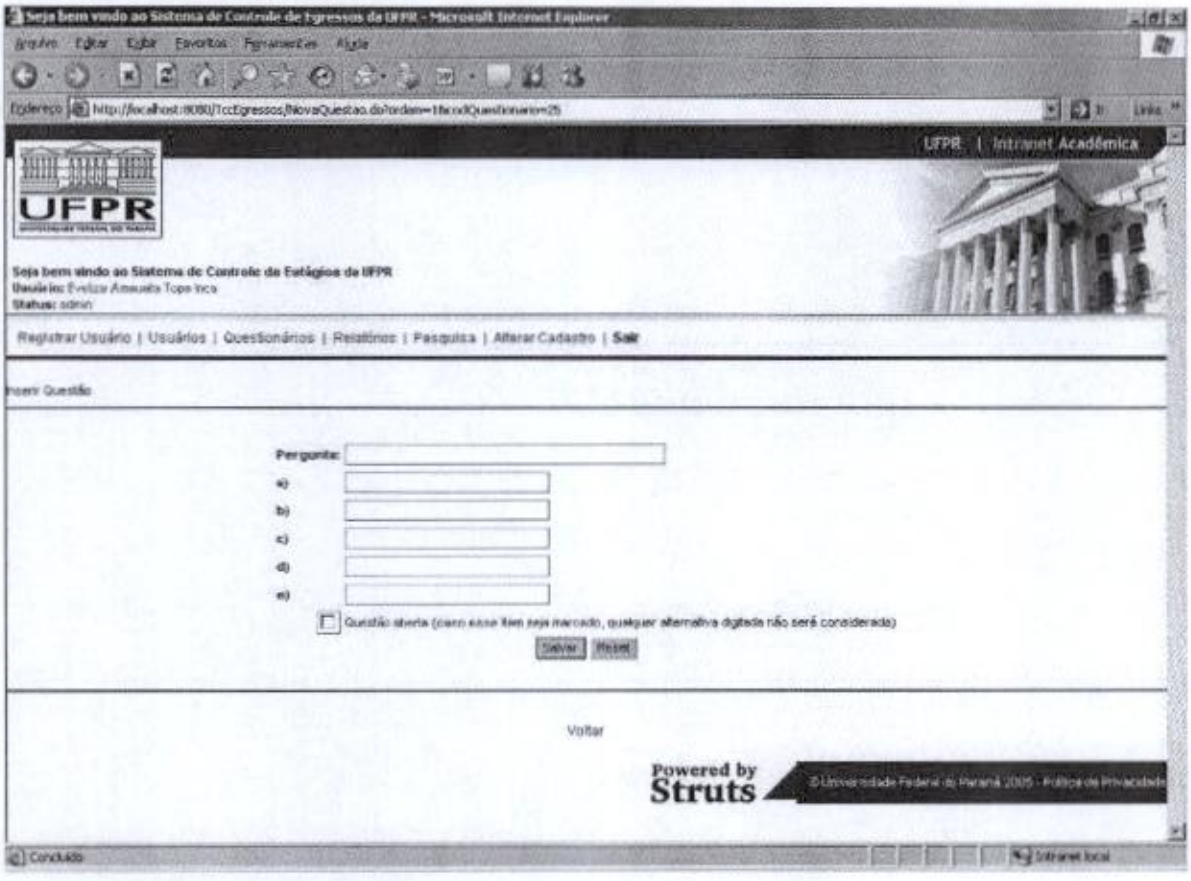
6.2.5.2.11 Fluxo Questionários – Visualizar Questão Inserida



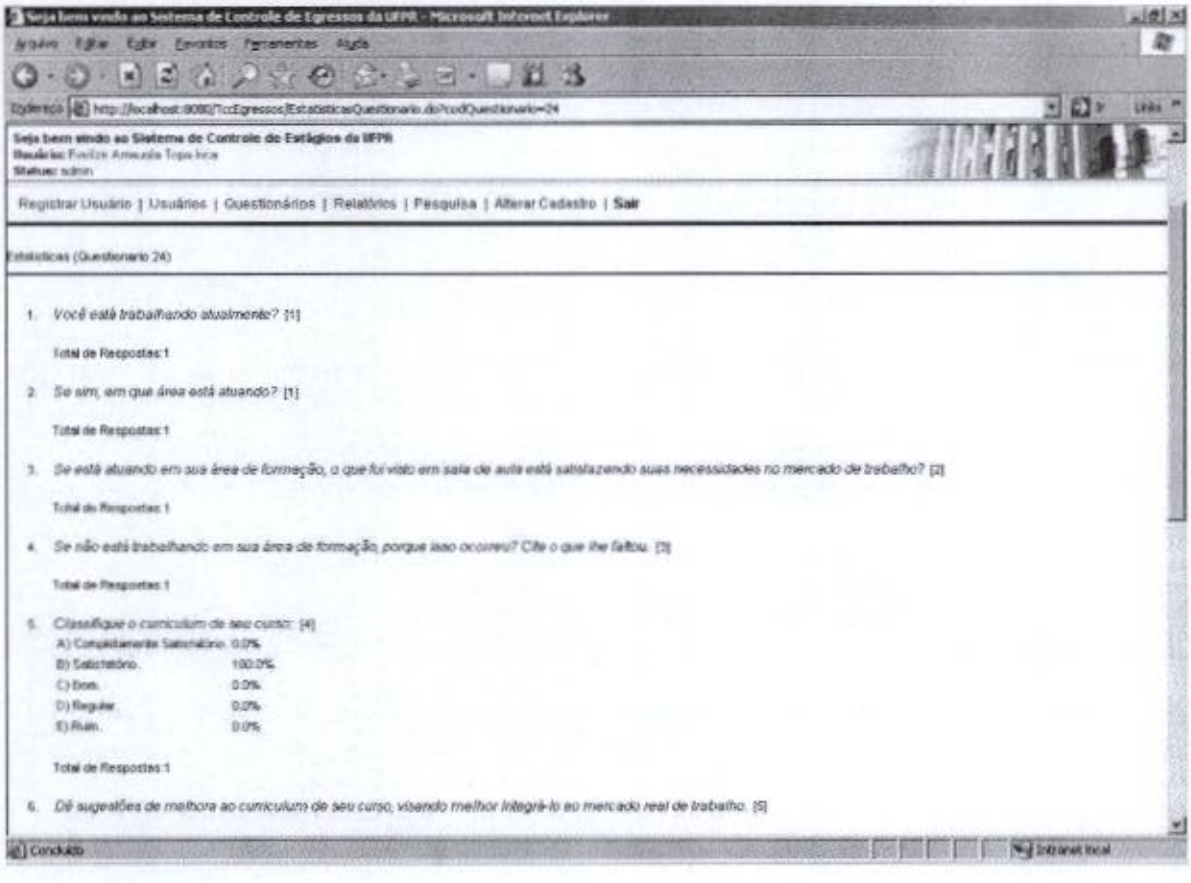
6.2.5.2.12 Fluxo Questionários – Inserir Novo Questionário



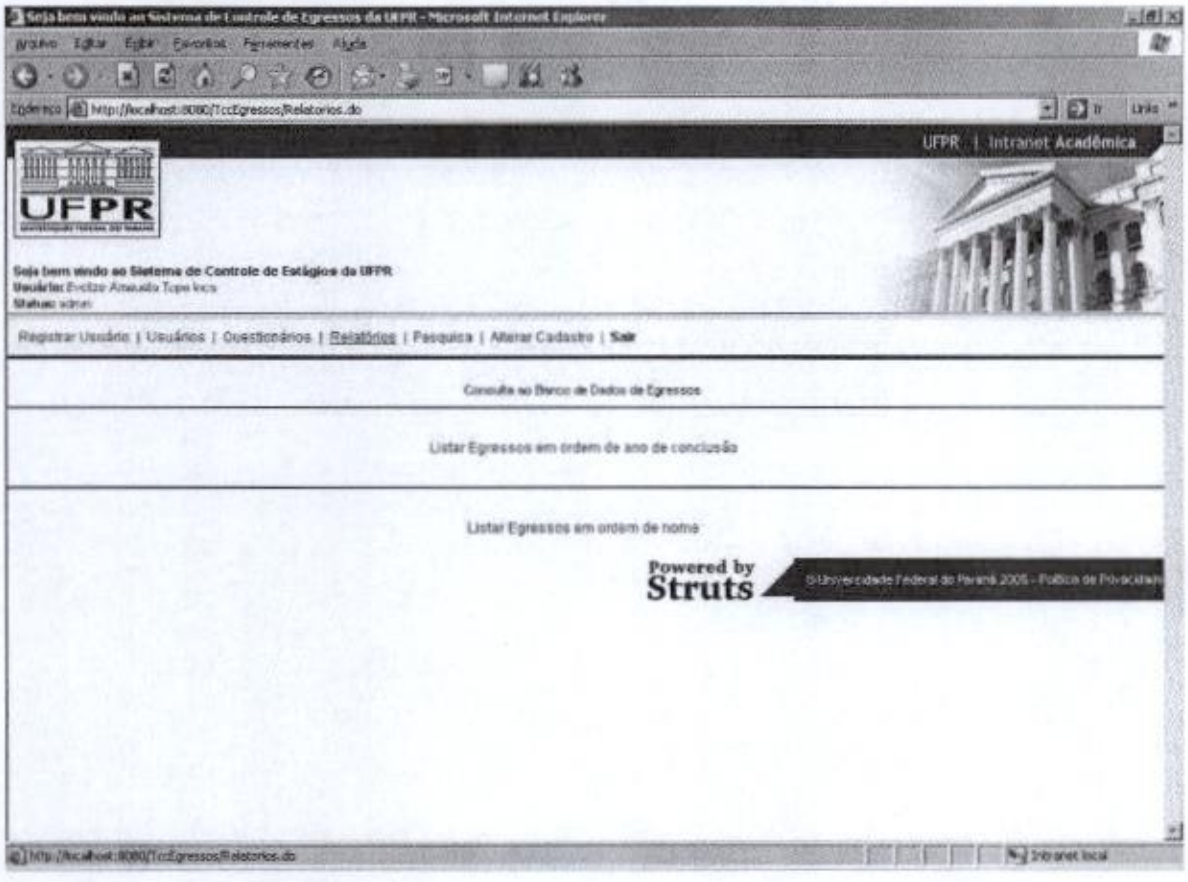
6.2.5.2.13 Fluxo Questionários – Inserir Nova Questão



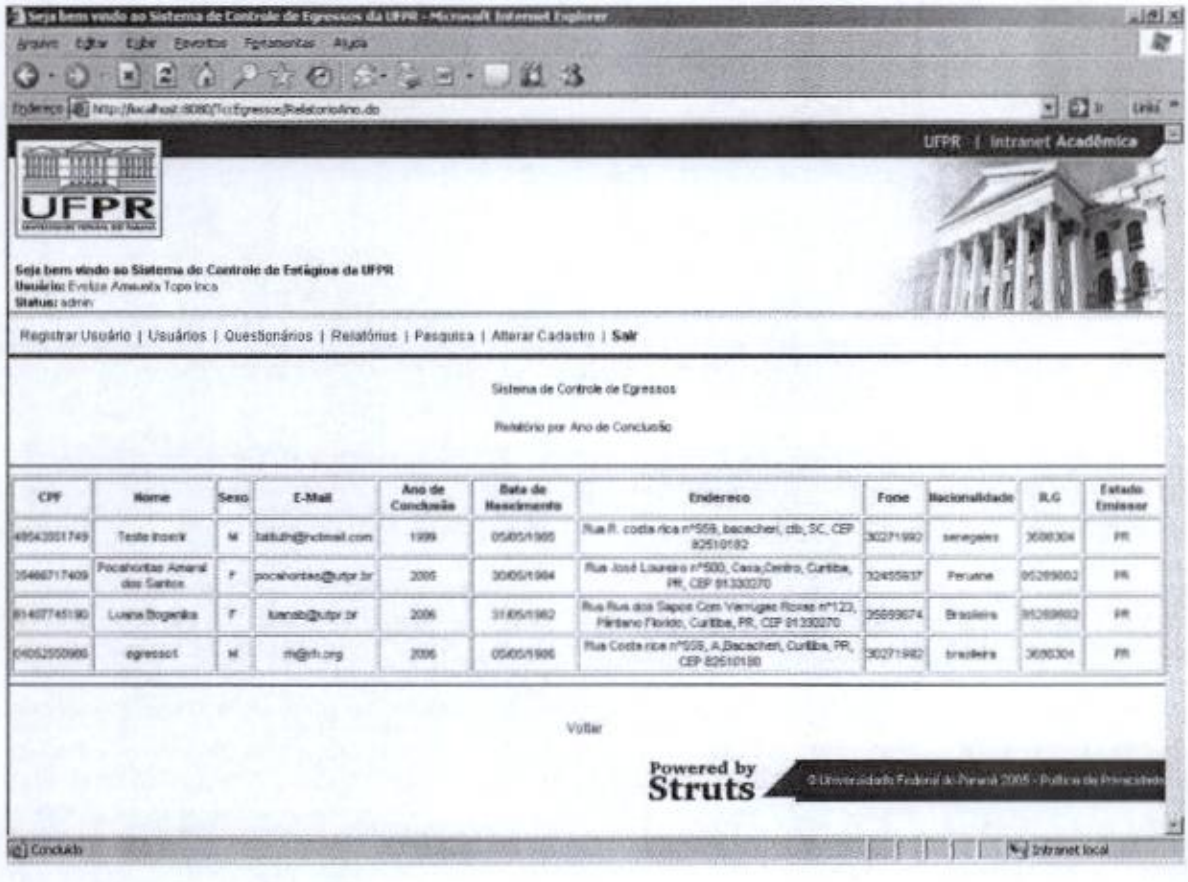
6.2.5.2.14 Fluxo Questionários – Visualizando Estatísticas



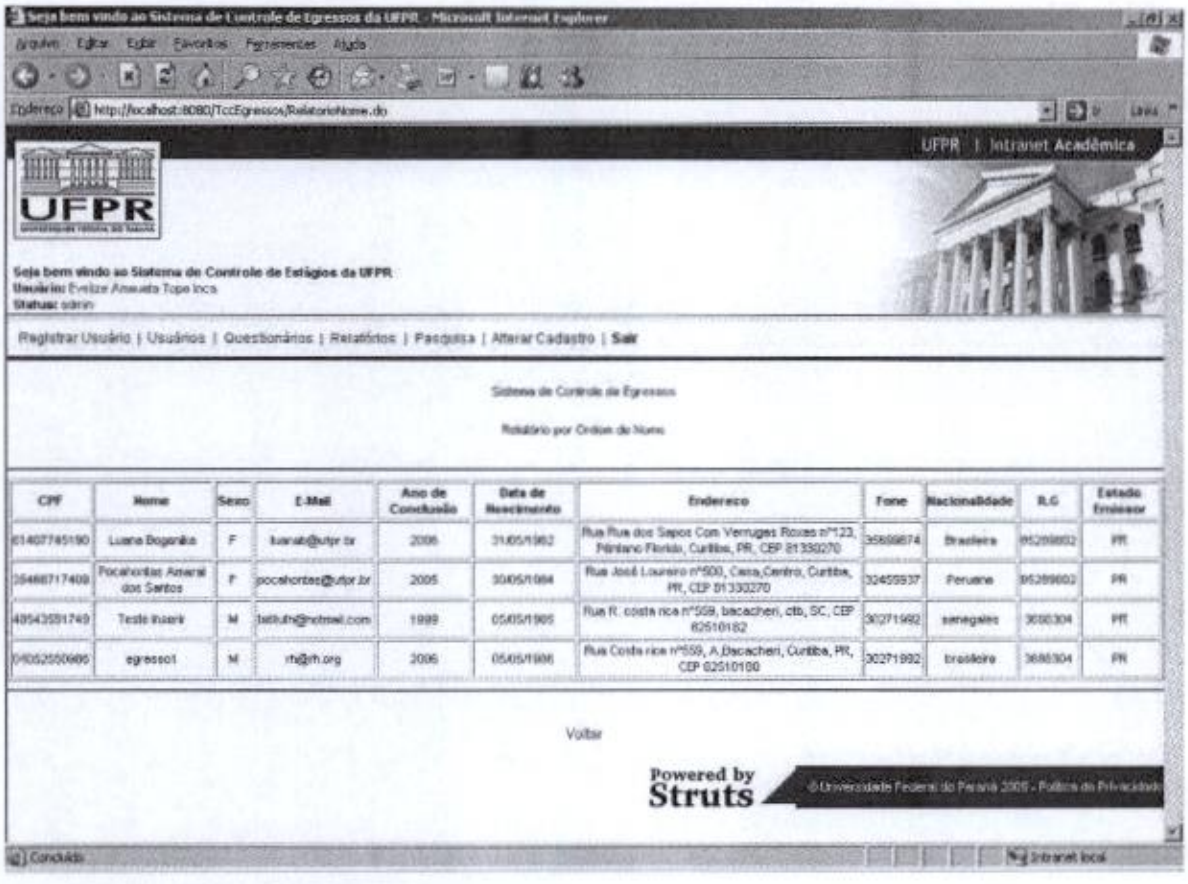
6.2.5.2.15 Fluxo Relatórios – Seleciona Tipo de Relatório



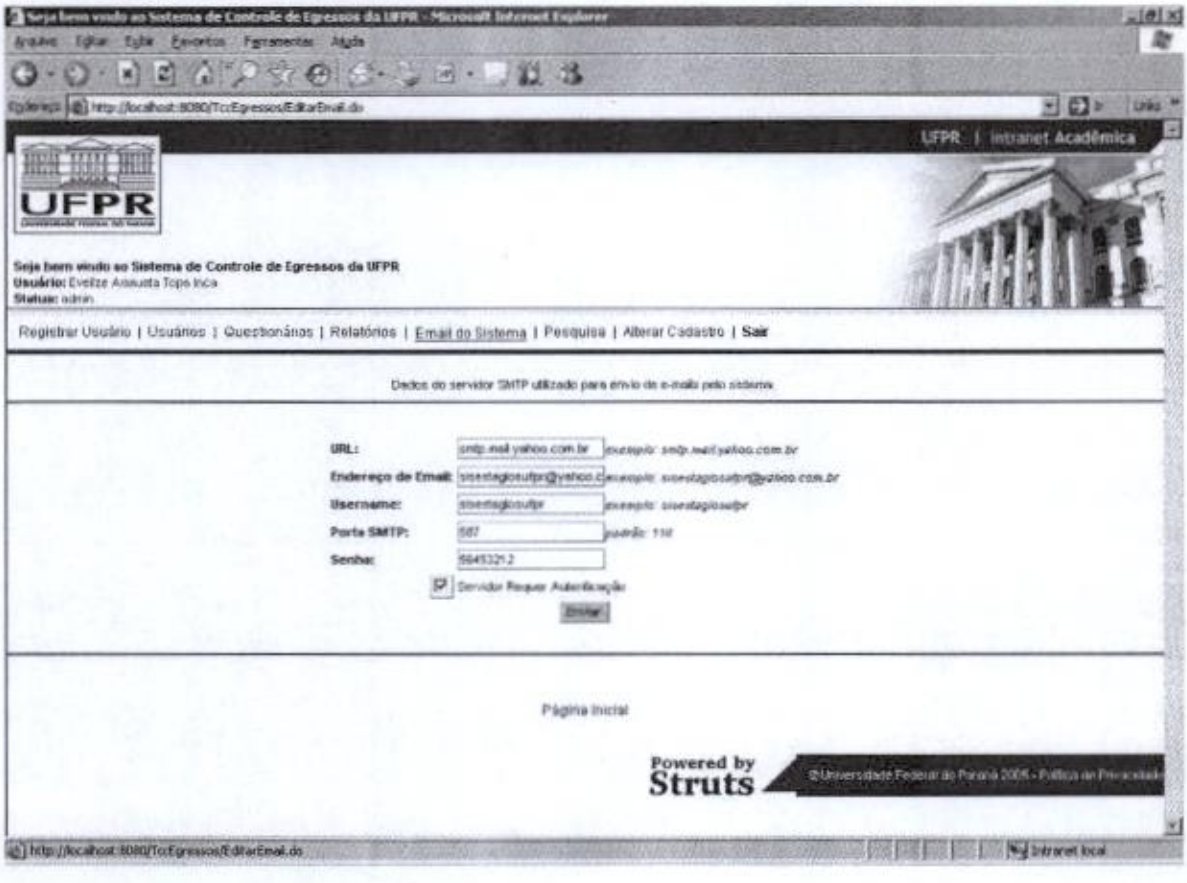
6.2.5.2.16 Fluxo Relatórios – Listando Egressos Por ano de conclusão



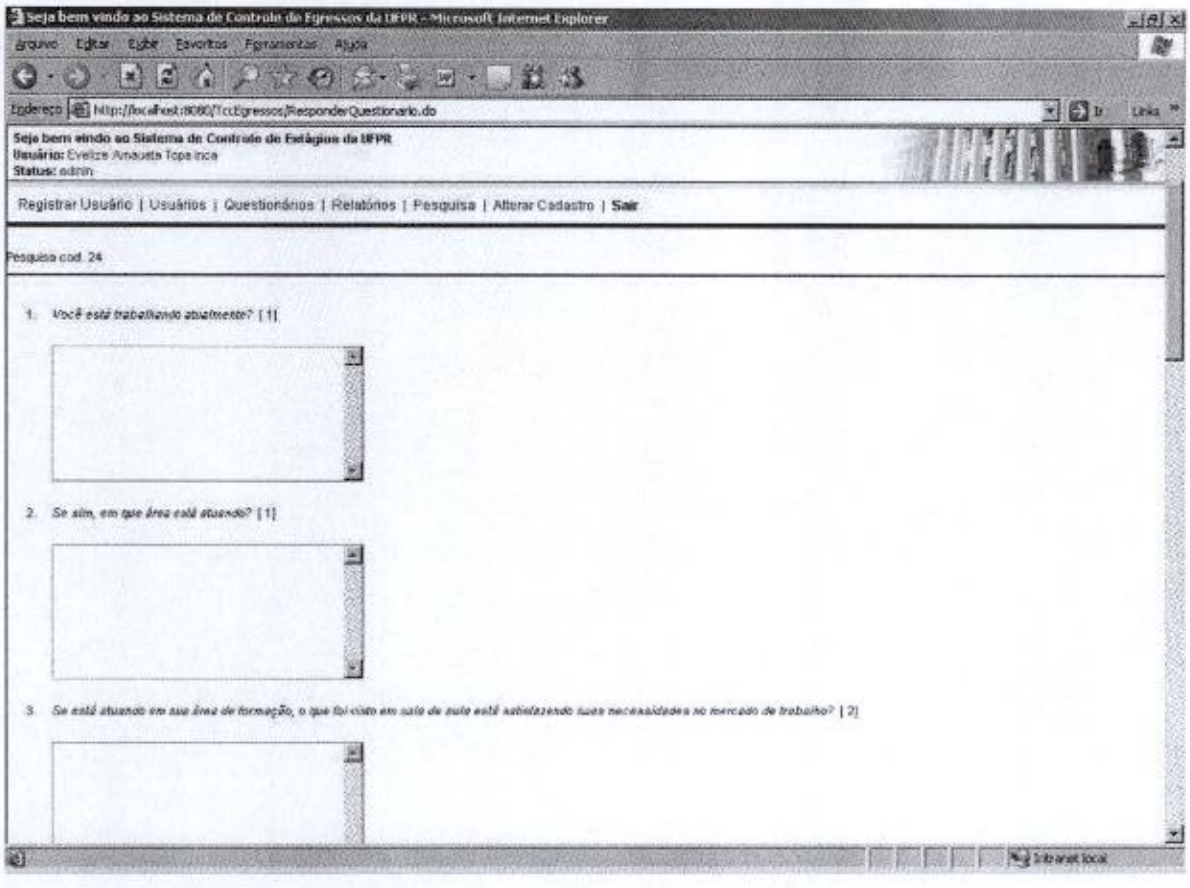
6.2.5.2.17 Fluxo Relatórios – Listando Egressos Por Nome



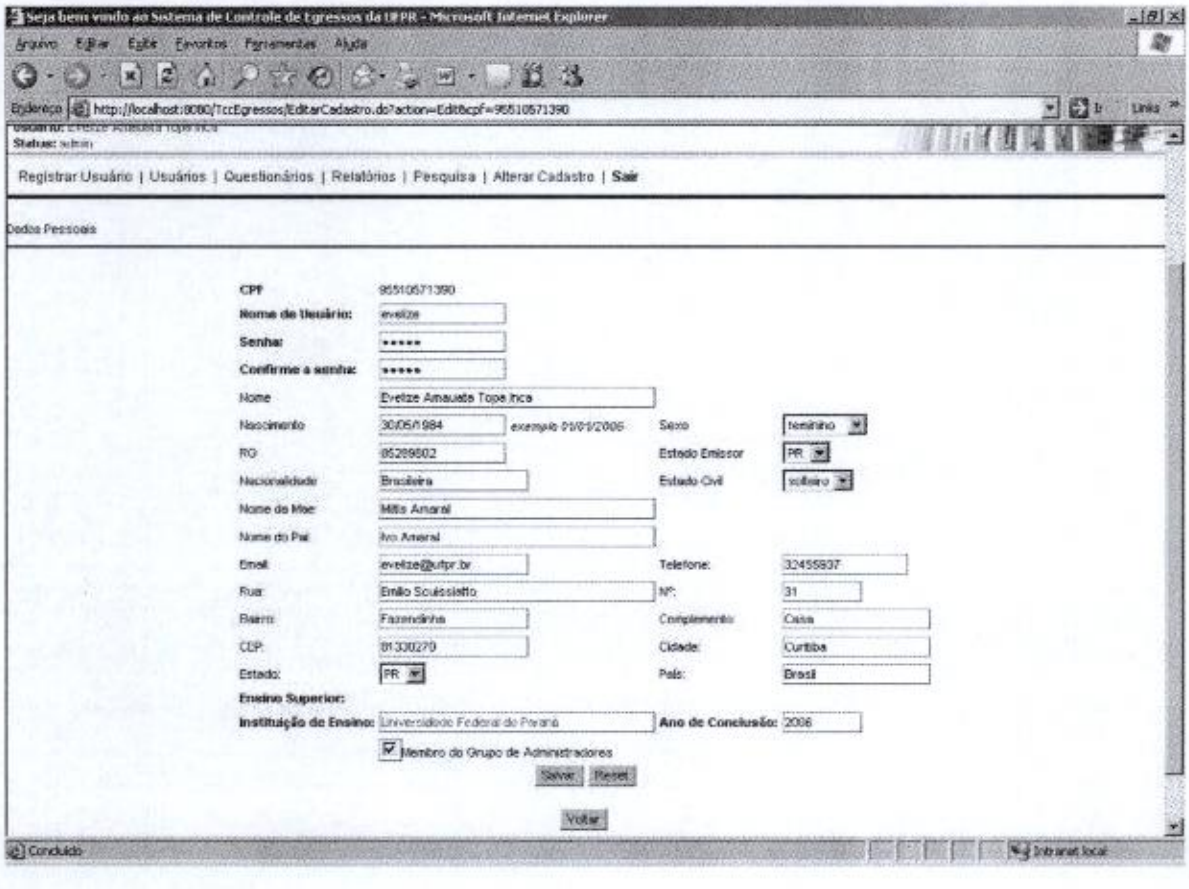
6.2.5.2.18 Fluxo Email do Sistema



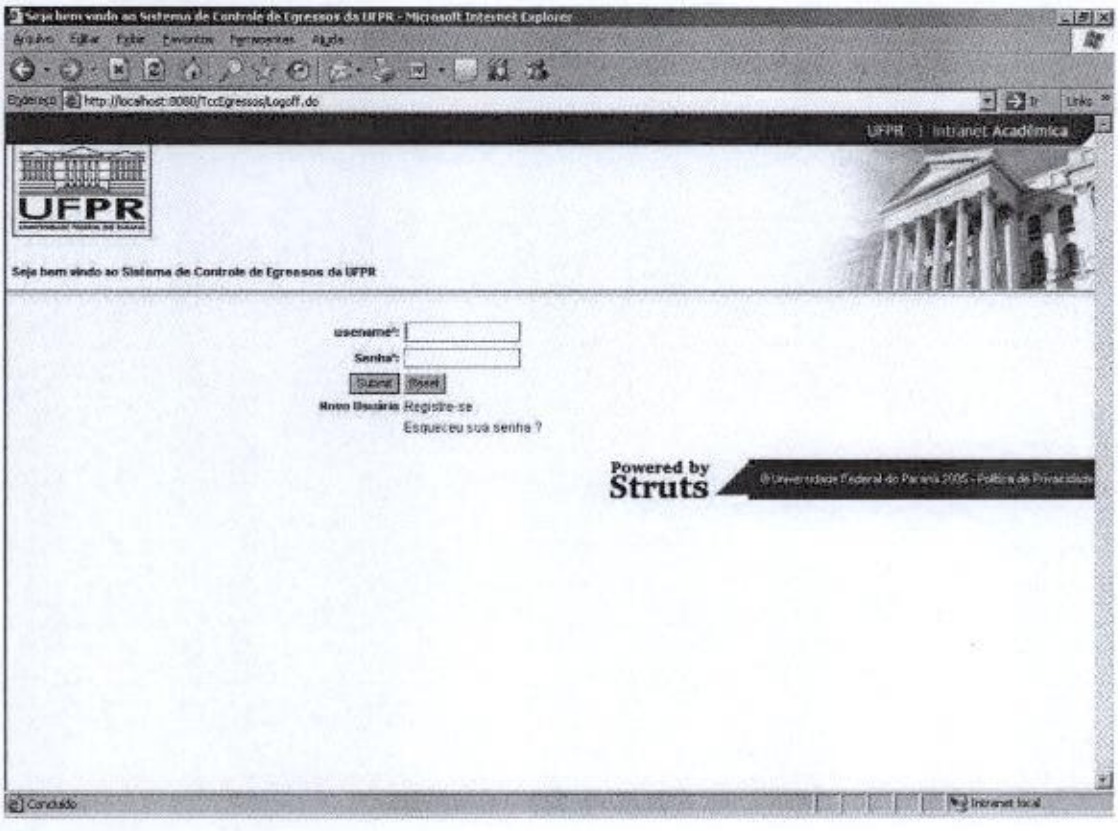
6.2.5.2.19 Fluxo Pesquisa – Visualizando



6.2.5.2.20 Fluxo Alterar Cadastro – Editando Dados Pessoais

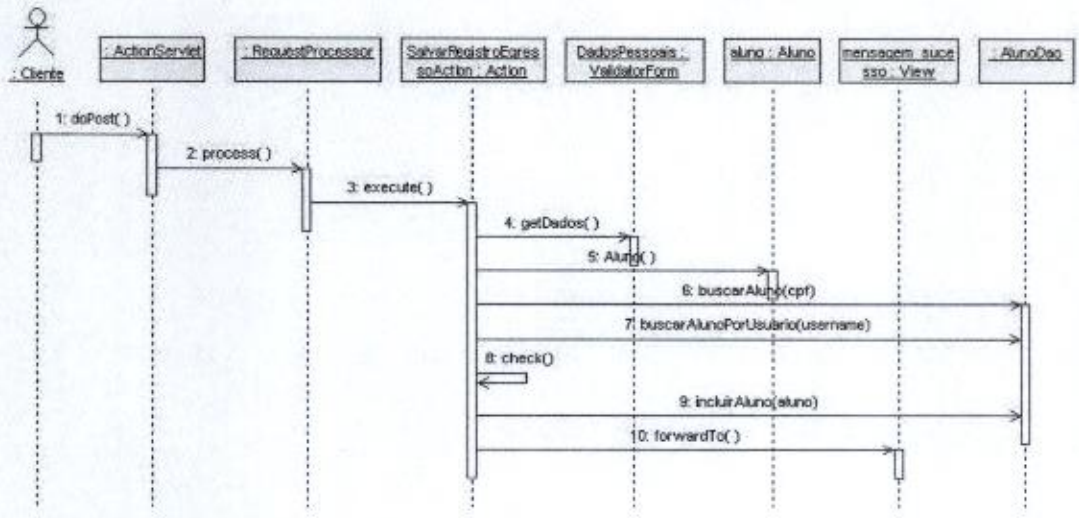


6.2.5.2.21 Sair

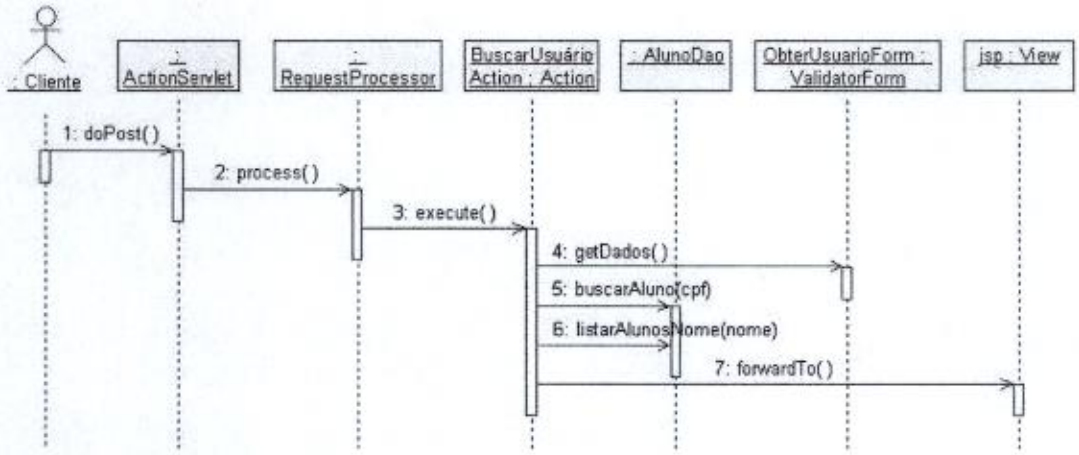

Seja bem vindo ao Sistema de Controle de Egressos da UFPR - Microsoft Internet Explorer
http://localhost:8080/TccEgressos/Logoff.do
UFPR | Intranet Acadêmica
UFPR
Seja bem vindo ao Sistema de Controle de Egressos da UFPR
username*:
Senha*:
Submit
Reset
Novo Usuário Registre-se
Esqueceu sua senha ?
Powered by
Struts
Concluído
Internet local

6.2.6 Diagramas de Sequência

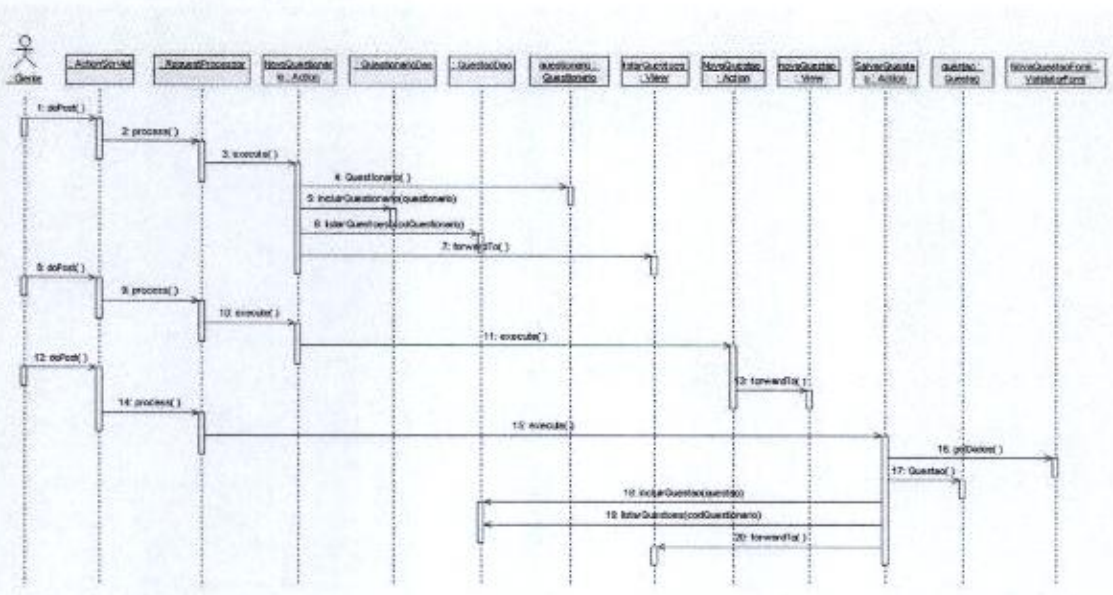
6.2.6.1 Inserir Usuário



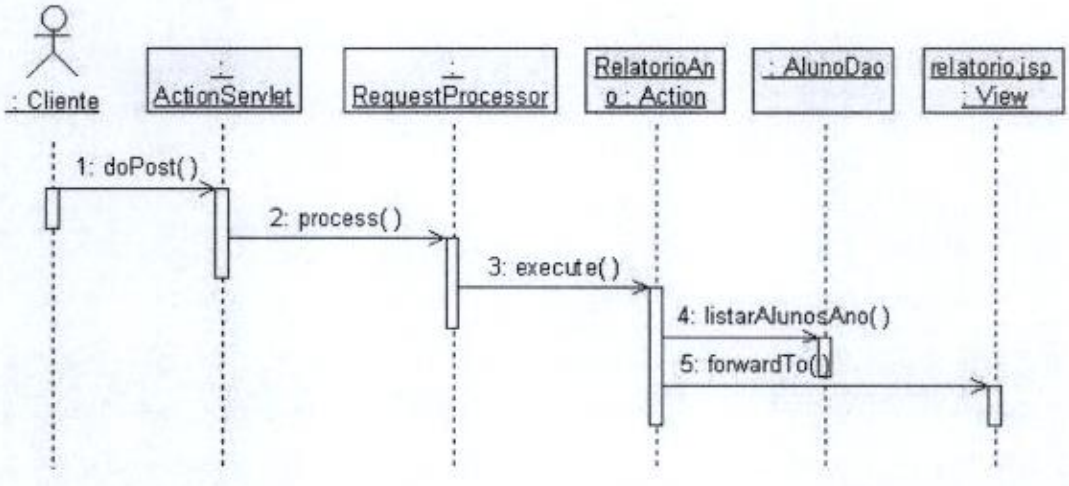
6.2.6.2 Buscar Usuários



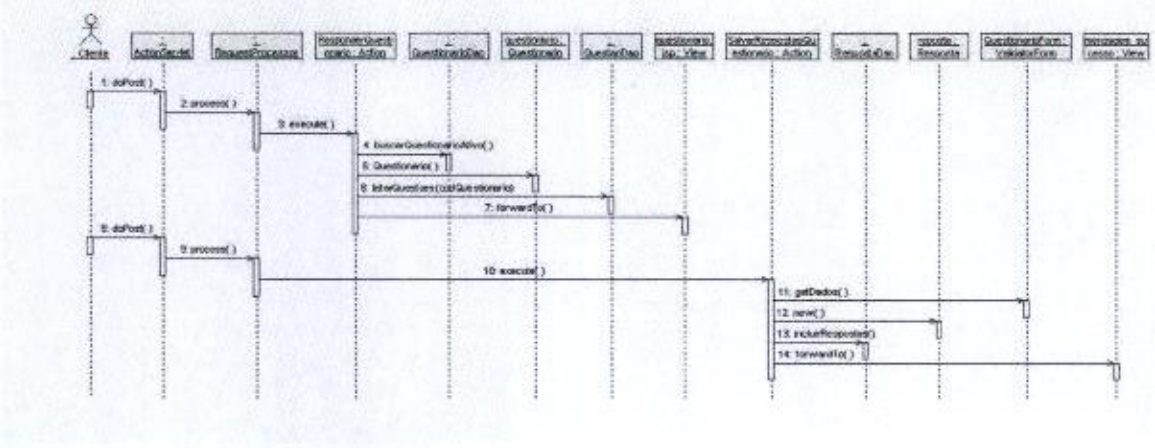
6.2.6.3 Inserir Questionário



6.2.6.4 Relatório

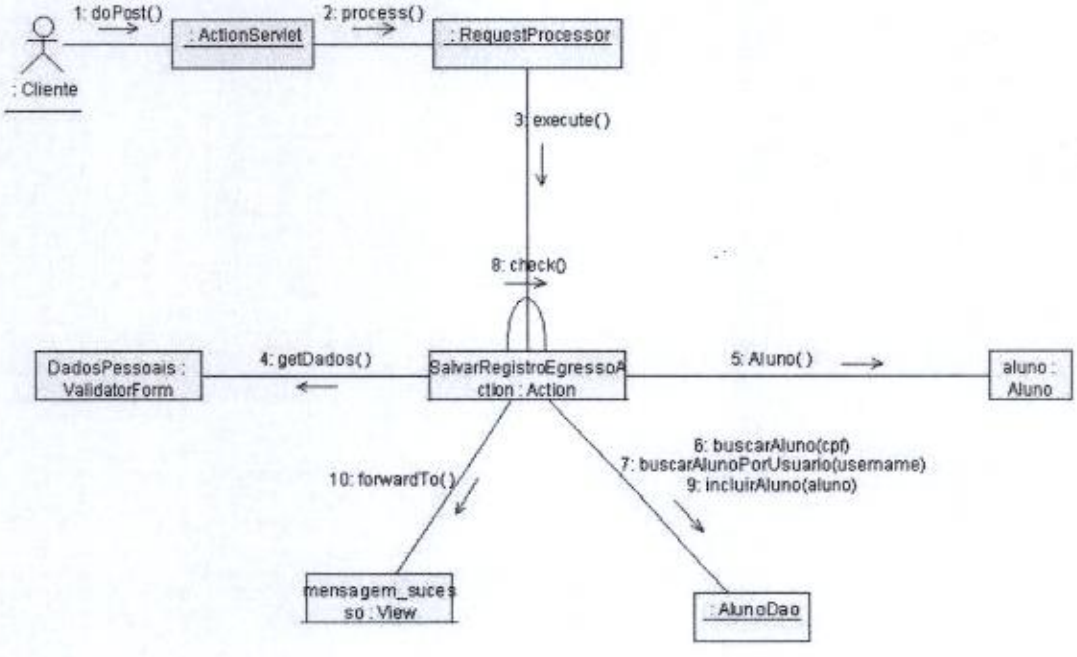


6.2.6.5 Responder Questionário

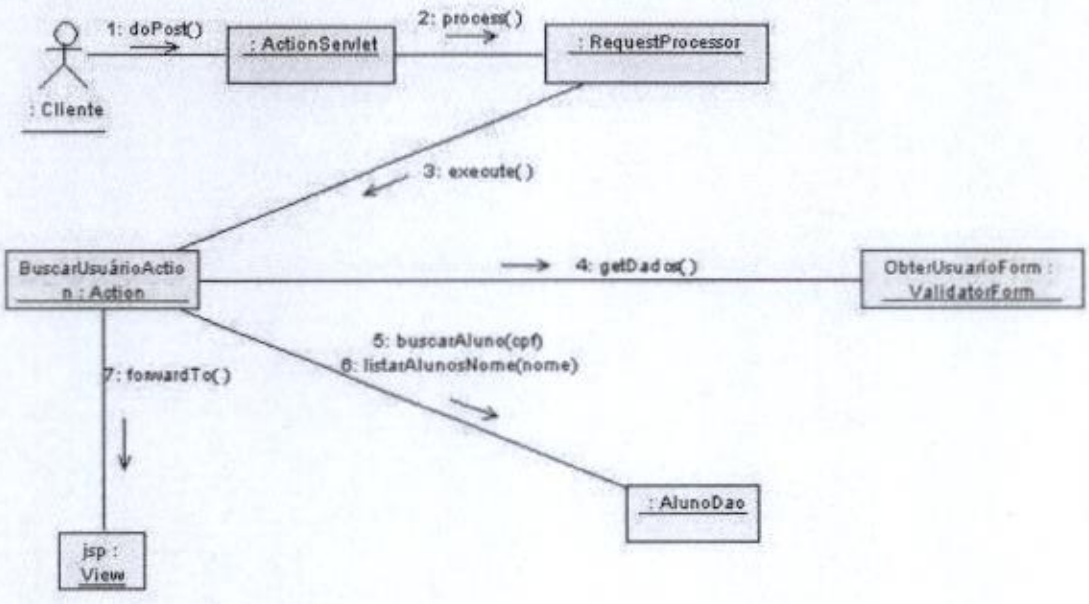


6.2.7 Diagramas de Colaboração

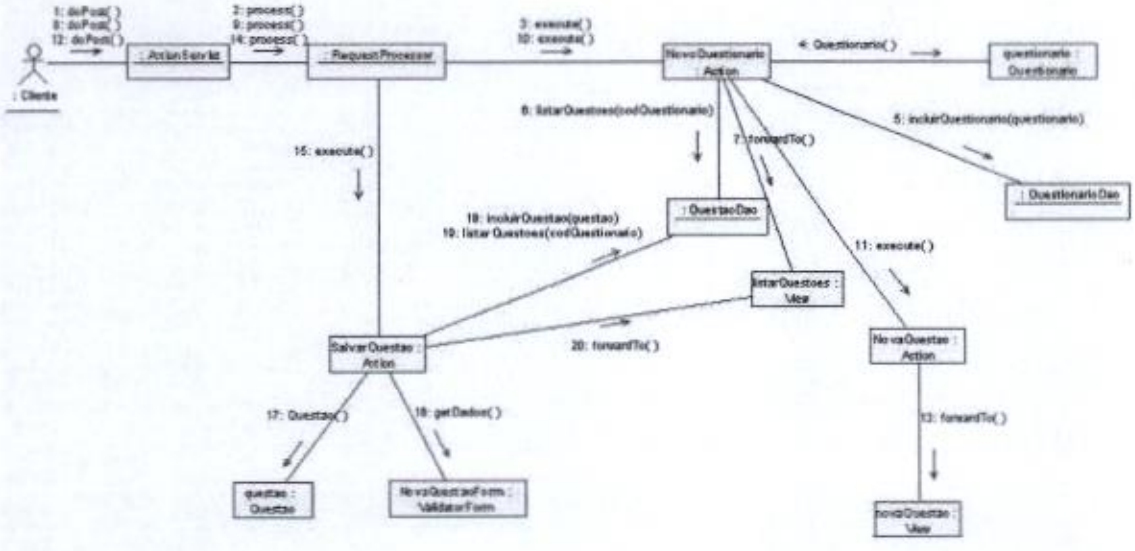
6.2.7.1 Inserir Usuário



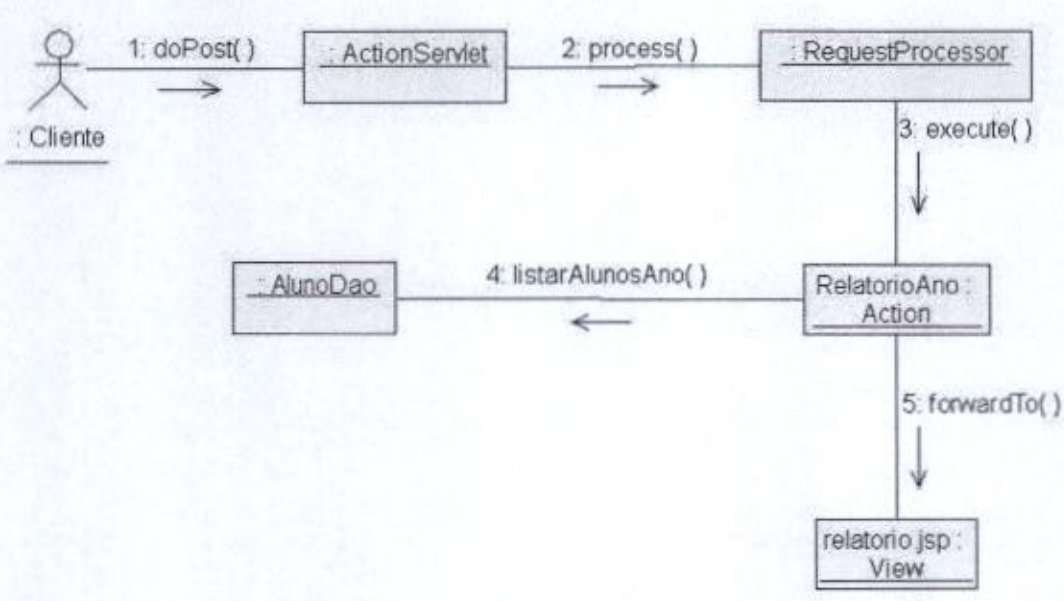
6.2.7.2 Buscar Usuário



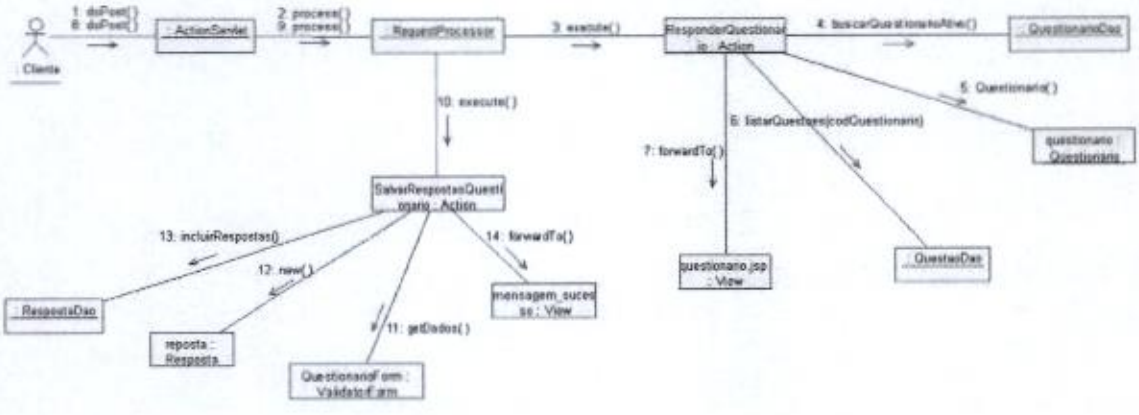
6.2.7.3 Inserir Questionário



6.2.7.4 Relatório



6.2.7.5 Responder Questionário



6.3 Apêndice 3

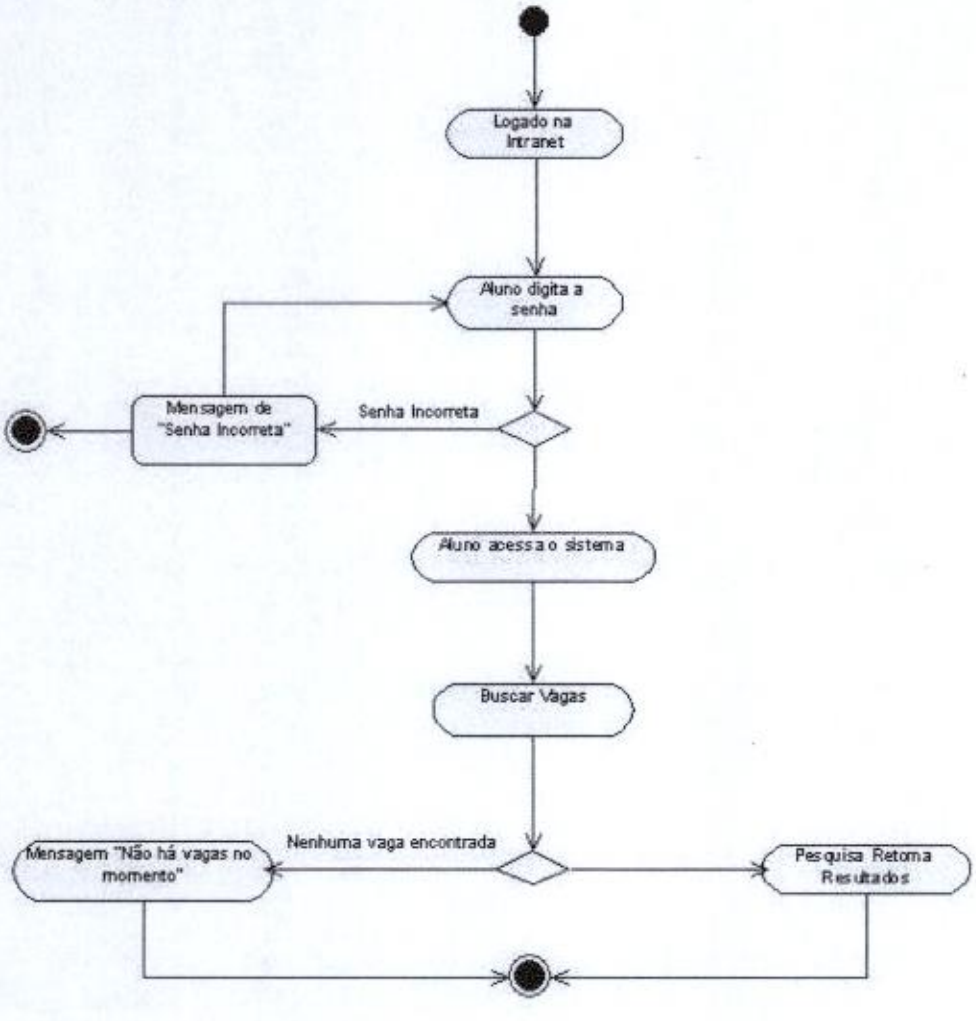
Modelagem de Dados

6.3.1 MÓDULO ESTÁGIO

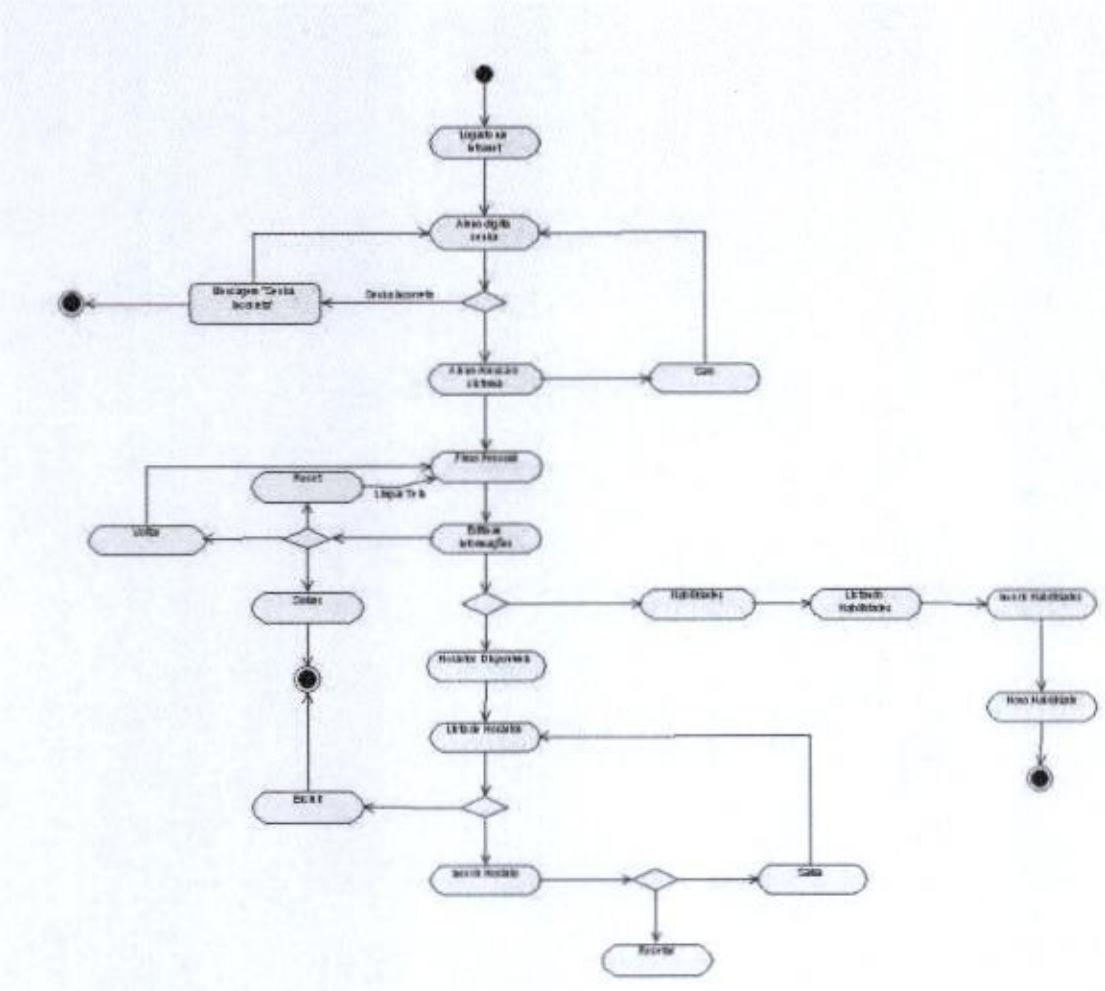
6.3.1.1 Diagramas de Atividade

6.3.1.1.1 Visão Aluno

6.3.1.1.1.1 Buscar Vagas

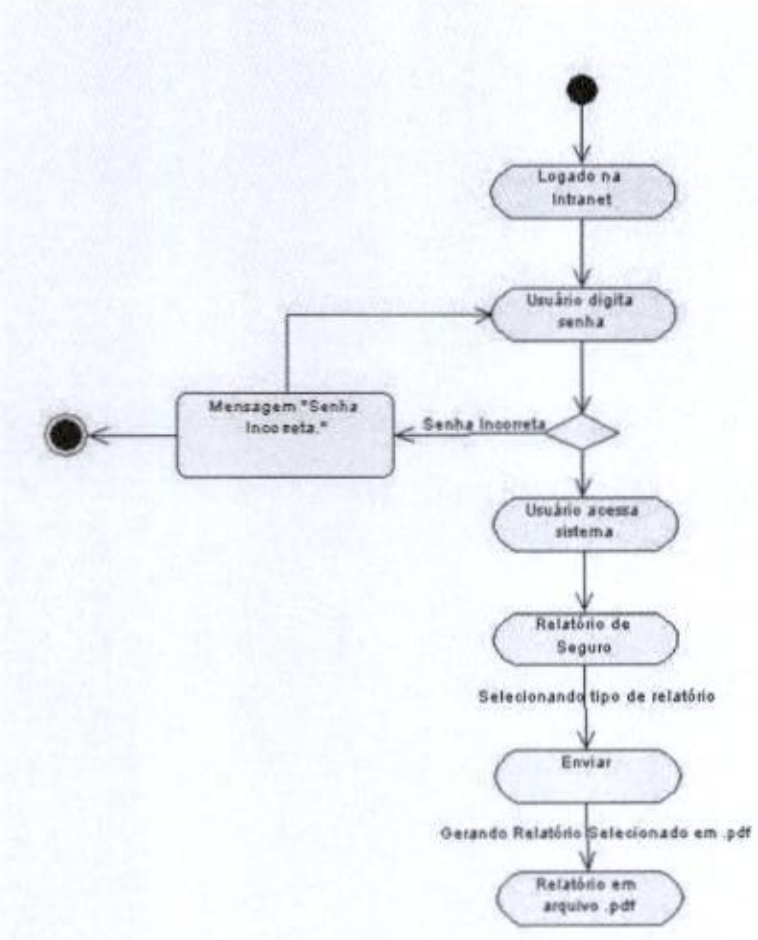


6.3.1.1.1.2 Pessoal

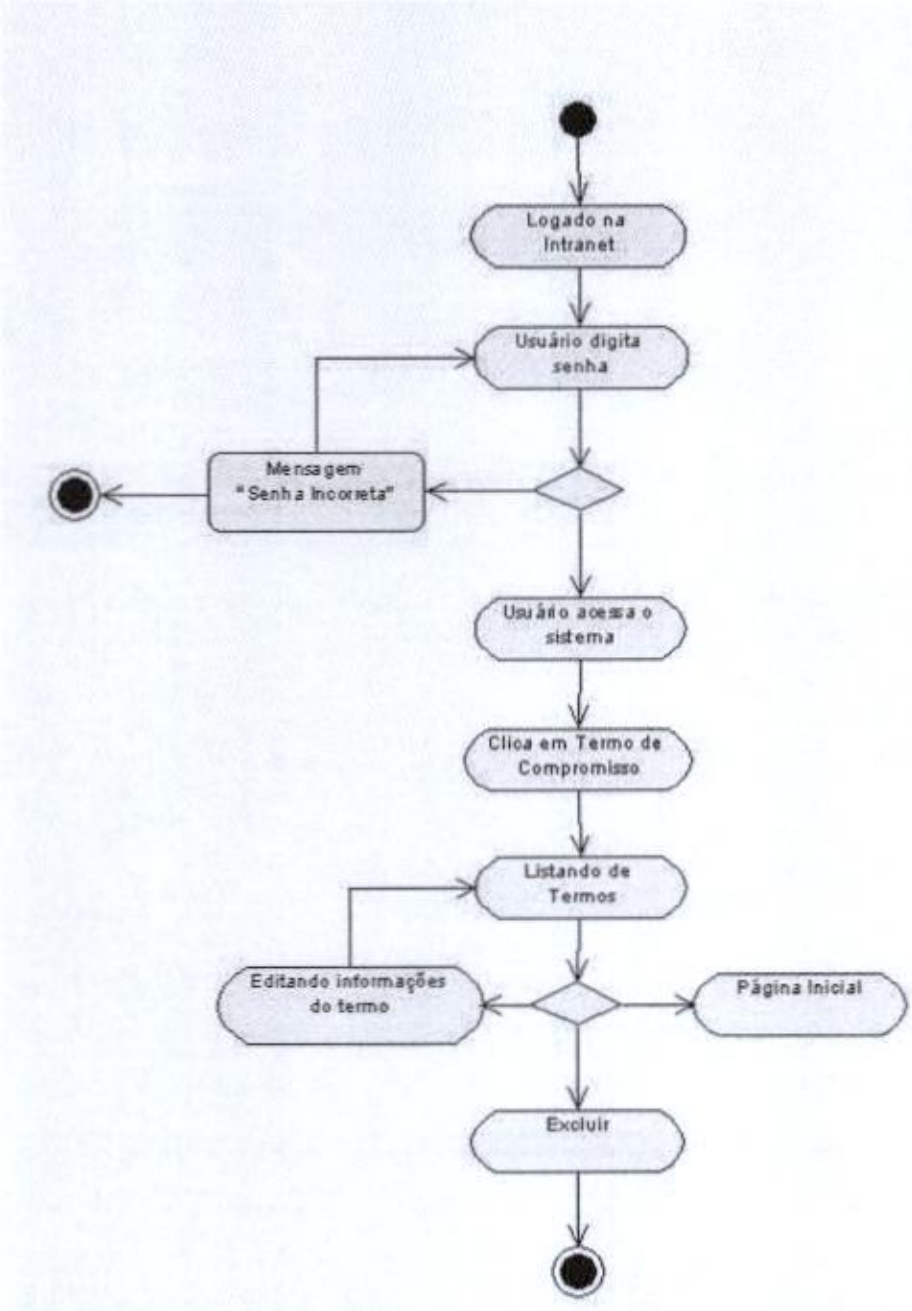


6.3.1.1.2 Visão Empresa

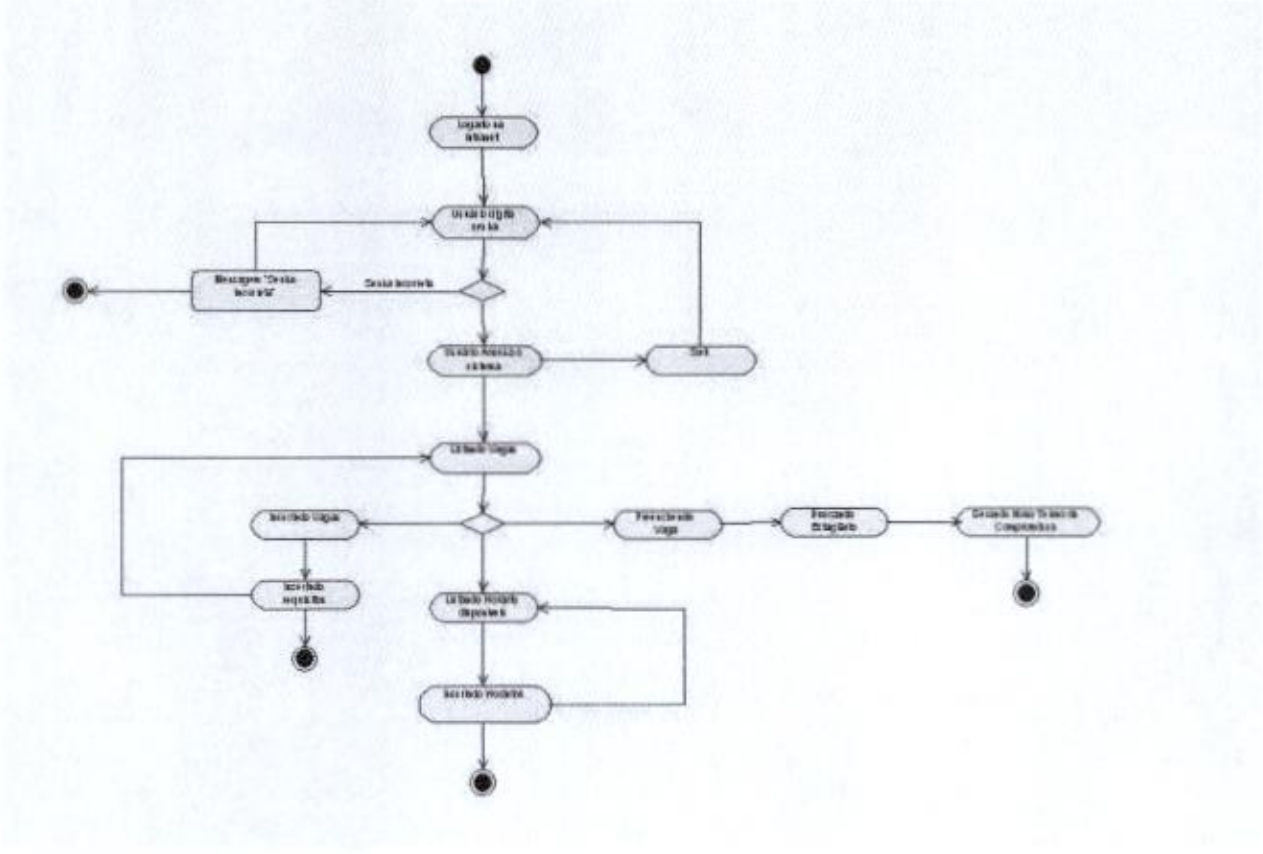
6.3.1.1.2.1 Relatório de Seguro



6.3.1.1.2.2 Termos de Compromisso

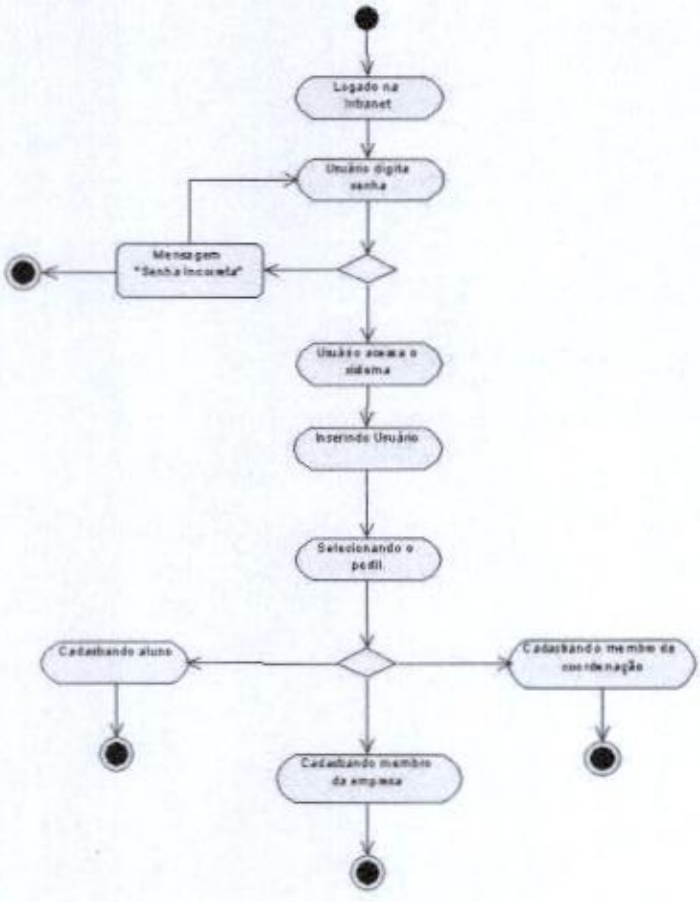


6.3.1.1.2.3 Vagas

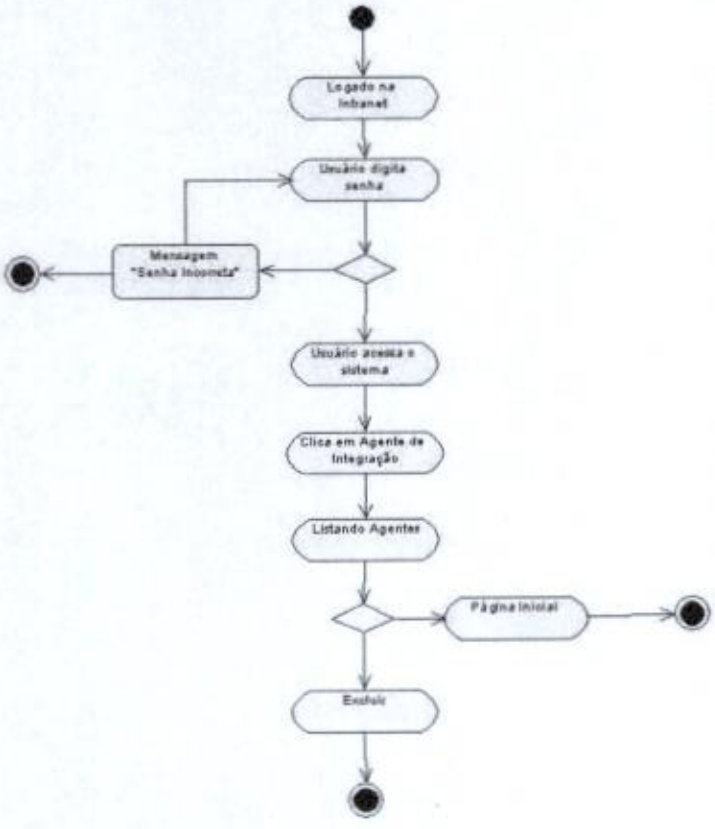


6.3.1.1.3 Visão Coordenação de Estágios

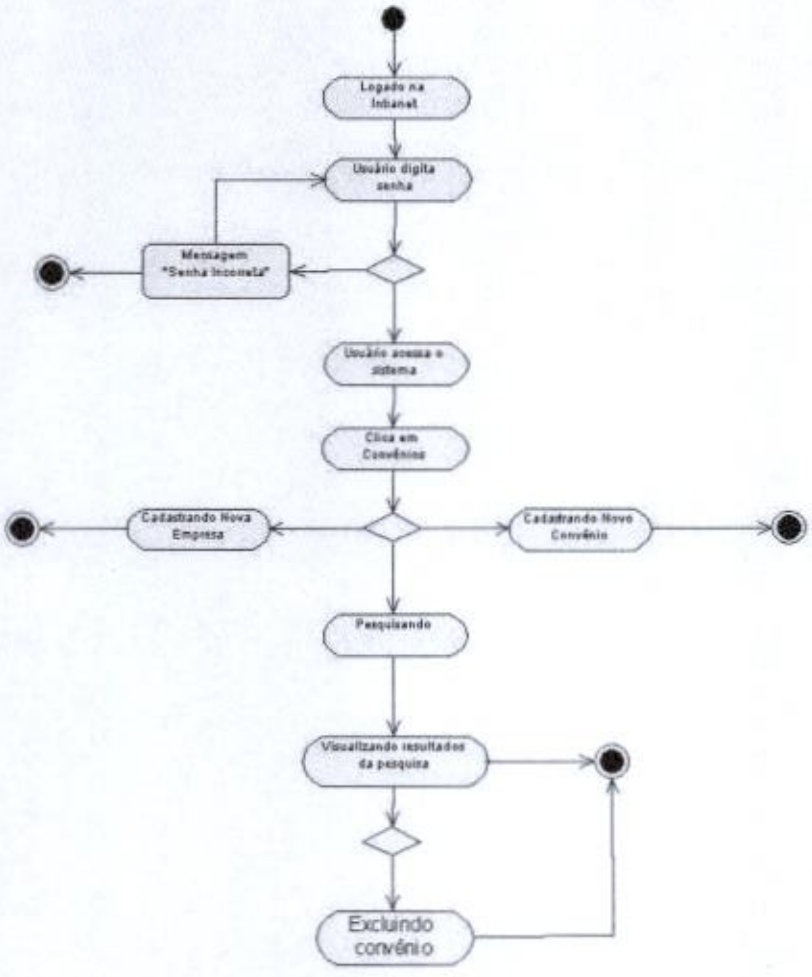
6.3.1.1.3.1 Inserir Usuário



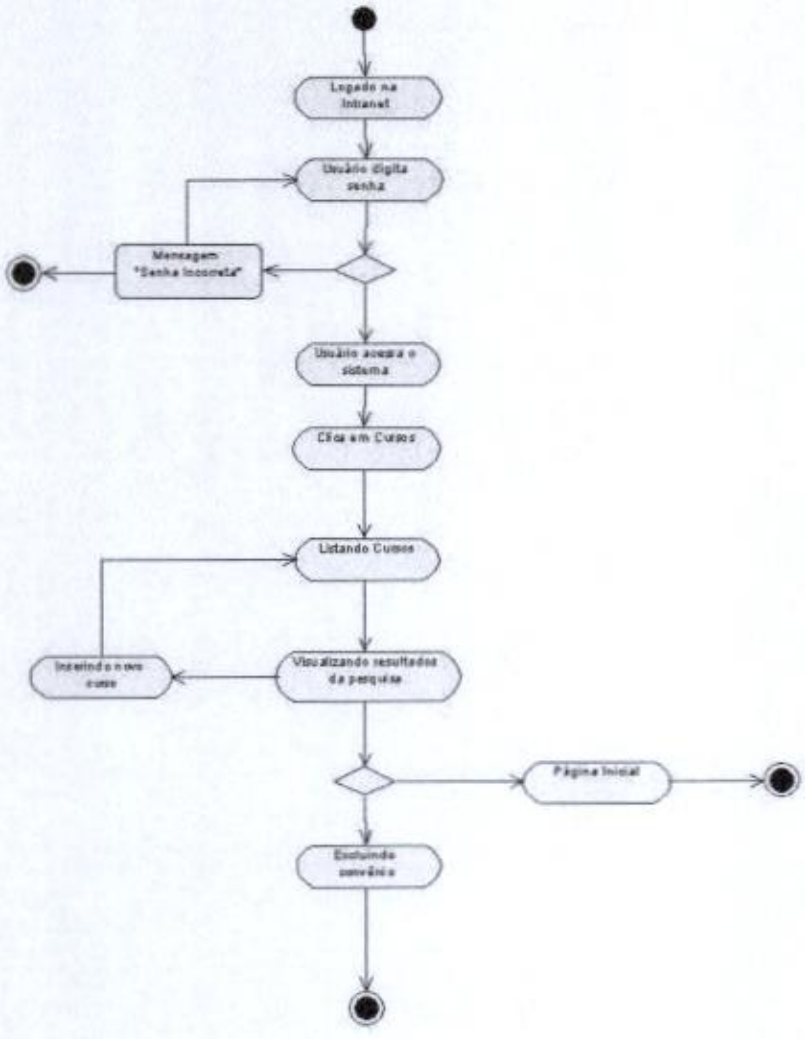
6.3.1.1.1.2 Agentes de Integração



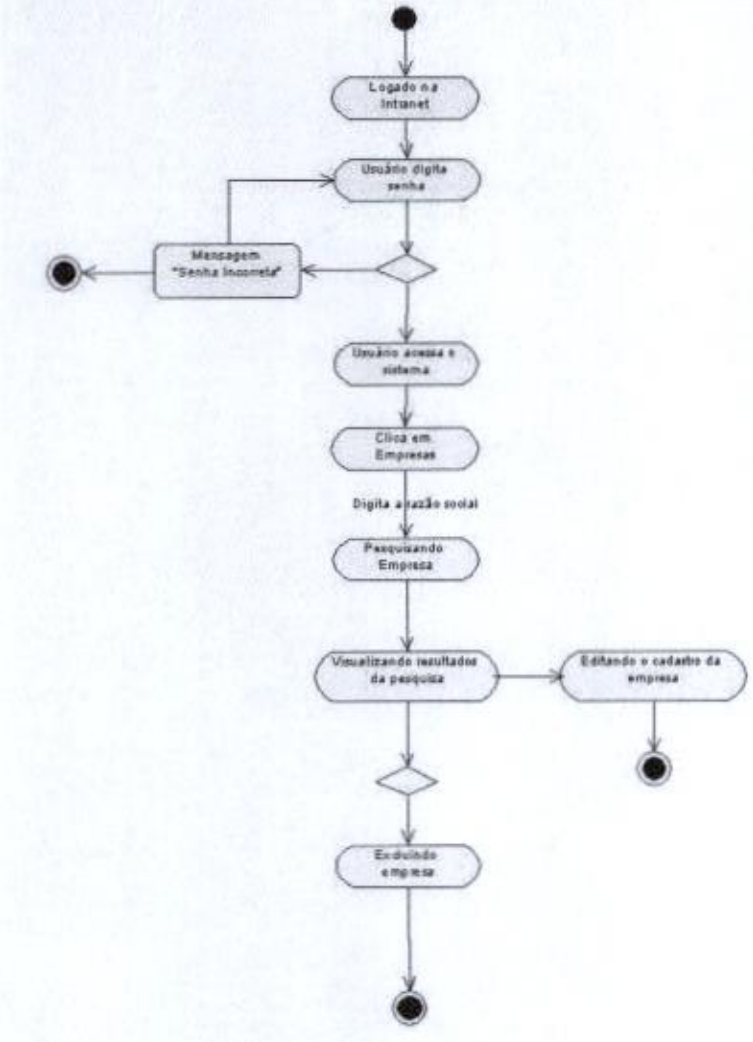
6.3.1.1.1.3 Convênios



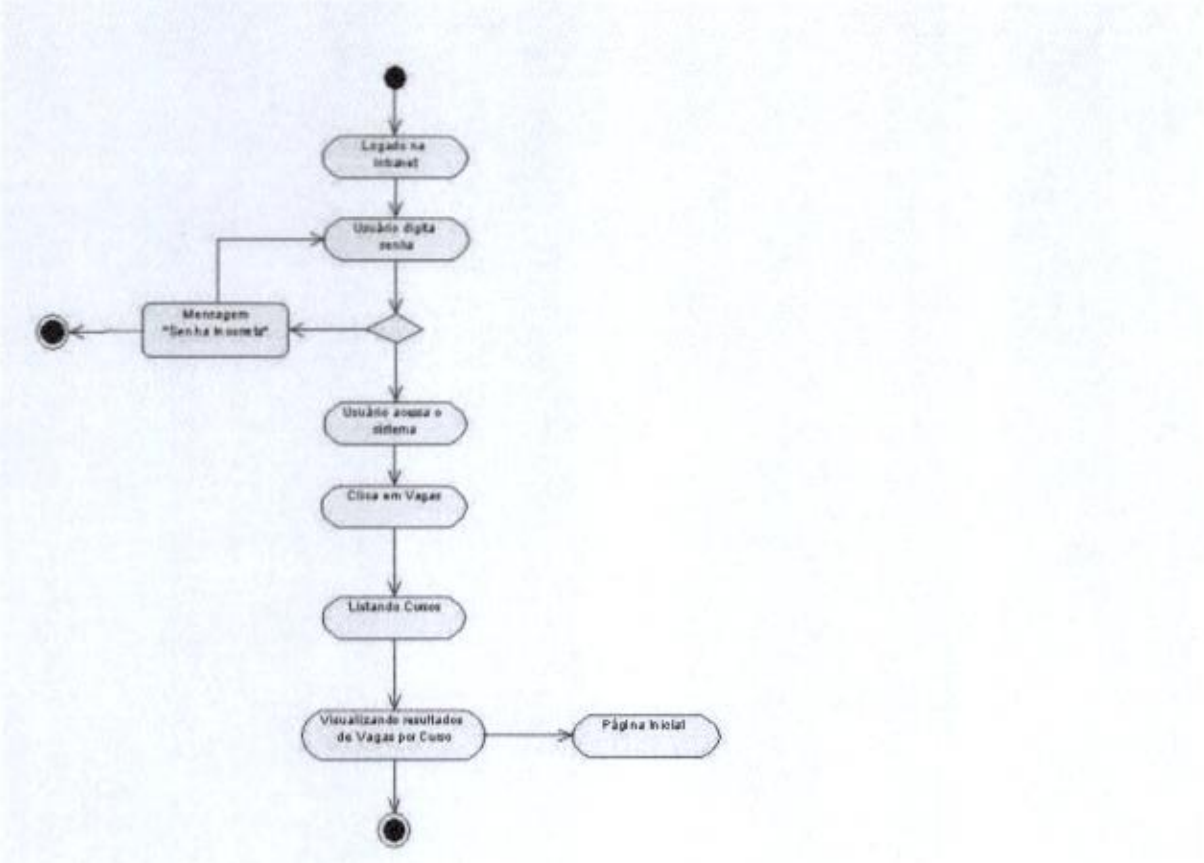
6.3.1.1.1.4 Cursos



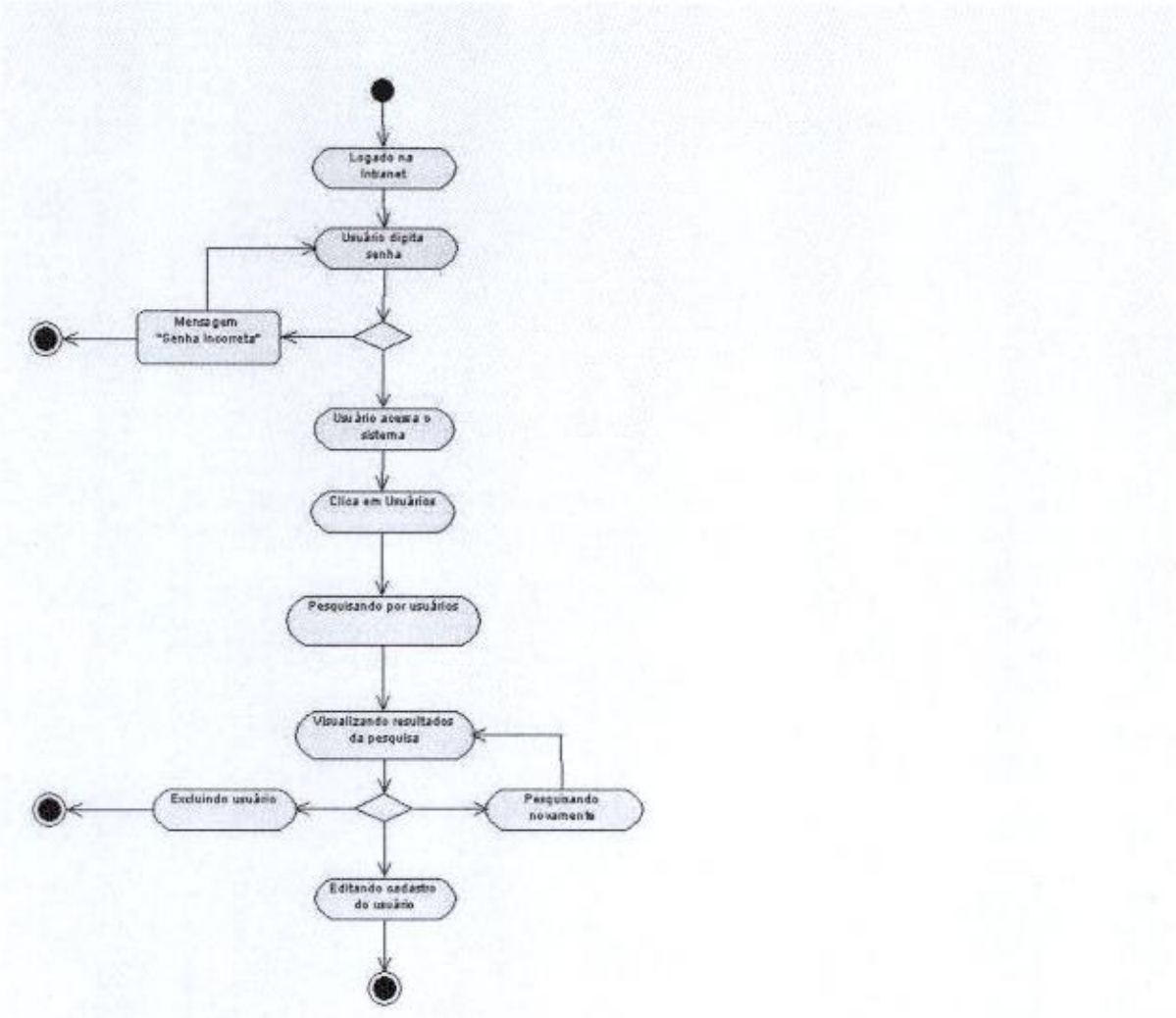
6.3.1.1.1.5 Empresas



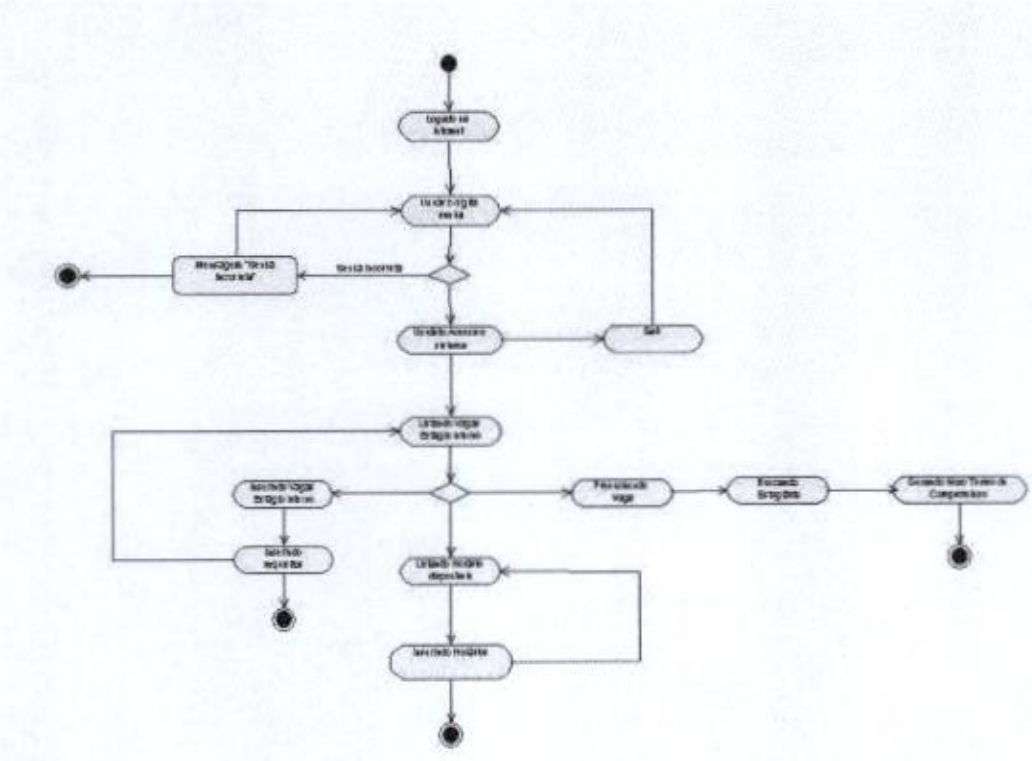
6.3.1.1.1.6 Vagas



6.3.1.1.1.7 Usuários

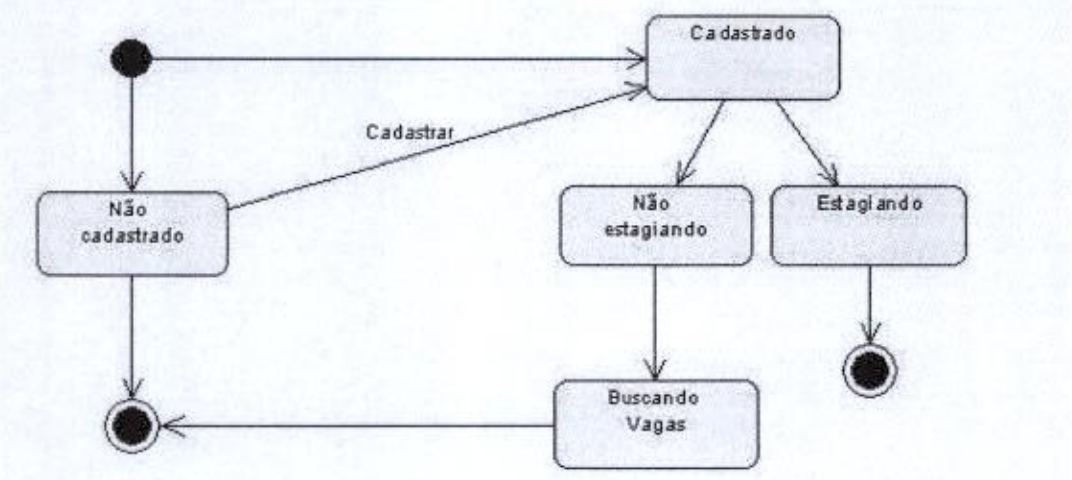


6.3.1.1.1.9 Estágio Interno

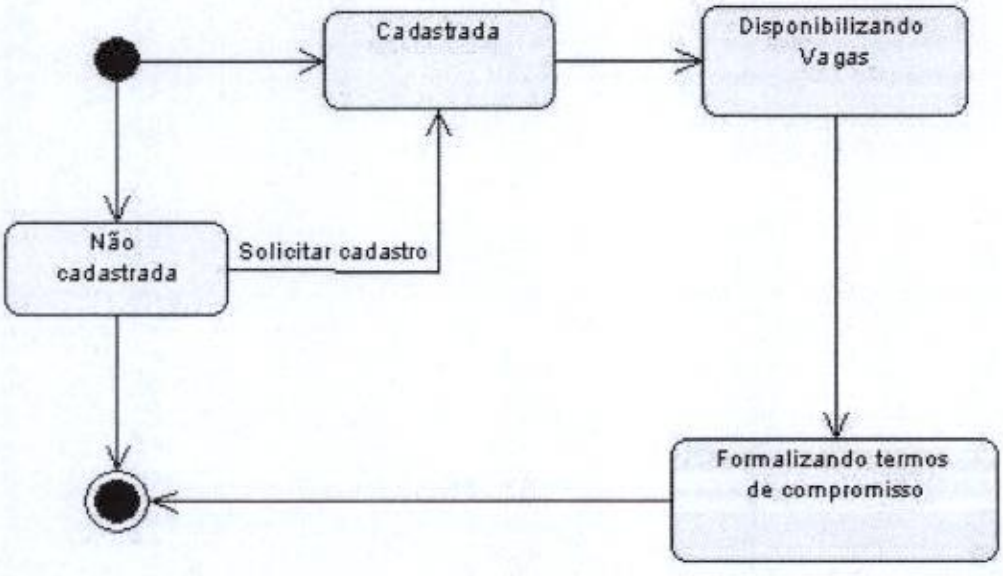


6.3.1.2 Diagramas de Estados

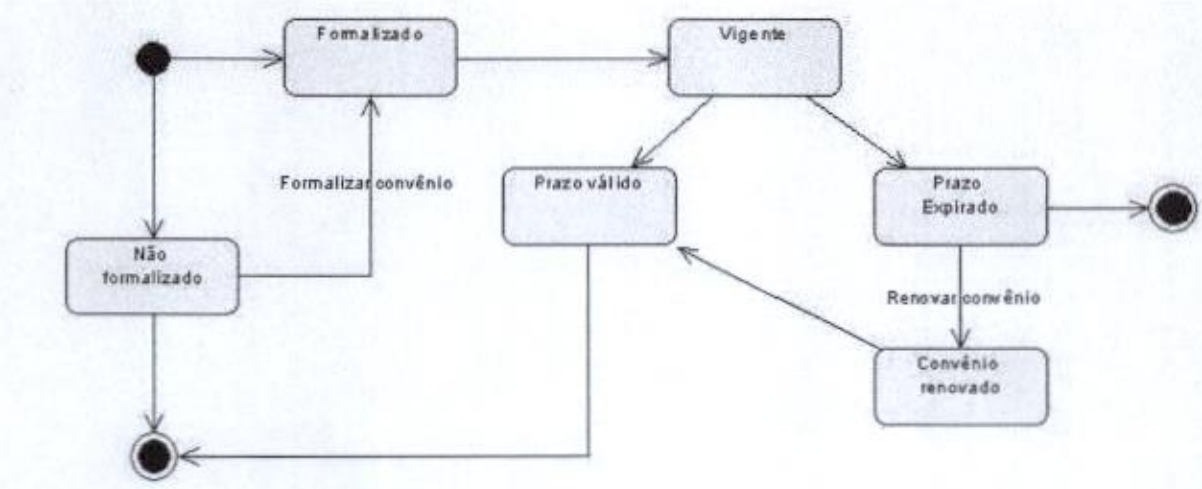
6.3.1.2.1 Classe Aluno



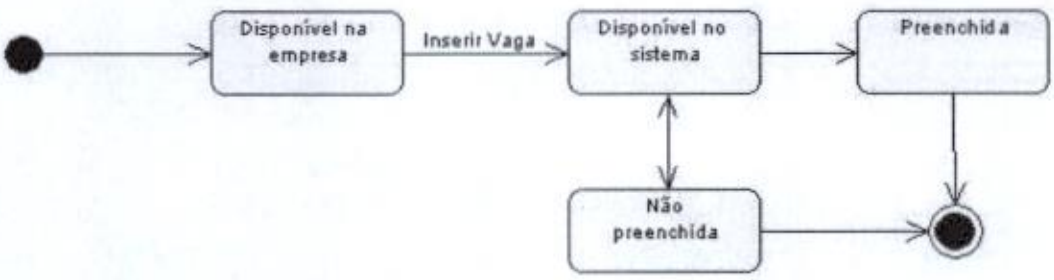
6.3.1.2.2 Classe Empresa



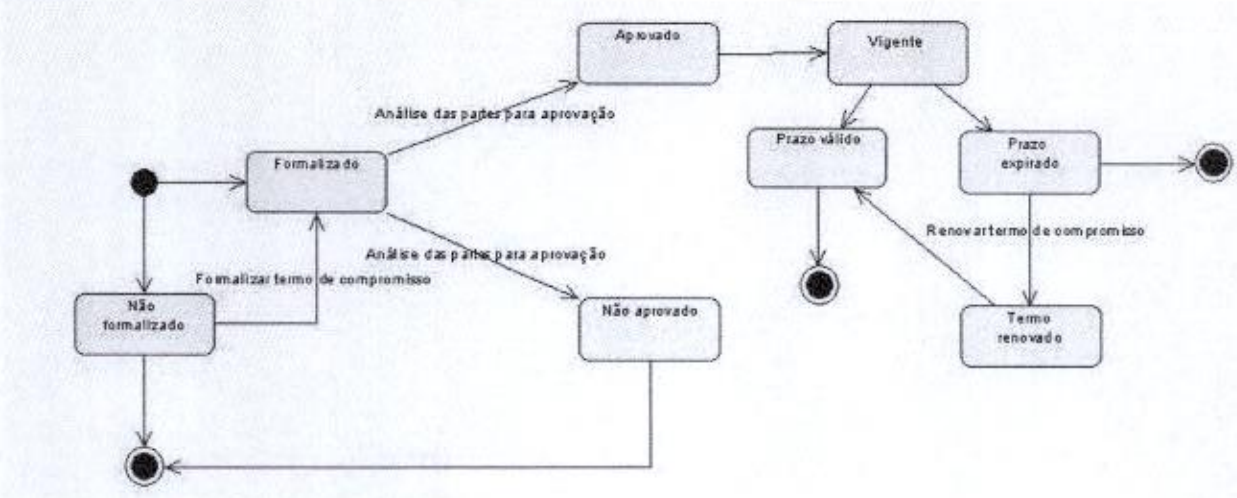
6.3.1.2.3 Classe Convênio



6.3.1.2.4 Classe Vagas



6.3.1.2.5 Classe Termo de Compromisso



6.3.1.3 Dicionário de Dados

6.3.1.3.1 Tabela: **AlunoEstágio**

Função: Armazenar e associar dados específicos do aluno.

Campo	Chave	Valor	Tipo (Tamanho)	Descrição
CodAlunoEstagio	PK	NOT NULL	int4	Código do aluno
pessoa_codpessoa	FK	NOT NULL	int4	Código pessoa
cursos_CodCurso	FK	NOT NULL	int4	Código do curso
disponivelFds			CHAR	Disponibilidade de estagiar nos finais de semana (S/N)
fumante			CHAR	Se o aluno é fumante (S/N)
deficiente			CHAR	Se o aluno possui alguma deficiência (S/N)
locomocao			VARCHAR	Meio de locomoção próprio que o aluno possui
sexo			CHAR	Sexo do aluno (M/F)
Email			VARCHAR	E-mail do aluno
fone			VARCHAR	Telefone do aluno
status			VARCHAR	Status do aluno no sistema
semestre			int4	Semestre em que o aluno está matriculado
rg			NUMERIC	Número do RG do aluno
rg_naturalidade			CHAR(2)	Naturalidade do aluno
cpf			int4	Número de CPF do aluno
matricula			NUMERIC	Número de matrícula do aluno
cep			VARCHAR(8)	CEP da residência do aluno
rua			VARCHAR()	Nome da rua onde reside o aluno
numero			int4	Número da residência do aluno
complemento			VARCHAR	Complemento do endereço
bairro			VARCHAR	Bairro da residência do aluno
municipio			VARCHAR	Município da residência do aluno
estado			CHAR(2)	Estado da residência do aluno

pais			VARCHAR	País da residência do aluno

6.3.1.3.2 Tabela: **Cursos**

Função: Armazenar e associar dados específicos do Curso.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
CodCurso	PK	NOT NULL	int4	Código do Curso
nomecurso			VARCHAR	Nome do Curso
duração			int4	Duração do Curso

6.3.1.3.3 Tabela: **Habilidades**

Função: **Armazenar todas as habilidades do sistema.**

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
CodHabilidade	PK	NOT NULL	int4	Código das Habilidades
nomeHabilidade			VARCHAR(20)	Nome da Habilidade

6.3.1.3.4 Tabela: **HabilidadesAluno**

Função: **Associar habilidades específicas do aluno.**

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
CodHabilidadeAluno	PK	NOT NULL	int4	Código das Habilidades do aluno
Habilidades_codHabilidade	FK	NOT NULL	int4	Código das Habilidades
AlunoEstagio_codAlunoEstagio	FK	NOT NULL	int4	Código do aluno

6.3.1.3.5 Tabela: **HabilidadesVaga**

Função: **Armazenar e associar habilidades específicas do aluno.**

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codhabilidadesvaga	PK	NOT NULL	int4	Código das Habilidades exigidas pela vaga
Habilidades_codHabilidade	FK	NOT NULL	int4	Código das Habilidades
Vagas_CodVaga	FK	NOT NULL	int4	Código da Vaga

6.3.1.3.6 Tabela: **Pessoa**

Função: **Armazenar todas as pessoas do sistema.**

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
Cod_pessoa	PK	NOT NULL	int4	Código da pessoa
nome_pessoa			VARCHAR	Nome da pessoa
nasc_pessoa			DATE	Data de nascimento da pessoa
usuario			VARCHAR	Nome usado pela pessoa para acesso ao sistema
senha			VARCHAR	Senha da pessoa para acesso ao sistema

6.3.1.3.7 Tabela: **TurnoDisponivel**

Função: Armazenar e associar os turnos disponíveis específicos do aluno.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codTurnoDisponivel	PK	NOT NULL	int4	Código do turno disponível
AlunoEstagio_codAlunoEstagio	FK	NOT NULL	int4	Código do aluno
turnoDisponivel			CHAR	Descrição do turno disponível
horainicio			TIME	Início da hora disponível
horafim			TIME	Final da hora disponível

6.3.1.3.8 Tabela: **planos_estagio**

Função: Armazenar e associar as informações sobre os planos de estágio.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codplanoestagio	PK	NOT NULL	int4	Código do plano de estágio
TermoCompromisso_codTermo	FK	NOT NULL	int4	Código do termo de compromisso
orientador			VARCHAR	Nome do orientador de estágio em caso de estágio voluntário.
formação_orientador			VARCHAR	Formação acadêmica do orientador em caso de estágio voluntário.
area_atividade_estagiario			VARCHAR	Área de atividade onde irá atuar o estagiário em caso de estágio voluntário.
objetivos			VARCHAR	Objetivos do estágio em caso de estágio voluntário.

atividades			TEXT	Atividade que serão exercidas pelo estagiário em caso de estágio voluntário.
cargo_orientador			VARCHAR	Cargo do orientador em caso de estágio interno.
ramal			VARCHAR	Ramal do departamento em que ocorrerá o estágio interno
departamento			VARCHAR	Departamento em que ocorrerá o estágio interno

6.3.1.3.9 Tabela: **Convenios**

Função: Armazenar e associar dados sobre os convênios entre as empresas e a UFPR.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codConvenio	PK	NOT NULL	int4	Código do Convênio
Empresa_CNPJ	FK	NOT NULL	VARCHAR	CNPJ da empresa
dataInicio			DATE	Data de início do Convênio
descricao			VARCHAR	Descrição do Convênio
dataFim			DATE	Data do fim do Convênio

6.3.1.3.10 Tabela: **Empresa**

Função: Armazenar e associar dados específicos da empresa.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
CNPJ	PK	NOT NULL	VARCHAR	CNPJ da empresa
razaoSocial			VARCHAR	Razão Social da empresa
ramo			VARCHAR	Ramo em que a empresa atua
site			VARCHAR	Web Site da empresa
email			VARCHAR	E-mail da empresa
nomeFantasia			VARCHAR	Nome fantasia da empresa

fone			VARCHAR	Telefone da empresa
representante			VARCHAR	Representante da empresa em relação aos estágios
cargoRepresentante			VARCHAR	Cargo que o representante na empresa
fax			VARCHAR	Fax da empresa
numprocesso			VARCHAR	Número do processo da empresa na UFPR.
agente_integracao			CHAR	Define se a empresa é um agente integrador ou não (S/N)

6.3.1.3.11 Tabela: **EnderecoEmpresa**

Função: Armazenar e associar dados específicos sobre o endereço da empresa.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codEnderecoEmpresa	PK	NOT NULL	int4	Código do endereço da empresa
Empresa_CNPJ	FK	NOT NULL	VARCHAR	CNPJ da empresa
cep			VARCHAR	Número do CEP
rua			VARCHAR	Nome da Rua
numero			int4	Número do prédio da empresa
complemento			VARCHAR	Complemento de endereço
bairro			VARCHAR	Bairro da empresa
municipio			VARCHAR	Munícipio da empresa
estado			VARCHAR	Estado da empresa
pais			VARCHAR	País da empresa

6.3.1.3.12 Tabela: **TermoCompromisso**

Função: Armazenar e associar dados específicos sobre o termo de compromisso.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codTermo	PK	NOT NULL	int4	Código do termo de compromisso
AlunoEstagio_ codAlunoEstagio	FK	NOT NULL	int4	Código do aluno
Vagas_codVaga	FK	NOT NULL	int4	Código da vaga
Empresa_CNPJ	FK	NOT NULL	VARCHAR	CNPJ da empresa
DataInicio			DATE	Data de início do termo de compromisso
DataFIm			DATE	Data do fim do termo de compromisso
Agente_integração_ CNPJ			VARCHAR	CNPJ do agente de integração
obrigatorio			CHAR	Se o estagio é obrigatório (S/N)
seguro_providenciado			VARCHAR	Entidade responsável por providenciar o seguro.
seguro_num_apoloce			NUMERIC	Número da apólice do seguro
Seguro_companhia			VARCHAR	Companhia Seguradora.

6.3.1.3.13 Tabela: **Turno**

Função: Armazenar e associar dados específicos sobre o turno da vaga oferecida pela empresa.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codTurno	PK	NOT NULL	int4	Código do turno
Vagas_codVaga	FK	NOT NULL	int4	Código da vaga
turno			CHAR	Turno (M/T/N)
horaInicio			TIME	Hora de início do estágio
horaFim			TIME	Hora do fim do estágio

6.3.1.3.14 Tabela: **Turnodisponivel**

Função: Armazenar e associar dados específicos sobre o turno que o estagiário está disponível.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codTurnoDisponivel	PK	NOT NULL	int4	Código do turno disponível
AlunoEstagio_ codAlunoEstagio	FK	NOT NULL	int4	Código do Aluno
turnoDisponivel			CHAR	Turno (M/T/N)
horaInicio			TIME	Hora de início do estágio
horaFim			TIME	Hora do fim do estágio

6.3.1.3.15 Tabela: **UsuarioEmpresa**

Função: Associar as empresas com seus respectivos usuários.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
Empresa_CNPJ	FK	NOT NULL	VARCHAR	CNPJ da empresa
AlunoEstagio_ codAlunoEstagio	FK	NOT NULL	int4	Código do Estagiário

6.3.1.3.16 Tabela: **Vagas_cursos**

Função: Associar a vaga aos cursos a que se destina.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
Codvagas_cursos	PK	NOT NULL	int4	Código Vagas Cursos
Cursos_codCurso	FK	NOT NULL	int4	Código do curso
Vagas_codVaga	FK	NOT NULL	int4	Código da Vaga

6.3.1.3.17 Tabela: **Vagas**

Função: Armazenar e associar dados específicos sobre a vaga oferecida pela empresa.

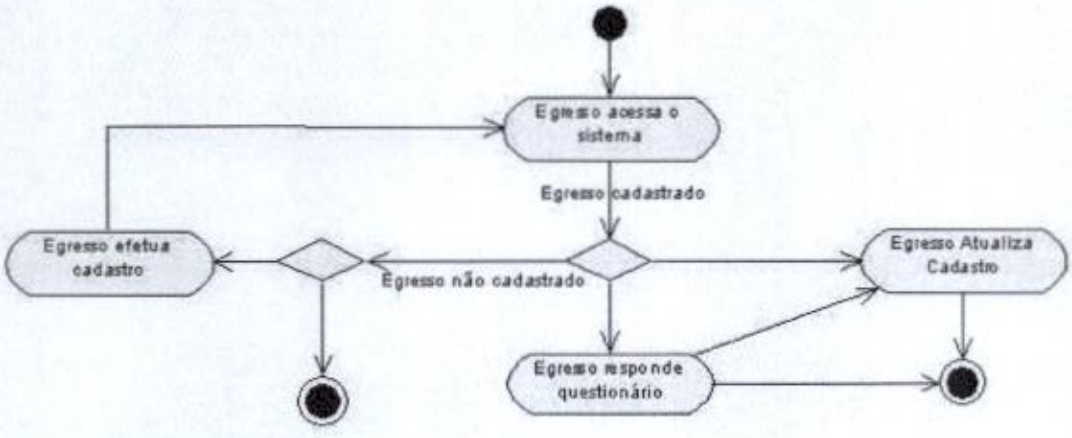
Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codVaga	PK	NOT NULL	int4	Código da vaga
Empresa_CNPJ	FK	NOT NULL	VARCHAR	CNPJ da empresa

setor			VARCHAR(20)	Setor da empresa onde a vaga está disponível
horasSemana			int4	Horas por semana de estágio
salario			VARCHAR	Salário oferecido, valor da hora, ou "à combinar".
sexo			CHAR	Sexo a que se destina a vaga
Inclusao			DATE	Data da inclusão da vaga no sistema
Preenchimento			DATE	Data do preenchimento da vaga
Fumante			CHAR	Se a vaga aceita estagiário fumante (S/N)
Deficiente			CHAR	Vaga para deficiente (S/N)
AtividadesFDS			CHAR	Vaga exige disponibilidade de finais de semana (S/N)
Locomoção			VARCHAR	Meio de locomoção exigido para a Vaga
Semestre			Int4	Semestre mínimo exigido para o estágio
vale_transporte			char	Se a empresa fornece vale transporte (S/N)
outros_beneficios			VARCHAR	Algum outro benefício que a empresa possa oferecer
estagio_interno			CHAR	Define a vaga como sendo para estágio interno na Universidade (S/N)

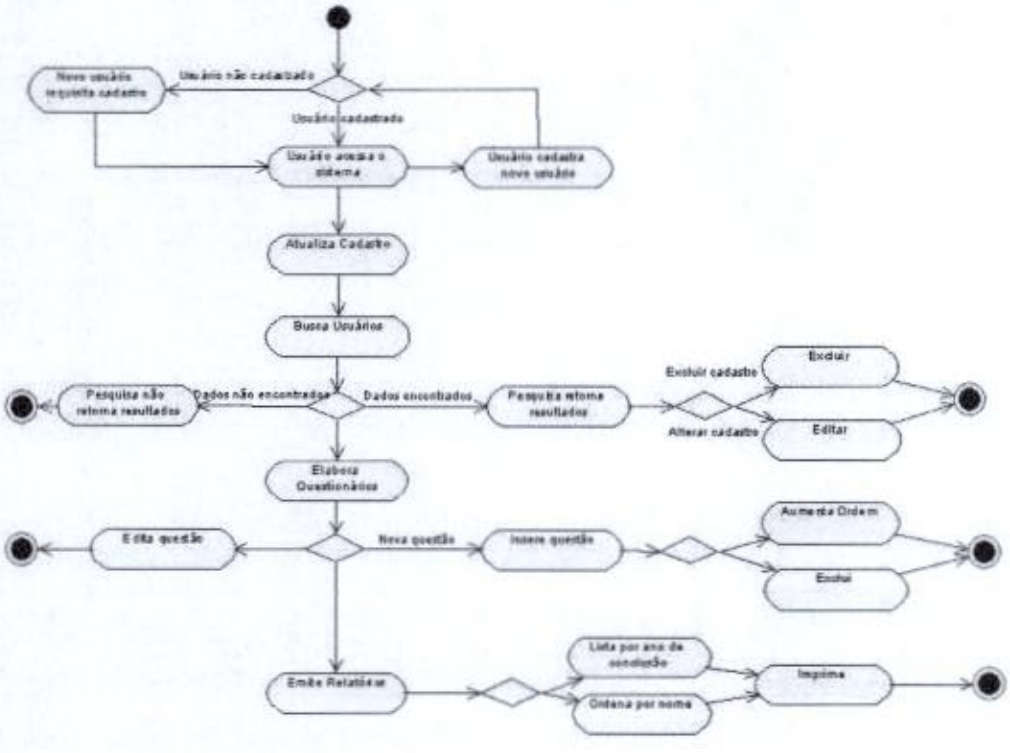
6.3.2 MÓDULO EGRESSOS

6.3.2.1 Diagramas de Atividades

6.3.2.1 Visão Aluno

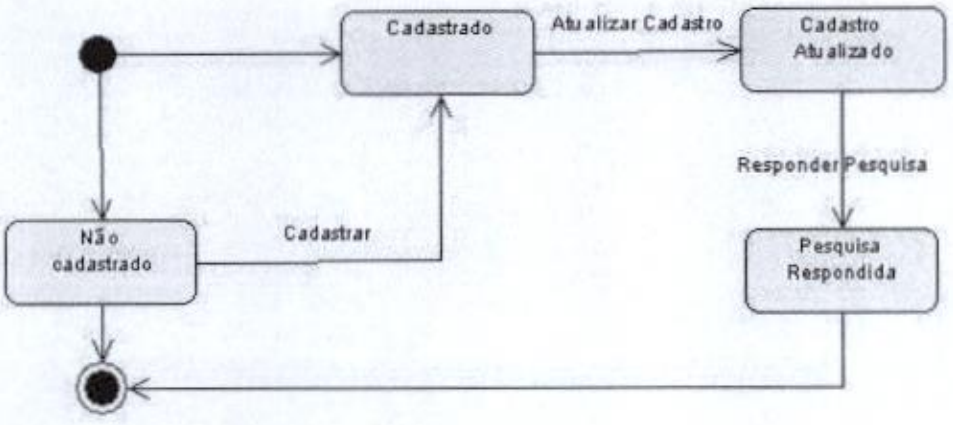


6.3.2.2 Visão Coordenação

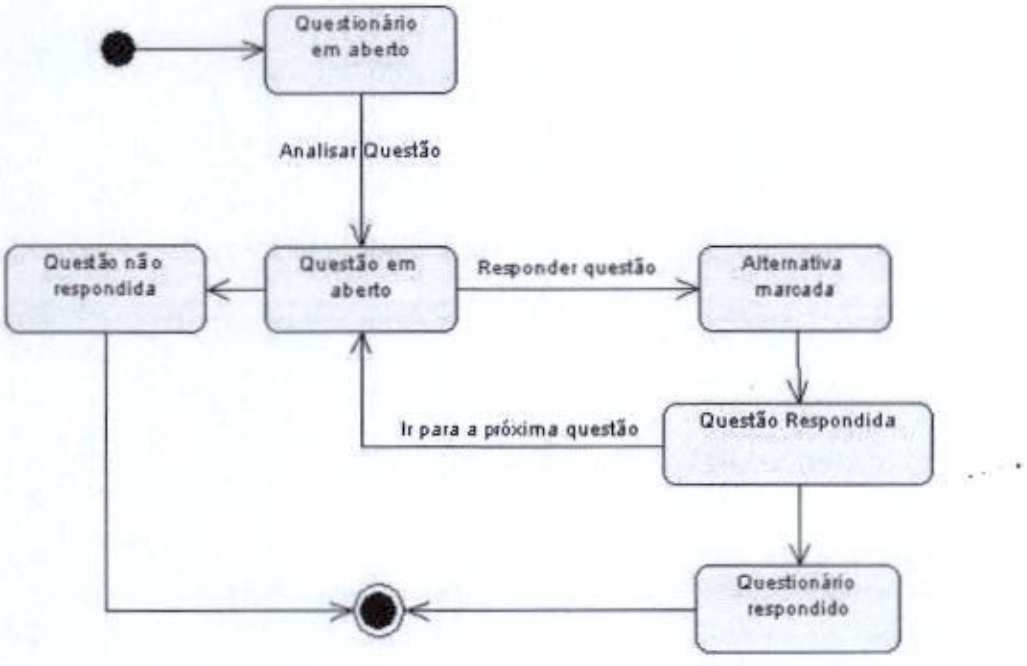


6.3.3 Diagramas de Estados

6.3.3.1 Classe Aluno



6.3.3.2 Classe Questionário



6.3.4 Dicionário de Dados

6.3.4.1 Tabela: **Aluno**

Função: Armazenar e associar dados específicos do aluno egresso.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
Cpf	PK	NOT NULL	CHAR(11)	CPF do egresso
pessoa_codpessoa	FK	NOT NULL	int4	Código da pessoa
Nome			VARCHAR	Nome do egresso
Rg			VARCHAR	Número do RG do egresso
rgNaturalidade			VARCHAR	Naturalidade do egresso
dataNascimento			DATE	Data de nascimento do egresso
EstadoCivil			VARCHAR	Estado Civil do egresso
NomePai			VARCHAR	Nome do Pai do egresso
NomeMae			VARCHAR	Nome da Mãe do egresso
Email			VARCHAR	E-mail do egresso
Fone			VARCHAR	Telefone do egresso
Situação			VARCHAR	Situação do egresso
Sexo			CHAR	Sexo do egresso (M/F)
nacionalidade			VARCHAR	Nacionalidade do egresso
Senha			VARCHAR	Senha de acesso ao sistema
anoconcsuperior			int4	Ano de conclusão do curso superior

6.3.4.2 Tabela: **EnderecoPessoa**

Função: Armazenar e associar o endereço do aluno egresso.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
Aluno_cpf	FK	NOT NULL	CHAR(11)	CPF do egresso
cep			VARCHAR	Número do CEP
rua			VARCHAR	Nome da Rua
numero			int4	Número da residência
complemento			VARCHAR	Complemento do endereço
bairro			VARCHAR	Nome do Bairro
município			VARCHAR	Nome do Município
estado			VARCHAR	Nome do Estado
pais			VARCHAR	Nome do País

6.3.4.3 Tabela: **Questionario**

Função: Armazenar e associar dados dos questionários.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codQuestionario	PK	NOT NULL	int4	Código do Questionário
DataCriacao			DATE	Data de criação do questionário.
ativo			int4	Situação do questionário

6.3.4.4 Tabela: **Questao**

Função: Armazenar e associar dados sobre as questões dos questionários.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
idQuestao	PK	NOT NULL	int4	Identificação da questão
questao			VARCHAR	Enunciado da questão
peso			int4	Peso da questão
alternativaA			VARCHAR	Descrição da Alternativa A

alternativaB			VARCHAR	Descrição da Alternativa B
alternativaC			VARCHAR	Descrição da Alternativa C
alternativaD			VARCHAR	Descrição da Alternativa D
alternativaE			VARCHAR	Descrição da Alternativa E
gabarito			VARCHAR	Gabarito da questão
questionário_ codquestionario	FK	NOT NULL	int4	Código do questionário
ordem			int4	Ordem da questão no questionário

6.3.4.5 Tabela: **Respostas**

Função: Armazenar e associar dados sobre as respostas das questões dos questionários.

Campo	Chave	Valor	Tipo(Tamanho)	Descrição
codRespostas	PK	NOT NULL	int4	Código da Resposta
Aluno_cpf	FK	NOT NULL	CHAR(11)	CPF do egresso
Questao_idQuestao	FK	NOT NULL	int4	Identificação da questão
Questionario_ codQuestionario	FK	NOT NULL	int4	Código da questão
resposta			VARCHAR	Resposta da questão

6.4 Apêndice 4

Códigos Fonte

6.4.0 Módulo Egressos - Código Fonte

6.4.1 Estrutura de diretórios:

```
RootDir
|
+-- META-INF
|   |
|   +-- context.xml
|
+-- pages
|   |
|   +-- *.jsp
|
+-- WEB-INF
    |
    +-- classes
    |   |
    |   +-- estagio
    |   |   |
    |   |   +-- action
    |   |   +-- bean
    |   |   +-- dão
    |   |   +-- form
    |   |   +-- validation
    |   +-- jasper
    |   +--MessageResources.properties
    |
    +-- lib
    |   |
    |   +-- commons-beanutils.jar
    |   +-- commons-validator.jar
    |   +-- commons-digester.jar
    |   +-- commons-fileupload.jar
    |   +-- commons-logging.jar
    |   +-- antl.jar
    |   +-- jakarta-oro.jar
    |   +-- struts.jar
    |   +-- activation.jar
    |   +-- mail.jar
    |
    +-- tiles-defs.xml
    +-- struts-config.xml
    +-- validation.xml
    +-- validation-rules.xml
    +-- web.xml
    +-- struts-bean.xml
    +-- struts-logic.xml
    +-- struts-html.xml
    +-- struts-nested.xml
    +-- struts-tiles.xml
```

6.4.2 tiles-defs.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE tiles-definitions PUBLIC
    "-//Apache Software Foundation//DTD Tiles
Configuration 1.1//EN"
    "http://jakarta.apache.org/struts/dtds/tiles-
config_1_1.dtd">
<!--
   This is a blank Tiles definition file with a commented
example.-->
<tiles-definitions>
  <!-- sample tiles definitions
  <definition name=".mainLayout"
path="/common/layouts/classicLayout.jsp">
     <put name="title"  value="Sample Page Title" />
     <put name="header" value="/common/header.jsp" />
     <put name="menu"   value=".mainMenu" />
     <put name="footer" value="/common/footer.jsp" />
     <put name="body"   value=".portal.body" />
  </definition>
  <definition name=".mainMenu"
path="/common/layouts/vboxLayout.jsp" >
     <putList name="list" >
        <add value=".menu.links" />
        <add value=".menu.taglib.references" />
        <add value=".menu.printer.friendly" />
        <add value=".menu.old.documents" />
    </putList>
  </definition>
  <definition name="aPage" extends=".mainLayout">
     <put name="title"  value="Another Title" />
     <put name="body"   value=".aPage.body" />
  </definition>
  end samples -->
  <definition name="mainLayout"
path="/pages/index.jsp">
    <put name="menu"   value="/pages/mainMenu.jsp" />
     <put name="body"   value="" />
     <put name="footer"   value="/pages/footer.jsp" />
  </definition>
  <definition name="dadospessoais"
extends="mainLayout">
     <put name="body"
value="/pages/dadosPessoais.jsp" />
  </definition>
  <definition name="inicial" extends="mainLayout">
     <put name="body"   value="/pages/inicial.jsp" />
  </definition>
  <definition name="erro" extends="mainLayout">
     <put name="body"   value="/pages/erro.jsp" />
  </definition>
  <definition name="sucesso" extends="mainLayout">
     <put name="body"   value="/pages/sucesso.jsp" />
  </definition>
  <definition name="buscarusuario"
extends="mainLayout">
     <put name="body"
value="/pages/buscarUsuario.jsp" />
  </definition>
  <definition name="resultadosdabusca"
extends="mainLayout">
     <put name="body"
value="/pages/resultadosBuscaUsuarios.jsp" />
  </definition>
     <definition name="gerenciarQuestionarios"
extends="mainLayout">
     <put name="body"
value="/pages/gerenciarQuestionarios.jsp" />
  </definition>
    <definition name="responderQuestionario"
extends="mainLayout">
     <put name="body"
value="/pages/responderQuestionario.jsp" />
  </definition>
    <definition name="listarQuestoes"
extends="mainLayout">
     <put name="body"
value="/pages/listarQuestoes.jsp" />
  </definition>
    <definition name="estatisticasQuestionario"
extends="mainLayout">
     <put name="body"
value="/pages/estatisticasQuestionario.jsp" />
  </definition>
    <definition name="mostrarQuestao"
extends="mainLayout">
     <put name="body"
value="/pages/mostrarQuestao.jsp" />
  </definition>
    <definition name="inserequestao"
extends="mainLayout">
  <put name="body"   value="/pages/inserequestao.jsp" />
  </definition>
    <definition name="tiporelatorio"
extends="mainLayout">
   <put name="body"   value="/pages/tiporelatorio.jsp" />
  </definition>
   <definition name="relatorionome"
extends="mainLayout">
     <put name="body"
value="/pages/relatorionome.jsp" />
  </definition>
  <definition name="relatorioano"
extends="mainLayout">
   <put name="body"   value="/pages/relatorioano.jsp" />
  </definition>
  <definition name="welcome" extends="mainLayout">
     <put name="body"   value="/pages/Welcome.jsp" />
  </definition>
  <definition name="novoservidorsmtp"
extends="mainLayout">
     <put name="body"
value="/pages/novoservidorsmtp.jsp" />
  </definition>
  <definition name="informausername"
extends="mainLayout">
     <put name="body"
value="/pages/informausername.jsp" />
  </definition>
  <definition name="alterarsenha"
extends="mainLayout">
  <put name="body"   value="/pages/alterarsenha.jsp" />
  </definition>
 </tiles-definitions>
```

6.4.3 struts-config.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

```
<!DOCTYPE struts-config PUBLIC
        "-//Apache Software Foundation//DTD Struts
Configuration 1.2//EN"
        "http://jakarta.apache.org/struts/dtds/struts-
config_1_2.dtd">
<!--
    This is a blank Struts configuration file with an
example
    welcome action/page and other commented sample
elements.

    Tiles and the Struts Validator are configured using the
factory defaults
    and are ready-to-use.

    NOTE: If you have a generator tool to create the
corresponding Java classes
    for you, you could include the details in the "form-
bean" declarations.
    Otherwise, you would only define the "form-bean"
element itself, with the
    corresponding "name" and "type" attributes, as shown
here.
-->
<struts-config>
<!-- ============== Data Source Configuration -->
<!--
<data-sources>
<data-source
type="org.apache.commons.dbcp.BasicDataSource">
    <set-property
     property="driverClassName"
     value="org.postgresql.Driver" />
    <set-property
     property="url"
     value="jdbc:postgresql://localhost/mydatabase" />
    <set-property
     property="username"
     value="me" />
    <set-property
     property="password"
     value="test" />
    <set-property
     property="maxActive"
     value="10" />
    <set-property
     property="maxWait"
     value="5000" />
    <set-property
     property="defaultAutoCommit"
     value="false" />
    <set-property
     property="defaultReadOnly"
     value="false" />
    <set-property
     property="validationQuery"
     value="SELECT COUNT(*) FROM market" />
</data-source>
</data-sources>
-->
<!-- ========== Form Bean Definitions -->
    <form-beans>
    <form-bean      name="RegistrationForm"
type="egressos.form.RegistrationForm"/>
    <form-bean      name="LogonForm"
type="egressos.form.LogonForm"/>
    <form-bean      name="DadosPessoaisForm"
type="egressos.form.DadosPessoaisForm"/>
    <form-bean      name="NovaQuestaoForm"
type="egressos.form.NovaQuestaoForm"/>
    <form-bean      name="ResponderQuestionarioForm"
type="egressos.form.ResponderQuestionarioForm"/>
    <form-bean      name="ObterUsuarioForm"
type="egressos.form.ObterUsuarioForm"/>
    <form-bean      name="ServidorSmtpForm"
type="egressos.form.ServidorSmtpForm"/>
    <form-bean      name="InformaUserNameForm"
type="egressos.form.InformaUserNameForm"/>
    <form-bean      name="AlterarSenhaForm"
type="egressos.form.AlterarSenhaForm"/>
    </form-beans>
<!--
========================================
Global Exception Definitions -->

    <global-exceptions>
      <!-- sample exception handler
      <exception
        key="expired.password"
        type="app.ExpiredPasswordException"
        path="/changePassword.jsp"/>
      end sample -->
    </global-exceptions>
<!--
========================================
= Global Forward Definitions -->
    <global-forwards>
      <!-- Default forward to "Welcome" action -->
      <!-- Demonstrates using index.jsp to forward -->
      <forward name="welcome"
path="/Welcome.do"/>
      <forward name="logon"          path="/Logon.do"/>
      <forward name="logoff"         path="/Logoff.do"/>

    </global-forwards>
<!--
========================================
= Action Mapping Definitions -->
    <action-mappings>
        <!-- Default "Welcome" action -->
        <!-- Forwards to Welcome.jsp -->
      <action
        path="/Welcome"
        type="egressos.action.WelcomeAction"
        scope="request"
        validate="false">
        <forward name="success" path="inicial" />
        <forward name="error" path="welcome" />
      </action>
       <action
        path="/Logon"
        forward="welcome"/>
      <action
            path="/MainMenu"
            type="egressos.action.MainMenuAction"
            scope="request"
            validate="false">
       <forward name="success"          path="inicial" />
```

```
      <forward name="error"          path="erro" />
    </action>
    <action
      path="/Registration"
      type="egressos.action.NovoUsuarioAdminAction"
      scope="request"
      validate="false"
      input="inicial">
      <forward name="success"
path="dadospessoais" />
      <forward name="error"          path="erro" />
    </action>
    <action
      path="/NovoUsuario"
      type="egressos.action.NovoUsuarioAction"
      name="LogonForm"
      scope="request"
      validate="false"
      input="welcome">
      <forward name="success"
path="dadospessoais" />
      <forward name="error"          path="welcome" />
    </action>
    <action
      path="/NovoEgresso"
      type="egressos.action.RegistrarEgressoAction"
      name="LogonForm"
      scope="request"
      validate="false"
      input="welcome">
      <forward name="success"
path="dadospessoais" />
      <forward name="error"          path="welcome" />
    </action>
    <action
      path="/SalvarRegistroEgressoAction"
type="egressos.action.SalvarRegistroEgressoAction"
      name="DadosPessoaisForm"
      scope="request"
      validate="true"
      input="dadospessoais">
      <forward name="success"     path="sucesso" />
      <forward name="error"
path="dadospessoais" />
    </action>
    <action
      path="/ExcluirEgresso"
      type="egressos.action.ExcluirEgressoAction"
      scope="request"
      validate="false"
      input="resultadosdabusca">
      <forward name="success"
path="buscarusuario" />
      <forward name="error"
path="buscarusuario" />
    </action>
    <action
      path="/LogonSubmit"
      type="egressos.action.LogonAction"
      name="LogonForm"
      scope="request"
      validate="true"
      input="welcome">
      <forward name="success" path="inicial" />
      <forward name="error" path="welcome" />
    </action>
    <action
      path="/Logoff"
      type="egressos.action.LogoffAction">
      <forward
        name="success"
        path="welcome"/>
    </action>
    <action
      path="/SalvarCadastro"
      type="egressos.action.SalvarCadastroAction"
      name="DadosPessoaisForm"
      scope="request"
      validate="true"
      input="dadospessoais">
      <forward name="success"   path="sucesso" />
      <forward name="error"
path="dadospessoais" />
    </action>
    <action
        path="/EditarCadastro"
        type="egressos.action.EditarCadastroAction"
        name="DadosPessoaisForm"
        input="inicial"
        scope="request"
        validate="false">
      <forward name="success"
path="dadospessoais" />
      <forward name="error"   path="erro" />
    </action>
    <action
        path="/GerenciarQuestionarios"
type="egressos.action.GerenciarQuestionariosAction"
        input="inicial"
        scope="request"
        validate="false">
      <forward name="success"
path="gerenciarQuestionarios" />
      <forward name="error"          path="erro" />
    </action>
    <action
        path="/ResponderQuestionario"
type="egressos.action.ResponderQuestionarioAction"
        input="inicial"
        scope="request"
        validate="false">
      <forward name="success"
path="responderQuestionario" />
      <forward name="error"    path="erro" />
    </action>
    <action
        path="/EstatisticasQuestionario"
type="egressos.action.EstatisticasQuestionarioAction"
        input="listarQuestoes"
        scope="request"
        validate="false">
      <forward name="success"
path="estatisticasQuestionario" />
      <forward name="error"
path="listarQuestoes" />
    </action>
    <action
        path="/MostrarRespostasQuestao"
```

```
type="egressos.action.MostrarRespostasQuestaoAction"
            input="estatisticasQuestionario"
            scope="request"
            validate="false">
      <forward name="success"
path="mostrarQuestao" />
      <forward name="error"
path="estatisticasQuestionario" />
    </action>
    <action
            path="/SalvarRespostasQuestionario"
            name="ResponderQuestionarioForm"
type="egressos.action.SalvarRespostasQuestionarioAction
"            input="responderQuestionario"
            scope="request"
            validate="true">
      <forward name="success"          path="sucesso"
/>
      <forward name="error"          path="erro" />
    </action>
    <action
            path="/DefinirAtivo"
            type="egressos.action.DefinirAtivoAction"
            input="gerenciarQuestionarios"
            scope="request"
            validate="false">
      <forward name="success"
path="gerenciarQuestionarios" />
      <forward name="error"
path="gerenciarQuestionarios" />
    </action>
    <action
            path="/NovoQuestionario"
type="egressos.action.NovoQuestionarioAction"
            input="gerenciarQuestionarios"
            scope="request"
            validate="false">
      <forward name="success"
path="listarQuestoes" />
      <forward name="error"
path="gerenciarQuestionarios" />
    </action>
    <action
            path="/ExcluirQuestionario"
type="egressos.action.ExcluirQuestionarioAction"
            input="gerenciarQuestionarios"
            scope="request"
            validate="false">
      <forward name="success"
path="gerenciarQuestionarios" />
      <forward name="error"
path="gerenciarQuestionarios" />
    </action>
    <action
            path="/NovaQuestao"
            name="NovaQuestaoForm"
            type="egressos.action.NovaQuestaoAction"
            input="listarQuestoes"
            scope="request"
            validate="false">
      <forward name="success"
path="inserequestao" />
      <forward name="error"
path="listarQuestoes" />
    </action>
    <action
            path="/SalvarQuestao"
            name="NovaQuestaoForm"
            type="egressos.action.SalvarQuestaoAction"
            input="inserequestao"
            scope="request"
            validate="true">
      <forward name="success"
path="listarQuestoes" />
      <forward name="error"
path="inserequestao" />
    </action>
    <action
            path="/ListarQuestoes"
            type="egressos.action.ListarQuestoesAction"
            input="gerenciarQuestionarios"
            scope="request"
            validate="false">
      <forward name="success"
path="listarQuestoes" />
      <forward name="error"          path="erro" />
    </action>
    <action
            path="/EditarQuestao"
            type="egressos.action.EditarQuestaoAction"
            name="NovaQuestaoForm"
            input="listarQuestoes"
            scope="request"
            validate="false">
      <forward name="success"
path="inserequestao" />
      <forward name="error"
path="listarQuestoes" />
    </action>
    <action
            path="/ExcluirQuestao"
            type="egressos.action.ExcluirQuestaoAction"
            input="listarQuestoes"
            scope="request"
            validate="false">
      <forward name="success"
path="listarQuestoes" />
      <forward name="error"
path="listarQuestoes" />
    </action>
    <action
            path="/AumentarOrdem"
            type="egressos.action.AumentarOrdemAction"
            input="listarQuestoes"
            scope="request"
            validate="false">
      <forward name="success"
path="listarQuestoes" />
      <forward name="error"
path="listarQuestoes" />
    </action>
    <action
            path="/Relatorios"
            type="egressos.action.RelatoriosAction"
            input="inicial"
            scope="request"
```

```
            validate="false">
        <forward name="success"
path="tiporelatorio" />
        <forward name="error"          path="erro" />
      </action>
      <action
            path="/RelatorioAno"
            type="egressos.action.RelatorioAnoAction"
            input="tiporelatorio"
            scope="request"
            validate="false">
        <forward name="success"
path="relatorioano" />
        <forward name="error"          path="tiporelatorio"
/>
      </action>
      <action
            path="/RelatorioNome"
            type="egressos.action.RelatorioNomeAction"
            input="tiporelatorio"
            scope="request"
            validate="false">
        <forward name="success"
path="relatorionome" />
        <forward name="error"          path="tiporelatorio"
/>
      </action>
      <action
            path="/BuscarUsuario"
            type="egressos.action.BuscarUsuarioAction"
            input="inicial"
            scope="request"
            validate="false">
        <forward name="success"
path="buscarusuario" />
        <forward name="error"          path="erro" />
      </action>
      <action
            path="/ObterUsuario"
            type="egressos.action.ObterUsuarioAction"
            input="inicial"
            name="ObterUsuarioForm"
            scope="request"
            validate="true">
        <forward name="success"
path="resultadosdabusca" />
        <forward name="error"
path="buscarusuario" />
      </action>
      <action
        path="/InformaUserName"
        forward="informausername" />
      <action
        path="/EnviaSenha"
        type="egressos.action.EnviaSenhaAction"
        scope="request"
        validate="true"
        name="InformaUserNameForm"
        input="informausername">
        <forward name="success" path="welcome" />
        <forward name="error" path="informausername"
/>
      </action>
      <action
            path="/EditarEmail"
            type="egressos.action.EditarEmailAction"
            scope="request"
            validate="false"
            name="ServidorSmtpForm"
            input="inicial">
            <forward name="success"
path="novoservidorsmtp" />
            <forward name="error" path="erro" />
        </action>
        <action
          path="/SalvarServidorSmtp"
          type="egressos.action.SalvarServidorSmtpAction"
          scope="request"
          validate="true"
          name="ServidorSmtpForm"
          input="novoservidorsmtp">
          <forward name="success" path="sucesso" />
          <forward name="error" path="erro" />
        </action>

        <action
          path="/AlterarSenha"
          forward="alterarsenha" />

        <action
          path="/SalvarSenhaAlterada"
          type="egressos.action.SalvarSenhaAlteradaAction"
          scope="request"
          validate="true"
          name="AlterarSenhaForm"
          input="inicial">
          <forward name="success" path="sucesso" />
          <forward name="error" path="erro" />
        </action>
        <!-- sample input and input submit actions
        <action
          path="/Input"
          type="org.apache.struts.actions.ForwardAction"
          parameter="/pages/Input.jsp"/>
        <action
          path="/InputSubmit"
          type="app.InputAction"
          name="inputForm"
          scope="request"
          validate="true"
          input="/pages/Input.jsp"/>
          <action
            path="/edit*"
            type="app.Edit{1}Action"
            name="inputForm"
            scope="request"
            validate="true"
            input="/pages/Edit{1}.jsp"/>
      end samples -->
      </action-mappings>
<!-- ================ Controller Configuration -->
      <controller
processorClass="org.apache.struts.tiles.TilesRequestProce
ssor"/>
<!--
============================================
Message Resources Definitions -->
```

```
  <message-resources parameter="MessageResources"
/><!-- ========== Plug Ins Configuration -->
 <!-- ============== Tiles plugin -->
 <!--
   This plugin initialize Tiles definition factory. This later
can takes some parameters explained here after. The
plugin first read parameters from  web.xml, thenoverload
them with parameters defined here. All parameters are
optional.  The plugin should be declared in each struts-
config file.    - definitions-config: (optional)
Specify configuration file names. There can be several
comma separated file names (default: ?? )   -
moduleAware: (optional - struts1.1)  Specify if the Tiles
definition factory is module aware. If true (default), there
will be one factory for each Struts module.If false, there
will be one common factory for all module. In this
later case, it is still needed to declare one plugin per
module.  The factory will be initialized with parameters
found in the first  initialized plugin (generally the one
associated with the default module). true : One factory per
module. (default)  false : one single shared factory for all
modules   - definitions-parser-validate: (optional)
Specify if xml parser should validate the Tiles onfiguration
file.  true : validate. DTD should be specified in file header
(default)  false : no validation  Paths found in Tiles
definitions are relative to the main context. -->
<plug-in classNamе="org.apache.struts.tiles.TilesPlugin"
>
 <!-- Path to XML definition file -->
 <set-property property="definitions-config"
value="/WEB-INF/tiles-defs.xml" />
<!-- Set Module-awareness to true -->
<set-property property="moduleAware" value="true" />
</plug-in>
 <!-- ================== Validator plugin -->
 <plug-in
classNamе="org.apache.struts.validator.ValidatorPlugIn">
   <set-property
     property="pathnames"
     value="/WEB-INF/validator-rules.xml,/WEB-
INF/validation.xml"/>
 </plug-in>
</struts-config>
```

6.4.4 validation.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE form-validation PUBLIC
       "-//Apache Software Foundation//DTD Commons
Validator Rules Configuration 1.1.3//EN"
"http://jakarta.apache.org/commons/dtds/validator_1_1_3.
dtd">
<form-validation>
<!--
This is a minimal Validator form file with a couple of
examples.-->  <global>
     <!-- An example global constant
     <constant>
       <constant-name>postalCode</constant-name>
       <constant-value>^\d{5}\d*$</constant-value>
     </constant>
     end example-->
   </global>
   <formset>
     <form name="LogonForm">
       <field
         property="username"
         depends="required, maxlength">
        <arg    key="prompt.username"/>
        <arg1     name="maxlength"
key="${var:maxlength}" resource="false"/>
        <var><var-name>maxlength</var-name><var-
value>11</var-value></var>
       </field>
       <field
         property="password"
         depends="required,mask">
           <arg key="prompt.password"/>
           <var>
             <var-name>mask</var-name>
             <var-value>^[0-9a-zA-Z]*$</var-value>
           </var>
       </field>
     </form>
     <form name="RegistrationForm">
       <field property="email"
           depends="required,email">
         <arg   key="prompt.email"/>
       </field>
       <field property="fullName"
           depends="required">
        <arg    key="prompt.fullName"/>
       </field>
       <field property="usuario"
           depends="required">
        <arg    key="prompt.username"/>
       </field>
       <field property="userName"
           depends="required">
        <arg    key="prompt.username"/>
       </field>
       <field property="pin"
           depends="required">
        <arg    key="prompt.password"/>
       </field>
     </form>
     <form name="DadosPessoaisForm">
       <field property="email"
           depends="required,email">
         <arg   key="prompt.email"/>
       </field>
       <field property="nome"
           depends="required">
        <arg    key="prompt.fullName"/>
       </field>
       <field property="senha"
             depends="required,mask">
           <arg key="prompt.password"/>
           <var>
             <var-name>mask</var-name>
             <var-value>^[0-9a-zA-Z]*$</var-value>
           </var>
       </field>
       <field property="anoConclSuperior"
           depends="required,integer,intRange">
         <arg0   key="prompt.anoConclSuperior"/>
         <arg1 name="intRange" key="${var:min}"
resource="false"/>
```

```
        <arg2 name="intRange" key="${var:max}"
resource="false"/>
        <var><var-name>min</var-name><var-
value>1950</var-value></var>
        <var><var-name>max</var-name><var-
value>2050</var-value></var>
      </field>
      <field property="bairro"
          depends="required">
       <arg    key="prompt.bairro"/>
      </field>
      <field property="cep"
            depends="required,minlength">
       <arg0    key="prompt.cep"/>
       <arg1    name="minlength"
key="${var:minlength}" resource="false"/>
       <var><var-name>minlength</var-name><var-
value>8</var-value></var>
      </field>
      <field property="cpf"
          depends="required,cpf, maxlength">
       <arg0    key="prompt.cpf"/>
       <arg1    name="maxlength"
key="${var:maxlength}" resource="false"/>
       <var><var-name>maxlength</var-name><var-
value>11</var-value></var>
      </field>
      <field property="dataNascimento"
          depends="required,date">
        <arg    key="prompt.dataNascimento"/>
        <var>
          <var-name>datePatternStrict</var-name>
          <var-value>dd/MM/yyyy</var-value>
        </var>
      </field>
      <field property="estado"
          depends="required">
        <arg  key="prompt.estado"/>
      </field>
      <field property="estadoCivil"
          depends="required">
       <arg    key="prompt.estadoCivil"/>
      </field>
      <field property="fone"
                depends="required,minlength">
       <arg0    key="prompt.fone"/>
       <arg1    name="minlength"
key="${var:minlength}" resource="false"/>
       <var><var-name>minlength</var-name><var-
value>8</var-value></var>
      </field>
      <field property="institSuperior"
          depends="required">
        <arg  key="prompt.institSuperior"/>
      </field>

      <field property="municipio"
          depends="required">
       <arg    key="prompt.municipio"/>
      </field>
      <field property="nacionalidade"
          depends="required">
       <arg    key="prompt.nacionalidade"/>
      </field>
      <field property="nomeMae"
          depends="required">
        <arg  key="prompt.mae"/>
      </field>
      <field property="nomePai"
          depends="required">
       <arg    key="prompt.pai"/>
      </field>
      <field property="numero"
          depends="required,integer,intRange">
        <arg0    key="prompt.numero"/>
        <arg1 name="intRange" key="${var:min}"
resource="false"/>
        <arg2 name="intRange" key="${var:max}"
resource="false"/>
        <var><var-name>min</var-name><var-
value>0</var-value></var>
        <var><var-name>max</var-name><var-
value>20000</var-value></var>
      </field>
      <field property="pais"
          depends="required">
        <arg  key="prompt.pais"/>
      </field>
      <field property="rg"
          depends="required, minlength">
       <arg    key="prompt.rg"/>
       <arg1    name="minlength"
key="${var:minlength}" resource="false"/>
        <var><var-name>minlength</var-name><var-
value>6</var-value></var>
      </field>
      <field property="rgNaturalidade"
          depends="required">
       <arg    key="prompt.rgNaturalidade"/>
      </field>
      <field property="rua"
          depends="required">
        <arg  key="prompt.rua"/>
      </field>
      <field property="sexo"
          depends="required">
       <arg    key="prompt.sexo"/>
      </field>
    </form>
    <form name="NovaQuestaoForm">
      <field
        property="pergunta"
        depends="required">
          <arg key="prompt.pergunta"/>
      </field>
    </form>
    <form name="ServidorSmtpForm">
      <field
        property="endereco_email"
        depends="required">
       <arg    key="prompt.email"/>
      </field>
      <field
        property="porta_smtp"
        depends="required">
       <arg    key="prompt.porta_smtp"/>
      </field>
      <field
```

```
        property="senha"
        depends="required">
       <arg    key="prompt.password"/>
      </field>
      <field
        property="url"
        depends="required">
       <arg    key="prompt.url"/>
      </field>
      <field
        property="username"
        depends="required">
       <arg    key="prompt.username"/>
      </field>
    </form>
    <form name="InformaUserNameForm">
      <field
        property="username"
        depends="required">
       <arg    key="prompt.username"/>
      </field>
      <field
        property="email"
        depends="required, email">
       <arg    key="prompt.email"/>
      </field>
    </form>
    <form name="AlterarSenhaForm">
      <field
        property="senha"
        depends="required">
       <arg    key="prompt.password"/>
      </field>
      <field
        property="senha2"
        depends="required">
       <arg    key="prompt.password2"/>
      </field>
    </form>
  </formset>
</form-validation>
```

6.4.5 validation-rules.xml

```
<!DOCTYPE form-validation PUBLIC
      "-//Apache Software Foundation//DTD Commons
Validator Rules Configuration 1.1.3//EN"
http://jakarta.apache.org/commons/dtds/validator_1_1_3.dt
d">
<!--
  $Id: validator-rules.xml 159291 2005-03-28 20:19:29Z
niallp $
  This file contains the default Struts Validator pluggable
validator
  definitions.  It should be placed somewhere under
/WEB-INF and
  referenced in the struts-config.xml under the plug-in
element
  for the ValidatorPlugIn.
    <plug-in
className="org.apache.struts.validator.ValidatorPlugIn">
      <set-property property="pathnames" value="/WEB-
INF/validator-rules.xml,
                                /WEB-
INF/validation.xml"/>
    </plug-in>
  These are the default error messages associated with
  each validator defined in this file.  They should be
  added to your projects ApplicationResources.properties
  file or you can associate new ones by modifying the
  pluggable validators msg attributes in this file.
  # Struts Validator Error Messages
  errors.required={0} is required.
  errors.minlength={0} can not be less than {1}
characters.
  errors.maxlength={0} can not be greater than {1}
characters.
  errors.invalid={0} is invalid.
  errors.byte={0} must be a byte.
  errors.short={0} must be a short.
  errors.integer={0} must be an integer.
  errors.long={0} must be a long.
  errors.float={0} must be a float.
  errors.double={0} must be a double.
  errors.date={0} is not a date.
  errors.range={0} is not in the range {1} through {2}.
  errors.creditcard={0} is an invalid credit card number.
  errors.email={0} is an invalid e-mail address.
  Note: Starting in Struts 1.2.0 the default javascript
definitions have
      been consolidated to commons-validator. The default
can be overridden
      by supplying a <javascript> element with a CDATA
section, just as
      in struts 1.1.-->
<form-validation>
  <global>
    <validator name="required"
classname="org.apache.struts.validator.FieldChecks"
          method="validateRequired"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
             msg="errors.required"/>
    <validator name="requiredif"
classname="org.apache.struts.validator.FieldChecks"
            method="validateRequiredIf"
            methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
org.apache.struts.action.ActionMessages,
org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
<validator name="validwhen"
       msg="errors.required"
classname="org.apache.struts.validator.validwhen.ValidW
hen" method="validateValidWhen"
            methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"/>
    <validator name="minlength"
classname="org.apache.struts.validator.FieldChecks"
          method="validateMinLength"
```

```
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.minlength"
jsFunction="org.apache.commons.validator.javascript.vali
dateMinLength"/>
    <validator name="maxlength"
classname="org.apache.struts.validator.FieldChecks"
          method="validateMaxLength"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.maxlength"
jsFunction="org.apache.commons.validator.javascript.vali
dateMaxLength"/>
    <validator name="mask"
classname="org.apache.struts.validator.FieldChecks"
          method="validateMask"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.invalid"/>
    <validator name="byte"
classname="org.apache.struts.validator.FieldChecks"
          method="validateByte"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.byte"
     jsFunctionName="ByteValidations"/>
    <validator name="short"
classname="org.apache.struts.validator.FieldChecks"
          method="validateShort"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.short"
     jsFunctionName="ShortValidations"/>
    <validator name="integer"
classname="org.apache.struts.validator.FieldChecks"
          method="validateInteger"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.integer"
     jsFunctionName="IntegerValidations"/>
    <validator name="long"
classname="org.apache.struts.validator.FieldChecks"
          method="validateLong"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.long"/>
    <validator name="float"
classname="org.apache.struts.validator.FieldChecks"
          method="validateFloat"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.float"
     jsFunctionName="FloatValidations"/>
    <validator name="double"
classname="org.apache.struts.validator.FieldChecks"
          method="validateDouble"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.double"/>
    <validator name="date"
classname="org.apache.struts.validator.FieldChecks"
          method="validateDate"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.date"
     jsFunctionName="DateValidations"/>
    <validator name="intRange"
classname="org.apache.struts.validator.FieldChecks"
          method="validateIntRange"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends="integer"
            msg="errors.range"/>
```

```
    <validator name="floatRange"
classname="org.apache.struts.validator.FieldChecks"
          method="validateFloatRange"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends="float"
            msg="errors.range"/>
    <validator name="doubleRange"
classname="org.apache.struts.validator.FieldChecks"
          method="validateDoubleRange"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends="double"
            msg="errors.range"/>
    <validator name="creditCard"
classname="org.apache.struts.validator.FieldChecks"
          method="validateCreditCard"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.creditcard"/>
    <validator name="email"
classname="org.apache.struts.validator.FieldChecks"
          method="validateEmail"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.email"/>
    <validator name="url"
classname="org.apache.struts.validator.FieldChecks"
          method="validateUrl"
      methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.url"/>
   <validator name="cpf"
          classname="egressos.validation.Valida"
          method="validateCpf"
          methodParams="java.lang.Object,
org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
          msg="errors.cpf">
   </validator>
<!--
    This simply allows struts to include the
validateUtilities into a page, it should
    not be used as a validation rule.
   -->
   <validator name="includeJavaScriptUtilities"
       classname=""
         method=""
     methodParams=""
       depends=""
         msg=""
jsFunction="org.apache.commons.validator.javascript.vali
dateUtilities"/>
  </global>
</form-validation>
```

6.4.6 web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
 PUBLIC "-//Sun Microsystems, Inc.//DTD Web
Application 2.2//EN"
 "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
<web-app>
 <display-name>Struts Blank Application</display-
name>
 <!-- Standard Action Servlet Configuration (with
debugging) -->
 <servlet>
  <servlet-name>action</servlet-name>
  <servlet-
class>org.apache.struts.action.ActionServlet</servlet-
class>
   <init-param>
    <param-name>config</param-name>
    <param-value>/WEB-INF/struts-config.xml</param-
value>
   </init-param>
   <init-param>
    <param-name>debug</param-name>
    <param-value>2</param-value>
   </init-param>
   <init-param>
    <param-name>detail</param-name>
    <param-value>2</param-value>
   </init-param>
   <load-on-startup>2</load-on-startup>
 </servlet>
 <!-- Standard Action Servlet Mapping -->
 <servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
 </servlet-mapping>
 <!-- The Usual Welcome File List -->
 <welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
 </welcome-file-list>
  <!-- Struts Tag Library Descriptors -->
 <taglib>
  <taglib-uri>/tags/struts-bean</taglib-uri>
```

```
  <taglib-location>/WEB-INF/struts-bean.tld</taglib-
location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-html</taglib-uri>
  <taglib-location>/WEB-INF/struts-html.tld</taglib-
location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-logic</taglib-uri>
  <taglib-location>/WEB-INF/struts-logic.tld</taglib-
location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-nested</taglib-uri>
  <taglib-location>/WEB-INF/struts-nested.tld</taglib-
location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-tiles</taglib-uri>
  <taglib-location>/WEB-INF/struts-tiles.tld</taglib-
location>
 </taglib>
</web-app>
```

6.4.7 package egressos.action

6.4.8 AumentarOrdemAction.java

```
package egressos.action;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.dao.QuestaoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class AumentarOrdemAction extends Action{
   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {
      ActionMessages errors = new ActionMessages();
      String codQuestao = "";
      String codQuestionario = "";
      QuestaoDao qDao = new QuestaoDao();
      try{
         HttpSession session = request.getSession();
         Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
         codQuestao = (String)
request.getParameter("codQuestao");
         codQuestionario = (String)
request.getParameter("codQuestionario");
         //request.setAttribute("codQuestao", codQuestao);
         request.setAttribute("codQuestionario",
codQuestionario);
         HashMap map = new HashMap();
         map.put("codQuestionario", codQuestionario);
         request.setAttribute("map", map);
// seta ordem
         HashMap map1 = new HashMap();
         Questao questao = null;
         try {
            questao =
qDao.buscarUltimaQuestao(Integer.valueOf(codQuestiona
rio));
         } catch (NumberFormatException e){
errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.convnumerica"));
         }
         if (questao == null){
            questao = new Questao();
            questao.setOrdem(Integer.valueOf(0));
         }
         Integer ultimaOrdem = questao.getOrdem();
         if(Integer.valueOf(0).equals(ultimaOrdem)){
            map1.put("ordem","1");
         } else {
            ++ultimaOrdem;
            String ordem = ultimaOrdem.toString();
            map1.put("ordem",ordem);
         }
         request.setAttribute("map1", map1);
//ordem setada
         if (user == null){
            return mapping.findForward("logon");
         }
      } catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE,
               new ActionMessage("error.generico"));
      }
      Questao questaoAlterada = new Questao();
      List<Questao> questoes = new
ArrayList<Questao>();
      try{
         questaoAlterada =
qDao.buscarQuestao(Integer.valueOf(codQuestao));
         if(questaoAlterada != null){
            Integer ordem = 0;
            Integer novaOrdem = 0;
            ordem = questaoAlterada.getOrdem();
            Questao proxima =
qDao.buscarProximaQuestao(Integer.valueOf(codQuestao
));            if (proxima == null){
errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.ultima.questao"));
               proxima = new Questao();
            } else {
               novaOrdem =  proxima.getOrdem();
               questaoAlterada.setOrdem(novaOrdem);
               proxima.setOrdem(ordem);
               qDao.alterarQuestao(proxima);
               qDao.alterarQuestao(questaoAlterada);
            }
         }
```

```
        questoes =
qDao.listarQuestoes(Integer.valueOf(codQuestionario));
      } catch (NumberFormatException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.generico"));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
      }
      if (questoes == null){
        questoes = new ArrayList<Questao>();
      }
      request.setAttribute("questoes", questoes);
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return
mapping.findForward(Constants.FAILURE);
      } else {
        return
mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.9 BuscarUsuarioAction.java

```
package egressos.action;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class BuscarUsuarioAction extends Action{
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
           return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
     }
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.10 DefinirAtivoAction.java

```
package egressos.action;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questionario;
import egressos.bean.QuestionarioDTO;
import egressos.dao.QuestionarioDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class DefinirAtivoAction extends Action{
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     try{
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
           return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
     }
     QuestionarioDao qDao = new QuestionarioDao();
     List<Questionario> questionarios = new
ArrayList<Questionario>();
     List<QuestionarioDTO> qDtos = new ArrayList();
     String codQuestionario = (String)
request.getParameter("codQuestionario");
     Questionario questionario = null;
     Questionario questionarioAtivo = null;
     try{
        questionario =
qDao.buscarQuestionario(Integer.valueOf(codQuestionari
o));
        if (questionario != null){
           questionario.setStatus(Integer.valueOf(1));
           questionarioAtivo =
qDao.buscarQuestionarioAtivo();
           if((questionarioAtivo != null) &&
!(questionario.equals(questionarioAtivo))){
questionarioAtivo.setStatus(Integer.valueOf(0));
              qDao.alterarQuestionario(questionario);
              if(questionarioAtivo.getCodQuestionario() !=
null){
qDao.alterarQuestionario(questionarioAtivo);
              }
```

```
        } else
          if(questionarioAtivo == null){
            qDao.alterarQuestionario(questionario);
          }
      }
    } catch (NumberFormatException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
    }catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.delete.questionario"));
    }
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    questionarios = qDao.listarQuestionarios();
    for(Questionario q:questionarios){
      QuestionarioDTO qDto = new QuestionarioDTO();
      qDto.setCodQuestionario(q.getCodQuestionario());

qDto.setDataCriacao(sdf.format(q.getDataCriacao().getTi
me()));
      qDto.setStatus("");
      if(Integer.valueOf(1).equals(q.getStatus())){
        qDto.setStatus("Ativo");
      }
      qDtos.add(qDto);
    }
    session.removeAttribute("questionarios");
    session.setAttribute("questionarios", qDtos);
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.11 EditarCadastroAction.java

```
/*
 * EditarCadastroAction.java
 * Created on 12 de Novembro de 2005, 19:08
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import egressos.form.DadosPessoaisForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
public class EditarCadastroAction extends Action {
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {

    HttpSession session = request.getSession();
    DadosPessoaisForm regform = (DadosPessoaisForm)
form;
    AlunoDao alunoDao = new AlunoDao();
    Aluno userCheck = (Aluno)
session.getAttribute(Constants.USER_KEY);
    String action = request.getParameter("action");
    if (userCheck == null) {
      return (mapping.findForward("logon"));
    }
    String cpf = request.getParameter("cpf");
    if("".equals(cpf) || cpf == null){
      cpf = "";
    }

    ActionMessages errors = new ActionMessages();
    String[] estados = { "PR","SC", "RS",
"RR","TO","AM","PB","PE","AP","AC","SP","RJ","MG"
,
        "ES","BA" };
        String[] estadosCivis = {"solteiro","casado"};
        ArrayList options = new ArrayList();
        options.add(new LabelValueBean("masculino",
"M"));
        options.add(new LabelValueBean("feminino",
"F"));
        session.setAttribute("estados", estados);
        session.setAttribute("estadosCivis",
estadosCivis);
        session.setAttribute("sexos", options);
        Aluno user = null;
        try {
          user = alunoDao.buscarAluno(cpf);
        }catch (SQLException sqle) {

errors.add(ActionMessages.GLOBAL_MESSAGE,new
ActionMessage("sqlerror.buscar.aluno"));
        }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.aluno"));
        }

regform.setAnoConclSuperior(user.getAnoConcSuperior()
);

regform.setBairro(user.getEndereco().getBairro());
        regform.setCep(user.getEndereco().getCep());

regform.setComplemento(user.getEndereco().getComplem
ento());
        regform.setCpf(user.getCpf());
        SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
```

```
regform.setDataNascimento(sdf.format(user.getDataNasci
mento().getTime()));
            regform.setEmail(user.getEmail());

regform.setEstado(user.getEndereco().getEstado());
            regform.setEstadoCivil(user.getEstadoCivil());
            regform.setFone(user.getFone());
            regform.setInstitSuperior("Universidade Federal
do Paraná");

regform.setMunicipio(user.getEndereco().getMunicipio());
            regform.setNome(user.getNomePessoa());

regform.setNacionalidade(user.getNacionalidade());
            regform.setNomeMae(user.getNomeMae());
            regform.setNomePai(user.getNomePai());

regform.setNumero(user.getEndereco().getNumero());
            regform.setPais(user.getEndereco().getPais());
            regform.setRg(user.getRg());

regform.setRgNaturalidade(user.getRgNaturalidade());
            regform.setRua(user.getEndereco().getRua());
            regform.setSexo(user.getSexo());
            regform.setAction(action);
            regform.setSenha(user.getSenha());
            regform.setSenha2(user.getSenha());
            regform.setSituacao(user.getSituacao());
            regform.setUsuario(user.getUsuario());
            if (regform.getCpf().equals(null)){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.username.required"));
            }
            if (errors.isEmpty()){
               return mapping.findForward("success");
            } else {
               return mapping.findForward("error");
            }
   }

}
```

6.4.12 EditarEmailAction.java

```
/*
 * EditarEmailAction.java
 * Created on 18 de Maio de 2006, 14:06
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.ServidorSmtp;
import egressos.dao.Servidor_smtp_egressosDao;
import egressos.form.ServidorSmtpForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class EditarEmailAction extends Action {

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {

      HttpSession session = request.getSession();
      ServidorSmtpForm serverform =
(ServidorSmtpForm) form;
      Aluno userCheck = (Aluno)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      if (userCheck == null) {
         return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      Servidor_smtp_egressosDao servidorDao = new
Servidor_smtp_egressosDao();
      ServidorSmtp smtpServer = new ServidorSmtp();
      List<ServidorSmtp> servidores = new
ArrayList<ServidorSmtp>();
      ServidorSmtp servidor = new ServidorSmtp();
      try {
         servidores = servidorDao.listarServidor_smtp();
         if(servidores.isEmpty()){

servidor.setCod_servidor_smtp(Integer.valueOf(1));
            servidor.setEndereco_email("");
            servidor.setPorta_smtp(Integer.valueOf(110));

servidor.setRequer_autenticacao(Boolean.valueOf(false));
            servidor.setSenha("");
            servidor.setUrl("");
            servidor.setUsername("");
            try{
               servidorDao.incluirServidorSmtp(servidor);
            } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.incluir.servidor"));
            }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.incluir.servidor"));
            }
         } else {
            servidor =
servidorDao.buscarServidorSmtp(Integer.valueOf(1));
         }
```

```
serverform.setCod_servidor_smtp(servidor.getCod_servid
or_smtp());

serverform.setEndereco_email(servidor.getEndereco_emai
l());

serverform.setPorta_smtp(servidor.getPorta_smtp());

serverform.setRequer_autenticacao(servidor.getRequer_au
tenticacao());
        serverform.setSenha(servidor.getSenha());
        serverform.setUrl(servidor.getUrl());
        serverform.setUsername(servidor.getUsername());
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.recuperar.dados"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.recuperar.dados"));
      }

      if (errors.isEmpty()){
        return mapping.findForward("success");
      } else {
        return mapping.findForward("error");
      }
    }

}
```

6.4.13 EditarQuestaoAction.java

```
/*
 * EditarQuestaoAction.java
 * Created on 19 de Dezembro de 2005, 19:53
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;

import egressos.dao.QuestaoDao;
import egressos.form.NovaQuestaoForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;


/**
 *
 *
 */
public class EditarQuestaoAction extends Action {

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {


      HttpSession session = request.getSession();
      NovaQuestaoForm questaoform =
(NovaQuestaoForm) form;
      Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      String codQuestao = (String)
request.getParameter("codQuestao");
      QuestaoDao questaoDao = new QuestaoDao();
      Questao novaQuestao = new Questao();
      try {
        novaQuestao =
questaoDao.buscarQuestao(Integer.valueOf(codQuestao));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
      }
      if(novaQuestao == null){
        novaQuestao = new Questao();
      }
      questaoform.setCodQuestao(codQuestao);

questaoform.setCodQuestionario(novaQuestao.getCodQue
stionario().toString());

questaoform.setPergunta(novaQuestao.getPergunta());

questaoform.setAlternativaA(novaQuestao.getAlternativa
A());

questaoform.setAlternativaB(novaQuestao.getAlternativa
B());

questaoform.setAlternativaC(novaQuestao.getAlternativa
C());

questaoform.setAlternativaD(novaQuestao.getAlternativa
D());

questaoform.setAlternativaE(novaQuestao.getAlternativaE
());

questaoform.setOrdem(novaQuestao.getOrdem().toString(
));
      questaoform.setAction(action);
      if("".equals(novaQuestao.getAlternativaA()) &&
"".equals(novaQuestao.getAlternativaB()) &&
          "".equals(novaQuestao.getAlternativaC()) &&
"".equals(novaQuestao.getAlternativaD()) &&
"".equals(novaQuestao.getAlternativaE())){
        questaoform.setQuestaoaberta("sim");
      }
```

```
        request.setAttribute("codQuestionario",
novaQuestao.getCodQuestionario());
        if (errors.isEmpty()){
          return mapping.findForward("success");
        } else {
          return mapping.findForward("error");
        }
    }

}
```

6.4.14 EnviaSenhaAction.java

```
/*
 * EnviaSenhaAction.java
 * Created on 18 de Maio de 2006, 14:16
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.ServidorSmtp;
import egressos.dao.AlunoDao;
import egressos.dao.Servidor_smtp_egressosDao;
import egressos.mail.MailManager;
import egressos.mail.MailServer;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.commons.beanutils.PropertyUtils;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 * @author Ricardo
 */
public class EnviaSenhaAction extends Action{

  /** Creates a new instance of LogonAction */
  public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {


    ActionMessages errors = new ActionMessages();
    String userName = null;
    String email = null;
    userName = (String)
PropertyUtils.getSimpleProperty(form,"username");
    email = (String)
PropertyUtils.getSimpleProperty(form,"email");
    AlunoDao alunoDao = new AlunoDao();
    Servidor_smtp_egressosDao servidorDao = new
Servidor_smtp_egressosDao();
    Aluno user = new Aluno();
    try {
        user =
alunoDao.buscarAlunoPorUsuario(userName);
      } catch (SQLException e){
        errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new
ActionMessage("error.recuperar.dados"));
      } catch (Exception e){
        errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new ActionMessage("error.generico"));
      }
      if (user != null) {
        if(!email.equals(user.getEmail())) {
          errors.add(
                ActionMessages.GLOBAL_MESSAGE,
                new
ActionMessage("error.email.incorreto"));
        }
      } else {
        errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new
ActionMessage("error.usuario.naoexiste"));
      }
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return (mapping.findForward("error"));
      }else {
        try{
          ServidorSmtp servidorSmtp =
servidorDao.listarServidor_smtp().get(0);
          if(servidorSmtp != null){
            MailServer server = new
MailServer(servidorSmtp.getUrl(),servidorSmtp.getUserna
me(),servidorSmtp.getSenha(),servidorSmtp.getRequer_au
tenticacao().booleanValue());
            MailManager m = new MailManager();
            //altera senha e armazenha o hashCode da
nova senha
            String novasenha =
String.valueOf(user.getUsuario().hashCode());
            if(user.getUsuario().hashCode() < 0){
              novasenha =
String.valueOf(user.getUsuario().hashCode()*-1);
            }

user.setSenha(String.valueOf(novasenha.hashCode()));
            try{
              alunoDao.alterarAluno(user);
            }catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.alterar.senha"));
            }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.senha"));
            }
            try{

m.send(server,user.getEmail(),"","",servidorSmtp.getEnder
eco_email(),"Sistema de Controle de Estágios -
UFPR","Recuperação de senha.\nSua senha foi alterada.
```

```
Para acessar o sistema utiliza a senha:\n"+novasenha,
servidorSmtp);
                request.setAttribute("mensagem", "Sua
senha foi enviada. Verifique seu e-mail.");
            }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.enviar.senha"));
            }
          } else {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.servidorsmtp.naoexiste"));
          }
        }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.recuperar.dados"));
        }
        if(errors.isEmpty()){
          return
(mapping.findForward(Constants.SUCCESS));
        } else {
          saveErrors(request, errors );
          return
(mapping.findForward(Constants.FAILURE));
        }
      }
    }
}
```

6.4.15 EstatisticasQuestionarioAction.java

```
/*
 * EstatisticasQuestionarioAction.java
 * Created on 6 de Janeiro de 2006, 14:56
 */

package egressos.action;

import egressos.bean.Alternativa;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.bean.Questionario;
import egressos.bean.Resposta;
import egressos.dao.QuestaoDao;
import egressos.dao.QuestionarioDao;
import egressos.dao.RespostaDao;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class EstatisticasQuestionarioAction extends
Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      ActionMessages errors = new ActionMessages();
      String codQuestionario = "";
      QuestaoDao qDao = new QuestaoDao();
      RespostaDao rDao = new RespostaDao();
      QuestionarioDao questionarioDao = new
QuestionarioDao();
      Questionario questionarioAtivo = new Questionario();
      try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        codQuestionario =
request.getParameter("codQuestionario");
        questionarioAtivo =
questionarioDao.buscarQuestionario(Integer.valueOf(cod
Questionario));
        request.setAttribute("codQuestionario",
codQuestionario);
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
      }
      List<Questao> questoes = new ArrayList<Questao>();
      List<Resposta> respostas = new
ArrayList<Resposta>();
      try{
        questoes =
qDao.listarQuestoes(Integer.valueOf(codQuestionario));
        for(Questao q:questoes){
          Alternativa alternativaA = new Alternativa();
          Alternativa alternativaB = new Alternativa();
          Alternativa alternativaC = new Alternativa();
          Alternativa alternativaD = new Alternativa();
          Alternativa alternativaE = new Alternativa();
          List<Alternativa> alternativas = new
ArrayList<Alternativa>();
          if(!"".equals(q.getAlternativaA()) &&
q.getAlternativaA() != null){

alternativaA.setAlternativa(q.getAlternativaA());
            alternativaA.setIndice("A");
            alternativas.add(alternativaA);
          }
          if(!"".equals(q.getAlternativaB()) &&
q.getAlternativaB() != null){

alternativaB.setAlternativa(q.getAlternativaB());
            alternativaB.setIndice("B");
            alternativas.add(alternativaB);
          }
```

```
            if(!"".equals(q.getAlternativaC()) &&
q.getAlternativaC() != null){

alternativaC.setAlternativa(q.getAlternativaC());
                alternativaC.setIndice("C");
                alternativas.add(alternativaC);
            }
            if(!"".equals(q.getAlternativaD()) &&
q.getAlternativaD() != null){

alternativaD.setAlternativa(q.getAlternativaD());
                alternativaD.setIndice("D");
                alternativas.add(alternativaD);
            }
            if(!"".equals(q.getAlternativaE()) &&
q.getAlternativaE() != null){

alternativaE.setAlternativa(q.getAlternativaE());
                alternativaE.setIndice("E");
                alternativas.add(alternativaE);
            }
            q.setAlternativas(alternativas);
          }
        } catch (NumberFormatException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
        }
        for(Questao q: questoes){
          if(q.getAlternativas().size()>0){
            Integer totalRespostas = q.getTotalRespostas();
            for(Resposta r:q.getRespostas()){
              if("A".equals(r.getResposta())){
                if("".equals(q.getAlternativaA()) ||
q.getAlternativaA() == null){

q.setTotalAlternativaA(Double.valueOf(0));
                  --totalRespostas;
                }
              } else
              if("B".equals(r.getResposta())){
                if("".equals(q.getAlternativaB()) ||
q.getAlternativaB() == null){

q.setTotalAlternativaB(Double.valueOf(0));
                  --totalRespostas;
                }
              } else
              if("C".equals(r.getResposta())){
                if("".equals(q.getAlternativaC()) ||
q.getAlternativaC() == null){

q.setTotalAlternativaC(Double.valueOf(0));
                  --totalRespostas;
                }
              } else
              if("D".equals(r.getResposta())){
                if("".equals(q.getAlternativaD()) ||
q.getAlternativaD() == null){

q.setTotalAlternativaD(Double.valueOf(0));
                  --totalRespostas;
                }
              } else
              if("E".equals(r.getResposta())){
                if("".equals(q.getAlternativaE()) ||
q.getAlternativaE() == null){

q.setTotalAlternativaE(Double.valueOf(0));
                  --totalRespostas;
                }
              }
            }
            if(totalRespostas.intValue() <= 0){
              totalRespostas = Integer.valueOf(1);
            }
            q.setTotalRespostas(totalRespostas);
            Double totalAlternativaA =
q.getTotalAlternativaA();
            Double totalAlternativaB =
q.getTotalAlternativaB();
            Double totalAlternativaC =
q.getTotalAlternativaC();
            Double totalAlternativaD =
q.getTotalAlternativaD();
            Double totalAlternativaE =
q.getTotalAlternativaE();
            Double porcentA =
totalAlternativaA/totalRespostas.doubleValue();
            Double porcentB =
totalAlternativaB/totalRespostas.doubleValue();
            Double porcentC =
totalAlternativaC/totalRespostas.doubleValue();
            Double porcentD =
totalAlternativaD/totalRespostas.doubleValue();
            Double porcentE =
totalAlternativaE/totalRespostas.doubleValue();
            porcentA = porcentA*100;
            porcentB = porcentB*100;
            porcentC = porcentC*100;
            porcentD = porcentD*100;
            porcentE = porcentE*100;
            DecimalFormat df = new
DecimalFormat("#,##0.0#;(#)");
            String totA =
df.format(porcentA.doubleValue());
            String totB = df.format(porcentB.doubleValue());
            String totC = df.format(porcentC.doubleValue());
            String totD =
df.format(porcentD.doubleValue());
            String totE = df.format(porcentE.doubleValue());

q.setTotalAlternativaA(Double.valueOf(totA.replace(',','.'))
);

q.setTotalAlternativaB(Double.valueOf(totB.replace(',','.'))
);

q.setTotalAlternativaC(Double.valueOf(totC.replace(',','.'))
);

q.setTotalAlternativaD(Double.valueOf(totD.replace(',','.'))
);

q.setTotalAlternativaE(Double.valueOf(totE.replace(',','.'))
);
```

```
        }
      }
      request.setAttribute("questoes", questoes);
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.16 ExcluirEgressoAction.java

```
/*
 * ExcluirEgressoAction.java
 * Created on 14 de Janeiro de 2006, 10:28
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questionario;
import egressos.bean.QuestionarioDTO;
import egressos.dao.AlunoDao;
import egressos.dao.QuestionarioDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class ExcluirEgressoAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     AlunoDao alunoDao = new AlunoDao();
     String cpf = "";
     try{
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        cpf = (String) request.getParameter("cpf");
        if (user == null){
          return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
      }
      try {
        Aluno a = new Aluno();
        a = alunoDao.buscarAluno(cpf);
        alunoDao.excluirAluno(a);
      }catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.delete.egresso"));
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.17 ExcluirQuestaoAction.java

```
/*
 * ExcluirQuestaoAction.java
 * Created on 28 de Dezembro de 2005, 13:53
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.dao.QuestaoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;


/**
 *
 *
 */
public class ExcluirQuestaoAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     String codQuestao = "";
```

```
    try{
      Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
      codQuestao = (String)
request.getParameter("codQuestao");
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
      new ActionMessage("error.generico"));
    }
    QuestaoDao qDao = new QuestaoDao();
    List<Questao> questoes = new ArrayList<Questao>();
    Questao questao = null;
    try{
      questao =
qDao.buscarQuestao(Integer.parseInt(codQuestao));
      qDao.excluirQuestao(questao);
      questoes =
qDao.listarQuestoes(questao.getCodQuestionario());
    } catch (NumberFormatException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.excluir.questao"));
    } catch (Exception e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.excluir.questao"));
    }

    if (questoes == null){
      questoes = new ArrayList<Questao>();
    }
    if(questao == null){
      questao = new Questao();
    }
    request.setAttribute("questoes",questoes);
    request.setAttribute("codQuestionario",
questao.getCodQuestionario());
    HashMap map = new HashMap();
    map.put("codQuestionario",
questao.getCodQuestionario());
    request.setAttribute("map", map);
// seta ordem
    HashMap map1 = new HashMap();
    Questao questaoOrdem = null;
    try {
      questaoOrdem =
qDao.buscarUltimaQuestao(Integer.valueOf(questao.getC
odQuestionario()));
    } catch (NumberFormatException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
    }
    if (questaoOrdem == null){
      questaoOrdem = new Questao();
      questaoOrdem.setOrdem(Integer.valueOf(0));
    }
    Integer ultimaOrdem = questaoOrdem.getOrdem();
    if(Integer.valueOf(0).equals(ultimaOrdem)){
      map1.put("ordem","1");
    } else {
      ++ultimaOrdem;
      String ordem = ultimaOrdem.toString();
      map1.put("ordem",ordem);
    }
    request.setAttribute("map1", map1);
//ordem setada
  if(!errors.isEmpty()){
    saveErrors(request, errors);
    return mapping.findForward(Constants.FAILURE);
  } else {
    return mapping.findForward(Constants.SUCCESS);
  }
}
}
```

6.4.18 ExcluirQuestionarioAction.java

```
/*
 * ExcluirQuestionarioAction.java
 * Created on 17 de Dezembro de 2005, 16:32
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questionario;
import egressos.bean.QuestionarioDTO;
import egressos.dao.QuestionarioDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;


/**
 *
 *
 */
public class ExcluirQuestionarioAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     String codQuestionario = "";
     try{
       Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
       codQuestionario = (String)
request.getParameter("codQuestionario");
       request.setAttribute("codQuestionario",
codQuestionario);
```

```
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
      }
      QuestionarioDao qDao = new QuestionarioDao();
      try {
        List<QuestionarioDTO> questionarios =
(ArrayList<QuestionarioDTO>)
session.getAttribute("questionarios");
        for(QuestionarioDTO q : questionarios) {
          if (q.getCodQuestionario() ==
Integer.parseInt(codQuestionario)) {
            Questionario questionario = new
Questionario();
            questionario =
qDao.buscarQuestionario(Integer.valueOf(codQuestionari
o));
            qDao.excluirQuestionario(questionario);
            questionarios.remove(q);
            break;
          }
        }
        session.removeAttribute("questionarios");
        session.setAttribute("questionarios", questionarios);
      }catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.delete.questionario"));
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.19 GerenciarQuestionariosAction.java

```
/*
 * GerenciarQuestionariosAction.java
 * Created on 14 de Dezembro de 2005, 23:53
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questionario;
import egressos.bean.QuestionarioDTO;
import egressos.dao.QuestionarioDao;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class GerenciarQuestionariosAction extends
Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      try{
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
      }
      QuestionarioDao qDao = new QuestionarioDao();
      List<Questionario> questionarios =
qDao.listarQuestionarios();
      List<QuestionarioDTO> qDtos = new ArrayList();
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      for(Questionario q:questionarios){
        QuestionarioDTO qDto = new QuestionarioDTO();
        qDto.setCodQuestionario(q.getCodQuestionario());

qDto.setDataCriacao(sdf.format(q.getDataCriacao().getTi
me()));
        qDto.setStatus("");
        if(Integer.valueOf(1).equals(q.getStatus())){
          qDto.setStatus("Ativo");
        }
        qDtos.add(qDto);
      }
      session.setAttribute("questionarios", qDtos);
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.20 ListarQuestoesAction.java

```
/*
 * ListarQuestoesAction.java
 * Created on 16 de Dezembro de 2005, 18:45
 */
```

```
package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.dao.QuestaoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class ListarQuestoesAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     String codQuestionario = "";
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        codQuestionario = (String)
request.getParameter("codQuestionario");
        request.setAttribute("codQuestionario",
codQuestionario);
        if (user == null){
          return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
     }
     QuestaoDao qDao = new QuestaoDao();
     List<Questao> questoes = new ArrayList<Questao>();
     try{
         questoes =
qDao.listarQuestoes(Integer.valueOf(codQuestionario));
     } catch (NumberFormatException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
     } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
     }
     if (questoes == null){
        questoes = new ArrayList<Questao>();
     }
     HashMap map1 = new HashMap();
     Questao questao =
qDao.buscarUltimaQuestao(Integer.valueOf(codQuestiona
rio));
     if (questao == null){
        questao = new Questao();
        questao.setOrdem(Integer.valueOf(0));
     }
     String ordem = questao.getOrdem().toString();
     if(Integer.valueOf(0).toString().equals(ordem)){
        map1.put("ordem","1");
     } else {
        map1.put("ordem",ordem);
     }
     request.setAttribute("map1", map1);
     HashMap map = new HashMap();
     map.put("codQuestionario", codQuestionario);
     request.setAttribute("map", map);
     request.setAttribute("questoes", questoes);
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.21 LogoffAction.java

```
/*
 * LogoffAction.java
 * Created on 8 de Novembro de 2005, 16:23
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

/**
 *
 *
 */
public class LogoffAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     HttpSession session = request.getSession();
     Aluno user = null;
     user =
(Aluno)session.getAttribute(Constants.USER_KEY);
     if(user != null){
        session.removeAttribute("user");
        session.removeAttribute("logado");
```

```
        session.removeAttribute("codPessoa");
        Cookie logado = new Cookie("logado", "false");
        // we want the cookie to be returned to any app
below root
        logado.setPath("/");
        response.addCookie(logado);
      }
      return (mapping.findForward(Constants.SUCCESS));
    }
}
```

6.4.22 LogonAction.java

```
/*
 * LogonAction.java
 * Created on 7 de Novembro de 2005, 21:04
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.commons.beanutils.PropertyUtils;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;


/**
 *
 *
 */
public class LogonAction extends Action{

  /** Creates a new instance of LogonAction */
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {

      // Local variables
      String userName = (String)
PropertyUtils.getSimpleProperty(form,"username");
      String password = (String)
PropertyUtils.getSimpleProperty(form,"password");
      ActionMessages errors = new ActionMessages();
      Aluno user = new Aluno();
      try {
        AlunoDao dao = new AlunoDao();
        user = dao.buscarAlunoPorUsuario(userName);
      } catch (SQLException e){
        errors.add(
        ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("sqlerror.buscar.usuario"));
      } catch (Exception e){
        errors.add(
        ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.buscar.usuario"));
      }
      if (user != null) {

if(!String.valueOf(password.hashCode()).equals(user.getS
enha())) {
          errors.add(
            ActionMessages.GLOBAL_MESSAGE,
            new
ActionMessage("error.password.incorreto"));
        } else {
          HttpSession session = request.getSession();
          if(session.getAttribute(Constants.USER_KEY)
!= null){

session.removeAttribute(Constants.USER_KEY);
          }
          session.setAttribute(Constants.USER_KEY,
user);
          Cookie codPessoaCookie = new
Cookie("codPessoa", user.getCodPessoa().toString());
          Cookie logado = new Cookie("logado", "true");
          // we want the cookie to be returned to any app
below root
          codPessoaCookie.setPath("/");
          logado.setPath("/");
          response.addCookie(codPessoaCookie);
          response.addCookie(logado);
        }
      } else {
        errors.add(
        ActionMessages.GLOBAL_MESSAGE,
        new
ActionMessage("error.usuario.naoencontrado"));
      }
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return (mapping.findForward("error"));
      }else {
        return (mapping.findForward("success"));
      }
    }
}
```

6.4.23 MainMenuAction.java

```
/*
 * MainMenuAction.java
 * Created on 12 de Janeiro de 2006, 19:20
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
```

```
/**
 *
 *
 */
public class MainMenuAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null ||
"guest".equals(user.getNomePessoa())){
           return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
     }
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.24 MostrarRespostasQuestaoAction.java

```
/*
 * MostrarRespostasQuestaoAction.java
 * Created on 6 de Janeiro de 2006, 17:42
 */

package egressos.action;

import egressos.bean.Alternativa;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.bean.Resposta;
import egressos.bean.RespostaDTO;
import egressos.dao.AlunoDao;
import egressos.dao.QuestaoDao;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class MostrarRespostasQuestaoAction  extends
Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     try{
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
           return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
     }
     QuestaoDao qDao = new QuestaoDao();
     String codQuestao = (String)
request.getParameter("codQuestao");
     Questao q =
qDao.buscarQuestao(Integer.valueOf(codQuestao));
     Alternativa alternativaA = new Alternativa();
     Alternativa alternativaB = new Alternativa();
     Alternativa alternativaC = new Alternativa();
     Alternativa alternativaD = new Alternativa();
     Alternativa alternativaE = new Alternativa();
     List<Alternativa> alternativas = new
ArrayList<Alternativa>();
     if(!"".equals(q.getAlternativaA()) &&
q.getAlternativaA() != null){
        alternativaA.setAlternativa(q.getAlternativaA());
        alternativaA.setIndice("A");
        alternativas.add(alternativaA);
     }
     if(!"".equals(q.getAlternativaB()) &&
q.getAlternativaB() != null){
        alternativaB.setAlternativa(q.getAlternativaB());
        alternativaB.setIndice("B");
        alternativas.add(alternativaB);
     }
     if(!"".equals(q.getAlternativaC()) &&
q.getAlternativaC() != null){
        alternativaC.setAlternativa(q.getAlternativaC());
        alternativaC.setIndice("C");
        alternativas.add(alternativaC);
     }
     if(!"".equals(q.getAlternativaD()) &&
q.getAlternativaD() != null){
        alternativaD.setAlternativa(q.getAlternativaD());
        alternativaD.setIndice("D");
        alternativas.add(alternativaD);
     }
```

```
    if(!"".equals(q.getAlternativaE()) &&
q.getAlternativaE() != null){
      alternativaE.setAlternativa(q.getAlternativaE());
      alternativaE.setIndice("E");
      alternativas.add(alternativaE);
    }
    q.setAlternativas(alternativas);
    if(q.getAlternativas().size()>0){
      Integer totalRespostas = q.getTotalRespostas();
      for(Resposta r:q.getRespostas()){
        if("A".equals(r.getResposta())){
          if("".equals(q.getAlternativaA()) ||
q.getAlternativaA() == null){
            q.setTotalAlternativaA(Double.valueOf(0));
            --totalRespostas;
          }
        }
        if("B".equals(r.getResposta())){
          if("".equals(q.getAlternativaB()) ||
q.getAlternativaB() == null){
            q.setTotalAlternativaB(Double.valueOf(0));
            --totalRespostas;
          }
        }
        if("C".equals(r.getResposta())){
          if("".equals(q.getAlternativaC()) ||
q.getAlternativaC() == null){
            q.setTotalAlternativaC(Double.valueOf(0));
            --totalRespostas;
          }
        }
        if("D".equals(r.getResposta())){
          if("".equals(q.getAlternativaD()) ||
q.getAlternativaD() == null){
            q.setTotalAlternativaD(Double.valueOf(0));
            --totalRespostas;
          }
        }
        if("E".equals(r.getResposta())){
          if("".equals(q.getAlternativaE()) ||
q.getAlternativaE() == null){
            q.setTotalAlternativaE(Double.valueOf(0));
            --totalRespostas;
          }
        }
      }
      if(totalRespostas.intValue() <= 0){
        totalRespostas = Integer.valueOf(1);
      }
      q.setTotalRespostas(totalRespostas);
      Double totalAlternativaA =
q.getTotalAlternativaA();
      Double totalAlternativaB =
q.getTotalAlternativaB();
      Double totalAlternativaC =
q.getTotalAlternativaC();
      Double totalAlternativaD =
q.getTotalAlternativaD();
      Double totalAlternativaE =
q.getTotalAlternativaE();
      Double porcentA =
totalAlternativaA/totalRespostas.doubleValue();
      Double porcentB =
totalAlternativaB/totalRespostas.doubleValue();
      Double porcentC =
totalAlternativaC/totalRespostas.doubleValue();
      Double porcentD =
totalAlternativaD/totalRespostas.doubleValue();
      Double porcentE =
totalAlternativaE/totalRespostas.doubleValue();
      porcentA = porcentA*100;
      porcentB = porcentB*100;
      porcentC = porcentC*100;
      porcentD = porcentD*100;
      porcentE = porcentE*100;
      DecimalFormat df = new
DecimalFormat("#,##0.0#;(#)");
      String totA = df.format(porcentA.doubleValue());
      String totB = df.format(porcentB.doubleValue());
      String totC = df.format(porcentC.doubleValue());
      String totD = df.format(porcentD.doubleValue());
      String totE = df.format(porcentE.doubleValue());

q.setTotalAlternativaA(Double.valueOf(totA.replace(',','.'))
);

q.setTotalAlternativaB(Double.valueOf(totB.replace(',','.'))
);

q.setTotalAlternativaC(Double.valueOf(totC.replace(',','.'))
);

q.setTotalAlternativaD(Double.valueOf(totD.replace(',','.'))
);

q.setTotalAlternativaE(Double.valueOf(totE.replace(',','.'))
);
    }
    List<RespostaDTO> respostas = new
ArrayList<RespostaDTO>();
    AlunoDao alunoDao = new AlunoDao();
    for(Resposta r:q.getRespostas()){
      RespostaDTO rDto = new RespostaDTO();
      Aluno aluno;
      if(r.getResposta().length()>1){
        try{
          aluno = alunoDao.buscarAluno(r.getCpf());
        } catch (Exception e) {
          aluno = new Aluno();
        }
        rDto.setNomeAluno(aluno.getNomePessoa());
        rDto.setResposta(r.getResposta());
        respostas.add(rDto);
      }
    }
    if(respostas.size()>0){
      request.setAttribute("embranco", "embranco");
    }
    request.setAttribute("questao", q);
    request.setAttribute("respostas", respostas);
    request.setAttribute("codQuestao",
q.getCodQuestao().toString());
    request.setAttribute("codQuestionario",
q.getCodQuestionario());
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
```

```
        return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.25 NovaQuestaoAction.java

```
/*
 * NovaQuestaoAction.java
 * Created on 17 de Dezembro de 2005, 18:26
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.bean.Questionario;
import egressos.bean.QuestionarioDTO;
import egressos.dao.QuestaoDao;
import egressos.dao.QuestionarioDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

/**
 *
 *
 */
public class NovaQuestaoAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     try{
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
     }
     String codQuestionario =
request.getParameter("codQuestionario");
     if (codQuestionario == null){
        codQuestionario = "";
     }
     request.setAttribute("codQuestionario",
codQuestionario);
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return
mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.26 NovoQuestionarioAction.java

```
/*
 * NovoQuestionarioAction.java
 * Created on 17 de Dezembro de 2005, 17:02
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.bean.Questionario;
import egressos.bean.QuestionarioDTO;
import egressos.dao.QuestaoDao;
import egressos.dao.QuestionarioDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.HashMap;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class NovoQuestionarioAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     try{
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("logon");
        }
     } catch (Exception e) {
```

```
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
      }
      List<QuestionarioDTO> questionariosDto =
(ArrayList<QuestionarioDTO>)
session.getAttribute("questionarios");
      QuestionarioDao qDao = new QuestionarioDao();
      try {
        Questionario quest = new Questionario();
        quest.setDataCriacao(new GregorianCalendar());
        quest.setStatus(0);
        qDao.incluirQuestionario(quest);
        quest = null;
        Questionario q = qDao.buscarNovoQuestionario();

        QuestionarioDTO qDto = new QuestionarioDTO();
        qDto.setCodQuestionario(q.getCodQuestionario());
        SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");

qDto.setDataCriacao(sdf.format(q.getDataCriacao().getTi
me()));
        qDto.setStatus("");
        if(Integer.valueOf(1).equals(q.getStatus())){
          qDto.setStatus("Ativo");
        }
        questionariosDto.add(qDto);
        session.removeAttribute("questionarios");
        session.setAttribute("questionarios",
questionariosDto);
        request.setAttribute("codQuestionario",
qDto.getCodQuestionario());
        QuestaoDao questaoDao = new QuestaoDao();
        List<Questao> questoes = new
ArrayList<Questao>();
        try{
          questoes =
questaoDao.listarQuestoes(q.getCodQuestionario());
        } catch (NumberFormatException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
        } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }
        if (questoes == null){
          questoes = new ArrayList<Questao>();
        }
        request.setAttribute("questoes", questoes);
// seta ordem
        HashMap map1 = new HashMap();
        Questao questao = null;
        try {
          questao =
questaoDao.buscarUltimaQuestao(Integer.valueOf(qDto.g
etCodQuestionario()));
        } catch (NumberFormatException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.convnumerica"));
        }
        if (questao == null){
          questao = new Questao();
          questao.setOrdem(Integer.valueOf(0));
        }
        Integer ultimaOrdem = questao.getOrdem();
        if(Integer.valueOf(0).equals(ultimaOrdem)){
          map1.put("ordem","1");
        } else {
          ++ultimaOrdem;
          String ordem = ultimaOrdem.toString();
          map1.put("ordem",ordem);
        }
        request.setAttribute("map1", map1);
//ordem setada
      }catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.delete.questionario"));
      }

      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return
mapping.findForward(Constants.FAILURE);
      } else {
        return
mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.27 NovoUsuarioAction.java

```
/*
 * NovoUsuarioAction.java
 * Created on 15 de Novembro de 2005, 11:12
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
/**
 *
 *
 */
public class NovoUsuarioAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
```

```
    ActionMessages errors = new ActionMessages();
    try{
      HttpSession session = request.getSession();
      Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        user = new Aluno();
        user.setSituacao("egresso");
        user.setNomePessoa("guest");

session.setAttribute(Constants.USER_KEY,user);
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
       new ActionMessage("error.generico"));
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.28 NovoUsuarioAdminAction.java

```
/*
 * NovoUsuarioAdminAction.java
 * Created on 12 de Janeiro de 2006, 23:04
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

/**
 *
 *
 */
public class NovoUsuarioAdminAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     HttpSession session = request.getSession();
     try{
       Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
       if (user == null){
         return mapping.findForward("logon");
       }
     } catch (Exception e) {
       errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
     }
     String[] estados = { "PR","SC", "RS",
"RR","TO","AM","PB","PE","AP","AC","SP","RJ","MG"
,
          "ES","BA" };
     String[] estadosCivis = {"solteiro","casado"};
     ArrayList options = new ArrayList();
     options.add(new LabelValueBean("masculino",
"M"));
     options.add(new LabelValueBean("feminino", "F"));
     session.setAttribute("estados", estados);
     session.setAttribute("estadosCivis", estadosCivis);
     session.setAttribute("sexos", options);
     if(!errors.isEmpty()){
       saveErrors(request, errors);
       return mapping.findForward(Constants.FAILURE);
     } else {
       return mapping.findForward(Constants.SUCCESS);
     }
   }
}
```

6.4.29 ObterUsuarioAction.java

```
/*
 * ObterUsuarioAction.java
 * Created on 12 de Janeiro de 2006, 20:33
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class ObterUsuarioAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
```

```
    try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
            return mapping.findForward("logon");
        }
    } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
    }
    AlunoDao alDao = new AlunoDao();
    List<Aluno> alunos = new ArrayList<Aluno>();
    String cpf = request.getParameter("cpf");
    String nome = request.getParameter("nome");
    if(cpf != null && !"".equals(cpf)){
        Aluno aluno = alDao.buscarAluno(cpf);
        if(aluno.getNomePessoa() != null &&
!"".equals(aluno.getNomePessoa())){
            alunos.add(aluno);
        }
    }
    if(nome != null && !"".equals(nome)){
        alunos.addAll(alDao.listarAlunosNome(nome));
        }
    if(alunos.size() == 0){
        request.setAttribute("semresultados", "Egresso não
encontrado");
    } else {
        request.setAttribute("alunos",alunos);
    }
    if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
    } else {
        return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.30 RegistrarEgressoAction.java

```
/*
 * RegistrarEgressoAction.java
 * Created on 20 de Novembro de 2005, 15:31
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

/**
 *
 */
public class RegistrarEgressoAction extends Action {

  public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {

  ActionMessages errors = new ActionMessages();

  String[] estados = { "PR","SC", "RS",
"RR","TO","AM","PB","PE","AP","AC","SP","RJ","MG"
,
    "ES","BA" };
  String[] estadosCivis = {"solteiro","casado"};
  ArrayList options = new ArrayList();
  options.add(new LabelValueBean("masculino", "M"));
  options.add(new LabelValueBean("feminino", "F"));
  try{
        HttpSession session = request.getSession();
        session.setAttribute("estados", estados);
        session.setAttribute("estadosCivis", estadosCivis);
        session.setAttribute("sexos", options);
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
            user = new Aluno();
            user.setNomePessoa("guest");
            user.setSituacao("egresso");

session.setAttribute(Constants.USER_KEY,user);
        }
    } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
      }
    if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
    }
    else{
       return mapping.findForward(Constants.SUCCESS);
    }
    }

}
```

6.4.31 RelatorioAnoAction.java

```
/*
 * RelatorioAnoAction.java
 * Created on 12 de Janeiro de 2006, 18:17
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.AlunoDTO;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import java.text.SimpleDateFormat;
```

```
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class RelatorioAnoAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     AlunoDao alunoDao = new AlunoDao();
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
     }
     List<Aluno> alunos = alunoDao.listarAlunosAno();
     if(alunos == null){
        alunos = new ArrayList<Aluno>();
     }
     List<AlunoDTO> alunosDTO = new
ArrayList<AlunoDTO>();
     SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
     for(Aluno al:alunos){
        AlunoDTO alDto = new AlunoDTO();
        if(!"admin".equals(al.getSituacao())){

alDto.setAnoConcSuperior(al.getAnoConcSuperior().toStr
ing());
           alDto.setCpf(al.getCpf());

alDto.setDataNascimento(sdf.format(al.getDataNasciment
o().getTime()));
           alDto.setEmail(al.getEmail());
           alDto.setEndereco(al.getEndereco().toString());
           alDto.setEstadoCivil(al.getEstadoCivil());
           alDto.setFone(al.getFone());
           alDto.setNacionalidade(al.getNacionalidade());
           alDto.setNomeAluno(al.getNomePessoa());
           alDto.setNomePai(al.getNomePai());
           alDto.setNomeMae(al.getNomeMae());
           alDto.setRg(al.getRg());
alDto.setRgNaturalidade(al.getRgNaturalidade());
           alDto.setSenha(al.getSenha());
           alDto.setSexo(al.getSexo());
           alunosDTO.add(alDto);
        }
     }
     request.setAttribute("alunos", alunosDTO);
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.32 RelatorioNomeAction.java

```
/*
 * RelatorioNomeAction.java
 * Created on 12 de Janeiro de 2006, 20:02
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.AlunoDTO;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class RelatorioNomeAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     AlunoDao alunoDao = new AlunoDao();
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("logon");
        }
     } catch (Exception e) {
```

```
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
      }
      List<Aluno> alunos = alunoDao.listarAlunosNome();
      if(alunos == null){
        alunos = new ArrayList<Aluno>();
      }
      List<AlunoDTO> alunosDTO = new
ArrayList<AlunoDTO>();
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      for(Aluno al:alunos){
        AlunoDTO alDto = new AlunoDTO();
        if(!"admin".equals(al.getSituacao())){

alDto.setAnoConcSuperior(al.getAnoConcSuperior().toStr
ing());
            alDto.setCpf(al.getCpf());

alDto.setDataNascimento(sdf.format(al.getDataNasciment
o().getTime()));
            alDto.setEmail(al.getEmail());
            alDto.setEndereco(al.getEndereco().toString());
            alDto.setEstadoCivil(al.getEstadoCivil());
            alDto.setFone(al.getFone());
            alDto.setNacionalidade(al.getNacionalidade());
            alDto.setNomeAluno(al.getNomePessoa());
            alDto.setNomePai(al.getNomePai());
            alDto.setNomeMae(al.getNomeMae());
            alDto.setRg(al.getRg());

alDto.setRgNaturalidade(al.getRgNaturalidade());
            alDto.setSenha(al.getSenha());
            alDto.setSexo(al.getSexo());
            alunosDTO.add(alDto);
        }
      }
      request.setAttribute("alunos", alunosDTO);
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.33 RelatoriosAction.java

```
/*
 * RelatoriosAction.java
 * Created on 12 de Janeiro de 2006, 18:10
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class RelatoriosAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
           return mapping.findForward("logon");
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
         new ActionMessage("error.generico"));
     }
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.34 ResponderQuestionarioAction.java

```
/*
 * ResponderQuestionarioAction.java
 * Created on 29 de Dezembro de 2005, 21:45
 */

package egressos.action;

import egressos.bean.Alternativa;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.bean.Questionario;
import egressos.dao.QuestaoDao;
import egressos.dao.QuestionarioDao;
import egressos.dao.RespostaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
```

```
/**
 *
 *
 */
public class ResponderQuestionarioAction extends
Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
     ActionMessages errors = new ActionMessages();
     String codQuestionario = "";
     QuestaoDao qDao = new QuestaoDao();
     QuestionarioDao questionarioDao = new
QuestionarioDao();
     Questionario questionarioAtivo = new Questionario();
     RespostaDao respostaDao = new RespostaDao();
     try{
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        questionarioAtivo =
questionarioDao.buscarQuestionarioAtivo();
        if(questionarioAtivo.getCodQuestionario() != null)
{
           codQuestionario =
questionarioAtivo.getCodQuestionario().toString();
        } else {
           errors.add(codQuestionario, new
ActionMessage("error.questionarioativo.naodefinido"));
           saveErrors(request, errors);
           return
mapping.findForward(Constants.FAILURE);
        }
        request.setAttribute("codQuestionario",
codQuestionario);
        if (user == null){
           return mapping.findForward("logon");
        }
        if(respostaDao.questaoRespondida(user.getCpf(),
questionarioAtivo.getCodQuestionario())){
           errors.add(codQuestionario, new
ActionMessage("error.questionariorespondido"));
           saveErrors(request, errors);
           return
mapping.findForward(Constants.FAILURE);
        }
     } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
        new ActionMessage("error.generico"));
     }
     List<Questao> questoes = new ArrayList<Questao>();
     try{
        questoes =
qDao.listarQuestoes(Integer.valueOf(codQuestionario));
        for(Questao q:questoes){
           Alternativa alternativaA = new Alternativa();
           Alternativa alternativaB = new Alternativa();
           Alternativa alternativaC = new Alternativa();
           Alternativa alternativaD = new Alternativa();
           Alternativa alternativaE = new Alternativa();
           List<Alternativa> alternativas = new
ArrayList<Alternativa>();
           if(!"".equals(q.getAlternativaA()) &&
q.getAlternativaA() != null){

alternativaA.setAlternativa(q.getAlternativaA());
              alternativaA.setIndice("A");
              alternativas.add(alternativaA);
           }
           if(!"".equals(q.getAlternativaB()) &&
q.getAlternativaB() != null){

alternativaB.setAlternativa(q.getAlternativaB());
              alternativaB.setIndice("B");
              alternativas.add(alternativaB);
           }
           if(!"".equals(q.getAlternativaC()) &&
q.getAlternativaC() != null){

alternativaC.setAlternativa(q.getAlternativaC());
              alternativaC.setIndice("C");
              alternativas.add(alternativaC);
           }
           if(!"".equals(q.getAlternativaD()) &&
q.getAlternativaD() != null){

alternativaD.setAlternativa(q.getAlternativaD());
              alternativaD.setIndice("D");
              alternativas.add(alternativaD);
           }
           if(!"".equals(q.getAlternativaE()) &&
q.getAlternativaE() != null){

alternativaE.setAlternativa(q.getAlternativaE());
              alternativaE.setIndice("E");
              alternativas.add(alternativaE);
           }
           q.setAlternativas(alternativas);
        }
     } catch (NumberFormatException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
     } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
     }
     if (questoes == null){
        questoes = new ArrayList<Questao>();
     }
     request.setAttribute("questoes", questoes);
     if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
     } else {
        return mapping.findForward(Constants.SUCCESS);
     }
  }
}
```

6.4.35 SalvarCadastroAction.java

```
/*
 * SalvarCadastroAction.java
 * Created on 11 de Novembro de 2005, 22:41
```

```
 */

package egressos.action;

import egressos.bean.Constants;
import egressos.bean.User;
import egressos.dao.UserDao;
import egressos.form.RegistrationForm;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.commons.beanutils.PropertyUtils;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarCadastroAction extends Action {

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {


      HttpSession session = request.getSession();
      RegistrationForm regform = (RegistrationForm)
form;
      User user = (User)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      UserDao dao = new UserDao();
      if (action == null) {
        action = "Create";
      }
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      if("Create".equals(action)){
        String value = null;
        value = regform.getUserName();
        User testUser = dao.buscarUser(value);
        if (testUser.getUserName() != null) {
          errors.add(
             "username",
             new ActionMessage("error.username.unique",
regform.getUserName()));
        }
        value = regform.getPin();
        if ((value == null) || (value.length() < 1)) {
          errors.add("password", new
ActionMessage("error.password.required"));
        }
        value = regform.getPassword2();
        if ((value == null) || (value.length() < 1)) {
          errors.add(
             "password2",
             new
ActionMessage("error.password2.required"));
        }
      }
      User novo = new User();
      novo.setUserName(regform.getUserName());
      novo.setPin(regform.getPin());
      novo.setFullName(regform.getFullName());
      novo.setEmail(regform.getEmail());
      if (regform.getGrupo() == null){
        novo.setGrupo("outro");
      } else {
        novo.setGrupo(regform.getGrupo());
      }
      String oldPassword = user.getPin();
      if ((regform.getPin() == null)) {
        novo.setPin(oldPassword);
      }
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        if (user.getUserName().equals("guest")){
          return mapping.findForward("logon");
        }
        return
(mapping.findForward(Constants.FAILURE));
      } else{
        if("Create".equals(action)) {
          dao.incluirUser(novo);
          if (user.getUserName().equals("guest")){
            return mapping.findForward("logon");
          }
          return
(mapping.findForward(Constants.SUCCESS));
        } else {
          dao.alterarUser(novo);

session.removeAttribute(Constants.USER_KEY);
          session.setAttribute(Constants.USER_KEY,
novo);
          return
(mapping.findForward(Constants.SUCCESS));
        }
      }
   }
}
```

6.4.36 SalvarQuestaoAction.java

```
/*
 * SalvarQuestaoAction.java
 * Created on 19 de Dezembro de 2005, 19:03
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.dao.QuestaoDao;
import egressos.form.NovaQuestaoForm;
import java.sql.SQLException;
import java.util.ArrayList;
```

```
import java.util.HashMap;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarQuestaoAction extends Action {

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {


      HttpSession session = request.getSession();
      NovaQuestaoForm questaoform =
(NovaQuestaoForm) form;
      Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      String questaoAberta =
request.getParameter("questaoaberta");
      QuestaoDao questaoDao = new QuestaoDao();
      if(questaoAberta == null){
        questaoAberta = "";
      }
      if (action == null) {
        action = "Create";
      }
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      String value = null;
      value = questaoform.getPergunta();
      if (value == null || "".equals(value)){
        errors.add(
              "pergunta",
              new
ActionMessage("error.pergunta.embranco",
questaoform.getPergunta()));
      }
      Questao novaQuestao = new Questao();
      try{

novaQuestao.setPergunta(questaoform.getPergunta());
         if("sim".equals(questaoAberta)){
           novaQuestao.setAlternativaA("");
           novaQuestao.setAlternativaB("");
           novaQuestao.setAlternativaC("");
           novaQuestao.setAlternativaD("");
           novaQuestao.setAlternativaE("");
         } else {

novaQuestao.setAlternativaA(questaoform.getAlternativa
A());

novaQuestao.setAlternativaB(questaoform.getAlternativa
B());

novaQuestao.setAlternativaC(questaoform.getAlternativa
C());

novaQuestao.setAlternativaD(questaoform.getAlternativa
D());

novaQuestao.setAlternativaE(questaoform.getAlternativaE
());
         }

novaQuestao.setCodQuestionario(Integer.valueOf(questao
form.getCodQuestionario()));
         novaQuestao.setGabarito("");

novaQuestao.setOrdem(Integer.valueOf(questaoform.getO
rdem()));
      } catch (NumberFormatException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
      }
      if (novaQuestao == null){
        novaQuestao = new Questao();
      }

request.setAttribute("codQuestionario",questaoform.getCo
dQuestionario());
      List<Questao> questoes = new
ArrayList<Questao>();
      HashMap map = new HashMap();
      map.put("codQuestionario",
novaQuestao.getCodQuestionario());
      request.setAttribute("map", map);
      if("Create".equals(action)) {
        questaoDao.incluirQuestao(novaQuestao);
        try{
          questoes =
questaoDao.listarQuestoes(Integer.valueOf(novaQuestao.g
etCodQuestionario()));
        } catch (NumberFormatException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.convnumerica"));
        } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }
        if (questoes == null){
          questoes = new ArrayList<Questao>();
        }
        HashMap map1 = new HashMap();
        Questao questao =
questaoDao.buscarUltimaQuestao(novaQuestao.getCodQu
estionario());
        if (questao == null){
          questao = new Questao();
```

```
            questao.setOrdem(Integer.valueOf(0));
        }
        Integer ultimaOrdem = questao.getOrdem();
        if(Integer.valueOf(0).equals(ultimaOrdem)){
            map1.put("ordem","1");
        } else {
            ++ultimaOrdem;
            String ordem = ultimaOrdem.toString();
            map1.put("ordem",ordem);
        }
        request.setAttribute("map1", map1);
        request.setAttribute("questoes", questoes);
    } else {

novaQuestao.setCodQuestao(Integer.valueOf(questaoform
.getCodQuestao()));
        questaoDao.alterarQuestao(novaQuestao);
        try{
            questoes =
questaoDao.listarQuestoes(Integer.valueOf(novaQuestao.g
etCodQuestionario()));
        } catch (NumberFormatException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
        } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }
        if (questoes == null){
            questoes = new ArrayList<Questao>();
        }
        request.setAttribute("questoes", questoes);
        HashMap map1 = new HashMap();
        Questao questao =
questaoDao.buscarUltimaQuestao(novaQuestao.getCodQu
estionario());
        if (questao == null){
            questao = new Questao();
            questao.setOrdem(Integer.valueOf(0));
        }
        Integer ultimaOrdem = questao.getOrdem();
        if(Integer.valueOf(0).equals(ultimaOrdem)){
            map1.put("ordem","1");
        } else {
            ++ultimaOrdem;
            String ordem = ultimaOrdem.toString();
            map1.put("ordem",ordem);
        }
        request.setAttribute("map1", map1);
    }
    if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return
(mapping.findForward(Constants.FAILURE));
    } else {
        return
(mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.37 SalvarRegistroEgressoAction.java

```
/*
 * SalvarRegistroEgressoAction.java
 * Created on 20 de Novembro de 2005, 16:06
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Endereco;
import egressos.bean.Pessoa;
import egressos.dao.AlunoDao;
import egressos.dao.EnderecoPessoaDao;
import egressos.dao.PessoaDao;
import egressos.form.DadosPessoaisForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

/**
 *
 * @author Ricardo
 */
public class SalvarRegistroEgressoAction extends Action
{

    public ActionForward execute(
            ActionMapping mapping,
            ActionForm form,
            HttpServletRequest request,
            HttpServletResponse response)
            throws Exception {


        HttpSession session = request.getSession();
        DadosPessoaisForm regform = (DadosPessoaisForm)
form;
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        String action = request.getParameter("action");
        //verifica se o usuario fez login na intranet academica
        String logou = (String)
session.getAttribute("logado");
        boolean logado = false;
        String codPessoa = null;
        if("true".equals(logou)){
            logado = true;
        }
        AlunoDao alunoDao = new AlunoDao();
        EnderecoPessoaDao enderecoDao = new
EnderecoPessoaDao();
        if (action == null) {
```

```
        action = "Create";
      }
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      String[] estados = { "PR","SC", "RS",
"RR","TO","AM","PB","PE","AP","AC","SP","RJ","MG"
,"ES","BA" };
      String[] estadosCivis = {"solteiro","casado"};
      ArrayList options = new ArrayList();
      options.add(new LabelValueBean("masculino",
"M"));
      options.add(new LabelValueBean("feminino", "F"));
      session.setAttribute("estados", estados);
      session.setAttribute("estadosCivis", estadosCivis);
      session.setAttribute("sexos", options);
      if("Create".equals(action)){
        String value = null;
        value = regform.getCpf();
        Aluno testAluno = alunoDao.buscarAluno(value);
        if (testAluno.getCpf() != null){
          errors.add(
              "cpf",
              new ActionMessage("error.cpf.unique",
regform.getCpf()));
        }
        PessoaDao pessoaDao = new PessoaDao();
        Pessoa pessoaCheck =
pessoaDao.buscarPessoa(regform.getUsuario());
        //verifica se o nome de usuario informado já está
sendo utilizado
        if (pessoaCheck != null){
          //se o nome de usuario já estiver sendo utilizado,
verifica se é o próprio usuário q está logado
          if(logado == true){
            codPessoa = (String)
session.getAttribute("codPessoa");
            Pessoa pessoaLogada =
pessoaDao.buscarPessoa(Integer.valueOf(codPessoa));
            //se não for o usuário logado que está tentando
utilizar o username emite um erro e não insere nada no
banco
            if(!pessoaLogada.equals(pessoaCheck)){
              errors.add(
                  "usuario",
                  new
ActionMessage("error.usuario.unique",
regform.getUsuario()));
            }
            /*se for o mesmo usuario logado que está
tentando cadastrar-se no modulo egressos,
                prossegue normalmente. O método
incluirAluno em AlunoDao previne
                a tentativa de nova inserção da mesma
pessoa na tabela pessoa.*/
          }
        }
        Integer ano = regform.getAnoConclSuperior();
        if((ano == null) || ( ano < Integer.valueOf(1950))){
          errors.add(
              "anoConcSuperior",
              new
ActionMessage("error.anoConcSuperior"));
        }
        ano = null;
      }
      Aluno novoAluno = new Aluno();
      novoAluno.setNomePessoa(regform.getNome());
      novoAluno.setEmail(regform.getEmail());
      novoAluno.setCpf(regform.getCpf());
      novoAluno.setFone(regform.getFone());
      novoAluno.setUsuario(regform.getUsuario());
      if (regform.getSituacao() == null ||
"".equals(regform.getSituacao())){
        novoAluno.setSituacao("egresso");
      } else {
        novoAluno.setSituacao(regform.getSituacao());
      }
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      GregorianCalendar dataNascimento = new
GregorianCalendar();

dataNascimento.setTime(sdf.parse(regform.getDataNasci
mento()));
      novoAluno.setDataNascimento(dataNascimento);
      novoAluno.setEstadoCivil(regform.getEstadoCivil());

novoAluno.setNacionalidade(regform.getNacionalidade())
;
      novoAluno.setNomeMae(regform.getNomeMae());
      novoAluno.setNomePai(regform.getNomePai());

novoAluno.setRgNaturalidade(regform.getRgNaturalidade
());
      novoAluno.setRg(regform.getRg());
      novoAluno.setSexo(regform.getSexo());
      Endereco endereco = new Endereco();
      endereco.setOwner(regform.getCpf());
      endereco.setBairro(regform.getBairro());
      endereco.setCep(regform.getCep());

endereco.setComplemento(regform.getComplemento());
      endereco.setEstado(regform.getEstado());
      endereco.setMunicipio(regform.getMunicipio());
      endereco.setNumero(regform.getNumero());
      endereco.setPais(regform.getPais());
      endereco.setRua(regform.getRua());
      novoAluno.setEndereco(endereco);

novoAluno.setAnoConcSuperior(regform.getAnoConclSu
perior());
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        if (user.getNomePessoa().equals("guest")){
          return mapping.findForward("logon");
        } else
          return
(mapping.findForward(Constants.FAILURE));
      }
      if("Create".equals(action)) {
        //insere no banco o codigo hash da senha
informada

novoAluno.setSenha(String.valueOf(regform.getSenha().h
ashCode()));
        try{
```

```
            alunoDao.incluirAluno(novoAluno);
        } catch (SQLException sqle) {

errors.add(ActionMessages.GLOBAL_MESSAGE,new
ActionMessage("sqlerror.incluir.aluno"));
        } catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.incluir.aluno"));
        }
      } else {
        try {
          //nao altera a senha nessa classe. Seta a senha
com a mesma senha de antes.
          Aluno alunoAntesAlterar =
alunoDao.buscarAluno(regform.getCpf());

novoAluno.setSenha(alunoAntesAlterar.getSenha());
        }catch (SQLException sqle) {

errors.add(ActionMessages.GLOBAL_MESSAGE,new
ActionMessage("sqlerror.buscar.aluno"));
        }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.aluno"));
        }
        try{
          alunoDao.alterarAluno(novoAluno);
        } catch (SQLException sqle) {

errors.add(ActionMessages.GLOBAL_MESSAGE,new
ActionMessage("sqlerror.alterar.aluno"));
        } catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.aluno"));
        }
        if(novoAluno.getCpf().equals(user.getCpf())){

session.removeAttribute(Constants.USER_KEY);
          session.setAttribute(Constants.USER_KEY,
novoAluno);
        }
      }

      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        if (user.getNomePessoa().equals("guest")){
          return mapping.findForward("logon");
        } else
          return
(mapping.findForward(Constants.FAILURE));
      } else {
        if (user.getNomePessoa().equals("guest")){
          return mapping.findForward("logon");
        } else
          return
(mapping.findForward(Constants.SUCCESS));
      }
    }
}
```

6.4.38 SalvarRespostasQuestionarioAction.java

```
/*
 * SalvarRespostasQuestionarioAction.java
 * Created on 4 de Janeiro de 2006, 18:12
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Questao;
import egressos.bean.Questionario;
import egressos.bean.Resposta;
import egressos.dao.QuestaoDao;
import egressos.dao.QuestionarioDao;
import egressos.dao.RespostaDao;
import egressos.form.ResponderQuestionarioForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarRespostasQuestionarioAction extends
Action {

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {


      HttpSession session = request.getSession();
      ResponderQuestionarioForm respostasForm =
(ResponderQuestionarioForm) form;
      Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
      String cpf = user.getCpf();
      QuestionarioDao questionarioDao = new
QuestionarioDao();
      QuestaoDao questaoDao = new QuestaoDao();
      RespostaDao respostaDao = new RespostaDao();
      String codQuestionario =
respostasForm.getCodQuestionario();
      Questionario questionario = new Questionario();
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      try{
```

```
        questionario =
questionarioDao.buscarQuestionario(Integer.valueOf(cod
Questionario));
      } catch (NumberFormatException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
      } catch (NullPointerException e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.null"));
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.generico"));
      }
      for(Questao q:questionario.getQuestoes()){
        String resposta =
request.getParameter("questao"+q.getOrdem());
        if(resposta == null){
          resposta = "";
        }
        if(!"".equals(resposta)){
          Resposta r = new Resposta();
          r.setCodQuestao(q.getCodQuestao());
          r.setCodQuestionario(q.getCodQuestionario());
          r.setCpf(cpf);
          r.setResposta(resposta);
          try {
            respostaDao.incluirResposta(r);
          } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.resposta.incluir"));
          }
        }
      }
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        request.setAttribute("codQuestionario",
codQuestionario);
        return
(mapping.findForward(Constants.FAILURE));
      } else {
        return
(mapping.findForward(Constants.SUCCESS));
      }
    }
}
```

6.4.39 SalvarSenhaAlteradaAction.java

```
/*
 * SalvarSenhaAlteradaAction.java
 * Created on 19 de Junho de 2006, 21:54
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import egressos.form.AlterarSenhaForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarSenhaAlteradaAction extends Action{

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      Aluno user = null;
      AlunoDao alunoDao = new AlunoDao();
      try{
        user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      AlterarSenhaForm alterarSenhaForm =
(AlterarSenhaForm) form;
      Aluno aluno = null;
      try{
        aluno =
alunoDao.buscarAluno(alterarSenhaForm.getCpf());
        //armazenha o hashCode da nova senha

aluno.setSenha(String.valueOf(alterarSenhaForm.getSenha
().hashCode()));
        try{
          alunoDao.alterarAluno(aluno);
        }catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.alterar.senha"));
        }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.senha"));
        }
      }catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuario"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuario"));
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return
mapping.findForward(Constants.FAILURE);
      } else {
```

```
        return
mapping.findForward(Constants.SUCCESS);
      }
   }

}
```

6.4.40 SalvarServidorSmtpAction.java

```
/*
 * SalvarServidorSmtpAction.java
 * Created on 18 de Maio de 2006, 14:12
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.ServidorSmtp;
import egressos.dao.Servidor_smtp_egressosDao;
import egressos.form.ServidorSmtpForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarServidorSmtpAction extends Action {

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {


      HttpSession session = request.getSession();
      ServidorSmtpForm serverForm =
(ServidorSmtpForm) form;
      Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
      Servidor_smtp_egressosDao serverDao = new
Servidor_smtp_egressosDao();
      if (user == null) {
         return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      ServidorSmtp servidorSmtp = new ServidorSmtp();

servidorSmtp.setCod_servidor_smtp(serverForm.getCod_s
ervidor_smtp());

servidorSmtp.setEndereco_email(serverForm.getEndereco
_email());

servidorSmtp.setPorta_smtp(serverForm.getPorta_smtp());

servidorSmtp.setRequer_autenticacao(Boolean.valueOf(se
rverForm.getRequer_autenticacao() != null?true:false));
      servidorSmtp.setSenha(serverForm.getSenha());
      servidorSmtp.setUrl(serverForm.getUrl());

servidorSmtp.setUsername(serverForm.getUsername());

      try{
         serverDao.alterarServidorSmtp(servidorSmtp);
      } catch (SQLException e){
         errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.editar.servidorsmtp"));
      }catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.editar.servidorsmtp"));
      }

      if (!errors.isEmpty()) {
         saveErrors(request, errors);
         return
(mapping.findForward(Constants.FAILURE));
      } else {
         return
(mapping.findForward(Constants.SUCCESS));
      }

   }
}
```

6.4.41 WelcomeAction.java

```
/*
 * WelcomeAction.java
 * Created on 28 de Junho de 2006, 10:26
 */

package egressos.action;

import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.dao.AlunoDao;
import java.sql.SQLException;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 * @author Ricardo
 */
public class WelcomeAction extends Action{

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
```

```
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      //busca dados de outro contexto - Intranet Academica
      Cookie[] cookies = request.getCookies();
      if (cookies != null) {
        for(int i = 0;i < cookies.length;i++) {
          if("logado".equals(cookies[i].getName()) ||
"codPessoa".equals(cookies[i].getName())){
            session.setAttribute(cookies[i].getName(),
cookies[i].getValue());
          }
        }
      }
      Aluno user = null;
      AlunoDao alunoDao = new AlunoDao();
      String logado = (String)
session.getAttribute("logado");
      if(logado == null){
        return
mapping.findForward(Constants.FAILURE);
      }
      if (logado.intern() == "true".intern()){
        String codPessoa = null;
        codPessoa = (String)
session.getAttribute("codPessoa");
        try {
          user =
alunoDao.buscarAluno(Integer.valueOf(codPessoa));
        } catch (SQLException e){
          errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new
ActionMessage("error.recuperar.dados"));
        } catch (Exception e){
          errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new ActionMessage("error.generico"));
        }
        if (user == null) {
          errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new
ActionMessage("error.usuario.naoexiste"));
        } else {
          if(session.getAttribute(Constants.USER_KEY)
!= null){

session.removeAttribute(Constants.USER_KEY);
          }
          session.setAttribute(Constants.USER_KEY,
user);
        }
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return
mapping.findForward(Constants.FAILURE);
      } else {
        return
mapping.findForward(Constants.SUCCESS);
      }
    }

}
```

6.4.42 package egressos.bean

6.4.43 Alternativa.java

```
/*
 * Alternativa.java
 * Created on 4 de Janeiro de 2006, 13:11
 */

package egressos.bean;

/**
 *
 *
 */
public class Alternativa {

  /** Creates a new instance of Alternativa */
  public Alternativa() {
  }

  private String indice;
  private String alternativa;

  public String getIndice() {
    return indice;
  }

  public void setIndice(String indice) {
    this.indice = indice;
  }

  public String getAlternativa() {
    return alternativa;
  }

  public void setAlternativa(String alternativa) {
    this.alternativa = alternativa;
  }

}
```

6.4.44 Aluno.java

```
package egressos.bean;

import java.util.Collection;
/**
 *Classe que representa um aluno.
 *
 *@since 12/10/05
 */
public class Aluno extends Pessoa {

  private String cpf;
  private String rg;
  private String sexo;

  private String nacionalidade;
  private String rgNaturalidade;
  private String estadoCivil;
  private String nomePai;
```

```
    private String nomeMae;
    private String email;
    private String fone;
    private String situacao;
    private Integer anoConcSuperior;
    private Endereco endereco;

    /**
     * Creates a new instance of Aluno.
     */
    public Aluno() {

    }

    public String getSituacao() {
        return situacao;
    }

    public void setSituacao(String situacao) {
        this.situacao = situacao;
    }

    public Integer getAnoConcSuperior() {
        return anoConcSuperior;
    }

    public void setAnoConcSuperior(Integer
anoConcSuperior) {
        this.anoConcSuperior = anoConcSuperior;
    }

    public String getCpf() {
        return cpf;
    }

    public void setCpf(String cpf) {
        this.cpf = cpf;
    }

    public String getRg() {
        return rg;
    }

    public void setRg(String rg) {
        this.rg = rg;
    }

    public String getSexo() {
        return sexo;
    }

    public void setSexo(String sexo) {
        this.sexo = sexo;
    }

    public String getNacionalidade() {
        return nacionalidade;
    }

    public void setNacionalidade(String nacionalidade) {
        this.nacionalidade = nacionalidade;
    }

    public String getRgNaturalidade() {
        return rgNaturalidade;
    }

    public void setRgNaturalidade(String rgNaturalidade) {
        this.rgNaturalidade = rgNaturalidade;
    }

    public String getEstadoCivil() {
        return estadoCivil;
    }

    public void setEstadoCivil(String estadoCivil) {
        this.estadoCivil = estadoCivil;
    }

    public String getNomePai() {
        return nomePai;
    }

    public void setNomePai(String nomePai) {
        this.nomePai = nomePai;
    }

    public String getNomeMae() {
        return nomeMae;
    }

    public void setNomeMae(String nomeMae) {
        this.nomeMae = nomeMae;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getFone() {
        return fone;
    }

    public void setFone(String fone) {
        this.fone = fone;
    }

    public Endereco getEndereco() {
        return endereco;
    }

    public void setEndereco(Endereco endereco) {
        this.endereco = endereco;
    }

}
```

6.4.45 AlunoDTO.java

```
/*
 * AlunoDTO.java
 * Created on 12 de Janeiro de 2006, 18:43
 */

package egressos.bean;
```

```
/**
 *
 *
 */
public class AlunoDTO {

  /** Creates a new instance of AlunoDTO */
  public AlunoDTO() {
  }
  private String cpf;
  private String senha;
  private String nomeAluno;
  private String sexo;
  private String rg;
  private String rgNaturalidade;
  private String nacionalidade;
  private String dataNascimento;
  private String estadoCivil;
  private String nomePai;
  private String nomeMae;
  private String email;
  private String fone;
  private String endereco;
  private String anoConcSuperior;

  public String getCpf() {
    return cpf;
  }

  public void setCpf(String cpf) {
    this.cpf = cpf;
  }

  public String getSenha() {
    return senha;
  }

  public void setSenha(String senha) {
    this.senha = senha;
  }

  public String getNomeAluno() {
    return nomeAluno;
  }

  public void setNomeAluno(String nomeAluno) {
    this.nomeAluno = nomeAluno;
  }

  public String getSexo() {
    return sexo;
  }

  public void setSexo(String sexo) {
    this.sexo = sexo;
  }

  public String getRg() {
    return rg;
  }

  public void setRg(String rg) {
    this.rg = rg;
  }

  public String getRgNaturalidade() {
    return rgNaturalidade;
  }

  public void setRgNaturalidade(String rgNaturalidade) {
    this.rgNaturalidade = rgNaturalidade;
  }

  public String getNacionalidade() {
    return nacionalidade;
  }

  public void setNacionalidade(String nacionalidade) {
    this.nacionalidade = nacionalidade;
  }

  public String getDataNascimento() {
    return dataNascimento;
  }

  public void setDataNascimento(String dataNascimento)
{
    this.dataNascimento = dataNascimento;
  }

  public String getEstadoCivil() {
    return estadoCivil;
  }

  public void setEstadoCivil(String estadoCivil) {
    this.estadoCivil = estadoCivil;
  }

  public String getNomePai() {
    return nomePai;
  }

  public void setNomePai(String nomePai) {
    this.nomePai = nomePai;
  }

  public String getNomeMae() {
    return nomeMae;
  }

  public void setNomeMae(String nomeMae) {
    this.nomeMae = nomeMae;
  }

  public String getEmail() {
    return email;
  }

  public void setEmail(String email) {
    this.email = email;
  }

  public String getFone() {
    return fone;
  }

  public void setFone(String fone) {
```

```
    this.fone = fone;
  }

  public String getEndereco() {
    return endereco;
  }

  public void setEndereco(String endereco) {
    this.endereco = endereco;
  }

  public String getAnoConcSuperior() {
    return anoConcSuperior;
  }

  public void setAnoConcSuperior(String
anoConcSuperior) {
    this.anoConcSuperior = anoConcSuperior;
  }

}
```

6.4.46 Constants.java

```
/*
 * Constants.java
 * Created on 11 de Novembro de 2005, 22:46
 */

package egressos.bean;

/**
 *
 *
 */
public final class Constants {


  /**
   * The token representing "failure" for this application.
   */
  public static final String FAILURE = "error";


  /**
   * The token representing "success" for this application.
   */
  public static final String SUCCESS = "success";


  /**
   * The session scope attribute under which the User
object
   * for the currently logged in user is stored.
   */
  public static final String USER_KEY = "user";

  public static final String OUTRO = "outro";
}
```

6.4.47 Endereco.java

```
package egressos.bean;
/**
 *Classe que representa o Endereco de uma pessoa.
 *
 *@since 12/10/05
 */

public class Endereco
{
  private String cep;
  private String rua;
  private int numero;
  private String complemento;
  private String bairro;
  private String municipio;
  private String estado;
  private String pais;
  private String owner;

  /**
   * Creates a new instance of Endereco
   */
  public Endereco()   {
  }

  public String getCep() {
    return cep;
  }

  public void setCep(String cep) {
    this.cep = cep;
  }

  public String getRua() {
    return rua;
  }

  public void setRua(String rua) {
    this.rua = rua;
  }

  public int getNumero() {
    return numero;
  }

  public void setNumero(int numero) {
    this.numero = numero;
  }

  public String getComplemento() {
    return complemento;
  }

  public void setComplemento(String complemento) {
    this.complemento = complemento;
  }

  public String getBairro() {
    return bairro;
  }

  public void setBairro(String bairro) {
    this.bairro = bairro;
  }

  public String getMunicipio() {
    return municipio;
  }
```

```
  public void setMunicipio(String municipio) {
    this.municipio = municipio;
  }

  public String getEstado() {
    return estado;
  }

  public void setEstado(String estado) {
    this.estado = estado;
  }

  public String getPais() {
    return pais;
  }

  public void setPais(String pais) {
    this.pais = pais;
  }

  public String getOwner() {
    return owner;
  }

  public void setOwner(String owner) {
    this.owner = owner;
  }

  public String toString(){
    return "Rua "+this.getRua()+"
nº"+this.getNumero()+",
"+(".equals(this.getComplemento())?"":this.getComplemento()+",")+this.getBairro()+", "+this.getMunicipio()+",
"+this.getEstado().toUpperCase()+", CEP "+this.getCep();
  }

}
```

6.4.48 Pessoa.java

```
package egressos.bean;

import java.util.GregorianCalendar;

/**
 *Classe que define uma pessoa.
 *
 *@since 12/10/05
 */

public class Pessoa {
  private String usuario;
  private String senha;
  private Integer codPessoa;
  private String nomePessoa;
  private GregorianCalendar dataNascimento;

  /**
   * Creates a new instance of Pessoa.
   */
  public Pessoa() {

  }

  public String getUsuario() {
    return usuario;
  }

  public void setUsuario(String usuario) {
    this.usuario = usuario;
  }

  public String getSenha() {
    return senha;
  }

  public void setSenha(String senha) {
    this.senha = senha;
  }

  public Integer getCodPessoa() {
    return codPessoa;
  }

  public void setCodPessoa(Integer codPessoa) {
    this.codPessoa = codPessoa;
  }

  public String getNomePessoa() {
    return nomePessoa;
  }

  public void setNomePessoa(String nomePessoa) {
    this.nomePessoa = nomePessoa;
  }

  public GregorianCalendar getDataNascimento() {
    return dataNascimento;
  }

  public void setDataNascimento(GregorianCalendar
dataNascimento) {
    this.dataNascimento = dataNascimento;
  }

  public int hashCode() {
    int ret = codPessoa.hashCode();
    ret = ret * 37 + nomePessoa.hashCode();
    ret = ret * 37 + usuario.hashCode();
    ret = ret * 37 + senha.hashCode();
    ret = ret * 37 + dataNascimento.hashCode();
    // se houver mais campos, vá repetindo a expressão
acima "ret = ret * 37 + hashCode do proximo campo"
    // por exemplo, ret = ret * 37 + telefone.hashCode();
    return ret;
  }

  public boolean equals(Object o) {
    if (!(o instanceof Pessoa)) return false;
    if (o == this) return true;
    Pessoa p = (Pessoa) o;
    return codPessoa.equals(p.getCodPessoa()) &&
usuario.equals(p.getUsuario());
  }

}
```

6.4.49 Questao.java

```
package egressos.bean;

import java.util.List;
```

```
/**
 *Classe que representa uma Questao.
 *
 *@since 12/10/05
 */


public class Questao
{
  private Integer codQuestao;
  private Integer codQuestionario;
  private String pergunta;
  private Integer ordem;
  private String alternativaA;
  private String alternativaB;
  private String alternativaC;
  private String alternativaD;
  private String alternativaE;
  private Double totalAlternativaA;
  private Double totalAlternativaB;
  private Double totalAlternativaC;
  private Double totalAlternativaD;
  private Double totalAlternativaE;
  private Integer totalRespostas;
  private String gabarito;
  private List<Alternativa> alternativas;
  private List<Resposta> respostas;

  /**
   *Creates a new instance of Questao.
   */
  public Questao()
  {

  }

  public Integer getCodQuestao() {
     return codQuestao;
  }

  public void setCodQuestao(Integer codQuestao) {

     this.codQuestao = codQuestao;
  }

  public String getPergunta() {
     return pergunta;
  }

  public void setPergunta(String pergunta) {
     this.pergunta = pergunta;
  }

  public Integer getOrdem() {
     return ordem;
  }

  public void setOrdem(Integer ordem) {
     this.ordem = ordem;
  }

  public String getAlternativaA() {
     return alternativaA;
  }

  public void setAlternativaA(String alternativaA) {
     this.alternativaA = alternativaA;
  }

  public String getAlternativaB() {
     return alternativaB;
  }

  public void setAlternativaB(String alternativaB) {
     this.alternativaB = alternativaB;
  }

  public String getAlternativaC() {
     return alternativaC;
  }

  public void setAlternativaC(String alternativaC) {
     this.alternativaC = alternativaC;
  }

  public String getAlternativaD() {
     return alternativaD;
  }

  public void setAlternativaD(String alternativaD) {
     this.alternativaD = alternativaD;
  }

  public String getAlternativaE() {
     return alternativaE;
  }

  public void setAlternativaE(String alternativaE) {
     this.alternativaE = alternativaE;
  }

  public String getGabarito() {
     return gabarito;
  }

  public void setGabarito(String gabarito) {
     this.gabarito = gabarito;
  }

  public Integer getCodQuestionario() {
     return codQuestionario;
  }

  public void setCodQuestionario(Integer
codQuestionario) {
     this.codQuestionario = codQuestionario;
  }

  public List<Alternativa> getAlternativas() {
     return alternativas;
  }

  public void setAlternativas(List<Alternativa>
alternativas) {
     this.alternativas = alternativas;
  }

  public Double getTotalAlternativaA() {
```

```
        return totalAlternativaA;
    }

    public void setTotalAlternativaA(Double
totalAlternativaA) {
        this.totalAlternativaA = totalAlternativaA;
    }

    public Double getTotalAlternativaB() {
        return totalAlternativaB;
    }

    public void setTotalAlternativaB(Double
totalAlternativaB) {
        this.totalAlternativaB = totalAlternativaB;
    }

    public Double getTotalAlternativaC() {
        return totalAlternativaC;
    }

    public void setTotalAlternativaC(Double
totalAlternativaC) {
        this.totalAlternativaC = totalAlternativaC;
    }

    public Double getTotalAlternativaD() {
        return totalAlternativaD;
    }

    public void setTotalAlternativaD(Double
totalAlternativaD) {
        this.totalAlternativaD = totalAlternativaD;
    }

    public Double getTotalAlternativaE() {
        return totalAlternativaE;
    }

    public void setTotalAlternativaE(Double
totalAlternativaE) {
        this.totalAlternativaE = totalAlternativaE;
    }

    public Integer getTotalRespostas() {
        return totalRespostas;
    }

    public void setTotalRespostas(Integer totalRespostas) {
        this.totalRespostas = totalRespostas;
    }

    public List<Resposta> getRespostas() {
        return respostas;
    }

    public void setRespostas(List<Resposta> respostas) {
        this.respostas = respostas;
    }
}
```

6.4.50 Questionario.java

```
/*
 * Questionario.java
 * Created on 14 de Dezembro de 2005, 22:36
 */

package egressos.bean;

import java.text.SimpleDateFormat;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class Questionario {

    /** Creates a new instance of Questionario */
    public Questionario() {
    }

    private Integer codQuestionario;
    private GregorianCalendar dataCriacao;
    private Integer status;
    private List<Questao> questoes;

    public boolean equals(Questionario q){
        boolean iguais = false;
        if(codQuestionario.equals(q.getCodQuestionario())){
            iguais = true;
        }
        return iguais;
    }

    public Integer getCodQuestionario() {
        return codQuestionario;
    }

    public void setCodQuestionario(Integer
codQuestionario) {
        this.codQuestionario = codQuestionario;
    }

    public GregorianCalendar getDataCriacao() {
        return dataCriacao;
    }

    public void setDataCriacao(GregorianCalendar
dataCriacao) {
        this.dataCriacao = dataCriacao;
    }

    public Integer getStatus() {
        return status;
    }

    public void setStatus(Integer status) {
        this.status = status;
    }

    public List<Questao> getQuestoes() {
        return questoes;
    }

    public void setQuestoes(List<Questao> questoes) {
        this.questoes = questoes;
```

```
    }
}
```

6.4.51 QuestionarioDTO.java

```
/*
 * QuestionarioDTO.java
 * Created on 16 de Dezembro de 2005, 18:25
 */

package egressos.bean;

import java.util.GregorianCalendar;

/**
 *
 *
 */
public class QuestionarioDTO {

    /** Creates a new instance of QuestionarioDTO */
    public QuestionarioDTO() {
    }

    private Integer codQuestionario;
    private String dataCriacao;
    private String status;

    public Integer getCodQuestionario() {
        return codQuestionario;
    }

    public void setCodQuestionario(Integer
codQuestionario) {
        this.codQuestionario = codQuestionario;
    }

    public String getDataCriacao() {
        return dataCriacao;
    }

    public void setDataCriacao(String dataCriacao) {
        this.dataCriacao = dataCriacao;
    }

    public String getStatus() {
        return status;
    }

    public void setStatus(String status) {
        this.status = status;
    }

}
```

6.4.52 Resposta.java

```
/*
 * Resposta.java
 * Created on 14 de Dezembro de 2005, 23:24
 */

package egressos.bean;

/**
 *
 *
 */
public class Resposta {

    /** Creates a new instance of Resposta */
    public Resposta() {
    }

    private Integer codResposta;
    private Integer codQuestao;
    private Integer codQuestionario;
    private String cpf;
    private String resposta;

    public Integer getCodResposta() {
        return codResposta;
    }

    public void setCodResposta(Integer codResposta) {
        this.codResposta = codResposta;
    }

    public Integer getCodQuestao() {
        return codQuestao;
    }

    public void setCodQuestao(Integer codQuestao) {
        this.codQuestao = codQuestao;
    }

    public String getCpf() {
        return cpf;
    }

    public void setCpf(String cpf) {
        this.cpf = cpf;
    }

    public String getResposta() {
        return resposta;
    }

    public void setResposta(String resposta) {
        this.resposta = resposta;
    }

    public Integer getCodQuestionario() {
        return codQuestionario;
    }

    public void setCodQuestionario(Integer
codQuestionario) {
        this.codQuestionario = codQuestionario;
    }

}
```

6.4.53 RespostaDTO.java

```
/*
 * QuestionarioDTO.java
 * Created on 16 de Dezembro de 2005, 18:25
 */

package egressos.bean;
```

```
import java.util.GregorianCalendar;

/**
 *
 *
 */
public class QuestionarioDTO {

    /** Creates a new instance of QuestionarioDTO */
    public QuestionarioDTO() {
    }

    private Integer codQuestionario;
    private String dataCriacao;
    private String status;

    public Integer getCodQuestionario() {
        return codQuestionario;
    }

    public void setCodQuestionario(Integer
codQuestionario) {
        this.codQuestionario = codQuestionario;
    }

    public String getDataCriacao() {
        return dataCriacao;
    }

    public void setDataCriacao(String dataCriacao) {
        this.dataCriacao = dataCriacao;
    }

    public String getStatus() {
        return status;
    }

    public void setStatus(String status) {
        this.status = status;
    }

}
```

6.4.54 ServidorSmtp.java

```
/*
 * ServidorSmtp.java
 * Created on 18 de Maio de 2006, 13:59
 */

package egressos.bean;

/**
 *
 *
 */
public class ServidorSmtp {

    private Integer cod_servidor_smtp;
    private String url;
    private String username;
    private String senha;
    private Integer porta_smtp;
    private String endereco_email;
    private Boolean requer_autenticacao;

    /** Creates a new instance of ServidorSmtp */
    public ServidorSmtp() {
    }

    public Integer getCod_servidor_smtp() {
        return cod_servidor_smtp;
    }

    public void setCod_servidor_smtp(Integer
cod_servidor_smtp) {
        this.cod_servidor_smtp = cod_servidor_smtp;
    }

    public String getUrl() {
        return url;
    }

    public void setUrl(String url) {
        this.url = url;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getSenha() {
        return senha;
    }

    public void setSenha(String senha) {
        this.senha = senha;
    }

    public Integer getPorta_smtp() {
        return porta_smtp;
    }

    public void setPorta_smtp(Integer porta_smtp) {
        this.porta_smtp = porta_smtp;
    }

    public String getEndereco_email() {
        return endereco_email;
    }

    public void setEndereco_email(String endereco_email)
{
        this.endereco_email = endereco_email;
    }

    public Boolean getRequer_autenticacao() {
        return requer_autenticacao;
    }

    public void setRequer_autenticacao(Boolean
requer_autenticacao) {
        this.requer_autenticacao = requer_autenticacao;
    }
```

}

6.4.55 User.java

```
/*
 * User.java
 * Created on 7 de Novembro de 2005, 18:31
 */

package egressos.bean;

/**
 *
 *
 */
public class User {

    private String userName;
    private String fullName;
    private String email;
    private String pin;
    private String grupo;

    /** Creates a new instance of User */
    public User() {
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getFullName() {
        return fullName;
    }

    public void setFullName(String fullName) {
        this.fullName = fullName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPin() {
        return pin;
    }

    public void setPin(String pin) {
        this.pin = pin;
    }

    public String getGrupo() {
        return grupo;
    }

    public void setGrupo(String grupo) {
        this.grupo = grupo;
    }
}
```

6.4.56 package egressos.dao

6.4.57 AlunoDao.java

```
/*
 * AlunoDao.java
 * Created on 25 de Outubro de 2005, 11:33
 */

package egressos.dao;
import egressos.bean.Aluno;
import egressos.bean.Endereco;
import egressos.bean.Pessoa;
import egressos.bean.Resposta;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 * @author Ricardo
 */
public class AlunoDao extends BaseDao{

    /** Creates a new instance of AlunoDao */
    private EnderecoPessoaDao enderecoDao;
    private PessoaDao pessoaDao;

    public AlunoDao() throws Exception{
        enderecoDao = new EnderecoPessoaDao();
        pessoaDao = new PessoaDao();
    }

    public Aluno buscarAluno(String cpf) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Aluno aluno = new Aluno();
        Pessoa pessoa = new Pessoa();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select cpf, rg, sexo, nacionalidade,");
            sb.append(" rgnaturalidade, estadocivil, nomepai,
nomemae, email, fone, situacao, anoconcsuperior,
pessoa_codpessoa ");
            sb.append("from aluno where cpf='");
            sb.append(cpf);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()){
                pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(rs.getInt("pessoa
_codpessoa")));
```

```
            Endereco endereco =
enderecoDao.buscarEnderecoPessoa(cpf);
            aluno.setNomePessoa(pessoa.getNomePessoa());
            aluno.setCpf(cpf);
            aluno.setSenha(pessoa.getSenha());
            aluno.setEndereco(endereco);

aluno.setNacionalidade(rs.getString("nacionalidade"));
            aluno.setRg(rs.getString("rg"));

aluno.setRgNaturalidade(rs.getString("rgNaturalidade"));

aluno.setDataNascimento(pessoa.getDataNascimento());

aluno.setEstadoCivil(rs.getString("estadoCivil"));
            aluno.setNomePai(rs.getString("nomePai"));
            aluno.setNomeMae(rs.getString("nomemae"));
            aluno.setEmail(rs.getString("email"));
            aluno.setFone(rs.getString("fone"));
            aluno.setSexo(rs.getString("sexo"));
            aluno.setSituacao(rs.getString("situacao"));

aluno.setAnoConcSuperior(rs.getInt("anoConcSuperior"));
            aluno.setUsuario(pessoa.getUsuario());
            aluno.setCodPessoa(pessoa.getCodPessoa());
          }
          return aluno;
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
    }

    public Aluno buscarAluno(Integer codPessoa) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Aluno aluno = new Aluno();
        Pessoa pessoa = new Pessoa();
        try {
          pessoa = pessoaDao.buscarPessoa(codPessoa);
          if(pessoa != null){
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select cpf, rg, sexo, nacionalidade,");
            sb.append(" rgnaturalidade, estadocivil,
nomepai, nomemae, email, fone, situacao,
anoconcsuperior, pessoa_codpessoa ");
            sb.append("from aluno where
pessoa_codpessoa='");
            sb.append(pessoa.getCodPessoa());
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()){
              pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(rs.getInt("pessoa
_codpessoa")));
              Endereco endereco =
enderecoDao.buscarEnderecoPessoa(rs.getString("cpf"));
aluno.setNomePessoa(pessoa.getNomePessoa());
              aluno.setCpf(rs.getString("cpf"));
              aluno.setSenha(pessoa.getSenha());
              aluno.setEndereco(endereco);

aluno.setNacionalidade(rs.getString("nacionalidade"));
              aluno.setRg(rs.getString("rg"));

aluno.setRgNaturalidade(rs.getString("rgNaturalidade"));

aluno.setDataNascimento(pessoa.getDataNascimento());

aluno.setEstadoCivil(rs.getString("estadoCivil"));
              aluno.setNomePai(rs.getString("nomePai"));

aluno.setNomeMae(rs.getString("nomemae"));
              aluno.setEmail(rs.getString("email"));
              aluno.setFone(rs.getString("fone"));
              aluno.setSexo(rs.getString("sexo"));
              aluno.setSituacao(rs.getString("situacao"));

aluno.setAnoConcSuperior(rs.getInt("anoConcSuperior"));
              aluno.setUsuario(pessoa.getUsuario());
              aluno.setCodPessoa(pessoa.getCodPessoa());
              return aluno;
            } else {
              return null;
            }
          } else {
            return null;
          }
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
    }


    public List<Aluno> listarAlunosAno() throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Aluno> alunos = new ArrayList<Aluno>();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select cpf, rg, sexo, nacionalidade,
rgnaturalidade, estadocivil, nomepai, nomemae, email,
fone, situacao, anoconcsuperior, pessoa_codpessoa")
          .append(" from aluno order by anoconcsuperior");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while (rs.next()){
            Aluno aluno = new Aluno();
            Pessoa pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(rs.getInt("pessoa
_codpessoa")));
            aluno.setNomePessoa(pessoa.getNomePessoa());
```

```
            aluno.setCpf(rs.getString("cpf"));
            Endereco endereco =
enderecoDao.buscarEnderecoPessoa(aluno.getCpf());
            aluno.setSenha(pessoa.getSenha());
            aluno.setEndereco(endereco);

aluno.setNacionalidade(rs.getString("nacionalidade"));
            aluno.setRg(rs.getString("rg"));

aluno.setRgNaturalidade(rs.getString("rgNaturalidade"));

aluno.setDataNascimento(pessoa.getDataNascimento());

aluno.setEstadoCivil(rs.getString("estadoCivil"));
            aluno.setNomePai(rs.getString("nomePai"));
            aluno.setNomeMae(rs.getString("nomemae"));
            aluno.setEmail(rs.getString("email"));
            aluno.setFone(rs.getString("fone"));
            aluno.setSexo(rs.getString("sexo"));
            aluno.setSituacao(rs.getString("situacao"));

aluno.setAnoConcSuperior(rs.getInt("anoConcSuperior"));
            aluno.setUsuario(pessoa.getUsuario());
            aluno.setCodPessoa(pessoa.getCodPessoa());
            alunos.add(aluno);
          }
          return alunos;
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
    }

    public List<Aluno> listarAlunosNome() throws
SQLException, Exception {
        List<Aluno> alunos = new ArrayList<Aluno>();
        try {
          List<Pessoa> pessoas = pessoaDao.listarPessoas();
          for(Pessoa pessoa: pessoas){
            Aluno alunoTest =
buscarAluno(pessoa.getCodPessoa());
            if(alunoTest.getCpf() != null){
              alunos.add(alunoTest);
            }
          }
          return alunos;
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        }
    }

    public List<Aluno> listarAlunosNome(String nome)
throws SQLException, Exception {
        List<Aluno> alunos = new ArrayList<Aluno>();
        try {
          List<Pessoa> pessoas =
pessoaDao.listarPessoasNome(nome);
          for(Pessoa pessoa:pessoas){
            Aluno alunoTest =
buscarAluno(pessoa.getCodPessoa());
            if(alunoTest != null){
              alunos.add(alunoTest);
            }
          }
          return alunos;
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        }
    }

    public void incluirAluno(Aluno aluno) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Pessoa pessoa = new Pessoa();
        try {
          pessoa =
pessoaDao.buscarPessoa(aluno.getUsuario());
          if(pessoa == null){
            pessoaDao.incluirPessoa(aluno);
            pessoa =
pessoaDao.buscarNovaPessoa(aluno.getUsuario());
          }
          Aluno alunoTeste =
buscarAlunoPorUsuario(aluno.getUsuario());
          if (pessoa != null && alunoTeste == null){
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into aluno (cpf, rg, sexo,
rgnaturalidade, estadoCivil, nomePai, nomeMae, email,
fone, situacao, anoconcsuperior, nacionalidade,
pessoa_codpessoa )");
            sb.append(" values ('");
            sb.append(aluno.getCpf());
            sb.append("','");
            sb.append(aluno.getRg());
            sb.append("','");
            sb.append(aluno.getSexo());
            sb.append("','");
            sb.append(aluno.getRgNaturalidade());
            sb.append("','");
            sb.append(aluno.getEstadoCivil());
            sb.append("','");
            sb.append(aluno.getNomePai());
            sb.append("','");
            sb.append(aluno.getNomeMae());
            sb.append("','");
            sb.append(aluno.getEmail());
            sb.append("','");
            sb.append(aluno.getFone());
            sb.append("','");
            sb.append(aluno.getSituacao());
            sb.append("','");
            sb.append(aluno.getAnoConcSuperior());
            sb.append("','");
            sb.append(aluno.getNacionalidade());
            sb.append("','");
            sb.append(pessoa.getCodPessoa());
            sb.append("')");
```

```
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();

enderecoDao.incluirEnderecoPessoa(aluno.getEndereco())
;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirAluno(Aluno aluno) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      EnderecoPessoaDao enderecoDao = new
EnderecoPessoaDao();
      RespostaDao respostaDao = new RespostaDao();
      List<Resposta> respostas = new
ArrayList<Resposta>();
      try {
        respostas =
respostaDao.buscarRespostas(aluno.getCpf());
        for(Resposta r: respostas){
          respostaDao.excluirResposta(r);
        }

enderecoDao.excluirEndereco(aluno.getEndereco());
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from aluno where cpf=");
        sb.append("'");
        sb.append(aluno.getCpf());
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
        pessoaDao.excluirPessoa(aluno.getUsuario());
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void alterarAluno(Aluno aluno) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        pessoaDao.alterarPessoa(aluno);
        StringBuffer sb = new StringBuffer();
        sb.append("update aluno set")
        .append(" rg ='").append(aluno.getRg()).append("',
rgnaturalidade ='")
        .append(aluno.getRgNaturalidade()).append("',
estadocivil ='" )
        .append(aluno.getEstadoCivil()).append("',
nomepai='").append(aluno.getNomePai()).append("',
nomemae ='")
        .append(aluno.getNomeMae()).append("',
email='").append(aluno.getEmail()).append("',
fone='").append(aluno.getFone())
        .append("', sexo='").append(aluno.getSexo())
        .append("', situacao
='").append(aluno.getSituacao()).append("',
anoconcsuperior='").append(aluno.getAnoConcSuperior())
        .append("', nacionalidade
='").append(aluno.getNacionalidade())
        .append("' where cpf
='").append(aluno.getCpf()).append("'");

        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
        Endereco endereco = new Endereco();

endereco.setBairro(aluno.getEndereco().getBairro());
        endereco.setCep(aluno.getEndereco().getCep());

endereco.setComplemento(aluno.getEndereco().getCompl
emento());

endereco.setEstado(aluno.getEndereco().getEstado());

endereco.setMunicipio(aluno.getEndereco().getMunicipio(
));

endereco.setNumero(aluno.getEndereco().getNumero());
        endereco.setOwner(aluno.getCpf());
        endereco.setPais(aluno.getEndereco().getPais());
        endereco.setRua(aluno.getEndereco().getRua());
        Endereco enderecoTest =
enderecoDao.buscarEnderecoPessoa(aluno.getCpf());

if(aluno.getCpf().equals(enderecoTest.getOwner())){
          enderecoDao.alterarEndereco(endereco);
        }
      } catch (SQLException e) {
        throw e;
      } catch (Exception e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }
```

```
  public Aluno buscarAlunoPorUsuario(String usuario)
throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Aluno aluno = new Aluno();
    Pessoa pessoa = new Pessoa();
    try {
      pessoa = pessoaDao.buscarPessoa(usuario);
      if(pessoa != null){
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select cpf, rg, sexo, nacionalidade,");
        sb.append(" rgnaturalidade, estadocivil,
nomepai, nomemae, email, fone, situacao,
anoconcsuperior ");
        sb.append("from aluno where
pessoa_codpessoa='");
        sb.append(pessoa.getCodPessoa());
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if (rs.next()){
          Endereco endereco =
enderecoDao.buscarEnderecoPessoa(rs.getString("cpf"));
          aluno.setCpf(rs.getString("cpf"));
          aluno.setEndereco(endereco);

aluno.setNacionalidade(rs.getString("nacionalidade"));
          aluno.setRg(rs.getString("rg"));

aluno.setRgNaturalidade(rs.getString("rgNaturalidade"));

aluno.setEstadoCivil(rs.getString("estadoCivil"));
          aluno.setNomePai(rs.getString("nomePai"));

aluno.setNomeMae(rs.getString("nomemae"));
          aluno.setEmail(rs.getString("email"));
          aluno.setFone(rs.getString("fone"));
          aluno.setSexo(rs.getString("sexo"));
          aluno.setSituacao(rs.getString("situacao"));

aluno.setAnoConcSuperior(rs.getInt("anoConcSuperior"));
          aluno.setSenha(pessoa.getSenha());

aluno.setNomePessoa(pessoa.getNomePessoa());

aluno.setDataNascimento(pessoa.getDataNascimento());
          aluno.setUsuario(pessoa.getUsuario());
          aluno.setCodPessoa(pessoa.getCodPessoa());
        } else {
          return null;
        }
      }
      return aluno;
    } catch (SQLException e) {
      throw e;
    } catch (Exception e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
  }
```

6.4.58 BaseDao.java

```
/*
 * BaseDao.java
 * Created on 25 de Outubro de 2005, 11:36
 */

package egressos.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;

/**
 *
 *
 */
public class BaseDao {

  protected static DataSource dataSource;

  public BaseDao(){
    if (dataSource == null) {
      try {
        Context initCtx = new InitialContext();
        dataSource = (DataSource)
initCtx.lookup("java:comp/env/jdbc/postgres");
      } catch (NamingException ex) {
        System.out.println(ex.getMessage());
      }
    }
  }

  protected Connection getConnection() throws
SQLException {
    Connection conn = null;
    try {
      conn = dataSource.getConnection();
    }
    catch (SQLException e) {
      throw e;
    }
    return conn;
  }


  protected void closeConnection(
    Connection conn,
    PreparedStatement stmt,
    ResultSet rs) {
    if (rs != null) {
      try {
        rs.close();
      } catch (SQLException e) {
      }
    }
    if (stmt != null) {
      try {
```

```
            stmt.close();
        } catch (SQLException e) {
        }
    }
    if (conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
        }
    }
  }

}
```

6.4.59 EnderecoPessoaDao.java

```
/*
 * EnderecoPessoaDao.java
 * Created on 6 de Novembro de 2005, 20:48
 */

package egressos.dao;

import egressos.bean.Endereco;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 *
 */
public class EnderecoPessoaDao extends BaseDao{

  /** Creates a new instance of
EnderecoEnderecoPessoaDao */
  public EnderecoPessoaDao() throws Exception{
  }

  public Endereco buscarEnderecoPessoa(String cpf)
throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Endereco enderecoPessoa= new Endereco();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select cep, rua, numero, complemento,
bairro, municipio, estado, pais, cpf from EnderecoPessoa
where cpf='");
      sb.append(cpf);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if(rs.next()){
        enderecoPessoa.setCep(rs.getString("cep"));
        enderecoPessoa.setRua(rs.getString("rua"));

enderecoPessoa.setNumero(rs.getInt("numero"));

enderecoPessoa.setComplemento(rs.getString("compleme
nto"));
        enderecoPessoa.setBairro(rs.getString("bairro"));

enderecoPessoa.setMunicipio(rs.getString("municipio"));

enderecoPessoa.setEstado(rs.getString("estado"));
        enderecoPessoa.setPais(rs.getString("pais"));
        enderecoPessoa.setOwner(rs.getString("cpf"));
      }
    } catch (SQLException e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return enderecoPessoa;
  }

  public void incluirEnderecoPessoa(Endereco endereco)
throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("insert into EnderecoPessoa (cep, rua,
numero, complemento, bairro, municipio, estado, pais,
cpf)");
      sb.append(" values ('");
      sb.append(endereco.getCep());
      sb.append("','");
      sb.append(endereco.getRua());
      sb.append("','");
      sb.append(endereco.getNumero());
      sb.append("','");
      sb.append(endereco.getComplemento());
      sb.append("','");
      sb.append(endereco.getBairro());
      sb.append("','");
      sb.append(endereco.getMunicipio());
      sb.append("','");
      sb.append(endereco.getEstado());
      sb.append("','");
      sb.append(endereco.getPais());
      sb.append("','");
      sb.append(endereco.getOwner());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirEndereco(Endereco endereco) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
```

```
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from EnderecoPessoa where
cpf=");
            sb.append("'");
            sb.append(endereco.getOwner());
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarEndereco(Endereco endereco) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update EnderecoPessoa set cep
='").append(endereco.getCep())
            .append("',
rua='").append(endereco.getRua()).append("', numero='")
            .append(endereco.getNumero()).append("',
complemento='").append(endereco.getComplemento())
            .append("',
bairro='").append(endereco.getBairro()).append("',
municipio='").append(endereco.getMunicipio())
            .append("',
estado='").append(endereco.getEstado()).append("',
pais='").append(endereco.getPais())
            .append("' where cpf
='").append(endereco.getOwner()).append("'");

            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }
}
```

6.4.60 PessoaDao.java

```
/*
 * PessoaDao.java
 * Created on 30 de Março de 2006, 19:30
 */

package egressos.dao;

import egressos.bean.Pessoa;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 * @author Ricardo
 */
public class PessoaDao extends BaseDao{

    /** Creates a new instance of PessoaDao */
    public PessoaDao() throws Exception{
    }


    public void incluirPessoa(Pessoa pessoa) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            SimpleDateFormat sdf = new
SimpleDateFormat("yyyy-MM-dd");
            String dataNasc =
sdf.format(pessoa.getDataNascimento().getTime());
            sb.append("insert into Pessoa (nome_pessoa,
nasc_pessoa, usuario, senha)");
            sb.append("  values ('");
            sb.append(pessoa.getNomePessoa());
            sb.append("','");
            sb.append(dataNasc);
            sb.append("','");
            sb.append(pessoa.getUsuario());
            sb.append("','");
            sb.append(pessoa.getSenha());
            sb.append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
```

```
            stmt.close();
        }
        closeConnection(conn, stmt, rs);
    }
  }

  public void excluirPessoa(String usuario) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from pessoa where usuario=");
      sb.append("'");
      sb.append(usuario);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void alterarPessoa(Pessoa pessoa) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      SimpleDateFormat sdf = new
SimpleDateFormat("yyyy-MM-dd");
      String dataNasc =
sdf.format(pessoa.getDataNascimento().getTime());
      StringBuffer sb = new StringBuffer();
      sb.append("update pessoa set
nome_pessoa='").append(pessoa.getNomePessoa())
      .append("', nasc_pessoa='").append(dataNasc)
      .append("', senha='").append(pessoa.getSenha())
      .append("' where usuario
='").append(pessoa.getUsuario()).append("'");
      conn = getConnection();
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } catch (Exception e){
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public Pessoa buscarPessoa(String usuario) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Pessoa pessoa = new Pessoa();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select cod_pessoa, nome_pessoa,
nasc_pessoa, senha, usuario from pessoa where
upper(usuario) = upper('")
      .append(usuario)
      .append("')");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if(rs.next()){

pessoa.setCodPessoa(Integer.valueOf(rs.getInt("cod_pesso
a")));
        GregorianCalendar dataNascimento = new
GregorianCalendar();

dataNascimento.setTime(rs.getDate("nasc_pessoa"));
        pessoa.setDataNascimento(dataNascimento);

pessoa.setNomePessoa(rs.getString("nome_pessoa"));
        pessoa.setSenha(rs.getString("senha"));
        pessoa.setUsuario(rs.getString("usuario"));
      } else {
        return null;
      }
    } catch (SQLException e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return pessoa;
  }

  public Pessoa buscarPessoa(Integer codPessoa) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Pessoa pessoa = new Pessoa();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select nome_pessoa, nasc_pessoa,
senha, usuario from pessoa where cod_pessoa='")
      .append(codPessoa)
      .append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if(rs.next()){
        pessoa.setCodPessoa(codPessoa);
        GregorianCalendar dataNascimento = new
GregorianCalendar();
```

```
dataNascimento.setTime(rs.getDate("nasc_pessoa"));
            pessoa.setDataNascimento(dataNascimento);

pessoa.setNomePessoa(rs.getString("nome_pessoa"));
            pessoa.setSenha(rs.getString("senha"));
            pessoa.setUsuario(rs.getString("usuario"));
          } else {
            return null;
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return pessoa;
    }

    public List<Pessoa> listarPessoasNome(String nome)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Pessoa> pessoas = new ArrayList<Pessoa>();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select cod_pessoa, nome_pessoa,
nasc_pessoa, senha, usuario")
          .append(" from pessoa where upper(nome_pessoa)
like upper('%")
          .append(nome)
          .append("%') order by nome_pessoa");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while (rs.next()){

pessoas.add(buscarPessoa(Integer.valueOf(rs.getInt("cod_
pessoa"))));
          }
          return pessoas;
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
    }

    public List<Pessoa> listarPessoas() throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Pessoa> pessoas = new ArrayList<Pessoa>();
        try {
          conn = getConnection();
          String query = "select cod_pessoa from pessoa
order by nome_pessoa";
          stmt = conn.prepareStatement(query);
          rs = stmt.executeQuery();
          while (rs.next()){

pessoas.add(buscarPessoa(Integer.valueOf(rs.getInt("cod_
pessoa"))));
          }
          return pessoas;
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
    }

    public Pessoa buscarNovaPessoa(String usuario) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Pessoa pessoa = new Pessoa();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select max(cod_pessoa) as codpessoa
from pessoa where upper(usuario) = upper('")
          .append(usuario)
          .append("')");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if(rs.next()){
            pessoa =
buscarPessoa(Integer.valueOf(rs.getInt("codpessoa")));
          } else {
            return null;
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return pessoa;
    }
}
```

6.4.61 QuestaoDao.java

```
/*
 * QuestaoDao.java
 * Created on 6 de Novembro de 2005, 20:54
 */

package egressos.dao;

import egressos.bean.Questao;
import egressos.bean.Resposta;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
```

```
public class QuestaoDao extends BaseDao{

   /** Creates a new instance of QuestaoDao */
   public QuestaoDao() throws Exception{
   }

   public Questao buscarQuestao(Integer codQuestao)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      RespostaDao respostaDao = new RespostaDao();
      Questao questao = new Questao();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select idquestao,
questionario_codquestionario, questao, ordem,")
              .append(" alternativaA, alternativaB,
alternativaC, alternativaD, alternativaE, gabarito")
              .append(" from questao where idQuestao='");
        sb.append(codQuestao);
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        List<Resposta> respostas =
respostaDao.buscarRespostasQuestao(codQuestao);
        if(rs.next()){

questao.setCodQuestao(Integer.valueOf(rs.getInt("idquesta
o")));

questao.setAlternativaA(rs.getString("alternativaA"));

questao.setAlternativaB(rs.getString("alternativaB"));

questao.setAlternativaC(rs.getString("alternativaC"));

questao.setAlternativaD(rs.getString("alternativaD"));

questao.setAlternativaE(rs.getString("alternativaE"));
          questao.setGabarito(rs.getString("gabarito"));

questao.setOrdem(Integer.valueOf(rs.getInt("ordem")));
          questao.setPergunta(rs.getString("questao"));

questao.setCodQuestionario(Integer.valueOf(rs.getInt("que
stionario_codquestionario")));

questao.setTotalAlternativaA(Double.valueOf(0));

questao.setTotalAlternativaB(Double.valueOf(0));

questao.setTotalAlternativaC(Double.valueOf(0));

questao.setTotalAlternativaD(Double.valueOf(0));

questao.setTotalAlternativaE(Double.valueOf(0));
          questao.setTotalRespostas(Integer.valueOf(0));
          questao.setRespostas(respostas);
        } else {
          return null;
        }
        for(Resposta r: respostas){
          Double totalA =
questao.getTotalAlternativaA();
          Double totalB =
questao.getTotalAlternativaB();
          Double totalC =
questao.getTotalAlternativaC();
          Double totalD =
questao.getTotalAlternativaD();
          Double totalE =
questao.getTotalAlternativaE();
          Integer totalRespostas =
questao.getTotalRespostas();
            if("A".equals(r.getResposta())){
              questao.setTotalAlternativaA(++totalA);
            } else
            if("B".equals(r.getResposta())){
              questao.setTotalAlternativaB(++totalB);
            } else
            if("C".equals(r.getResposta())){
              questao.setTotalAlternativaC(++totalC);
            } else
            if("D".equals(r.getResposta())){
              questao.setTotalAlternativaD(++totalD);
            } else
            if("E".equals(r.getResposta())){
              questao.setTotalAlternativaE(++totalE);
            }
            questao.setTotalRespostas(++totalRespostas);
          }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return questao;
   }


   public List<Questao> listarQuestoes(Integer
codQuestionario) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Questao> questoes = new
ArrayList<Questao>();
      RespostaDao respostaDao = new RespostaDao();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select idquestao,
questionario_codquestionario, questao, ordem, ")
              .append(" alternativaa, alternativab,
alternativac, alternativad, alternativae, gabarito")
              .append(" from questao where
questionario_codquestionario='")
              .append(codQuestionario)
              .append("' order by ordem");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while (rs.next()) {
          Questao questao = new Questao();
          questao.setCodQuestao(rs.getInt("idquestao"));
          questao.setPergunta(rs.getString("questao"));
          questao.setOrdem(rs.getInt("ordem"));
```

```
questao.setAlternativaA(rs.getString("alternativaA"));

questao.setAlternativaB(rs.getString("alternativaB"));

questao.setAlternativaC(rs.getString("alternativaC"));

questao.setAlternativaD(rs.getString("alternativaD"));

questao.setAlternativaE(rs.getString("alternativaE"));
        questao.setGabarito(rs.getString("gabarito"));

questao.setCodQuestionario(Integer.valueOf(rs.getInt("que
stionario_codquestionario")));
        List<Resposta> respostas =
respostaDao.buscarRespostasQuestao(questao.getCodQues
tao());
        questao.setRespostas(respostas);

questao.setTotalAlternativaA(Double.valueOf(0));

questao.setTotalAlternativaB(Double.valueOf(0));

questao.setTotalAlternativaC(Double.valueOf(0));

questao.setTotalAlternativaD(Double.valueOf(0));

questao.setTotalAlternativaE(Double.valueOf(0));
        questao.setTotalRespostas(Integer.valueOf(0));
        for(Resposta r: respostas){
          Double totalA =
questao.getTotalAlternativaA();
          Double totalB =
questao.getTotalAlternativaB();
          Double totalC =
questao.getTotalAlternativaC();
          Double totalD =
questao.getTotalAlternativaD();
          Double totalE =
questao.getTotalAlternativaE();
          Integer totalRespostas =
questao.getTotalRespostas();
          if("A".equals(r.getResposta())){
            questao.setTotalAlternativaA(++totalA);
          } else
          if("B".equals(r.getResposta())){
            questao.setTotalAlternativaB(++totalB);
          } else
          if("C".equals(r.getResposta())){
            questao.setTotalAlternativaC(++totalC);
          } else
          if("D".equals(r.getResposta())){
            questao.setTotalAlternativaD(++totalD);
          } else
          if("E".equals(r.getResposta())){
            questao.setTotalAlternativaE(++totalE);
          }
          questao.setTotalRespostas(++totalRespostas);
        }
        questoes.add(questao);
      }
    } catch (SQLException e) {
    throw e;
    } finally {
        closeConnection(conn, stmt, rs);
    }
    return questoes;
  }

  public void incluirQuestao(Questao questao) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("insert into questao (questao, ordem,")
          .append(" alternativaA, alternativaB,
alternativaC, alternativaD, alternativaE, gabarito,
questionario_codquestionario)")
          .append(" values ('");
      sb.append(questao.getPergunta());
      sb.append("','");
      sb.append(questao.getOrdem());
      sb.append("','");
      sb.append(questao.getAlternativaA());
      sb.append("','");
      sb.append(questao.getAlternativaB());
      sb.append("','");
      sb.append(questao.getAlternativaC());
      sb.append("','");
      sb.append(questao.getAlternativaD());
      sb.append("','");
      sb.append(questao.getAlternativaE());
      sb.append("','");
      sb.append(questao.getGabarito());
      sb.append("','");
      sb.append(questao.getCodQuestionario());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirQuestao(Questao questao) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    RespostaDao respostaDao = new RespostaDao();
    try {

respostaDao.excluirRespostas(questao.getCodQuestao());
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from Questao where
idQuestao=");
```

```
            sb.append("'");
            sb.append(questao.getCodQuestao());
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarQuestao(Questao questao) throws
SQLException {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update questao set questao
='").append(questao.getPergunta())
            .append("',
ordem='").append(questao.getOrdem()).append("',
alternativaa='")
            .append(questao.getAlternativaA()).append("',
alternativab='").append(questao.getAlternativaB())
            .append("',
alternativac='").append(questao.getAlternativaC()).append
("', alternativad='").append(questao.getAlternativaD())
            .append("',
alternativae='").append(questao.getAlternativaE())
            .append("',
gabarito='").append(questao.getGabarito())
            .append("',
questionario_codquestionario='").append(questao.getCodQ
uestionario())
            .append("' where idQuestao
='").append(questao.getCodQuestao()).append("'");

            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }


    public Questao buscarUltimaQuestao(Integer
codQuestionario) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Questao questao = new Questao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select idquestao,
questionario_codquestionario, questao, ordem, ")
                    .append(" alternativaa, alternativab,
alternativac, alternativad, alternativae, gabarito")
                    .append(" from questao where
questionario_codquestionario='")
            .append(codQuestionario).append("' and
ordem='").append(buscarMaiorOrdem(codQuestionario)).a
ppend("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()){
                questao.setPergunta(rs.getString("questao"));

questao.setAlternativaA(rs.getString("alternativaA"));

questao.setAlternativaA(rs.getString("alternativaA"));

questao.setAlternativaA(rs.getString("alternativaA"));

questao.setAlternativaA(rs.getString("alternativaA"));

questao.setAlternativaA(rs.getString("alternativaA"));

questao.setCodQuestao(Integer.valueOf(rs.getInt("idquesta
o")));
                questao.setCodQuestionario(codQuestionario);

questao.setOrdem(Integer.valueOf(rs.getInt("ordem")));
                questao.setGabarito("");
            } else {
                return null;
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
        return questao;
    }

    public Integer buscarMaiorOrdem(Integer
codQuestionario) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Integer ordem = 0;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
```

```
        sb.append("select max(ordem) as ordem from
questao where questionario_codquestionario='")
        .append(codQuestionario).append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          ordem = rs.getInt("ordem");
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
      return ordem;
    }

    public Questao buscarProximaQuestao(Integer
codQuestao) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Questao questao = new Questao();
      Questao proximaQuestao = new Questao();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        questao = buscarQuestao(codQuestao);
        sb.append("select idquestao,
questionario_codquestionario, questao, ordem,")
            .append(" alternativaA, alternativaB,
alternativaC, alternativaD, alternativaE, gabarito")
            .append(" from questao where ordem>'");
        sb.append(questao.getOrdem());
        sb.append("' and questionario_codquestionario='")
        .append(questao.getCodQuestionario()).append("'
order by ordem");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        //List<Questao> questoes = new
ArrayList<Questao>();
        if(rs.next()){

proximaQuestao.setCodQuestao(rs.getInt("idquestao"));

proximaQuestao.setAlternativaA(rs.getString("alternativa
A"));

proximaQuestao.setAlternativaB(rs.getString("alternativa
B"));

proximaQuestao.setAlternativaC(rs.getString("alternativa
C"));

proximaQuestao.setAlternativaD(rs.getString("alternativa
D"));

proximaQuestao.setAlternativaE(rs.getString("alternativaE
"));

proximaQuestao.setGabarito(rs.getString("gabarito"));

proximaQuestao.setOrdem(Integer.valueOf(rs.getInt("orde
m")));

proximaQuestao.setPergunta(rs.getString("questao"));

proximaQuestao.setCodQuestionario(Integer.valueOf(rs.ge
tInt("questionario_codquestionario")));
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return proximaQuestao;
    }

}
```

6.4.62 QuestionarioDao.java

```
/*
 * QuestionarioDao.java
 * Created on 14 de Dezembro de 2005, 22:36
 */

package egressos.dao;

import egressos.bean.Questao;
import egressos.bean.Questionario;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class QuestionarioDao  extends BaseDao{

    /** Creates a new instance of QuestionarioDao */
    public QuestionarioDao() throws Exception{
    }

    public void incluirQuestionario(Questionario
questionario) throws SQLException,Exception{
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("insert into questionario (datacriacao,
ativo)")
          .append(" values ('");
```

```
        SimpleDateFormat sdf = new
SimpleDateFormat("yyyy-MM-dd");

sb.append(sdf.format(questionario.getDataCriacao().getTi
me()))
        .append("', '")
        .append(questionario.getStatus())
        .append("')");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      }
      catch (SQLException e) {
        throw e;
      }
      finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirQuestionario(Questionario q) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      QuestaoDao questaoDao = new QuestaoDao();
      try {
        for(Questao questao: q.getQuestoes()){
          questaoDao.excluirQuestao(questao);
        }
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from Questionario where
codquestionario=");
        sb.append("'");
        sb.append(q.getCodQuestionario());
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void alterarQuestionario(Questionario q) throws
SQLException {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      SimpleDateFormat sdf = new
SimpleDateFormat("yyyy-MM-dd");
      try {
        StringBuffer sb = new StringBuffer();
        sb.append("update Questionario set")
        .append(" datacriacao
='").append(sdf.format(q.getDataCriacao().getTime()))
        .append("', ativo='").append(q.getStatus())
        .append("' where codquestionario
='").append(q.getCodQuestionario()).append("'");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public Questionario buscarQuestionario(Integer
codQuestionario) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Questionario q = new Questionario();
      GregorianCalendar c = new GregorianCalendar();
      q.setDataCriacao(c);
      List<Questao> questoes = new
ArrayList<Questao>();
      QuestaoDao qDao = new QuestaoDao();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codquestionario, datacriacao,
ativo from questionario where codquestionario='");
        sb.append(codQuestionario);
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          q.setCodQuestionario(codQuestionario);

q.getDataCriacao().setTime(rs.getDate("datacriacao"));
          questoes =
qDao.listarQuestoes(codQuestionario);
          q.setQuestoes(questoes);
          q.setStatus(rs.getInt("ativo"));
        }
        return q;
      } catch (SQLException e) {
        throw e;
      } catch (Exception e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
    }
```

```
   public Questionario buscarNovoQuestionario() throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Questionario q = new Questionario();
      GregorianCalendar c = new GregorianCalendar();
      q.setDataCriacao(c);
      QuestaoDao qDao = new QuestaoDao();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select max(codquestionario) as
codquestionario from questionario");
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         Integer codQuestionario = 0;
         if(rs.next()){
            codQuestionario = rs.getInt("codquestionario");
         }
         if (codQuestionario != null){
            q = buscarQuestionario(codQuestionario);
         }
         return q;
      } catch (SQLException e) {
         throw e;
      } catch (Exception e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
   }


   public ArrayList<Questionario> listarQuestionarios()
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      ArrayList<Questionario> questionarios = new
ArrayList<Questionario>();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select codquestionario from
Questionario order by codquestionario");
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         while (rs.next()) {
            Questionario questionario =
buscarQuestionario(rs.getInt("codquestionario"));
            questionarios.add(questionario);
         }
      } catch (SQLException e) {
         throw e;
      } catch (Exception e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
      return questionarios;
   }
```

```
   public Questionario buscarQuestionarioAtivo() throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Questionario q = new Questionario();
      GregorianCalendar c = new GregorianCalendar();
      q.setDataCriacao(c);
      List<Questao> questoes = new
ArrayList<Questao>();
      QuestaoDao qDao = new QuestaoDao();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select codquestionario, datacriacao,
ativo from questionario where ativo='1'");
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         if(rs.next()){

q.setCodQuestionario(Integer.valueOf(rs.getInt("codquesti
onario")));
            c.setTime(rs.getDate("datacriacao"));
            q.setDataCriacao(c);
            questoes =
qDao.listarQuestoes(Integer.valueOf(rs.getInt("codquestio
nario")));
            q.setQuestoes(questoes);
            q.setStatus(Integer.valueOf(rs.getInt("ativo")));
         }
         return q;
      } catch (SQLException e) {
         throw e;
      } catch (Exception e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
   }

}
```

6.4.63 RespostaDao.java

```
/*
 * RespostaDao.java
 * Created on 14 de Dezembro de 2005, 23:25
 */

package egressos.dao;
import egressos.bean.Resposta;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class RespostaDao extends BaseDao{

   /** Creates a new instance of RespostaDao */
```

```
    public RespostaDao() {
    }

    public Resposta buscarResposta(Integer codResposta)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Resposta resposta= new Resposta();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codresposta,
questao_codquestao, aluno_cpf,
questionario_codquestionario, resposta from respostas
where codresposta='");
          sb.append(codResposta);
          sb.append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if(rs.next()){

resposta.setCodResposta(Integer.valueOf(rs.getInt("codres
posta")));

resposta.setCodQuestao(Integer.valueOf(rs.getInt("questao
_codquestao")));

resposta.setCodQuestionario(Integer.valueOf(rs.getInt("qu
estionario_codquestionario")));
            resposta.setCpf(rs.getString("aluno_cpf"));
            resposta.setResposta(rs.getString("resposta"));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return resposta;
    }

    public boolean questaoRespondida(String cpf, Integer
codQuestionario) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        boolean questaoRespondida = false;
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select aluno_cpf,
questionario_codquestionario from respostas where
aluno_cpf='")
            .append(cpf)
            .append("' and
questionario_codquestionario='").append(codQuestionario
).append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if(rs.next()){
            questaoRespondida = true;
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return questaoRespondida;
    }


    public List<Resposta> buscarRespostas(Integer
codQuestionario) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Resposta> respostas = new
ArrayList<Resposta>();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codresposta,
questao_codquestao, aluno_cpf,
questionario_codquestionario, resposta from respostas
where questionario_codquestionario='");
          sb.append(codQuestionario);
          sb.append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while(rs.next()){
            Resposta resposta= new Resposta();

resposta.setCodResposta(Integer.valueOf(rs.getInt("codres
posta")));

resposta.setCodQuestao(Integer.valueOf(rs.getInt("questao
_codquestao")));

resposta.setCodQuestionario(Integer.valueOf(rs.getInt("qu
estionario_codquestionario")));
            resposta.setCpf(rs.getString("aluno_cpf"));
            resposta.setResposta(rs.getString("resposta"));
            respostas.add(resposta);
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return respostas;
    }

    public List<Resposta> buscarRespostas(String cpf)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Resposta> respostas = new
ArrayList<Resposta>();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codresposta,
questao_codquestao, aluno_cpf,
questionario_codquestionario, resposta from respostas
where aluno_cpf='");
          sb.append(cpf);
          sb.append("'");
```

```
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){
          Resposta resposta= new Resposta();

resposta.setCodResposta(Integer.valueOf(rs.getInt("codres
posta")));

resposta.setCodQuestao(Integer.valueOf(rs.getInt("questao
_codquestao")));

resposta.setCodQuestionario(Integer.valueOf(rs.getInt("qu
estionario_codquestionario")));
          resposta.setCpf(rs.getString("aluno_cpf"));
          resposta.setResposta(rs.getString("resposta"));
          respostas.add(resposta);
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return respostas;
    }

    public List<Resposta> buscarRespostasQuestao(Integer
codQuestao) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Resposta> respostas = new
ArrayList<Resposta>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codresposta,
questao_codquestao, aluno_cpf,
questionario_codquestionario, resposta from respostas
where questao_codquestao='");
        sb.append(codQuestao);
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){
          Resposta resposta= new Resposta();

resposta.setCodResposta(Integer.valueOf(rs.getInt("codres
posta")));

resposta.setCodQuestao(Integer.valueOf(rs.getInt("questao
_codquestao")));

resposta.setCodQuestionario(Integer.valueOf(rs.getInt("qu
estionario_codquestionario")));
          resposta.setCpf(rs.getString("aluno_cpf"));
          resposta.setResposta(rs.getString("resposta"));
          respostas.add(resposta);
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return respostas;
    }

    public void incluirResposta(Resposta resposta) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("insert into respostas
(questao_codquestao, aluno_cpf,
questionario_codquestionario, resposta)");
        sb.append(" values ('");
        sb.append(resposta.getCodQuestao());
        sb.append("','");
        sb.append(resposta.getCpf());
        sb.append("','");
        sb.append(resposta.getCodQuestionario());
        sb.append("','");
        sb.append(resposta.getResposta());
        sb.append("')");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirResposta(Resposta resposta) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from respostas where
codresposta=");
        sb.append("'");
        sb.append(resposta.getCodResposta());
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
```

```
    }

    public void excluirRespostas(Integer codQuestao)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from respostas where
questao_codquestao = ");
            sb.append("'");
            sb.append(codQuestao);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }


    public void alterarResposta(Resposta resposta) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update respostas set aluno_cpf
='").append(resposta.getCpf())

.append("',questao_codquestao='").append(resposta.getCo
dQuestao())
            .append("',
resposta='").append(resposta.getResposta())
            .append("',
questionario_codquestionario='").append(resposta.getCod
Questionario())
            .append("' where codresposta
='").append(resposta.getCodResposta()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }
}
```

6.4.64 Servidor_smtp_egressosDao.java

```
/*
 * Servidor_smtp_egressosDao.java
 * Created on 18 de Maio de 2006, 14:00
 */

package egressos.dao;

import egressos.bean.ServidorSmtp;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class Servidor_smtp_egressosDao extends
BaseDao{


    /** Creates a new instance of
Servidor_smtp_egressosDao */
    public Servidor_smtp_egressosDao() throws Exception{

    }

    public List<ServidorSmtp> listarServidor_smtp()
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        ArrayList<ServidorSmtp> servidores = new
ArrayList<ServidorSmtp>();
        try {
            conn = getConnection();
            stmt = conn.prepareStatement("select
cod_servidor_smtp_egressos from
servidor_smtp_egressos");
            rs = stmt.executeQuery();
            while (rs.next()) {

servidores.add(buscarServidorSmtp(Integer.valueOf(rs.get
Int("cod_servidor_smtp_egressos"))));
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return servidores;
    }

    public ServidorSmtp buscarServidorSmtp(Integer
codServidorSmtp) throws SQLException, Exception {
        Connection conn = null;
```

```
        PreparedStatement stmt = null;
        ResultSet rs = null;
        ServidorSmtp servidorSmtp = new ServidorSmtp();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select cod_servidor_smtp_egressos, url,
username, senha, porta_smtp, endereco_email,
requer_autenticacao from servidor_smtp_egressos where
cod_servidor_smtp_egressos ='")
          .append(codServidorSmtp)
          .append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if (rs.next()) {

servidorSmtp.setCod_servidor_smtp(codServidorSmtp);

servidorSmtp.setEndereco_email(rs.getString("endereco_e
mail"));

servidorSmtp.setPorta_smtp(Integer.valueOf(rs.getInt("por
ta_smtp")));

servidorSmtp.setRequer_autenticacao(rs.getBoolean("requ
er_autenticacao"));
            servidorSmtp.setSenha(rs.getString("senha"));
            servidorSmtp.setUrl(rs.getString("url"));

servidorSmtp.setUsername(rs.getString("username"));
          } else {
            return null;
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return servidorSmtp;
    }

    public void incluirServidorSmtp(ServidorSmtp
servidorSmtp) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("insert into servidor_smtp_egressos
(cod_servidor_smtp_egressos, url, username, senha,
porta_smtp, endereco_email, requer_autenticacao)")
          .append(" values ('")
          .append(servidorSmtp.getCod_servidor_smtp())
          .append("','")
          .append(servidorSmtp.getUrl())
          .append("','")
          .append(servidorSmtp.getUsername())
          .append("','")
          .append(servidorSmtp.getSenha())
          .append("','")
          .append(servidorSmtp.getPorta_smtp())
          .append("','")
          .append(servidorSmtp.getEndereco_email())
          .append("','")
          .append(servidorSmtp.getRequer_autenticacao())
          .append("')");
          stmt = conn.prepareStatement(sb.toString());
          stmt.executeUpdate();
        } catch (SQLException e) {
          throw e;
        } finally {
          if (rs != null) {
            rs.close();
          }
          if (stmt != null) {
            stmt.close();
          }
          closeConnection(conn, stmt, rs);
        }
    }

    public void excluirServidorSmtp(Integer
codServidorSmtp) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("delete from servidor_smtp_egressos
where cod_servidor_smtp_egressos = '")
          .append(codServidorSmtp)
          .append("'");
          stmt = conn.prepareStatement(sb.toString());
          stmt.executeUpdate();
        } catch (SQLException e) {
          throw e;
        } finally {
          if (rs != null) {
            rs.close();
          }
          if (stmt != null) {
            stmt.close();
          }
          closeConnection(conn, stmt, rs);
        }
    }

    public void alterarServidorSmtp(ServidorSmtp
servidorSmtp) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
          StringBuffer sb = new StringBuffer();
          sb.append("update servidor_smtp_egressos set
url='").append(servidorSmtp.getUrl())
          .append("', username =
'").append(servidorSmtp.getUsername())
          .append("', senha =
'").append(servidorSmtp.getSenha())
          .append("', porta_smtp =
'").append(servidorSmtp.getPorta_smtp())
          .append("', endereco_email =
'").append(servidorSmtp.getEndereco_email())
          .append("', requer_autenticacao =
'").append(servidorSmtp.getRequer_autenticacao())
```

```
        .append("' where cod_servidor_smtp_egressos =
").append(servidorSmtp.getCod_servidor_smtp()).append(
"");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } catch (Exception e){
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }
}
```

6.4.65 UserDao.java

```
/*
 * Userjava
 * Created on 7 de Novembro de 2005, 18:33
 */

package egressos.dao;

import egressos.bean.User;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 *
 */
public class UserDao extends BaseDao{

  /** Creates a new instance of UserDao */
  public UserDao(){
  }

  public User buscarUser(String userName) throws
SQLException {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    User user = new User();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select username, pin, fullusername,
email, grupo from users where username='");
      sb.append(userName);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if (rs.next()){
        user.setFullName(rs.getString("fullusername"));
        user.setUserName(rs.getString("username"));
        user.setPin(rs.getString("pin"));
        user.setEmail(rs.getString("email"));
        user.setGrupo(rs.getString("grupo"));
      }
    } catch (SQLException e) {
      e.printStackTrace();
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return user;
  }

  public void incluirUser(User user) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("insert into users (username, pin,
fullusername, email, grupo)");
      sb.append("  values ('");
      sb.append(user.getUserName());
      sb.append("','");
      sb.append(user.getPin());
      sb.append("','");
      sb.append(user.getFullName());
      sb.append("','");
      sb.append(user.getEmail());
      sb.append("','");
      sb.append(user.getGrupo());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirUser(User user) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from users where username=");
      sb.append("'");
      sb.append(user.getUserName());
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
```

```
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void alterarUser(User user) throws
SQLException {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      StringBuffer sb = new StringBuffer();
      sb.append("update users set
pin='").append(user.getPin()).append("', fullUserName='")
      .append(user.getFullName()).append("',
email='").append(user.getEmail())
      .append("', grupo='").append(user.getGrupo())
      .append("' where username
='").append(user.getUserName()).append("'");

      conn = getConnection();
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

}
```

6.4.66 package estagio.form

6.4.67 AlterarSenhaForm.java

```
/*
 * AlterarSenhaForm.java
 * Created on 19 de Junho de 2006, 21:52
 */

package egressos.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class AlterarSenhaForm extends ValidatorForm{

  private String senha;
  private String senha2;
  private String cpf;


  public void reset(ActionMapping mapping,
HttpServletRequest request) {
    setCpf(request.getParameter("cpf"));
  }

  public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
    ActionErrors errors = super.validate(mapping,
request);
    if (!senha.equals(senha2)) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionError("error.password.match"));
    }
    return errors;
  }

  public String getSenha() {
    return senha;
  }

  public void setSenha(String senha) {
    this.senha = senha;
  }

  public String getSenha2() {
    return senha2;
  }

  public void setSenha2(String senha2) {
    this.senha2 = senha2;
  }

  public String getCpf() {
    return cpf;
  }

  public void setCpf(String cpf) {
    this.cpf = cpf;
  }

}
```

6.4.68 DadosPessoaisForm.java

```
/*
 * DadosPessoaisForm.java
 * Created on 15 de Novembro de 2005, 22:05
 */

package egressos.form;
import egressos.bean.Aluno;
import egressos.bean.Constants;
import egressos.bean.Pessoa;
import egressos.dao.PessoaDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.GregorianCalendar;
```

```
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 * @author Ricardo
 */
public class DadosPessoaisForm extends ValidatorForm  {

    private String action;
    private String sexo;
    private String cpf;
    private String nome;
    private String rg;
    private String rgNaturalidade;
    private String nacionalidade;
    private String dataNascimento;
    private String estadoCivil;
    private String nomePai;
    private String nomeMae;
    private String email;
    private String fone;
    private String senha;
    private String senha2;
    private String bairro;
    private String cep;
    private String rua;
    private String complemento;
    private String estado;
    private String municipio;
    private Integer numero;
    private String pais;
    private Integer anoConclSuperior;
    private String institSuperior;
    private String situacao;
    private String usuario;
    public void reset(ActionMapping mapping,
HttpServletRequest request) {
        this.setAction("Create");
        this.setCpf(null);
        this.setRg(null);
        this.setRgNaturalidade(null);
        this.setNacionalidade(null);
        this.setEstadoCivil(null);
        this.setNomePai(null);
        this.setNomeMae(null);
        this.setEmail(null);
        this.setFone(null);
        this.setBairro(null);
        this.setCep(null);
        this.setRua(null);
        this.setComplemento(null);
        this.setEstado(null);
        this.setMunicipio(null);
        this.setNumero(null);
        this.setPais(null);
        this.setAnoConclSuperior(null);
        this.setInstitSuperior("Universidade Federal do
Paraná");
        HttpSession session = request.getSession();
        String logado = (String)
session.getAttribute("logado");
        if(logado != null){
          if ("true".equals(logado)){
            String codPessoa = (String)
session.getAttribute("codPessoa");
            Pessoa pessoa = null;
            try {
              PessoaDao pessoaDao = new PessoaDao();
              pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(codPessoa));
            } catch (SQLException e){
              e.printStackTrace();
            } catch (Exception e){
              e.printStackTrace();
            }
            if(pessoa != null){
              SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
              setUsuario(pessoa.getUsuario());
              setNome(pessoa.getNomePessoa());
              setSenha(pessoa.getSenha());

setDataNascimento(sdf.format(pessoa.getDataNascimento
().getTime()));
            }
          }
        }

    }

    public ActionErrors validate(ActionMapping mapping,
HttpServletRequest request) {

        // Perform validator framework validations
        ActionErrors errors = super.validate(mapping,
request);

        // Only need crossfield validations here
        HttpSession session = request.getSession();
        Aluno user = (Aluno)
session.getAttribute(Constants.USER_KEY);
        if(!"admin".equals(user.getSituacao())){
          if(getCep() != null && getCep().length() < 8){
            errors.add("cep",
                 new ActionMessage("error.cep.invalido"));
          }
        }
        GregorianCalendar hoje = new GregorianCalendar();
        Integer anoAtual =
Integer.valueOf(hoje.get(Calendar.YEAR));
        if(getAnoConclSuperior()>anoAtual){
          errors.add("AnoConclusaoSuperior",new
ActionMessage("error.anoConcSuperior"));
        }
        String logado = (String)
session.getAttribute("logado");
        if("Create".equals(getAction()) && logado == null){
          if(getSenha() == null || "".equals(getSenha())){
```

```
errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionError("prompt.senha.required"));
        }
        if(senha != null){
          if (!senha.equals(senha2)) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionError("error.password.match"));
          }
        }
      }
      return errors;
    }

    public String getAction() {
      return action;
    }

    public void setAction(String action) {
      this.action = action;
    }

    public String getSexo() {
      return sexo;
    }

    public void setSexo(String sexo) {
      this.sexo = sexo;
    }

    public String getCpf() {
      return cpf;
    }

    public void setCpf(String cpf) {
      this.cpf = cpf;
    }

    public String getNome() {
      return nome;
    }

    public void setNome(String nome) {
      this.nome = nome;
    }

    public String getRg() {
      return rg;
    }

    public void setRg(String rg) {
      this.rg = rg;
    }

    public String getRgNaturalidade() {
      return rgNaturalidade;
    }

    public void setRgNaturalidade(String rgNaturalidade) {
      this.rgNaturalidade = rgNaturalidade;
    }

    public String getNacionalidade() {
      return nacionalidade;
    }

    public void setNacionalidade(String nacionalidade) {
      this.nacionalidade = nacionalidade;
    }

    public String getDataNascimento() {
      return dataNascimento;
    }

    public void setDataNascimento(String dataNascimento)
  {
      this.dataNascimento = dataNascimento;
    }

    public String getEstadoCivil() {
      return estadoCivil;
    }

    public void setEstadoCivil(String estadoCivil) {
      this.estadoCivil = estadoCivil;
    }

    public String getNomePai() {
      return nomePai;
    }

    public void setNomePai(String nomePai) {
      this.nomePai = nomePai;
    }

    public String getNomeMae() {
      return nomeMae;
    }

    public void setNomeMae(String nomeMae) {
      this.nomeMae = nomeMae;
    }

    public String getEmail() {
      return email;
    }

    public void setEmail(String email) {
      this.email = email;
    }

    public String getFone() {
      return fone;
    }

    public void setFone(String fone) {
      this.fone = fone;
    }

    public String getSenha() {
      return senha;
    }

    public void setSenha(String senha) {
      this.senha = senha;
    }
```

```
public String getSenha2() {
    return senha2;
}

public void setSenha2(String senha2) {
    this.senha2 = senha2;
}

public String getBairro() {
    return bairro;
}

public void setBairro(String bairro) {
    this.bairro = bairro;
}

public String getCep() {
    return cep;
}

public void setCep(String cep) {
    this.cep = cep;
}

public String getRua() {
    return rua;
}

public void setRua(String rua) {
    this.rua = rua;
}

public String getComplemento() {
    return complemento;
}

public void setComplemento(String complemento) {
    this.complemento = complemento;
}

public String getEstado() {
    return estado;
}

public void setEstado(String estado) {
    this.estado = estado;
}

public String getMunicipio() {
    return municipio;
}

public void setMunicipio(String municipio) {
    this.municipio = municipio;
}

public Integer getNumero() {
    return numero;
}

public void setNumero(Integer numero) {
    this.numero = numero;
}

    public String getPais() {
        return pais;
    }

    public void setPais(String pais) {
        this.pais = pais;
    }

    public Integer getAnoConclSuperior() {
        return anoConclSuperior;
    }

    public void setAnoConclSuperior(Integer
anoConclSuperior) {
        this.anoConclSuperior = anoConclSuperior;
    }

    public String getInstitSuperior() {
        return institSuperior;
    }

    public void setInstitSuperior(String institSuperior) {
        this.institSuperior = institSuperior;
    }

    public String getSituacao() {
        return situacao;
    }

    public void setSituacao(String situacao) {
        this.situacao = situacao;
    }

    public String getUsuario() {
        return usuario;
    }

    public void setUsuario(String usuario) {
        this.usuario = usuario;
    }
}
```

InformaUserNameForm.java

```
/*
 * InformaUserNameForm.java
 * Created on 18 de Maio de 2006, 14:37
 */

package egressos.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class InformaUserNameForm extends
ValidatorForm{

    private String username;
    private String email;
```

```
  public void reset(ActionMapping mapping,
HttpServletRequest request) {
  }


  /**
   * Ensure that both fields have been input.
   *
   * @param mapping The mapping used to select this
instance
   * @param request The servlet request we are
processing
   */
  public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping,
request);
    return errors;
  }

  public String getUsername() {
    return username;
  }

  public void setUsername(String username) {
    this.username = username;
  }

  public String getEmail() {
    return email;
  }

  public void setEmail(String email) {
    this.email = email;
  }
}
```

6.4.69 LogonForm.java

```
/*
 * LogonForm.java
 * Created on 8 de Novembro de 2005, 15:55
 */

package egressos.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class LogonForm extends ValidatorForm{

  private String username;
  private String password;

  public String getUsername() {
    return username;
  }

  public void setUsername(String username) {
    this.username = username;
  }

  public String getPassword() {
    return password;
  }

  public void setPassword(String password) {
    this.password = password;
  }

  public void reset(ActionMapping mapping,
HttpServletRequest request) {
    setPassword("");
    setUsername("");
  }


  /**
   * Ensure that both fields have been input.
   *
   * @param mapping The mapping used to select this
instance
   * @param request The servlet request we are
processing
   */
  public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping,
request);

    if ((username == null) || (username.length() < 1))
      errors.add("username",
        new ActionError("error.username.required"));

    if ((password == null) || (password.length() < 1))
      errors.add("password",
        new ActionError("error.password.required"));

    return errors;
  }
}
```

6.4.70 NovaQuestaoForm.java

```
/*
 * NovaQuestaoForm.java
 * Created on 17 de Dezembro de 2005, 18:33
 */

package egressos.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
```

```
 *
 */
public class NovaQuestaoForm extends ValidatorForm {

    private String action;
    private String codQuestao;
    private String pergunta;
    private String alternativaA;
    private String alternativaB;
    private String alternativaC;
    private String alternativaD;
    private String alternativaE;
    private String codQuestionario;
    private String ordem;
    private String questaoaberta;

    public void reset(ActionMapping mapping,
HttpServletRequest request) {
        this.setAction("Create");
        this.setCodQuestao("");
        this.setPergunta("");
        this.setAlternativaA("");
        this.setAlternativaB("");
        this.setAlternativaC("");
        this.setAlternativaD("");
        this.setAlternativaE("");

this.setCodQuestionario(request.getParameter("codQuestio
nario"));
        this.setOrdem(request.getParameter("ordem"));
        this.setQuestaoaberta("");
    }

    public ActionErrors validate(ActionMapping mapping,
HttpServletRequest request) {

        // Perform validator framework validations
        ActionErrors errors = super.validate(mapping,
request);

        // Only need crossfield validations here
        return errors;
    }

    public String getAction() {
        return action;
    }

    public void setAction(String action) {
        this.action = action;
    }

    public String getCodQuestao() {
        return codQuestao;
    }

    public void setCodQuestao(String codQuestao) {
        this.codQuestao = codQuestao;
    }

    public String getPergunta() {
        return pergunta;
    }

    public void setPergunta(String pergunta) {
        this.pergunta = pergunta;
    }

    public String getAlternativaA() {
        return alternativaA;
    }

    public void setAlternativaA(String alternativaA) {
        this.alternativaA = alternativaA;
    }

    public String getAlternativaB() {
        return alternativaB;
    }

    public void setAlternativaB(String alternativaB) {
        this.alternativaB = alternativaB;
    }

    public String getAlternativaC() {
        return alternativaC;
    }

    public void setAlternativaC(String alternativaC) {
        this.alternativaC = alternativaC;
    }

    public String getAlternativaD() {
        return alternativaD;
    }

    public void setAlternativaD(String alternativaD) {
        this.alternativaD = alternativaD;
    }

    public String getAlternativaE() {
        return alternativaE;
    }

    public void setAlternativaE(String alternativaE) {
        this.alternativaE = alternativaE;
    }

    public String getCodQuestionario() {
        return codQuestionario;
    }

    public void setCodQuestionario(String codQuestionario)
{
        this.codQuestionario = codQuestionario;
    }

    public String getOrdem() {
        return ordem;
    }

    public void setOrdem(String ordem) {
        this.ordem = ordem;
    }

    public String getQuestaoaberta() {
        return questaoaberta;
    }
```

```
  public void setQuestaoaberta(String questaoaberta) {
    this.questaoaberta = questaoaberta;
  }
}
```

6.4.71 ObterUsuarioForm.java

```
/*
 * ObterUsuarioForm.java
 * Created on 12 de Janeiro de 2006, 20:24
 */

package egressos.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class ObterUsuarioForm extends ValidatorForm  {

  private String cpf;
  private String nome;

  public void reset(ActionMapping mapping,
HttpServletRequest request) {
    this.setCpf(null);
    this.setNome(null);
  }

  public ActionErrors validate(ActionMapping mapping,
HttpServletRequest request) {

    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping,
request);

    // Only need crossfield validations here
    if((getCpf() == null || "".intern() == getCpf().intern())
&& (getNome() == null || "".intern() ==
getNome().intern())){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionError("error.parametros.pesquisa"));
    }
    return errors;
  }

  public String getCpf() {
    return cpf;
  }

  public void setCpf(String cpf) {
    this.cpf = cpf;
  }

  public String getNome() {
    return nome;
  }

  public void setNome(String nome) {
    this.nome = nome;
  }
}
```

6.4.72 RegistrationForm.java

```
/*
 * RegistrationForm.java
 * Created on 7 de Novembro de 2005, 19:47
 */

package egressos.form;


import javax.servlet.http.HttpServletRequest;

import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.validator.ValidatorForm;



public class RegistrationForm extends ValidatorForm  {

  private String userName = null;
  private String pin = null;
  private String password2 = null;
  private String fullName = null;
  private String email = null;
  private String grupo = null;
  private String action = null;


  public String getUserName() {
    return userName;
  }

  public void setUserName(String userName) {
    this.userName = userName;
  }


  public String getPassword2() {
    return password2;
  }

  public void setPassword2(String password2) {
    this.password2 = password2;
  }

  public String getFullName() {
    return fullName;
  }

  public void setFullName(String fullName) {
    this.fullName = fullName;
  }

  public String getEmail() {
    return email;
  }
```

```
  public void setEmail(String email) {
    this.email = email;
  }

  public String getGrupo() {
    return grupo;
  }

  public void setGrupo(String grupo) {
    this.grupo = grupo;
  }
  public String getAction() {
    return action;
  }

  public void setAction(String action) {
    this.action = action;
  }

  public void reset(ActionMapping mapping,
HttpServletRequest request) {

    this.email= null;
    this.fullName = null;
    this.setPin(null);
    this.password2 = null;
    this.grupo = null;
    this.userName = null;
    this.action = "Create";

  }
  public ActionErrors validate(ActionMapping mapping,
                    HttpServletRequest request) {

    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping,
request);

    // Only need crossfield validations here
    if(getPin() != null){
      if (!getPin().equals(password2)) {
        errors.add("password2",
                new
ActionMessage("error.password.match"));
      }
    }
    return errors;
  }

  public String getPin() {
    return pin;
  }

  public void setPin(String pin) {
    this.pin = pin;
  }
}
```

6.4.73 ResponderQuestionarioForm.java

```
/*
 * ResponderQuestionarioForm.java
 * Created on 4 de Janeiro de 2006, 15:20
 */
package egressos.form;

import egressos.bean.Questao;
import egressos.bean.Questionario;
import egressos.dao.QuestionarioDao;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class ResponderQuestionarioForm extends
ActionForm {

  private String codQuestionario;

  public void reset(ActionMapping mapping,
HttpServletRequest request) {

this.setCodQuestionario((String)request.getAttribute("cod
Questionario"));
  }

  public ActionErrors validate(ActionMapping mapping,
HttpServletRequest request) {

    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping,
request);
    if(errors == null){
      errors = new ActionErrors();
    }
    String codQuestionario = getCodQuestionario();
    Questionario questionario = new Questionario();
    try{
      QuestionarioDao questionarioDao = new
QuestionarioDao();
      questionario =
questionarioDao.buscarQuestionario(Integer.valueOf(cod
Questionario));
    } catch (NumberFormatException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.convnumerica"));
    } catch (NullPointerException e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.null"));
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.generico"));
    }
    for(Questao q:questionario.getQuestoes()){
      String resposta =
request.getParameter("questao"+q.getOrdem());
      if(resposta == null || "".equals(resposta)){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.questionario.incompleto"));
        break;
      }
```

```
    }

    // Only need crossfield validations here
    return errors;
  }

  public String getCodQuestionario() {
    return codQuestionario;
  }

  public void setCodQuestionario(String codQuestionario)
{
    this.codQuestionario = codQuestionario;
  }
}
```

6.4.74 ServidorSmtpForm.java

```
/*
 * ServidorSmtpForm.java
 * Created on 18 de Maio de 2006, 14:10
 */

package egressos.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class ServidorSmtpForm extends ValidatorForm{

  private Integer cod_servidor_smtp;
  private String url;
  private String username;
  private String senha;
  private Integer porta_smtp;
  private String endereco_email;
  private Boolean requer_autenticacao;

  public void reset(ActionMapping mapping,
HttpServletRequest request) {
  }

  /**
   * Ensure that both fields have been input.
   *
   * @param mapping The mapping used to select this
instance
   * @param request The servlet request we are
processing
   */
  public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping,
request);
    return errors;
  }

  public Integer getCod_servidor_smtp() {
    return cod_servidor_smtp;
  }

  public void setCod_servidor_smtp(Integer
cod_servidor_smtp) {
    this.cod_servidor_smtp = cod_servidor_smtp;
  }

  public String getUrl() {
    return url;
  }

  public void setUrl(String url) {
    this.url = url;
  }

  public String getUsername() {
    return username;
  }

  public void setUsername(String username) {
    this.username = username;
  }

  public String getSenha() {
    return senha;
  }

  public void setSenha(String senha) {
    this.senha = senha;
  }

  public Integer getPorta_smtp() {
    return porta_smtp;
  }

  public void setPorta_smtp(Integer porta_smtp) {
    this.porta_smtp = porta_smtp;
  }

  public String getEndereco_email() {
    return endereco_email;
  }

  public void setEndereco_email(String endereco_email)
{
    this.endereco_email = endereco_email;
  }

  public Boolean getRequer_autenticacao() {
    return requer_autenticacao;
  }

  public void setRequer_autenticacao(Boolean
requer_autenticacao) {
    this.requer_autenticacao = requer_autenticacao;
  }


}
```

6.4.75 package estagio.mail

6.4.76 MailManager.java

```
/*
 * MailManager.java
 * Created on 18 de Maio de 2006, 14:19
 */

package egressos.mail;

import egressos.bean.ServidorSmtp;
import java.util.Properties;
import javax.mail.Authenticator;
import javax.mail.Message;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

/**
 *
 *
 */
public class MailManager {

   /** Creates a new instance of MailManager */
   public MailManager() {
   }

   public void send(MailServer server, MailMessage msg,
ServidorSmtp servidorSmtp) throws Exception {
      try {
         Properties mailProps = new Properties();
         mailProps.put("mail.smtp.host",
server.getMailServer());
         SimpleAuth auth = null;

if(!servidorSmtp.getPorta_smtp().equals(Integer.valueOf(1
10))){
            mailProps.put("mail.smtp.port",
servidorSmtp.getPorta_smtp().intValue() );
         }
         if (!server.isNeedsAuth()) {
            // [#] No authorization needed for this mail
server
            mailProps.put("mail.smtp.auth", "false");
         } else {
            // [#] Authorization needed for this mail server
            auth = new SimpleAuth(server.getUsername(),
server.getPassword());
            mailProps.put("mail.smtp.auth", "true");
            mailProps.put("mail.user", auth.username);
            mailProps.put("mail.from", msg.getFrom());
            mailProps.put("mail.to", msg.getTo());
         }
         Session mailSession =
Session.getInstance(mailProps, auth);
         mailSession.setDebug(false);
         Message email = new MimeMessage(mailSession);
         email.setRecipients( Message.RecipientType.TO,
InternetAddress.parse(msg.getTo()) );
         if (!msg.getCc().trim().equals("")) {
            email.setRecipients(
Message.RecipientType.CC,
InternetAddress.parse(msg.getCc()) );
         }
         if (!msg.getBcc().trim().equals("")) {
            email.setRecipients(
Message.RecipientType.BCC,
InternetAddress.parse(msg.getBcc()) );
         }
         email.setFrom( new
InternetAddress(msg.getFrom()));
         email.setSubject(msg.getSubject());
         email.setContent( msg.getText(), "text/plain" );

         Transport.send( email );
      } catch (Exception e) {
         System.out.println(" [E] ERROR: "+e);
         e.printStackTrace(System.out);
         throw e;
      }
      System.out.println(" [#] Email successfully sent");
      return;
   }

   public void send(MailServer server, String to, String cc,
String bcc, String from, String subject, String text,
ServidorSmtp servidorSmtp) throws Exception {
      MailMessage msg = new
MailMessage(to,cc,bcc,from,subject,text);
      this.send(server, msg, servidorSmtp);
      return;
   }

   class SimpleAuth extends Authenticator {
      public String username = null;
      public String password = null;
      public SimpleAuth(String user, String pwd) {
         username = user;
         password = pwd;
      }
      protected PasswordAuthentication
getPasswordAuthentication() {
         return new
PasswordAuthentication(username,password);
      }
   }
}
```

6.4.77 MailMessage.java

```
/*
 * MailMessage.java
 * Created on 18 de Maio de 2006, 14:19
 */

package egressos.mail;

/**
 *
 *
 */
public class MailMessage {

   private String to;
   private String cc;
```

```
  private String bcc;
  private String from;
  private String subject;
  private String text;

  /** Creates a new instance of MailMessage */
  public MailMessage() {
  }

  /** Creates a new instance of MailMessage */
  public MailMessage(String to,String cc,String
bcc,String from,String subject,String text) {
    this.to=to;
    this.cc=cc;
    this.bcc=bcc;
    this.from=from;
    this.subject=subject;
    this.text=text;
  }

  /** Getter for property bcc.
   * @return Value of property bcc.
   *
   */
  public java.lang.String getBcc() {
    return bcc;
  }

  /** Setter for property bcc.
   * @param bcc New value of property bcc.
   *
   */
  public void setBcc(java.lang.String bcc) {
    this.bcc = bcc;
  }

  /** Getter for property cc.
   * @return Value of property cc.
   *
   */
  public java.lang.String getCc() {
    return cc;
  }

  /** Setter for property cc.
   * @param cc New value of property cc.
   *
   */
  public void setCc(java.lang.String cc) {
    this.cc = cc;
  }

  /** Getter for property from.
   * @return Value of property from.
   *
   */
  public java.lang.String getFrom() {
    return from;
  }

  /** Setter for property from.
   * @param from New value of property from.
   *
   */
  public void setFrom(java.lang.String from) {
    this.from = from;
  }

  /** Getter for property subject.
   * @return Value of property subject.
   *
   */
  public java.lang.String getSubject() {
    return subject;
  }

  /** Setter for property subject.
   * @param subject New value of property subject.
   *
   */
  public void setSubject(java.lang.String subject) {
    this.subject = subject;
  }

  /** Getter for property text.
   * @return Value of property text.
   *
   */
  public java.lang.String getText() {
    return text;
  }

  /** Setter for property text.
   * @param text New value of property text.
   *
   */
  public void setText(java.lang.String text) {
    this.text = text;
  }

  /** Getter for property to.
   * @return Value of property to.
   *
   */
  public java.lang.String getTo() {
    return to;
  }

  /** Setter for property to.
   * @param to New value of property to.
   *
   */
  public void setTo(java.lang.String to) {
    this.to = to;
  }

}
```

6.4.78 MailServer.java

```
/*
 * MailServer.java
 * Created on 18 de Maio de 2006, 14:19
 */

package egressos.mail;

/**
 *
 *
```

```
*/
public class MailServer {

    private String mailServer = null;
    private String username = null;
    private String password = null;
    private boolean needsAuth = true;

    /** Creates a new instance of MailServer */
    public MailServer() {
    }

    /** Creates a new instance of MailServer */
    public MailServer(String mailServer, String username,
String password, boolean needsAuth) {
        this.mailServer=mailServer;
        this.username=username;
        this.password=password;
        this.needsAuth=needsAuth;
    }

    /** Getter for property mailServer.
     * @return Value of property mailServer.
     *
     */
    public java.lang.String getMailServer() {
        return mailServer;
    }

    /** Setter for property mailServer.
     * @param mailServer New value of property
mailServer.
     *
     */
    public void setMailServer(java.lang.String mailServer)
{
        this.mailServer = mailServer;
    }

    /** Getter for property needsAuth.
     * @return Value of property needsAuth.
     *
     */
    public boolean isNeedsAuth() {
        return needsAuth;
    }

    /** Setter for property needsAuth.
     * @param needsAuth New value of property
needsAuth.
     *
     */
    public void setNeedsAuth(boolean needsAuth) {
        this.needsAuth = needsAuth;
    }

    /** Getter for property password.
     * @return Value of property password.
     *
     */
    public java.lang.String getPassword() {
        return password;
    }

    /** Setter for property password.
     * @param password New value of property password.
     *
     */
    public void setPassword(java.lang.String password) {
        this.password = password;
    }

    /** Getter for property username.
     * @return Value of property username.
     *
     */
    public java.lang.String getUsername() {
        return username;
    }

    /** Setter for property username.
     * @param username New value of property username.
     *
     */
    public void setUsername(java.lang.String username) {
        this.username = username;
    }

}
```

6.4.79 package estagio.validation

6.4.80 Valida.java

```
/*
 * Valida.java
 * Created on 14 de Janeiro de 2006, 20:02
 */

package egressos.validation;

import javax.servlet.http.HttpServletRequest;
import org.apache.commons.validator.Field;
import org.apache.commons.validator.ValidatorAction;
import org.apache.commons.validator.ValidatorUtil;
import org.apache.struts.action.ActionMessages;
import org.apache.commons.validator.Validator;
import org.apache.struts.validator.Resources;

/**
 *
 *
 */
public class Valida {

    public static boolean validateCpf(Object bean,
            ValidatorAction va, Field field, ActionMessages
errors, Validator validator,HttpServletRequest request){
            String xCPF = "";
            if (field.getProperty() != null &&
field.getProperty().length() > 0) {
                xCPF=
ValidatorUtil.getValueAsString(bean,field.getProperty());
            }
            try {
                //Testa se o CPF é válido ou não
                int d1,d4,xx,nCount,resto,digito1,digito2;
                String Check;
```

```
            String Separadores = "/-.";
            d1 = 0; d4 = 0; xx = 1;
            for (nCount = 0; nCount < xCPF.length() -2;
nCount++) {
              String s_aux = xCPF.substring(nCount,
nCount+1);
              if (Separadores.indexOf(s_aux) == -1) {
                d1 = d1 + ( 11 - xx ) *
Integer.valueOf(s_aux).intValue();
                d4 = d4 + ( 12 - xx ) *
Integer.valueOf(s_aux).intValue();
                xx++;
              };
            };
            resto = (d1 % 11);
            if (resto < 2) {
              digito1 = 0;
            } else {
              digito1 = 11 - resto;
            }

            d4 = d4 + 2 * digito1;
            resto = (d4 % 11);
            if (resto < 2) {
              digito2 = 0;
            } else {
              digito2 = 11 - resto;
            }
            Check = String.valueOf(digito1) +
String.valueOf(digito2);
            String s_aux2 = xCPF.substring(xCPF.length()-
2, xCPF.length());
            if (s_aux2.compareTo(Check) != 0) {
              errors.add(field.getKey(),
Resources.getActionError(request,va,field));
              return false;
            }
            return true;
          } catch (Exception e) {
            errors.add(field.getKey(),
Resources.getActionError(request,va,field));
            return false;
          }
    }

}
```

6.4.81 Diretório pages

6.4.82 alterarsenha.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Formulário Para Alteração de Senha
  <hr align="left" class="textos"/>
</p>
  <html:form action="SalvarSenhaAlterada">
  <html:hidden property="cpf"/>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        Nova Senha:
      </th>
      <td align="left">
        <html:password property="senha" size="16"
maxlength="16" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Confirme a Senha:
      </th>
      <td align="left">
        <html:password property="senha2" size="16"
maxlength="16" styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        Enviar
      </html:submit>
      <input type=button value=Voltar
onclick="history.go(-1)" class="formulario">
    </div>
  </div>
</html:form>
```

6.4.83 buscarUsuario.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<div align="center">
  <p align="center" class="textos">
    Busca por Egressos
  </p>
  <hr align="left" class="textos"/>
  <p class="textos">
    <em class="textos">Informe o CPF correto do
Egresso que deseja buscar ou qualquer parte do
nome.</em>
  </p>
  <html:form action="ObterUsuario">
    <p class="textos">
      CPF: <html:text property="cpf"
styleClass="formulario"/>
    </p>
    <p class="textos">
      Nome:<html:text property="nome"
styleClass="formulario"/>
    </p>
    <html:submit styleClass="formulario">
    Buscar
    </html:submit>
  </html:form>
</div>
<table align="center">
  <tr align="center">
    <td class="menu" align="center"><html:link
action="MainMenu" styleClass="menu">Página
Inicial</html:link></td>
  </tr>
</table>
```

6.4.84 dadosPessoais.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="left" class="textos">
  <bean:message key="prompt.dadosPessoais"/>
  <hr align="left" class="textos"/>
</p>

<html:form action="/SalvarRegistroEgressoAction"
focus="cpf">
  <html:hidden property="action"/>
  <table align="center">
    <tr>
      <th align="left" class="textos"><bean:message
key="prompt.cpf"/>
      </th>
      <td align="left" class="textos">
        <logic:equal name="DadosPessoaisForm"
property="action"
          scope="request" value="Create">
          <html:text property="cpf" size="16"
styleClass="formulario"/> <em>Informe o CPF sem
pontos ou traços</em>
        </logic:equal>
        <logic:equal name="DadosPessoaisForm"
property="action"
          scope="request" value="Edit">
          <html:hidden property="cpf" write="true"
styleClass="formulario"/>
        </logic:equal>
      </td>
    </tr>

    <tr>
      <th align="left" class="textos">
        Nome de Usuário:
      </th>
      <td align="left" class="textos">
        <logic:equal name="DadosPessoaisForm"
property="action"
          scope="request" value="Create">
          <logic:notEqual name="logado"
scope="session" value="true">
            <html:text property="usuario" size="16"
maxlength="16" styleClass="formulario"/>
          </logic:notEqual>
          <logic:equal name="logado" scope="session"
value="true">
            <html:hidden property="usuario"
write="true" styleClass="formulario"/>
          </logic:equal>
        </logic:equal>
        <logic:equal name="DadosPessoaisForm"
property="action"
          scope="request" value="Edit">
          <html:hidden property="usuario" write="true"
styleClass="formulario"/>
        </logic:equal>
      </td>
    </tr>
    <logic:equal name="DadosPessoaisForm"
property="action" scope="request" value="Create">
      <logic:notPresent name="logado"
scope="session">
        <tr>
          <th align="left" class="textos">
            <bean:message key="prompt.password"/>:
          </th>
          <td align="left" class="textos">
            <html:password property="senha"
size="16" maxlength="16" styleClass="formulario"/>
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            <bean:message
key="prompt.password2"/>:
          </th>
          <td align="left" class="textos">
            <html:password property="senha2"
size="16" maxlength="16" styleClass="formulario"/>
          </td>
        </tr>
      </logic:notPresent>
    </logic:equal>
    <logic:notEqual name="DadosPessoaisForm"
property="action" scope="request" value="Create">
      <html:hidden property="senha" />
      <html:hidden property="senha2" />
      <tr>
        <th align="left" class="textos">
          <html:link action="AlterarSenha"
paramId="cpf"
            paramName="DadosPessoaisForm"
paramProperty="cpf" styleClass="menu">
            Alterar Senha
          </html:link>
        </th>
      </tr>
    </logic:notEqual>
    <tr>
      <td align="left" class="textos"><bean:message
key="prompt.nome"/></td>
      <td align="left" class="textos">
        <logic:notEqual name="logado"
scope="session" value="true">
          <html:text property="nome" size="42,5"
styleClass="formulario"/>
        </logic:notEqual>
        <logic:equal name="logado" scope="session"
value="true">
          <html:hidden property="nome" write="true"
styleClass="formulario"/>
        </logic:equal>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        <bean:message key="prompt.dataNascimento"/>
      </td>
      <td align="left" class="textos">
      <logic:notEqual name="logado" scope="session"
value="true">
          <html:text property="dataNascimento"
size="16" styleClass="formulario"/> <em>exemplo
01/01/2006</em>
```

```
        </logic:notEqual>
        <logic:equal name="logado" scope="session"
value="true">
          <html:hidden property="dataNascimento"
write="true" styleClass="formulario"/>
        </logic:equal>
      <td align="left" class="textos"><bean:message
key="prompt.sexo"/></td>
      <td align="left">
        <html:select property="sexo"
styleClass="formulario">
          <html:options collection="sexos"
property="value" labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos"><bean:message
key="prompt.rg"/></td>
      <td align="left"><html:text property="rg"
size="16" styleClass="formulario"/></td>
      <td align="left" class="textos"><bean:message
key="prompt.rgNaturalidade"/></td>
      <td align="left">
        <html:select property="rgNaturalidade"
styleClass="formulario">
          <html:options name="estados" />
        </html:select>
      </td>
    </tr>
    <tr>
      <td class="textos"><bean:message
key="prompt.nacionalidade"/>
      </td>
      <td align="left"><html:text
property="nacionalidade" size="20"
styleClass="formulario"/>
      </td>
      <td class="textos"><bean:message
key="prompt.estadoCivil"/>
      </td>
      <td align="left">
        <html:select property="estadoCivil"
styleClass="formulario">
          <html:options name="estadosCivis"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        <bean:message key="prompt.mae"/>:
      </td>
      <td align="left">
        <html:text property="nomeMae" size="42,5"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        <bean:message key="prompt.pai"/>:
      </td>
      <td align="left">
        <html:text property="nomePai" size="42,5"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        <bean:message key="prompt.email"/>:
      </td>
      <td align="left">
        <html:text property="email" size="20"
styleClass="formulario"/>
      </td>
      <td align="left" class="textos">
        <bean:message key="prompt.fone"/>:
      </td>
      <td align="left">
        <html:text property="fone" size="16"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">Rua:</td>
      <td align="left">
        <html:text property="rua" size="42,5"
styleClass="formulario"/>
      </td>
      <td align="left" class="textos">N&ordm;:</td>
      <td align="left">
        <html:text property="numero" size="8"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        Bairro:
      </td>
      <td align="left">
        <html:text property="bairro"
styleClass="formulario"/>
      </td>
      <td align="left"
class="textos">Complemento:</td>
      <td align="left">
        <html:text property="complemento"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">CEP:</td>
      <td align="left">
        <html:text property="cep"
styleClass="formulario"/>
      </td>
      <td align="left" class="textos">Cidade:</td>
      <td align="left">
        <html:text property="municipio"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">Estado:</td>
      <td align="left">
        <html:select property="estado"
styleClass="formulario">
          <html:options name="estados" />
        </html:select>
```

```
        </td>
        <td align="left" class="textos">Pa&iacute;s:</td>
        <td align="left">
          <html:text property="pais"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">Ensino
Superior:</th>
      </tr>
      <tr>
        <th align="left" class="textos">Instituição de
Ensino:</th>
        <td align="left">
        <html:text property="institSuperior" size="42,5"
disabled="true" styleClass="formulario"/></td>
        <th align="left" class="textos">Ano de
Conclusão:</th>
        <td align="left">
        <html:text property="anoConclSuperior" size="08"
styleClass="formulario"/></td>
      </tr>
      <logic:equal name="user" value="admin"
property="situacao">
        <tr>
          <th align="right">
          </th>
          <td align="left" class="textos">
            <html:checkbox property="situacao"
value="admin" styleClass="formulario"><bean:message
key="user.admin"/></html:checkbox>
          </td>
        </tr>
      </logic:equal>
      <logic:equal name="user" value="egresso"
property="situacao">
        <html:hidden property="situacao"/>
      </logic:equal>

    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
          <bean:message key="button.save"/>
        </html:submit>
        <html:reset styleClass="formulario">
          <bean:message key="button.reset"/>
        </html:reset>
      </div>
    </div>
</html:form>
<p align="center">
  <input type=button value=Voltar onclick="history.go(-
1)" class="formulario">
</p>
```

6.4.85 erro.jsp

```
<hr align="left" class="textos"/>
<p align="center" class="textos">
ERRO
<hr align="left" class="textos"/>
</p>
```

6.4.86 estatisticasQuestionario.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>

<p align="left" class="textos">
Estatisticas (Questionario <bean:write
name="codQuestionario" scope="request"/>)
<hr align="left" class="textos"/>
</p>
<ol>
<logic:present name="questoes">
<logic:iterate name="questoes" id="questao">
  <li class="textos">
    <em class="textos">
    <html:link action="MostrarRespostasQuestao"
paramId="codQuestao" paramName="questao"
paramProperty="codQuestao" styleClass="menu">
    <bean:write name="questao" property="pergunta"/>
    </html:link>
    </em> [<bean:write name="questao"
property="ordem" />]
    <table>
    <logic:iterate name="questao"
property="alternativas" id="alternativa">
      <tr>
        <td class="textos">
        <bean:write name="alternativa"
property="indice"/>)
        <bean:write name="alternativa"
property="alternativa"/>
        </td>
        <td class="textos">
        <logic:equal name="alternativa"
property="indice" value="A">
          <bean:write name="questao"
property="totalAlternativaA"/>%
        </logic:equal>
        <logic:equal name="alternativa"
property="indice" value="B">
          <bean:write name="questao"
property="totalAlternativaB"/>%
        </logic:equal>
        <logic:equal name="alternativa"
property="indice" value="C">
          <bean:write name="questao"
property="totalAlternativaC"/>%
        </logic:equal>
        <logic:equal name="alternativa"
property="indice" value="D">
          <bean:write name="questao"
property="totalAlternativaD"/>%
        </logic:equal>
        <logic:equal name="alternativa"
property="indice" value="E">
          <bean:write name="questao"
property="totalAlternativaE"/>%
        </logic:equal>
        </td>
      </tr>
    </logic:iterate>
    </table>
    <p class="textos">
```

```
    Total de Respostas:<bean:write name="questao"
property="totalRespostas"/>
    </p>
  </li>
</logic:iterate>
</logic:present>
<logic:empty name="questoes">
    <p class="textos" align="center">
      Questionário vazio.
    </p>
</logic:empty>
</ol>
<p align="center">
<html:link action="ListarQuestoes"
paramId="codQuestionario"
paramName="codQuestionario" styleClass="menu">
  Listar Questões
</html:link>
</p>
```

6.4.87 footer.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>

<table width="100%" border="0" cellspacing="0"
cellpadding="0">
  <tr>
    <td><html:img src="struts-power.gif" alt="Powered
By Struts" align="right"/></td>
    <td width="30" align="right"><img
src="images/canto.gif" width="30" height="30"></td>
    <td width="320" align="right"
class="fundoazul">&copy; Universidade Federal
    do Paran&aacute; 2005 - Pol&iacute;tica de
Privacidade </td>
  </tr>
</table>
```

6.4.88 gerenciarQuestionarios.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="left" class="textos">
  Gerenciar Questionarios
  <hr align="left" class="textos"/>
</p>
<table border="1" align="center">
  <tr>
    <th align="center" class="textos">Codigo</th>
    <th align="center" class="textos">Criação</th>
    <th align="center" class="textos">Status</th>
  </tr>
  <logic:iterate name="questionarios" id="questionario">
    <tr>
      <td align="center" class="textos">
        <html:link action="ListarQuestoes"
paramId="codQuestionario"
          paramName="questionario"
paramProperty="codQuestionario" styleClass="menu">
          <bean:write name="questionario"
property="codQuestionario"/>
        </html:link>

      </td>
      <td align="center" class="textos">
        <bean:write name="questionario"
property="dataCriacao"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="questionario"
property="status"/>
      </td>

      <td class="menu">
        <html:link action="ExcluirQuestionario"
paramId="codQuestionario"
          paramName="questionario"
paramProperty="codQuestionario" styleClass="menu">
          Excluir
          </html:link>
      </td>
      <td class="menu">
        <html:link action="DefinirAtivo"
paramId="codQuestionario"
          paramName="questionario"
paramProperty="codQuestionario" styleClass="menu">
          Ativo
          </html:link>
      </td>

    </tr>
  </logic:iterate>
  <tr>
    <td class="menu"><html:link
action="NovoQuestionario"
styleClass="menu">Novo</html:link></td>
  </tr>
</table>
<table align="center">
  <tr align="center">
    <td class="menu" align="center"><html:link
action="MainMenu" styleClass="menu">Página
Inicial</html:link></td>
  </tr>
</table>
```

6.4. 89 index.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles"
%>
<html:html>
  <head>
    <title>Seja bem vindo ao Sistema de Controle de
Egressos da UFPR</title>
    <meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
    <html:base/>
    <link href="images/tcc.css" rel="stylesheet"
type="text/css">
  </head>

  <body leftmargin="0" topmargin="0" marginwidth="0"
marginheight="0">
    <table width="100%" height="161" border="0"
cellpadding="0" cellspacing="0">
      <tr>
```

```
          <td width="7" height="28"
class="fundoazul"> </td>
          <td width="129" rowspan="2" align="center"
valign="bottom" class="fundoazul"><img
src="images/logo.gif" width="129" height="85"></td>
          <td width="713"
class="fundoazul"> </td>
          <td width="288" class="fundoazul"><img
src="images/tit_sist_control_estagio.gif" width="288"
height="28"></td>
        </tr>
        <tr>
          <td height="65"> </td>
          <td> </td>
          <td rowspan="5"><img
src="images/img_federal.jpg" width="288"
height="131"></td>
        </tr>
        <tr>
          <td height="20"> </td>
          <td align="center"
valign="bottom"> </td>
          <td> </td>
        </tr>
        <tr>
          <td height="19" rowspan="3"> </td>
          <td colspan="2" align="left" valign="top"
class="tcc">Seja bem vindo ao Sistema de Controle de
Egressos
            da UFPR</td>
        </tr>
        <logic:present name="user">
          <tr>
            <td height="18" colspan="2" align="left"
valign="middle" class="tcc">Usuário:
              <span class="textoCinza"><bean:write
name="user" property="nomePessoa" /></span></td>
          </tr>
          <tr>
            <td height="14" colspan="2" align="left"
valign="middle" class="tcc">Status: <span
class="textoCinza"><bean:write name="user"
property="situacao"/></span></td>
          </tr>
        </logic:present>
      </table>
      <table width="100%" height="1" border="0"
cellpadding="0" cellspacing="0">
        <tr>
          <td class="fundoazul"> </td>
        </tr>
      </table>
      <table width="100%" height="5" border="0"
cellpadding="0" cellspacing="0">
        <tr>
          <td class="fundoCinza">          </td>
        </tr>
      </table>
      <tiles:insert attribute="menu"/>
      <tiles:insert attribute="body"/>
      <tiles:insert attribute="footer"/>
    </body>
</html:html>
```

6.4.90 informausername.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>

<html:form action="EnviaSenha">
  <p align="center" class="textos">
    Sua senha será enviada para o email cadastrado no
sistema.
  </p>
  <table border="0" width="100%">
    <tr>
      <th align="right" class="textos">
        username*:
      </th>
      <td align="left">
        <html:text property="username" size="16"
maxlength="18" styleClass="input"/>
      </td>
    </tr>
    <tr>
      <th align="right" class="textos">
        e-mail*:
      </th>
      <td align="left">
        <html:text property="email" size="16"
styleClass="input"/>
      </td>
    </tr>
    <tr>
      <td align="right">
        <html:submit property="Submit"
styleClass="formulario">
          Enviar
        </html:submit>
      </td>
    </tr>
  </table>

</html:form>
```

6.4.91 inicial.jsp

6.4.92 inserequestao.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="left" class="textos">

  <logic:equal name="NovaQuestaoForm"
property="action" value="Create">
  Inserir Questão
  </logic:equal>
  <logic:equal name="NovaQuestaoForm"
property="action" value="Editar">
  Editar Questão
  </logic:equal>

<hr align="left" class="textos"/>
</p>

<html:form action="/SalvarQuestao" focus="pergunta">
  <html:hidden property="action"/>
  <html:hidden property="codQuestionario"/>
```

```
  <logic:equal name="user" value="admin"
property="situacao">
    <table align="center">
     <logic:equal name="NovaQuestaoForm"
property="action" value="Editar">
      <tr>
      <th align="left" class="textos">
      Codigo da Questão:
      </th>
      <td align="left" class="textos" class="textos">
      <html:hidden property="codQuestao" write="true"
styleClass="formulario"/>
      </td>
      </tr>
     </logic:equal>
      <tr>
      <th align="left" class="textos">
      Pergunta:
      </th>
      <td align="left" class="textos">
      <html:text property="pergunta" size="50"
styleClass="formulario"/>
      </td>
      </tr>
      <tr>
      <th align="left" class="textos">
      a)
      </th>
      <td align="left" class="textos">
      <html:text property="alternativaA" size="30"
styleClass="formulario"/>
      </td>
      </tr>
      <tr>
      <th align="left" class="textos">
      b)
      </th>
      <td align="left" class="textos">
      <html:text property="alternativaB" size="30"
styleClass="formulario"/>
      </td>
      </tr>
      <tr>
      <th align="left" class="textos" class="textos">
      c)
      </th>
      <td align="left" class="textos">
      <html:text property="alternativaC" size="30"
styleClass="formulario"/>
      </td>
      </tr>
      <tr>
      <th align="left" class="textos">
      d)
      </th>
      <td align="left" class="textos">
      <html:text property="alternativaD" size="30"
styleClass="formulario"/>
      </td>
      </tr>
      <tr>
      <th align="left" class="textos">
      e)
      </th>
      <td align="left" class="textos">
      <html:text property="alternativaE" size="30"
styleClass="formulario"/>
      </td>
      </tr>
      <tr>
      <th align="right" class="textos">
      <html:checkbox property="questaoaberta"
value="sim" styleClass="formulario"/>
      </th>
      <td align="left" class="textos">
      Questão aberta (caso esse item seja marcado,
qualquer alternativa digitada não será considerada)
      </td>
      </tr>
      <html:hidden property="ordem"
styleClass="formulario"/>
    </table>
    <div class="button" id="botao">
     <div align="center">
      <html:submit styleClass="formulario">
       <bean:message key="button.save"/>
      </html:submit>
      <html:reset styleClass="formulario">
       <bean:message key="button.reset"/>
      </html:reset>
        </div>
    </div>
  </logic:equal>
</html:form>
<hr class="textos"/>
<p align="center" class="textos">
<html:link page="/ListarQuestoes.do"
paramId="codQuestionario"
paramName="codQuestionario" styleClass="menu">
Voltar
</html:link>
</p>
```

6.4.93 listarQuestoes.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="left" class="textos">
  Questionario <bean:write name="codQuestionario"
scope="request"/>
  <hr align="left" class="textos"/>
</p>

<table border="1" align="center">
  <tr>
    <th align="center" class="textos">Questao</th>
    <th align="center" class="textos">Questionario</th>
    <th align="center" class="textos">Pergunta</th>
  </tr>
  <logic:iterate name="questoes" id="questao">
    <tr>
      <td align="center" class="textos">
        <html:link
page="/EditarQuestao.do?action=Editar"
paramId="codQuestao"
            paramName="questao"
paramProperty="codQuestao" styleClass="menu">
```

```
            <bean:write name="questao"
property="codQuestao"/>
            </html:link>
          </td>
          <td align="center" class="textos">
            <bean:write name="questao"
property="codQuestionario"/>
          </td>
          <td align="center" class="textos">
            <bean:write name="questao"
property="pergunta"/>
          </td>
          <td align="center">
            <html:link page="/AumentarOrdem.do"
paramId="codQuestao"
              paramName="questao"
paramProperty="codQuestao" name="map"
styleClass="menu">
            Aumentar Ordem
            </html:link>
          </td>
          <td align="center" class="textos">
            <html:link page="/ExcluirQuestao.do"
paramId="codQuestao"
              paramName="questao"
paramProperty="codQuestao" styleClass="menu">
            Excluir
            </html:link>
          </td>


      </tr>
    </logic:iterate>
    <tr>
      <td class="textos">
        <html:link action="NovaQuestao"
paramId="codQuestionario"
          paramName="codQuestionario" name="map1"
styleClass="menu">
        Inserir Questão
        </html:link>
      </td>
    </tr>
</table>
<hr class="textos"/>
<p align="center">
    <html:link action="GerenciarQuestionarios"
styleClass="menu">
    Questionarios
    </html:link>
    <html:link action="EstatisticasQuestionario"
paramId="codQuestionario"
paramName="codQuestionario" styleClass="menu">
    Estatísticas
    </html:link>
    <html:link action="MainMenu"
styleClass="menu">Página Inicial</html:link>
</p>
```

6.4.94 mainMenu.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<logic:present name="user">
  <logic:notEqual name="user" property="nomePessoa"
value="guest">
    <table height="30" border="0" cellpadding="0"
cellspacing="0">
      <tr align="center" valign="middle">
        <td width="10"> </td>
        <logic:equal name="user" value="admin"
property="situacao">
          <td><html:link action="Registration"
styleClass="menu"><bean:message
key="cadastro.salvar"/></html:link></td>
          <td width="15" class="menu">|</td>
          <td><html:link action="BuscarUsuario"
styleClass="menu">Usuários</html:link></td>
          <td width="15" class="menu">|</td>
          <td><html:link
action="GerenciarQuestionarios"
styleClass="menu">Questionários</html:link></td>
          <td width="15" class="menu">|</td>
          <td><html:link action="Relatorios"
styleClass="menu">Relatórios</html:link></td>
          <td width="15" class="menu">|</td>
          <td align="center" >
            <html:link action="EditarEmail"
styleClass="menu">Email do Sistema</html:link>
          </td>
          <td width="15" class="menu">|</td>
        </logic:equal>
        <td><html:link action="ResponderQuestionario"
styleClass="menu">Pesquisa</html:link></td>
        <td width="15" class="menu">|</td>
        <td><html:link
action="EditarCadastro?action=Edit" paramId="cpf"
paramName="user" paramProperty="cpf"
paramScope="session"
styleClass="menu"><bean:message
key="cadastro.alterar"/></html:link></td>
        <td width="15" class="menu">|</td>
        <td><html:link forward="logoff"
styleClass="menu"><strong><bean:message
key="prompt.sair"/></strong></html:link></td>
      </tr>
    </table>
    <table width="100%" height="4" border="0"
cellpadding="5" cellspacing="0">
      <tr>
        <td class="fundoazul"> </td>
      </tr>
    </table>
  </logic:notEqual>
</logic:present>
<html:errors/>
```

6.4.95 mostrarQuestao.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>

<p align="left" class="textos">
  Questao <bean:write name="codQuestao"
scope="request"/>
  <hr align="left" class="textos"/>
</p>
```

```
  <p class="textos">
    <em class="textos">
      <bean:write name="questao"
property="pergunta"/>

    </em> [<bean:write name="questao"
property="ordem" />]
    <table>
      <logic:iterate name="questao"
property="alternativas" id="alternativa">
        <tr>
          <td class="textos">
            <bean:write name="alternativa"
property="indice"/>)
            <bean:write name="alternativa"
property="alternativa"/>
          </td>
          <td class="textos">
            <logic:equal name="alternativa"
property="indice" value="A">
              <bean:write name="questao"
property="totalAlternativaA"/>%
            </logic:equal>
            <logic:equal name="alternativa"
property="indice" value="B">
              <bean:write name="questao"
property="totalAlternativaB"/>%
            </logic:equal>
            <logic:equal name="alternativa"
property="indice" value="C">
              <bean:write name="questao"
property="totalAlternativaC"/>%
            </logic:equal>
            <logic:equal name="alternativa"
property="indice" value="D">
              <bean:write name="questao"
property="totalAlternativaD"/>%
            </logic:equal>
            <logic:equal name="alternativa"
property="indice" value="E">
              <bean:write name="questao"
property="totalAlternativaE"/>%
            </logic:equal>
          </td>
        </tr>
      </logic:iterate>
      <ol>
        <logic:present name="embranco">
          <p class="textos">Respostas salvas para essa
questão:</p>
          <logic:iterate name="respostas"
id="respostadto">
            <li class="textos">
              <em class="textos"><bean:write
name="respostadto" property="resposta"/></em>
              <p class="textos">Postado por
                <bean:write name="respostadto"
property="nomeAluno"/>.
              </p>
            </li>
          </logic:iterate>
        </logic:present>
      </ol>
    </table>
    <p class="textos">
      Total de Respostas:<bean:write name="questao"
property="totalRespostas"/>
    </p>
    <p align="center">
      <html:link action="EstatisticasQuestionario"
paramId="codQuestionario"
paramName="codQuestionario" styleClass="menu">
      Voltar
      </html:link>
    </p>
  </p>
```

6.4.96 novoservidorsmtp.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<html:form action="SalvarServidorSmtp">
  <logic:equal name="user" property="situacao"
value="admin">
  <html:hidden property="cod_servidor_smtp" />
    <p align="center" class="textos">
      Dados do servidor SMTP utilizado para envio de e-
mails pelo sistema.
      <hr align="left" class="textos"/>
    </p>
    <table align="center">
      <tr>
        <th align="left" class="textos">
          URL:
        </th>
        <td align="left" class="textos">
          <html:text property="url"
styleClass="formulario"/><em class="textos">exemplo:
smtp.mail.yahoo.com.br</em>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Endereço de Email:
        </th>
        <td align="left" class="textos">
          <html:text property="endereco_email"
styleClass="formulario"/><em class="textos">exemplo:
sisestagiosufpr@yahoo.com.br</em>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Username:
        </th>
        <td align="left" class="textos">
          <html:text property="username"
styleClass="formulario"/><em class="textos">exemplo:
sisestagiosufpr</em>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Porta SMTP:
        </th>
        <td align="left" class="textos">
```

```
            <html:text property="porta_smtp"
styleClass="formulario"/><em class="textos">padrão:
110</em>
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Senha:
          </th>
          <td align="left" class="textos">
            <html:text property="senha"
styleClass="formulario"/>
          </td>
        </tr>
        <tr>
          <th align="right" class="textos">
            <html:checkbox
property="requer_autenticacao" value="true"
styleClass="formulario"/>
          </th>
          <td align="left" class="textos">
            Servidor Requer Autenticação
          </td>
        </tr>
      </table>
      <div class="button" id="botao">
        <div align="center">
          <html:submit styleClass="formulario">
            Enviar
          </html:submit>
        </div>
      </div>
    </logic:equal>
</html:form>

<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
        Página Inicial
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.97 relatorioano.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
  <bean:message key="index.heading"/>
</p>
<p align="center" class="textos">
  Relatório por Ano de Conclusão
</p>
<hr align="left" class="textos"/>

<table border="1" align="center">
  <tr>
    <th align="center" class="textos">CPF</th>
    <th align="center" class="textos">Nome</th>
    <th align="center" class="textos">Sexo</th>
    <th align="center" class="textos">E-Mail</th>
    <th align="center" class="textos">Ano de
Conclusão</th>
    <th align="center" class="textos">Data de
Nascimento</th>
    <th align="center" class="textos">Endereco</th>
    <th align="center" class="textos">Fone</th>
    <th align="center"
class="textos">Nacionalidade</th>
    <th align="center" class="textos">R.G</th>
    <th align="center" class="textos">Estado
Emissor</th>
  </tr>
  <logic:iterate name="alunos" id="aluno">
  <tr>
    <td align="center" class="textos">
      <bean:write name="aluno" property="cpf"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno"
property="nomeAluno"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno" property="sexo"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno" property="email"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno"
property="anoConcSuperior"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno"
property="dataNascimento"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno" property="endereco"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno" property="fone"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno"
property="nacionalidade"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno" property="rg"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="aluno"
property="rgNaturalidade"/>
    </td>
  </tr>
  </logic:iterate>
</table>
<hr class="textos"/>
<p align="center">
<html:link action="Relatorios" styleClass="menu">
Voltar
</html:link>
</p>
```

6.4.98 relatorionome.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
  <bean:message key="index.heading"/>
</p>
<p align="center" class="textos">
  Relatório por Ordem de Nome
</p>
<hr align="left" class="textos"/>

<table border="1" align="center">
  <tr>
    <th align="center" class="textos">CPF</th>
    <th align="center" class="textos">Nome</th>
    <th align="center" class="textos">Sexo</th>
    <th align="center" class="textos">E-Mail</th>
    <th align="center" class="textos">Ano de
Conclusão</th>
    <th align="center" class="textos">Data de
Nascimento</th>
    <th align="center" class="textos">Endereco</th>
    <th align="center" class="textos">Fone</th>
    <th align="center"
class="textos">Nacionalidade</th>
    <th align="center" class="textos">R.G</th>
    <th align="center" class="textos">Estado
Emissor</th>
  </tr>
  <logic:iterate name="alunos" id="aluno">
    <tr>
      <td align="center" class="textos">
        <bean:write name="aluno" property="cpf"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno"
property="nomeAluno"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno" property="sexo"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno" property="email"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno"
property="anoConcSuperior"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno"
property="dataNascimento"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno"
property="endereco"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno" property="fone"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno"
property="nacionalidade"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno" property="rg"/>
      </td>
      <td align="center" class="textos">
        <bean:write name="aluno"
property="rgNaturalidade"/>
      </td>
    </tr>
  </logic:iterate>
</table>
<hr class="textos"/>
<p align="center">
  <html:link action="Relatorios" styleClass="menu">
  Voltar
  </html:link>
</p>
```

6.4.99 responderQuestionario.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="left" class="textos">
  Pesquisa cod. <bean:write name="codQuestionario"
scope="request"/>
  <hr align="left" class="textos"/>
</p>

<html:form action="/SalvarRespostasQuestionario">
  <html:hidden property="codQuestionario"/>
  <ol>
    <%--logic:iterate name="questoes" id="questao"--%>
    <%
    java.util.ArrayList questoes = new
java.util.ArrayList();
    questoes = (java.util.ArrayList)
request.getAttribute("questoes");
    for (int i=0;i<questoes.size();++i){
      egressos.bean.Questao q = (egressos.bean.Questao)
questoes.get(i);
    %>
    <li class="textos">
      <em class="textos">
        <%--bean:write name="questao"
property="pergunta"/--%>
        <%= q.getPergunta()%>
      </em> [<%--bean:write name="questao"
property="ordem" /--%> <%= q.getOrdem()%>]

      <%--logic:iterate name="questao"
property="alternativas" id="alternativa"--%>
      <%
      java.util.ArrayList alternativas = new
java.util.ArrayList();
      alternativas = (java.util.ArrayList)
q.getAlternativas();
      if (q.getAlternativas() == null || alternativas.size()
== 0){%>
      <p class="textos">
```

```
            <textarea name="<%=
"questao"+q.getOrdem()%>" rows="8" cols="50"
class="formulario"></textarea>
          </p>
          <%} else {
            for (int ia=0;ia<alternativas.size();++ia){
              egressos.bean.Alternativa a =
(egressos.bean.Alternativa) alternativas.get(ia);
          %>
          <p class="textos">
            <%--bean:write name="alternativa"
property="indice"/--%>
            <%= a.getIndice()%>)
            <input type=radio name="<%=
"questao"+q.getOrdem()%>" value="<%=
a.getIndice()%>" class="formulario">
            <%--bean:write name="alternativa"
property="alternativa"/--%>
            <%= a.getAlternativa()%>
          </p>
          <%}%>
          <%}%>
          <%--/logic:iterate--%>

        </li>
        <%}%>
        <%--/logic:iterate--%>
      </ol>
      <div align="center">
        <html:submit styleClass="formulario">
          <bean:message key="button.save"/>
        </html:submit>
      </div>
</html:form>
```

6.4.100 resultadosBuscaUsuarios.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="left" class="textos">
  Resultados da Busca
  <hr align="left" class="textos"/>
</p>

<logic:present name="alunos">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">CPF</th>
      <th align="center" class="textos">Nome</th>
    </tr>
    <logic:iterate name="alunos" id="aluno">
      <tr>
        <td align="center" class="textos">
          <html:link
action="EditarCadastro?action=Edit" paramName="aluno"
paramProperty="cpf" paramId="cpf" styleClass="menu">
            <bean:write name="aluno"
property="cpf"/>
          </html:link>
        </td>
        <td align="center" class="textos">
          <bean:write name="aluno"
property="nomePessoa"/>
        </td>
        <td align="center" class="textos">
          <html:link action="ExcluirEgresso"
paramName="aluno" paramId="cpf" paramProperty="cpf"
styleClass="menu">
            Excluir
          </html:link>
        </td>
      </tr>
    </logic:iterate>
  </table>
</logic:present>
<logic:present name="semresultados">
  <p align="center" class="textos"><bean:write
name="semresultados"/></p>
</logic:present>
<hr class="textos"/>
<p align="center" class="textos">
  <html:link action="BuscarUsuario"
styleClass="menu">
  Buscar Novamente
  </html:link>
</p>
```

6.4.101 sucesso.jsp

```
<p align="center" class="textos">
OPERAÇÃO REALIZADA COM SUCESSO !
<hr align="left" class="textos"/>
</p>
```

6.4.102 tiporelatorio.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

  <p align="center" class="textos">
    Consulta ao Banco de Dados de Egressos
    <hr align="left" class="textos"/>
  </p>

  <p class="textos" align="center">
    <html:link action="/RelatorioAno"
styleClass="menu">
    Listar Egressos em ordem de ano de conclusão
    </html:link>
  </p>
  <hr class="textos"/>
  <p class="textos" align="center">
    <html:link action="/RelatorioNome"
styleClass="menu">
    Listar Egressos em ordem de nome
    </html:link>
  </p>
```

6.4.103 tcc.css

```
/* CSS Document */

.tcc {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #084F98;
        font-style: normal;
        font-weight: bold;
```

```
        text-decoration: none;
        background-color: #FFFFFF;
        border: none;
        background-position: 0px 0px;
        height: 0px;
        width: 0px;
        margin: 0px;
        padding: 0px;
        letter-spacing: normal;
        text-align: justify;
        word-spacing: normal;
        list-style-type: square;
}
.fundoazul {
        background-color: #084F98;
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #FFFFFF;
}
.textoCinza {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #666666;
        font-weight: normal;
}
.fundoCinza {
        background-color: #EFEFEF;
}
.menu {
        font-family: Arial, Helvetica, sans-serif;
        color: #084F98;
        font-size: 12px;
        background-color: #FFFFFF;
        text-decoration: none;
}
.menu:Hover {
        font-family: Arial, Helvetica, sans-serif;
        color: #084F98;
        font-size: 12px;
        background-color: #FFFFFF;
        text-decoration: underline;
}
.textos {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #084F98;
}
.formulario {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #333333;
        border: 1px outset #EFEFEF;
}
```

6.4.104 Welcome.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<logic:present name="mensagem">
   <p align="center" class="textos"><bean:write
name="mensagem"/></p>
</logic:present>
```

```
<html:form action="/LogonSubmit" focus="username">
  <table border="0" width="100%">

    <tr>
      <th align="right" class="textos">
        usename*:
      </th>
      <td align="left" class="textos">
        <html:text property="username" size="16"
maxlength="18" styleClass="formulario"/>
      </td>
    </tr>

    <tr>
      <th align="right" class="textos">
        <bean:message key="prompt.password" />*:
      </th>
      <td align="left" class="textos">
        <html:password property="password" size="16"
maxlength="18" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <td align="right">
        <html:submit property="Submit"
value="Submit" styleClass="formulario"/>
      </td>
      <td align="left">
        <html:reset styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="right" class="textos">
        <bean:message key="prompt.novousuario"/>
      </th>
      <td align="left" class="textos">
        <P><html:link action="/NovoEgresso"
styleClass="menu"><bean:message
key="prompt.registrarnovo"/></html:link></P>
      </td>
    </tr>
    </tr>
    <tr>
      <th align="right" class="textos">
      </th>
      <td align="left">
        <P><html:link action="InformaUserName"
styleClass="menu">Esqueceu sua senha
?</html:link></P>
      </td>
    </tr>
  </table>

</html:form>
```

6.4.105 MóduloEstágios - Código Fonte

6.4.105 Estrutura de diretórios:

```
RootDir
|
+-- META-INF
|   |
|   +-- context.xml
|
+-- pages
|   |
|   +-- *.jsp
|
+-- WEB-INF
    |
    +-- classes
    |   |
    |   +-- estagio
    |   |   |
    |   |   +-- action
    |   |   +-- bean
    |   |   +-- dão
    |   |   +-- form
    |   |   +-- validation
    |   +-- jasper
    |   +--MessageResources.properties
    |
    +--- lib
    |   |
    |   +-- commons-beanutils-1.5.jar
    |   +-- commons-beanutils.jar
    |   +-- commons-collection.jar
    |   +-- commons-digester.jar
    |   +-- commons-digester-1.7.jar
    |   +-- commons-fileupload.jar
    |   +-- commons-logging.jar
    |   +-- commons-logging-api-1.0.2.jar
    |   +-- commons-logging-1.0.2.jar
    |   +-- antl.jap
    |   +-- itext.jar
    |   +-- jakarta-oro.jar
    |   +-- jasperreports-1.2.2.jar
    |   +-- struts.jar
    |   +-- activation.jar
    |   +-- mail.jar
    |
    +-- tiles-defs.xml
    +-- struts-config.xml
    +-- validation.xml
    +-- validation-rules.xml
    +-- web.xml
    +-- struts-bean.xml
    +-- struts-logic.xml
    +-- struts-html.xml
    +-- struts-nested.xml
    +-- struts-tiles.xml
```

6.4.106 tiles-defs.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
 <!DOCTYPE tiles-definitions PUBLIC
    "-//Apache Software Foundation//DTD Tiles Configuration
1.1//EN"
    "http://jakarta.apache.org/struts/dtds/tiles-config_1_1.dtd">
<!--
   This is a blank Tiles definition file with a commented
example.-->
<tiles-definitions>
   <definition name="mainLayout" path="/pages/index.jsp">
      <put name="menu"   value="/pages/mainMenu.jsp" />
      <put name="body"   value="" />
      <put name="footer"   value="/pages/footer.jsp" />
   </definition>
  <definition name="login" path="/pages/login.jsp">
      <put name="body"   value="" />
   </definition>
   <definition name="welcome" extends="login">
      <put name="body"   value="/pages/Welcome.jsp" />
   </definition>
   <definition name="informausername" extends="login">
      <put name="body"   value="/pages/informausername.jsp" />
   </definition>
   <definition name="inicial" extends="mainLayout">
      <put name="body"   value="/pages/inicial.jsp" />
   </definition>
   <definition name="sucesso" extends="mainLayout">
      <put name="body"   value="/pages/sucesso.jsp" />
   </definition>
   <definition name="erro" extends="mainLayout">
      <put name="body"   value="/pages/erro.jsp" />
   </definition>
   <definition name="listarconvenios" extends="mainLayout">
      <put name="body"   value="/pages/listarConvenios.jsp" />
   </definition>
   <definition name="novoconvenio" extends="mainLayout">
      <put name="body"   value="/pages/novoConvenio.jsp" />
   </definition>
   <definition name="novaempresa" extends="mainLayout">
      <put name="body"   value="/pages/novaEmpresa.jsp" />
   </definition>
   <definition name="dadospessoais" extends="mainLayout">
      <put name="body"   value="/pages/dadospessoais.jsp" />
   </definition>
   <definition name="listarvagas" extends="mainLayout">
      <put name="body"   value="/pages/listarvagas.jsp" />
   </definition>
   <definition name="tipousuario" extends="mainLayout">
      <put name="body"   value="/pages/tipousuario.jsp" />
   </definition>
   <definition name="novavaga" extends="mainLayout">
      <put name="body"   value="/pages/novavaga.jsp" />
   </definition>
   <definition name="listarhabilidadesvaga"
extends="mainLayout">
<put name="body"   value="/pages/listarhabilidadesvaga.jsp" />
   </definition>
   <definition name="novahabilidadevaga"
extends="mainLayout">
<put name="body"   value="/pages/novahabilidadevaga.jsp" />
   </definition>
   <definition name="novahabilidadevagageral"
extends="mainLayout">
      <put name="body"
value="/pages/novahabilidadevagageral.jsp" />
   </definition>
   <definition name="listarhabilidadesaluno"
extends="mainLayout">
      <put name="body"
value="/pages/listarhabilidadesaluno.jsp" />
   </definition>
   <definition name="novahabilidadealuno"
extends="mainLayout">
      <put name="body"   value="/pages/novahabilidadealuno.jsp"
/>
   </definition>
   <definition name="novahabilidadealunogeral"
extends="mainLayout">
      <put name="body"
value="/pages/novahabilidadealunogeral.jsp" />
   </definition>
   <definition name="listarturnosdisponiveis"
extends="mainLayout">
      <put name="body"
value="/pages/listarturnosdisponiveis.jsp" />
   </definition>
   <definition name="novoturnodisponivel"
extends="mainLayout">
  <put name="body"   value="/pages/novoturnodisponivel.jsp" />
   </definition>
   <definition name="listarturnos" extends="mainLayout">
      <put name="body"   value="/pages/listarturnos.jsp" />
   </definition>
   <definition name="novoturno" extends="mainLayout">
      <put name="body"   value="/pages/novoturno.jsp" />
   </definition>
   <definition name="buscarvagas" extends="mainLayout">
    <put name="body"   value="/pages/buscarvagas.jsp" />
   </definition>
     <definition name="selecionarestagiario"
extends="mainLayout">
    <put name="body"   value="/pages/selecionarestagiario.jsp" />
   </definition>
    <definition name="novotermocompromisso"
extends="mainLayout">
      <put name="body"
value="/pages/novotermocompromisso.jsp" />
   </definition>
    <definition name="listartermos" extends="mainLayout">
      <put name="body"   value="/pages/listartermos.jsp" />
   </definition>
    <definition name="listaragentes" extends="mainLayout">
      <put name="body"   value="/pages/listaragentes.jsp" />
   </definition>
    <definition name="novoagente" extends="mainLayout">
      <put name="body"   value="/pages/novoagente.jsp" />
   </definition>
    <definition name="listarhabilidades" extends="mainLayout">
      <put name="body"   value="/pages/listarhabilidades.jsp" />
   </definition>
    <definition name="buscarestagiario" extends="mainLayout">
      <put name="body"   value="/pages/buscarestagiario.jsp" />
   </definition>
    <definition name="listarcursos" extends="mainLayout">
      <put name="body"   value="/pages/listarcursos.jsp" />
   </definition>
    <definition name="novocurso" extends="mainLayout">
      <put name="body"   value="/pages/novocurso.jsp" />
   </definition>
    <definition name="consultarcursosvagas"
extends="mainLayout">
      <put name="body"
value="/pages/consultarcursosvagas.jsp" />
   </definition>
    <definition name="buscarusuario" extends="mainLayout">
      <put name="body"   value="/pages/buscarusuario.jsp" />
   </definition>
    <definition name="listarusuarios" extends="mainLayout">
      <put name="body"   value="/pages/listarusuarios.jsp" />
   </definition>
    <definition name="listartermoscompromisso"
extends="mainLayout">
```

```
    <put name="body"
value="/pages/listartermoscompromisso.jsp" />
  </definition>
   <definition name="listarempresas" extends="mainLayout">
    <put name="body"  value="/pages/listarempresas.jsp" />
  </definition>
   <definition name="buscarempresa" extends="mainLayout">
    <put name="body"  value="/pages/buscarempresa.jsp" />
  </definition>
   <definition name="pesquisarempresas"
extends="mainLayout">
    <put name="body"  value="/pages/pesquisarempresas.jsp"
/>
  </definition>
   <definition name="pesquisartermoscompromisso"
extends="mainLayout">
    <put name="body"
value="/pages/pesquisartermoscompromisso.jsp" />
  </definition>
   <definition name="pesquisarconvenios"
extends="mainLayout">
  <put name="body"  value="/pages/pesquisarconvenios.jsp" />
  </definition>
   <definition name="pesquisarempresasemconvenio"
extends="mainLayout">
    <put name="body"
value="/pages/pesquisarempresasemconvenio.jsp" />
  </definition>
   <definition name="listarempresassemconvenio"
extends="mainLayout">
    <put name="body"
value="/pages/listarempresassemconvenio.jsp" />
  </definition>
   <definition name="pesquisarhabilidadealuno"
extends="mainLayout">
    <put name="body"
value="/pages/pesquisarhabilidadealuno.jsp" />
  </definition>
   <definition name="buscarrequisitos" extends="mainLayout">
    <put name="body"  value="/pages/buscarrequisitos.jsp" />
  </definition>
   <definition name="termocompromisso"
extends="mainLayout">
    <put name="body"  value="/pages/termocompromisso.jsp" />
  </definition>
   <definition name="buscarcnpj" extends="mainLayout">
    <put name="body"  value="/pages/buscarcnpj.jsp" />
  </definition>
   <definition name="listarunidade" extends="mainLayout">
    <put name="body"  value="/pages/listarunidade.jsp" />
  </definition>
   <definition name="linkrelatoriopdf" extends="mainLayout">
    <put name="body"  value="/pages/linkrelatoriopdf.jsp" />
  </definition>
   <definition name="selecionarrelatorio"
extends="mainLayout">
    <put name="body"  value="/pages/selecionarrelatorio.jsp" />
  </definition>
   <definition name="novoservidorsmtp"
extends="mainLayout">
    <put name="body"  value="/pages/novoservidorsmtp.jsp" />
  </definition>
   <definition name="alterarsenha" extends="mainLayout">
    <put name="body"  value="/pages/alterarsenha.jsp" />
  </definition>
   <!-- sample tiles definitions
  <definition name=".mainLayout"
path="/common/layouts/classicLayout.jsp">
    <put name="title"  value="Sample Page Title" />
    <put name="header" value="/common/header.jsp" />
    <put name="menu"  value=".mainMenu" />
    <put name="footer" value="/common/footer.jsp" />
    <put name="body"  value=".portal.body" />
  </definition>
  <definition name=".mainMenu"
path="/common/layouts/vboxLayout.jsp" >
    <putList name="list" >
      <add value=".menu.links" />
      <add value=".menu.taglib.references" />
      <add value=".menu.printer.friendly" />
      <add value=".menu.old.documents" />
    </putList>
  </definition>
  <definition name="aPage" extends=".mainLayout">
    <put name="title"  value="Another Title" />
    <put name="body"  value=".aPage.body" />
  </definition>
  end samples -->
</tiles-definitions>
```

6.4.107 struts-config.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE struts-config PUBLIC
      "-//Apache Software Foundation//DTD Struts Configuration
1.2//EN"
      "http://jakarta.apache.org/struts/dtds/struts-
config_1_2.dtd"><!--
   This is a blank Struts configuration file with an example
   welcome action/page and other commented sample elements.
   Tiles and the Struts Validator are configured using the factory
defaults
   and are ready-to-use.
   NOTE: If you have a generator tool to create the corresponding
Java classes
   for you, you could include the details in the "form-bean"
declarations.
   Otherwise, you would only define the "form-bean" element
itself, with the
   corresponding "name" and "type" attributes, as shown here.-->
<struts-config>
<!-- ============================================
Data Source Configuration -->
<!--
<data-sources>
<data-source
type="org.apache.commons.dbcp.BasicDataSource">
  <set-property
   property="driverClassName"
   value="org.postgresql.Driver" />
  <set-property
   property="url"
   value="jdbc:postgresql://localhost/mydatabase" />
  <set-property
   property="username"
   value="me" />
  <set-property
   property="password"
   value="test" />
  <set-property
   property="maxActive"
   value="10" />
  <set-property
   property="maxWait"
   value="5000" />
  <set-property
   property="defaultAutoCommit"
   value="false" />
  <set-property
   property="defaultReadOnly"
   value="false" />
  <set-property
   property="validationQuery"
   value="SELECT COUNT(*) FROM market" />
</data-source>
```

```
</data-sources>--> <!--
========================================================
= Form Bean Definitions -->
  <form-beans>
  <form-bean     name="LogonForm"
type="estagio.form.LogonForm"/>
  <form-bean     name="NovoConvenioForm"
type="estagio.form.NovoConvenioForm"/>
  <form-bean     name="NovaEmpresaForm"
type="estagio.form.NovaEmpresaForm"/>
  <form-bean     name="DadosPessoaisForm"
type="estagio.form.DadosPessoaisForm"/>
  <form-bean     name="TipoUsuarioForm"
type="estagio.form.TipoUsuarioForm"/>
  <form-bean     name="NovaVagaForm"
type="estagio.form.NovaVagaForm"/>
  <form-bean     name="NovaHabilidadeVagaForm"
type="estagio.form.NovaHabilidadeVagaForm"/>
  <form-bean     name="NovaHabilidadeForm"
type="estagio.form.NovaHabilidadeForm"/>
  <form-bean     name="NovaHabilidadeAlunoForm"
type="estagio.form.NovaHabilidadeAlunoForm"/>
  <form-bean     name="NovoTurnoDisponivelForm"
type="estagio.form.TurnoDisponivelForm"/>
  <form-bean     name="NovoTurnoForm"
type="estagio.form.NovoTurnoForm"/>
  <form-bean     name="SelecionarEstagiarioForm"
type="estagio.form.SelecionarEstagiarioForm"/>
  <form-bean     name="NovoTermoCompromissoForm"
type="estagio.form.NovoTermoCompromissoForm"/>
  <form-bean     name="NovoAgenteIntegracaoForm"
type="estagio.form.NovoAgenteIntegracaoForm"/>
  <form-bean     name="BuscarEstagiarioForm"
type="estagio.form.BuscarEstagiarioForm"/>
  <form-bean     name="NovoCursoForm"
type="estagio.form.NovoCursoForm"/>
  <form-bean     name="ConsultarCursosVagasForm"
type="estagio.form.ConsultarCursosVagasForm"/>
  <form-bean     name="BuscarUsuarioForm"
type="estagio.form.BuscarUsuarioForm"/>
  <form-bean     name="BuscarEmpresaForm"
type="estagio.form.BuscarEmpresaForm"/>
  <form-bean     name="PesquisarEmpresasForm"
type="estagio.form.PesquisarEmpresasForm"/>
  <form-bean     name="PesquisarTermosCompromissoForm"
type="estagio.form.PesquisarTermosCompromissoForm"/>
  <form-bean     name="PesquisarConveniosForm"
type="estagio.form.PesquisarConveniosForm"/>
  <form-bean     name="PesquisarHabilidadeAlunoForm"
type="estagio.form.PesquisarHabilidadeAlunoForm"/>
  <form-bean     name="ConteudoTermoForm"
type="estagio.form.ConteudoTermoForm"/>
  <form-bean     name="NovaVagaEstagioInternoForm"
type="estagio.form.NovaVagaEstagioInternoForm"/>
  <form-bean     name="SelecionarRelatorioForm"
type="estagio.form.SelecionarRelatorioForm"/>
  <form-bean     name="InformaUserNameForm"
type="estagio.form.InformaUserNameForm"/>
  <form-bean     name="ServidorSmtpForm"
type="estagio.form.ServidorSmtpForm"/>
  <form-bean     name="AlterarSenhaForm"
type="estagio.form.AlterarSenhaForm"/>
  <!-- sample form bean descriptor for an ActionForm
    <form-bean
      name="inputForm"
      type="app.InputForm"/>
  end sample -->
  <!-- sample form bean descriptor for a DynaActionForm
    <form-bean
      name="logonForm"
      type="org.apache.struts.action.DynaActionForm">
      <form-property
        name="username"
        type="java.lang.String"/>
      <form-property
        name="password"
        type="java.lang.String"/>
    </form-bean>
  end sample -->
  </form-beans>
<!-- ========================================================
Global Exception Definitions -->
  <global-exceptions>
    <!-- sample exception handler
    <exception
      key="expired.password"
      type="app.ExpiredPasswordException"
      path="/changePassword.jsp"/>
    end sample -->
  </global-exceptions>
<!-- ========================================================
Global Forward Definitions -->
  <global-forwards>
    <!-- Default forward to "Welcome" action -->
    <!-- Demonstrates using index.jsp to forward -->inicial
    <forward name="welcome"        path="/Welcome.do"/>
    <forward name="logon"          path="/Logon.do"/>
    <forward name="logoff"         path="/Logoff.do"/>
  </global-forwards>
<!-- ========================================================
Action Mapping Definitions -->
  <action-mappings>
      <!-- Default "Welcome" action -->
      <!-- Forwards to Welcome.jsp -->
    <action
      path="/Welcome"
      type="estagio.action.WelcomeAction"
      scope="request"
      validate="false">
      <forward name="success" path="inicial" />
      <forward name="error" path="welcome" />
    </action>
    <action
      path="/Logon"
      forward="welcome"/>
    <action
      path="/Inicial"
      forward="inicial"/>
    <action
      path="/LogonSubmit"
      type="estagio.action.LogonAction"
      name="LogonForm"
      scope="request"
      validate="true"
      input="welcome">
      <forward name="success" path="inicial" />
      <forward name="error" path="welcome" />
    </action>
    <action
      path="/Logoff"
      type="estagio.action.LogoffAction">
      <forward
        name="success"
        path="welcome"/>
    </action>
    <action
      path="/ListarConvenios"
      type="estagio.action.ListarConveniosAction"
      scope="request"
      validate="false"
      input="mainLayout">
      <forward name="success" path="listarconvenios" />
      <forward name="error" path="erro" />
    </action>
```

```
<action
  path="/ExcluirConvenio"
  type="estagio.action.ExcluirConvenioAction"
  scope="request"
  validate="false"
  input="pesquisarconvenios">
  <forward name="success" path="sucesso" />
  <forward name="error" path="erro" />
</action>
<action
  path="/NovoConvenio"
  type="estagio.action.NovoConvenioAction"
  scope="request"
  validate="false"
  name="NovoConvenioForm"
  input="listarconvenios">
  <forward name="success" path="novoconvenio" />
  <forward name="error" path="listarconvenios" />
</action>
<action
  path="/SalvarConvenio"
  type="estagio.action.SalvarConvenioAction"
  scope="request"
  validate="true"
  name="NovoConvenioForm"
  input="pesquisarempresasemconvenio">
  <forward name="success" path="sucesso" />
  <forward name="error" path="erro" />
</action>
<action
  path="/NovaEmpresa"
  type="estagio.action.NovoConvenioAction"
  scope="request"
  validate="false"
  name="NovaEmpresaForm"
  input="novaempresa">
  <forward name="success" path="novaempresa" />
  <forward name="error" path="pesquisarconvenios" />
</action>
<action
  path="/SalvarEmpresa"
  type="estagio.action.SalvarEmpresaAction"
  scope="request"
  validate="true"
  name="NovaEmpresaForm"
  input="novaempresa">
  <forward name="success" path="sucesso" />
  <forward name="error" path="novaempresa" />
</action>
<action
  path="/EditarEmpresa"
  type="estagio.action.EditarEmpresaAction"
  scope="request"
  validate="false"
  name="NovaEmpresaForm"
  input="inicial">
  <forward name="success" path="novaempresa" />
  <forward name="error" path="erro" />
</action>
<action
  path="/EditarCadastro"
  type="estagio.action.EditarCadastroAction"
  scope="request"
  validate="false"
  name="DadosPessoaisForm"
  input="inicial">
  <forward name="success" path="dadospessoais" />
  <forward name="error" path="inicial" />
</action>
<action
  path="/SalvarCadastro"
  type="estagio.action.SalvarCadastroAction"
  scope="request"
  validate="true"
  name="DadosPessoaisForm"
  input="dadospessoais">
  <forward name="success" path="sucesso" />
  <forward name="error" path="dadospessoais" />
</action>
<action
  path="/NovoUsuario"
  type="estagio.action.NovoUsuarioAction"
  scope="request"
  validate="false"
  input="inicial">
  <forward name="success" path="tipousuario" />
  <forward name="error" path="inicial" />
</action>
<action
  path="/TipoUsuario"
  type="estagio.action.TipoUsuarioAction"
  scope="request"
  validate="false"
  name="TipoUsuarioForm"
  input="tipousuario">
  <forward name="success" path="dadospessoais" />
  <forward name="error" path="tipousuario" />
  <forward name="buscarempresa" path="buscarempresa" >
</action>
<action
  path="/BuscarEmpresa"
  type="estagio.action.BuscarEmpresaAction"
  scope="request"
  validate="true"
  name="BuscarEmpresaForm"
  input="buscarempresa">
  <forward name="success" path="dadospessoais" />
  <forward name="error" path="buscarempresa" />
</action>
<action
  path="/ListarVagas"
  type="estagio.action.ListarVagasAction"
  scope="request"
  validate="false"
  input="inicial">
  <forward name="success" path="listarvagas" />
  <forward name="error" path="inicial" />
</action>
<action
  path="/ExcluirVaga"
  type="estagio.action.ExcluirVagaAction"
  scope="request"
  validate="false"
  input="listarvagas">
  <forward name="success" path="listarvagas" />
  <forward name="error" path="erro" />
</action>
<action
  path="/NovaVaga"
  type="estagio.action.NovaVagaAction"
  scope="request"
  validate="false"
  input="listarvagas">
  <forward name="success" path="novavaga" />
  <forward name="error" path="listarvagas" />
  <forward name="buscarcnpj" path="buscarcnpj" />
</action>
<action
  path="/NovaVagaEstagioInterno"
  type="estagio.action.NovaVagaEstagioInternoAction"
  scope="request"
  name="NovaVagaEstagioInternoForm"
  validate="false"
  input="listarvagas">
```

```
      <forward name="success" path="novavaga" />
      <forward name="error" path="listarvagas" />
    </action>
    <action
      path="/SalvarVaga"
      type="estagio.action.SalvarVagaAction"
      scope="request"
      validate="true"
      name="NovaVagaForm"
      input="novavaga">
      <forward name="success" path="listarvagas" />
      <forward name="error" path="novavaga" />
    </action>
    <action
      path="/EditarVaga"
      type="estagio.action.EditarVagaAction"
      scope="request"
      validate="false"
      name="NovaVagaForm"
      input="mainLayout">
      <forward name="success" path="novavaga" />
      <forward name="error" path="inicial" />
    </action>
    <action
      path="/ListarHabilidadesVaga"
      type="estagio.action.ListarHabilidadesVagaAction"
      scope="request"
      validate="false"
      input="listarvagas">
      <forward name="success" path="listarhabilidadesvaga" />
      <forward name="error" path="listarvagas" />
    </action>
    <action
      path="/NovaHabilidadeVagaGeral"
      type="estagio.action.NovaHabilidadeVagaGeralAction"
      scope="request"
      validate="false"
      input="novahabilidadevaga">
      <forward name="success"
path="novahabilidadevagageral" />
      <forward name="error" path="novahabilidadevaga" />
    </action>
    <action
      path="/NovaHabilidadeAlunoGeral"
      type="estagio.action.NovaHabilidadeAlunoGeralAction"
      scope="request"
      validate="false"
      input="novahabilidadealuno">
      <forward name="success"
path="novahabilidadealunogeral" />
      <forward name="error" path="novahabilidadealuno" />
    </action>
    <action
      path="/NovaHabilidadeVaga"
      type="estagio.action.NovaHabilidadeVagaAction"
      scope="request"
      validate="false"
      name="NovaHabilidadeVagaForm"
      input="listarhabilidadesvaga">
      <forward name="success" path="novahabilidadevaga" />
      <forward name="error" path="listarhabilidadesvaga" />
    </action>
    <action
      path="/SalvarHabilidadeVaga"
      type="estagio.action.SalvarHabilidadeVagaAction"
      scope="request"
      validate="true"
      name="NovaHabilidadeVagaForm"
      input="erro">
      <forward name="success" path="listarhabilidadesvaga" />
      <forward name="error" path="novahabilidadevaga" />
    </action>
    <action
      path="/SalvarHabilidadeAluno"
      type="estagio.action.SalvarHabilidadeAlunoAction"
      scope="request"
      validate="true"
      name="PesquisarHabilidadeAlunoForm"
      input="erro">
    <forward name="success" path="listarhabilidadesaluno" >
      <forward name="error" path="novahabilidadealunogeral"
    </action>
    <action
      path="/SalvarHabilidadeVagaGeral"
      type="estagio.action.SalvarHabilidadeVagaGeralAction"
      scope="request"
      validate="true"
      name="NovaHabilidadeForm"
      input="erro">
      <forward name="success" path="listarhabilidadesvaga" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/SalvarHabilidadeAlunoGeral"
      type="estagio.action.SalvarHabilidadeAlunoGeralAction"
      scope="request"
      validate="true"
      name="NovaHabilidadeForm"
      input="erro">
      <forward name="success" path="listarhabilidadesaluno" >
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ExcluirHabilidadeVaga"
      type="estagio.action.ExcluirHabilidadeVagaAction"
      scope="request"
      validate="false"
      input="listarhabilidadesvaga">
      <forward name="success" path="listarhabilidadesvaga" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ExcluirHabilidadeAluno"
      type="estagio.action.ExcluirHabilidadeAlunoAction"
      scope="request"
      validate="false"
      input="listarhabilidadesaluno">
      <forward name="success" path="listarhabilidadesaluno" >
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ListarHabilidadesAluno"
      type="estagio.action.ListarHabilidadesAlunoAction"
      scope="request"
      validate="false"
      input="dadospessoais">
      <forward name="success" path="listarhabilidadesaluno" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/NovaHabilidadeAluno"
      type="estagio.action.NovaHabilidadeAlunoAction"
      scope="request"
      validate="false"
      name="NovaHabilidadeAlunoForm"
      input="listarhabilidadesaluno">
      <forward name="success" path="novahabilidadealuno" />
      <forward name="error" path="listarhabilidadesaluno" />
    </action>
    <action
      path="/ListarTurnosDisponiveis"
      type="estagio.action.ListarTurnosDisponiveisAction"
      scope="request"
      validate="false"
```

```
    input="dadospessoais">
   <forward name="success" path="listarturnosdisponiveis" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/NovoTurnoDisponivel"
    type="estagio.action.NovoTurnoDisponivelAction"
    scope="request"
    validate="false"
    input="listarturnosdisponiveis">
    <forward name="success" path="novoturnodisponivel" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/SalvarTurnoDisponivel"
    type="estagio.action.SalvarTurnoDisponivelAction"
    scope="request"
    validate="true"
    name="NovoTurnoDisponivelForm"
    input="erro">
   <forward name="success" path="listarturnosdisponiveis" />
    <forward name="error" path="novoturnodisponivel" />
  </action>
  <action
    path="/ExcluirTurnoDisponivel"
    type="estagio.action.ExcluirTurnoDisponivelAction"
    scope="request"
    validate="false"
    input="listarturnosdisponiveis">
   <forward name="success" path="listarturnosdisponiveis" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/ListarTurnos"
    type="estagio.action.ListarTurnosAction"
    scope="request"
    validate="false"
    input="listarvagas">
    <forward name="success" path="listarturnos" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/NovoTurno"
    type="estagio.action.NovoTurnoAction"
    scope="request"
    validate="false"
    input="listarturnos">
    <forward name="success" path="novoturno" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/SalvarTurno"
    type="estagio.action.SalvarTurnoAction"
    scope="request"
    validate="true"
    name="NovoTurnoForm"
    input="erro">
    <forward name="success" path="listarturnos" />
    <forward name="error" path="novoturno" />
  </action>
  <action
    path="/ExcluirTurno"
    type="estagio.action.ExcluirTurnoAction"
    scope="request"
    validate="false"
    input="listarturnos">
    <forward name="success" path="listarturnos" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/BuscarVagas"
    type="estagio.action.BuscarVagasAction"
    scope="request"
    validate="false"
    input="inicial">
    <forward name="success" path="buscarvagas" />
    <forward name="error" path="inicial" />
  </action>
  <action
    path="/CriarTermoCompromisso"
    type="estagio.action.CriarTermoCompromissoAction"
    scope="request"
    validate="false"
    input="inicial">
    <forward name="success" path="buscarestagiario" />
    <forward name="error" path="inicial" />
  </action>
  <action
    path="/BuscarEstagiario"
    type="estagio.action.BuscarEstagiarioAction"
    scope="request"
    validate="true"
    name="BuscarEstagiarioForm"
    input="erro">
    <forward name="success" path="selecionarestagiario" />
    <forward name="error" path="inicial" />
  </action>
  <action
    path="/NovoTermoCompromisso"
    type="estagio.action.NovoTermoCompromissoAction"
    scope="request"
    validate="false"
    name="SelecionarEstagiarioForm"
    input="selecionarestagiario">
  <forward name="success" path="novotermocompromisso" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/SalvarTermoCompromisso"
    type="estagio.action.SalvarTermoCompromissoAction"
    scope="request"
    validate="true"
    name="NovoTermoCompromissoForm"
    input="novotermocompromisso">
    <forward name="success" path="termocompromisso" />
    <forward name="error" path="novotermocompromisso" />
  </action>
  <action
    path="/EditarTermo"
    type="estagio.action.EditarTermoAction"
    scope="request"
    validate="false"
    name="NovoTermoCompromissoForm"
    input="listartermos">
  <forward name="success" path="novotermocompromisso" />
    <forward name="error" path="erro" />
  </action>
  <action
    path="/ListarTermos"
    type="estagio.action.ListarTermosAction"
    scope="request"
    validate="false"
    input="inicial">
    <forward name="success" path="listartermos" />
    <forward name="error" path="inicial" />
  </action>
  <action
    path="/ExcluirTermo"
    type="estagio.action.ExcluirTermoAction"
    scope="request"
    validate="false"
    input="listartermos">
    <forward name="success" path="listartermos" />
    <forward name="error" path="erro" />
```

```
    </action>
    <action
      path="/ExcluirTermoCompromisso"
      type="estagio.action.ExcluirTermoCompromissoAction"
      scope="request"
      validate="false"
      input="listartermos">
      <forward name="success"
path="pesquisartermoscompromisso" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ListarAgentesIntegracao"
      type="estagio.action.ListarAgentesIntegracaoAction"
      scope="request"
      validate="false"
      input="inicial">
      <forward name="success" path="listaragentes" />
      <forward name="error" path="inicial" />
    </action>
    <action
      path="/ExcluirAgente"
      type="estagio.action.ExcluirAgenteAction"
      scope="request"
      validate="false"
      input="listaragentes">
      <forward name="success" path="listaragentes" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ListarHabilidades"
      type="estagio.action.ListarHabilidadesAction"
      scope="request"
      validate="false"
      input="inicial">
      <forward name="success" path="listarhabilidades" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ExcluirHabilidade"
      type="estagio.action.ExcluirHabilidadeAction"
      scope="request"
      validate="false"
      input="listarhabilidades">
      <forward name="success" path="listarhabilidades" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ListarCursos"
      type="estagio.action.ListarCursosAction"
      scope="request"
      validate="false"
      input="inicial">
      <forward name="success" path="listarcursos" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/NovoCurso"
      forward="novocurso"/>
    <action
      path="/SalvarCurso"
      type="estagio.action.SalvarCursoAction"
      scope="request"
      validate="true"
      name="NovoCursoForm"
      input="novocurso">
      <forward name="success" path="listarcursos" />
      <forward name="error" path="novocurso" />
    </action>
    <action
      path="/ExcluirCurso"
      type="estagio.action.ExcluirCursoAction"
      scope="request"
      validate="false"
      input="listarcursos">
      <forward name="success" path="listarcursos" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ConsultarCursosVagas"
      type="estagio.action.ConsultarCursosVagasAction"
      scope="request"
      validate="false"
      name="ConsultarCursosVagasForm"
      input="inicial">
      <forward name="success" path="consultarcursosvagas" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/BuscarUsuario"
      forward="buscarusuario"/>
    <action
      path="/PesquisarUsuario"
      type="estagio.action.PesquisarUsuarioAction"
      scope="request"
      validate="false"
      name="BuscarUsuarioForm"
      input="inicial">
      <forward name="success" path="listarusuarios" />
      <forward name="error" path="inicial" />
    </action>
    <action
      path="/ExcluirCadastro"
      type="estagio.action.ExcluirCadastroAction"
      scope="request"
      input="listarusuarios">
      <forward name="success" path="buscarusuario" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ListarTermosCompromisso"
      type="estagio.action.ListarTermosCompromissoAction"
      scope="request"
      validate="false"
      input="inicial">
      <forward name="success"
path="listartermoscompromisso" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/ListarEmpresas"
      type="estagio.action.ListarEmpresasAction"
      scope="request"
      validate="false"
      input="inicial">
      <forward name="success" path="listarempresas" />
      <forward name="error" path="erro" />
    </action>
    <action
      path="/BuscarEmpresas"
      forward="pesquisarempresas" />
    <action
      path="/PesquisarEmpresas"
      type="estagio.action.PesquisarEmpresasAction"
      scope="request"
      validate="true"
      name="PesquisarEmpresasForm"
      input="pesquisarempresas">
      <forward name="success" path="listarempresas" />
      <forward name="error" path="pesquisarempresas" />
    </action>
    <action
      path="/PesquisarUnidadeEstagioInterno"
      type="estagio.action.PesquisarUnidadeAction"
```

```
        scope="request"
        validate="true"
        name="PesquisarEmpresasForm"
        input="buscarcnpj">
        <forward name="success" path="listarunidade" />
        <forward name="error" path="buscarcnpj" />
      </action>
      <action
        path="/BuscarTermosCompromisso"
        forward="pesquisartermoscompromisso" />
      <action
        path="/PesquisarTermosCompromisso"
type="estagio.action.PesquisarTermosCompromissoAction"
        scope="request"
        validate="true"
        name="PesquisarTermosCompromissoForm"
        input="pesquisartermoscompromisso">
        <forward name="success"
path="listartermoscompromisso" />
        <forward name="error"
path="pesquisartermoscompromisso" />
      </action>
      <action
        path="/ExcluirEmpresa"
        type="estagio.action.ExcluirEmpresaAction"
        scope="request"
        input="erro">
        <forward name="success" path="pesquisarempresas" />
        <forward name="error" path="erro" />
      </action>
      <action
        path="/BuscarConvenios"
        forward="pesquisarconvenios" />
      <action
        path="/PesquisarConvenios"
        type="estagio.action.PesquisarConveniosAction"
        scope="request"
        validate="true"
        name="PesquisarConveniosForm"
        input="pesquisarconvenios">
        <forward name="success" path="listarconvenios" />
        <forward name="error" path="pesquisarconvenios" />
      </action>
      <action
        path="/BuscarEmpresaSemConvenio"
        forward="pesquisarempresasemconvenio" />
      <action
        path="/PesquisarEmpresasSemConvenio"
type="estagio.action.PesquisarEmpresasSemConvenioAction"
        scope="request"
        validate="true"
        name="PesquisarConveniosForm"
        input="pesquisarempresasemconvenio">
        <forward name="success"
path="listarempresassemconvenio" />
        <forward name="error"
path="pesquisarempresasemconvenio" />
      </action>
      <action
        path="/BuscarHabilidadeAluno"
        type="estagio.action.BuscarHabilidadeAlunoAction"
        scope="request"
        validate="true"
        input="novahabilidadealunogeral">
        <forward name="success"
path="pesquisarhabilidadealuno" />
        <forward name="error" path="erro" />
      </action>
      <action
        path="/PesquisarHabilidadeAluno"
        type="estagio.action.PesquisarHabilidadeAlunoAction"
        scope="request"
        validate="false"
        name="PesquisarHabilidadeAlunoForm"
        input="pesquisarhabilidadealuno">
        <forward name="success" path="novahabilidadealuno" />
      <forward name="error" path="pesquisarhabilidadealuno" />
    </action>
     <action
        path="/BuscarRequisitos"
        type="estagio.action.BuscarRequisitosAction"
        scope="request"
        validate="true"
        input="novahabilidadevagageral">
        <forward name="success" path="buscarrequisitos" />
        <forward name="error" path="erro" />
    </action>
    <action
        path="/SelecionarRelatorioSeguro"
        type="estagio.action.SelecionarRelatorioSeguroAction">
        <forward name="success" path="selecionarrelatorio" />
    </action>
    <action
        path="/RelatorioSeguro"
        type="estagio.action.RelatorioSeguroAction"
        scope="request"
        validate="false"
        name="SelecionarRelatorioForm"
        input="inicial">
        <forward name="success" path="linkrelatoriopdf" />
        <forward name="error" path="erro" />
    </action>
        <action
        path="/InformaUserName"
        forward="informausername" />
    <action
        path="/EnviaSenha"
        type="estagio.action.EnviaSenhaAction"
        scope="request"
        validate="true"
        name="InformaUserNameForm"
        input="informausername">
        <forward name="success" path="welcome" />
        <forward name="error" path="informausername" />
    </action>
     <action
        path="/EditarEmail"
        type="estagio.action.EditarEmailAction"
        scope="request"
        validate="false"
        name="ServidorSmtpForm"
        input="inicial">
        <forward name="success" path="novoservidorsmtp" />
        <forward name="error" path="erro" />
    </action>
    <action
        path="/SalvarServidorSmtp"
        type="estagio.action.SalvarServidorSmtpAction"
        scope="request"
        validate="true"
        name="ServidorSmtpForm"
        input="novoservidorsmtp">
        <forward name="success" path="sucesso" />
        <forward name="error" path="erro" />
    </action>
    <action
        path="/AlterarSenha"
        type="estagio.action.AlterarSenhaAction"
        scope="request"
        validate="false"
        input="inicial">
        <forward name="success" path="alterarsenha" />
        <forward name="error" path="erro" />
    </action>
```

```
    <action
      path="/SalvarSenhaAlterada"
      type="estagio.action.SalvarSenhaAlteradaAction"
      scope="request"
      validate="true"
      name="AlterarSenhaForm"
      input="inicial">
      <forward name="success" path="sucesso" />
      <forward name="error" path="erro" />
    </action>
  <!-- sample input and input submit actions
    <action
      path="/Input"
      type="org.apache.struts.actions.ForwardAction"
      parameter="/pages/Input.jsp"/>
    <action
      path="/InputSubmit"
      type="app.InputAction"
      name="inputForm"
      scope="request"
      validate="true"
      input="/pages/Input.jsp"/>
      <action
        path="/edit*"
        type="app.Edit{1}Action"
        name="inputForm"
        scope="request"
        validate="true"
        input="/pages/Edit{1}.jsp"/>
  end samples -->
  </action-mappings>
<!--
================================================
Controller Configuration -->
  <controller
processorClass="org.apache.struts.tiles.TilesRequestProcessor"/>
<!-- ========================================
Message Resources Definitions -->
  <message-resources parameter="MessageResources" />
<!--
================================================
Plug Ins Configuration -->
 <!-- ========================= Tiles plugin -->
 <!--    This plugin initialize Tiles definition factory. This later
can takes some
parameters explained here after. The plugin first read parameters
from  web.xml, thenoverload them with parameters defined here.
All parameters  are optional.
    The plugin should be declared in each struts-config file.
     - definitions-config: (optional)
          Specify configuration file names. There can be several
comma separated file names (default: ?? )
     - moduleAware: (optional - struts1.1)
          Specify if the Tiles definition factory is module aware. If
true (default), there will be one factory for each Struts module.
If false, there will be one common factory for all module. In this
later case, it is still needed to declare one plugin per module.
The factory will be initialized with parameters found in the first
initialized plugin (generally the one associated with the default
module).  true : One factory per module. (default)
false : one single shared factory for all modules
   - definitions-parser-validate: (optional)
Specify if xml parser should validate the Tiles configuration file.
 true : validate. DTD should be specified in file header (default)
false : no validation
 Paths found in Tiles definitions are relative to the main context.
 --> <plug-in className="org.apache.struts.tiles.TilesPlugin" >
 <!-- Path to XML definition file -->
    <set-property property="definitions-config"
    value="/WEB-INF/tiles-defs.xml" />
    <!-- Set Module-awareness to true -->
    <set-property property="moduleAware" value="true" />
  </plug-in>
 <!-- ========================= Validator plugin -->
  <plug-in
className="org.apache.struts.validator.ValidatorPlugIn">
    <set-property
      property="pathnames"
      value="/WEB-INF/validator-rules.xml,/WEB-
INF/validation.xml"/>
  </plug-in>

</struts-config>
```

6.4.108 validation.xml

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE form-validation PUBLIC
       "-//Apache Software Foundation//DTD Commons Validator
Rules Configuration 1.1.3//EN"
"http://jakarta.apache.org/commons/dtds/validator_1_1_3.dtd">
<form-validation> <!--
This is a minimal Validator form file with a couple of examples.
-->
   <global>
 <!-- An example global constant
      <constant>
        <constant-name>postalCode</constant-name>
        <constant-value>^\d{5}\d*$</constant-value>
      </constant>
      end example-->
  </global>
   <formset>
      <form name="LogonForm">
        <field
          property="username"
          depends="required">
            <arg0 key="prompt.username"/>
        </field>
        <field
          property="password"
          depends="required">
            <arg0 key="prompt.password"/>
        </field>
      </form>
      <form name="DadosPessoaisForm">
        <field property="email"
            depends="required,email">
          <arg  key="prompt.email"/>
        </field>
        <field property="nome_pessoa"
            depends="required">
         <arg    key="prompt.fullName"/>
        </field>
        <field property="usuario"
        depends="required,mask">
            <arg key="prompt.username"/>
            <var>
              <var-name>mask</var-name>
              <var-value>^[0-9a-zA-Z]*$</var-value>
            </var>
        </field>
        <field property="dataNascimento"
            depends="required,date">
          <arg    key="prompt.dataNascimento"/>
          <var>
            <var-name>datePatternStrict</var-name>
            <var-value>dd/MM/yyyy</var-value>
          </var>
        </field>
        <field property="fone"
      depends="required,minlength">
          <arg0    key="prompt.fone"/>
          <arg1    name="minlength" key="${var:minlength}"
resource="false"/>
```

```
        <var><var-name>minlength</var-name><var-
value>8</var-value></var>
       </field>
          <field property="cpf"
          depends="required,cpf, maxlength">
        <arg0    key="prompt.cpf"/>
        <arg1    name="maxlength" key="${var:maxlength}"
resource="false"/>
        <var><var-name>maxlength</var-name><var-
value>11</var-value></var>
       </field>
        <field property="bairro"
            depends="required">
        <arg    key="prompt.bairro"/>
       </field>
       <field property="matricula"
            depends="required">
         <arg  key="prompt.matricula"/>
       </field>
       <field property="cep"
              depends="required,minlength">
        <arg0    key="prompt.cep"/>
        <arg1    name="minlength" key="${var:minlength}"
resource="false"/>
        <var><var-name>minlength</var-name><var-
value>8</var-value></var>
       </field>
       <field property="estado"
            depends="required">
         <arg  key="prompt.estado"/>
       </field>
       <field property="municipio"
            depends="required">
        <arg    key="prompt.municipio"/>
       </field>
       <field property="numero"
            depends="required,integer,intRange">
         <arg0    key="prompt.numero"/>
         <arg1 name="intRange" key="${var:min}"
resource="false"/>
         <arg2 name="intRange" key="${var:max}"
resource="false"/>
         <var><var-name>min</var-name><var-value>0</var-
value></var>
         <var><var-name>max</var-name><var-
value>20000</var-value></var>
       </field>
       <field property="pais"
            depends="required">
         <arg  key="prompt.pais"/>
       </field>
       <field property="rg"
            depends="required, minlength">
        <arg    key="prompt.rg"/>
        <arg1    name="minlength" key="${var:minlength}"
resource="false"/>
        <var><var-name>minlength</var-name><var-
value>6</var-value></var>
       </field>
       <field property="rua"
            depends="required">
         <arg  key="prompt.rua"/>
       </field>
     </form>
     <form name="NovoConvenioForm">
       <field property="dataInicio"
            depends="required,date">
         <arg    key="prompt.dataInicio"/>
         <var>
           <var-name>datePatternStrict</var-name>
           <var-value>dd/MM/yyyy</var-value>
         </var>
       </field>
       <field property="dataFim"
            depends="required,date">
         <arg    key="prompt.dataFim"/>
         <var>
           <var-name>datePatternStrict</var-name>
           <var-value>dd/MM/yyyy</var-value>
         </var>
       </field>
     </form>
     <form name="/SalvarEmpresa">
      <field
          property="cnpj"
          depends="cnpj, required">
        <arg    key="prompt.cnpj"/>
       </field>
        <field property="razaoSocial"
            depends="required">
        <arg    key="prompt.razaoSocial"/>
       </field>
        <field property="representante"
            depends="required">
        <arg    key="prompt.representante"/>
       </field>
       <field property="ramo"
            depends="required">
        <arg    key="prompt.ramo"/>
       </field>
      <field property="fone"
       depends="required,minlength">
        <arg0    key="prompt.fone"/>
        <arg1    name="minlength" key="${var:minlength}"
resource="false"/>
        <var><var-name>minlength</var-name><var-
value>8</var-value></var>
       </field>
        <field property="fax"
                depends="required,minlength">
        <arg0    key="prompt.fax"/>
        <arg1    name="minlength" key="${var:minlength}"
resource="false"/>
        <var><var-name>minlength</var-name><var-
value>8</var-value></var>
       </field>
      <field property="email"
            depends="required,email">
        <arg  key="prompt.email"/>
       </field>
      <field property="rua"
            depends="required">
        <arg  key="prompt.rua"/>
       </field>
       <field property="numero"
            depends="required,integer,intRange">
         <arg0    key="prompt.numero"/>
         <arg1 name="intRange" key="${var:min}"
resource="false"/>
         <arg2 name="intRange" key="${var:max}"
resource="false"/>
         <var><var-name>min</var-name><var-value>0</var-
value></var>
         <var><var-name>max</var-name><var-
value>20000</var-value></var>
       </field>
      <field property="bairro"
            depends="required">
        <arg    key="prompt.bairro"/>
       </field>
      <field property="cep"
depends="required,minlength">
        <arg0    key="prompt.cep"/>
```

```
      <arg1    name="minlength" key="${var:minlength}"
resource="false"/>
        <var><var-name>minlength</var-name><var-
value>8</var-value></var>
       </field>
      <field property="municipio"
          depends="required">
       <arg    key="prompt.municipio"/>
      </field>

      <field property="pais"
          depends="required">
        <arg  key="prompt.pais"/>
      </field>
       </form>
      <form name="NovaVagaForm">
      <field property="setor"
          depends="required">
        <arg  key="prompt.setor"/>
      </field>
      <field property="salario"
          depends="required">
        <arg  key="prompt.salario"/>
      </field>
      <field property="horas"
          depends="required">
        <arg  key="prompt.horas"/>
      </field>
    </form>

    <form name="NovaHabilidadeVagaForm">
      <field property="codHabilidade"
          depends="required">
        <arg  key="prompt.codHabilidade"/>
      </field>
    </form>

    <form name="NovaHabilidadeAlunoForm">
      <field property="codHabilidade"
          depends="required">
        <arg  key="prompt.codHabilidade"/>
      </field>
    </form>

    <form name="NovaHabilidadeForm">
      <field property="descricaoHabilidade"
          depends="required">
        <arg  key="prompt.descricaoHabilidade"/>
      </field>
    </form>
    <form name="NovoTurnoDisponivelForm">
      <field property="horaInicio"
          depends="required, maxlength">
        <arg key="prompt.horaInicio"/>
        <arg1 name="maxlength" key="${var:maxlength}"
resource="false"/>
        <var><var-name>maxlength</var-name><var-
value>5</var-value></var>
       </field>
    <field property="horaFim"
          depends="required,maxlength">
        <arg key="prompt.horaFim"/>
        <arg1 name="maxlength" key="${var:maxlength}"
resource="false"/>
        <var><var-name>maxlength</var-name><var-
value>5</var-value></var>
       </field>
    </form>
    <form name="NovoTurnoForm">
      <field property="horaInicio"
          depends="required, maxlength">
        <arg key="prompt.horaInicio"/>
        <arg1 name="maxlength" key="${var:maxlength}"
resource="false"/>
        <var><var-name>maxlength</var-name><var-
value>5</var-value></var>
       </field>
    <field property="horaFim"
          depends="required,maxlength">
        <arg key="prompt.horaFim"/>
        <arg1 name="maxlength" key="${var:maxlength}"
resource="false"/>
        <var><var-name>maxlength</var-name><var-
value>5</var-value></var>
       </field>
    </form>
    <form name="NovoTermoCompromissoForm">
      <field property="dataInicio"
          depends="required,date">
        <arg    key="prompt.dataInicio"/>
        <var>
          <var-name>datePatternStrict</var-name>
          <var-value>dd/MM/yyyy</var-value>
        </var>
      </field>
      <field property="dataFim"
          depends="required,date">
        <arg    key="prompt.dataFim"/>
        <var>
          <var-name>datePatternStrict</var-name>
          <var-value>dd/MM/yyyy</var-value>
        </var>
      </field>
    </form>
    <form name="NovoAgenteIntegracaoForm">
      <field property="agente"
          depends="required">
        <arg  key="prompt.nome"/>
      </field>
    </form>
    <form name="BuscarEstagiarioForm">
      <field property="nomeEstagiario"
          depends="required">
        <arg  key="prompt.nome"/>
      </field>
    </form>
    <form name="NovoCursoForm">
      <field property="nomeCurso"
          depends="required">
        <arg  key="prompt.nome"/>
      </field>
      <field property="duracao"
          depends="required">
        <arg  key="prompt.duracao"/>
      </field>
    </form>
    <form name="BuscarEmpresaForm">
      <field
        property="cnpj"
        depends="cnpj, required">
       <arg    key="prompt.cnpj"/>
      </field>
    </form>
    <form name="BuscarUsuarioForm">
      <field
        property="nomeUsuario"
        depends="required">
       <arg    key="prompt.nome"/>
      </field>
    </form>
    <form name="PesquisarEmpresasForm">
      <field
        property="parteDoNome"
        depends="required">
```

```
      <arg    key="prompt.razaoSocial"/>
    </field>
  </form>
  <form name="PesquisarConveniosForm">
    <field
      property="valor"
      depends="required">
     <arg    key="prompt.valor"/>
    </field>
  </form>
  <form name="PesquisarTermosCompromissoForm">
    <field
      property="valor"
      depends="required">
     <arg    key="prompt.valor"/>
    </field>
  </form>
  <form name="ServidorSmtpForm">
    <field
      property="endereco_email"
      depends="required">
     <arg    key="prompt.email"/>
    </field>
    <field
      property="porta_smtp"
      depends="required">
     <arg    key="prompt.porta_smtp"/>
    </field>
    <field
      property="senha"
      depends="required">
     <arg    key="prompt.senha"/>
    </field>
    <field
      property="url"
      depends="required">
     <arg    key="prompt.url"/>
    </field>
    <field
      property="username"
      depends="required">
     <arg    key="prompt.username"/>
    </field>
  </form>
  <form name="InformaUserNameForm">
    <field
      property="username"
      depends="required">
     <arg    key="prompt.username"/>
    </field>
    <field
      property="email"
      depends="required, email">
     <arg    key="prompt.email"/>
    </field>
  </form>
  <form name="AlterarSenhaForm">
    <field
      property="senha"
      depends="required">
     <arg    key="prompt.password"/>
    </field>
    <field
      property="senha2"
      depends="required">
     <arg    key="prompt.password2"/>
    </field>
  </form>
 </formset>
</form-validation>
```

6.4.109 validation-rules.xml

```
<!DOCTYPE form-validation PUBLIC
"-//Apache Software Foundation//DTD Commons Validator Rules
Configuration 1.1.3//EN"
http://jakarta.apache.org/commons/dtds/validator_1_1_3.dtd">
<!-- $Id: validator-rules.xml 159291 2005-03-28 20:19:29Z
niallp $ This file contains the default Struts Validator pluggable
validator definitions. It should be placed somewhere under
/WEB-INF and referenced in the struts-config.xml under the
plug-in element for the ValidatorPlugIn.
    <plug-in
className="org.apache.struts.validator.ValidatorPlugIn">
      <set-property property="pathnames" value="/WEB-
INF/validator-rules.xml,
/WEB-INF/validation.xml"/>
    </plug-in>
  These are the default error messages associated with
  each validator defined in this file. They should be
  added to your projects ApplicationResources.properties
  file or you can associate new ones by modifying the
  pluggable validators msg attributes in this file.
  # Struts Validator Error Messages
  errors.required={0} is required.
  errors.minlength={0} can not be less than {1} characters.
  errors.maxlength={0} can not be greater than {1} characters.
  errors.invalid={0} is invalid.
  errors.byte={0} must be a byte.
  errors.short={0} must be a short.
  errors.integer={0} must be an integer.
  errors.long={0} must be a long.
  errors.float={0} must be a float.
  errors.double={0} must be a double.
  errors.date={0} is not a date.
  errors.range={0} is not in the range {1} through {2}.
  errors.creditcard={0} is an invalid credit card number.
  errors.email={0} is an invalid e-mail address.
  Note: Starting in Struts 1.2.0 the default javascript definitions
have been consolidated to commons-validator. The default can
be overridden by supplying a <javascript> element with a
CDATA section, just as in struts 1.1.->
<form-validation>
  <global>
    <validator name="required"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateRequired"
      methodParams="java.lang.Object,
                org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
            msg="errors.required"/>
    <validator name="requiredif"
          classname="org.apache.struts.validator.FieldChecks"
          method="validateRequiredIf"
          methodParams="java.lang.Object,
         org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          msg="errors.required"/>
    <validator name="validwhen"
       msg="errors.required"
classname="org.apache.struts.validator.validwhen.ValidWhen"
          method="validateValidWhen"
          methodParams="java.lang.Object,
              org.apache.commons.validator.ValidatorAction,
              org.apache.commons.validator.Field,
              org.apache.struts.action.ActionMessages,
              org.apache.commons.validator.Validator,
              javax.servlet.http.HttpServletRequest"/>
```

```
    <validator name="minlength"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateMinLength"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.minlength"
jsFunction="org.apache.commons.validator.javascript.validateMi
nLength"/>
    <validator name="maxlength"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateMaxLength"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.maxlength"
Function="org.apache.commons.validator.javascript.validateMax
Length"/>
    <validator name="mask"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateMask"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.invalid"/>
    <validator name="byte"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateByte"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.byte"
     jsFunctionName="ByteValidations"/>
    <validator name="short"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateShort"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.short"
     jsFunctionName="ShortValidations"/>
    <validator name="integer"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateInteger"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.integer"
     jsFunctionName="IntegerValidations"/>
    <validator name="long"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateLong"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.long"/>
    <validator name="float"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateFloat"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.float"
     jsFunctionName="FloatValidations"/>
    <validator name="double"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateDouble"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.double"/>
    <validator name="date"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateDate"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends=""
              msg="errors.date"
     jsFunctionName="DateValidations"/>
    <validator name="intRange"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateIntRange"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends="integer"
              msg="errors.range"/>
    <validator name="floatRange"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateFloatRange"
      methodParams="java.lang.Object,
                    org.apache.commons.validator.ValidatorAction,
                    org.apache.commons.validator.Field,
                    org.apache.struts.action.ActionMessages,
                    org.apache.commons.validator.Validator,
                    javax.servlet.http.HttpServletRequest"
            depends="float"
              msg="errors.range"/>
    <validator name="doubleRange"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateDoubleRange"
```

```
    methodParams="java.lang.Object,
                  org.apache.commons.validator.ValidatorAction,
                  org.apache.commons.validator.Field,
                  org.apache.struts.action.ActionMessages,
                  org.apache.commons.validator.Validator,
                  javax.servlet.http.HttpServletRequest"
          depends="double"
            msg="errors.range"/>
  <validator name="creditCard"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateCreditCard"
    methodParams="java.lang.Object,
                  org.apache.commons.validator.ValidatorAction,
                  org.apache.commons.validator.Field,
                  org.apache.struts.action.ActionMessages,
                  org.apache.commons.validator.Validator,
                  javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.creditcard"/>
  <validator name="email"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateEmail"
    methodParams="java.lang.Object,
                  org.apache.commons.validator.ValidatorAction,
                  org.apache.commons.validator.Field,
                  org.apache.struts.action.ActionMessages,
                  org.apache.commons.validator.Validator,
                  javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.email"/>
  <validator name="url"
        classname="org.apache.struts.validator.FieldChecks"
          method="validateUrl"
    methodParams="java.lang.Object,
                  org.apache.commons.validator.ValidatorAction,
                  org.apache.commons.validator.Field,
                  org.apache.struts.action.ActionMessages,
                  org.apache.commons.validator.Validator,
                  javax.servlet.http.HttpServletRequest"
          depends=""
            msg="errors.url"/>
           <validator name="cpf"
          classname="estagio.validation.Valida"
          method="validateCpf"
          methodParams="java.lang.Object,
                org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
          msg="errors.cpf"/>
  <validator name="cnpj"
          classname="estagio.validation.Valida"
          method="validateCnpj"
          methodParams="java.lang.Object,
                org.apache.commons.validator.ValidatorAction,
                org.apache.commons.validator.Field,
                org.apache.struts.action.ActionMessages,
                org.apache.commons.validator.Validator,
                javax.servlet.http.HttpServletRequest"
          depends=""
          msg="errors.cnpj"/>    <!--
    This simply allows struts to include the validateUtilities into a
page, it should
    not be used as a validation rule.
  -->
  <validator name="includeJavaScriptUtilities"
        classname=""
          method=""
    methodParams=""
          depends=""
              msg=""
jsFunction="org.apache.commons.validator.javascript.validateUtil
ities"/>
  </global>
</form-validation>
```

6.4.110 web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE web-app
 PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application
2.2//EN"  "http://java.sun.com/j2ee/dtds/web-app_2_2.dtd">
<web-app> <display-name>Struts Blank Application</display-
name>
  <!-- Standard Action Servlet Configuration (with debugging) --
> <servlet>
  <servlet-name>action</servlet-name>
  <servlet-class>org.apache.struts.action.ActionServlet</servlet-
class>
  <init-param>
   <param-name>config</param-name>
   <param-value>/WEB-INF/struts-config.xml</param-value>
  </init-param>
  <init-param>
   <param-name>debug</param-name>
   <param-value>2</param-value>
  </init-param>
  <init-param>
   <param-name>detail</param-name>
   <param-value>2</param-value>
  </init-param>
  <load-on-startup>2</load-on-startup>
 </servlet>
 <!-- Standard Action Servlet Mapping -->
 <servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
 </servlet-mapping>
 <!-- The Usual Welcome File List -->
 <welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
 </welcome-file-list>
<!-- Struts Tag Library Descriptors -->
 <taglib>
  <taglib-uri>/tags/struts-bean</taglib-uri>
  <taglib-location>/WEB-INF/struts-bean.tld</taglib-location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-html</taglib-uri>
  <taglib-location>/WEB-INF/struts-html.tld</taglib-location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-logic</taglib-uri>
  <taglib-location>/WEB-INF/struts-logic.tld</taglib-location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-nested</taglib-uri>
  <taglib-location>/WEB-INF/struts-nested.tld</taglib-location>
 </taglib>
 <taglib>
  <taglib-uri>/tags/struts-tiles</taglib-uri>
  <taglib-location>/WEB-INF/struts-tiles.tld</taglib-location>
 </taglib>
</web-app>
```

6.4.111 Diretório "Pages"

6.4.112 buscarcnpj.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
```

```
  Informe a Unidade Concedente do Estágio Interno
  <hr align="left" class="textos"/>
</p>
  <html:form action="PesquisarUnidadeEstagioInterno">
  <logic:present name="mensagem">
    <p align="center"><bean:write name="mensagem"/></p>
  </logic:present>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        Qualquer Parte da Razão Social:
      </th>
      <td align="left" class="textos">
        <html:text property="parteDoNome"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        Pesquisar
      </html:submit>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
          Página Inicial
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.113 buscarempresa.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Buscar Empresa Relacionada ao Novo Usuário
  <hr align="left" class="textos"/>
</p>
  <html:form action="BuscarEmpresa">
  <html:hidden property="status"/>
  <logic:present name="mensagem">
    <p align="center"><bean:write name="mensagem"/></p>
  </logic:present>
  <table align="center">
    <tr>
      <th align="left" class="textos" >
        CNPJ:
      </th>
      <td align="left" class="textos">
        <html:text property="cnpj" styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit>
        <bean:message key="button.save"/>
      </html:submit>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="NovoUsuario" styleClass="menu">
        Voltar
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.114 buscarestagiário.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="BuscarEstagiario">
  <p align="center" class="textos">
    Buscar Estagiário
    <hr align="left" class="textos"/>
  </p>
  <html:hidden property="codVaga"/>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        Qualquer parte do Nome:
      </th>
      <td align="left" class="textos">
        <html:text property="nomeEstagiario"
styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit>
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset>
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <logic:equal name="user" property="status"
value="empresa">
          <html:link action="ListarVagas"
styleClass="menu">Voltar para Vagas</html:link>
        </logic:equal>
        <logic:notEqual name="user" property="status"
value="empresa">
          <html:link action="ListarVagas?estagio=interno"
styleClass="menu">Voltar para Vagas</html:link>
        </logic:notEqual>
      </td>
    </tr>
  </table>
</p>
```

6.4.115 buscarrequisitos.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Busca por Requisitos
  <hr align="left" class="textos"/>
</p>
```

```
  <html:form action="SalvarHabilidadeVaga">
    <html:hidden property="codVaga"/>
    <table align="center">
    <logic:present name="requisitos">
    <p align="center">Requisitos encontrados em vagas de
estágio para <bean:write name="curso"/>:</p>
          <tr>
            <th align="left" class="textos">
              Requisitos:
            </th>
          </tr>
          <tr>
          <%! int multiplicando = 8;%>
          <%! int multiplicador = 1;%>
          <%! int multiploTres = 0;%>
          <%
          java.util.ArrayList requisitos = new
java.util.ArrayList();
          requisitos = (java.util.ArrayList)
request.getAttribute("requisitos");
          for (int i=0;i<requisitos.size();++i){
              estagio.bean.Habilidade requisito =
(estagio.bean.Habilidade) requisitos.get(i);
          %>
          <%--logic:iterate id="habilidade"
name="todashabilidades" indexId="indice"--%>
            <%multiploTres = multiplicando*multiplicador;%>
            <td align="right" class="textos">
              <%--html:multibox property="codHabilidade">
                <bean:write name="habilidade"
property="codHabilidade" />
              </html:multibox--%>
              <input type="checkbox" name="codHabilidade"
value="<%=requisito.getCodHabilidade()%>"
class="formulario">
            </td>
            <td align="left" class="textos">
              <%--bean:write name="habilidade"
property="descricaoHabilidade"/--%>
              <%=requisito.getDescricaoHabilidade()%>
            </td>
             <%if(i == multiploTres){
              ++multiplicador;%>
              </tr>
              <tr>
            <%}%>
          <%--/logic:iterate--%>
          <%}%>
          </tr>
        </logic:present>
        <logic:notPresent name="requisitos">
          <tr>
            <td align="left" class="textos">
              Não foram encontrados requisitos. Talvez todos já
tenham sido inseridos na vaga.
            </td>
          </tr>
        </logic:notPresent>
      </table>
      <logic:present name="requisitos">
        <div class="button" id="botao">
          <div align="center">
            <html:submit>
              Incluir na vaga
            </html:submit>
          </div>
        </div>
      </logic:present>
    </html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
      <tr>
        <td>
          <html:link action="ListarHabilidadesVaga"
paramId="codvaga"
            paramName="codvaga" styleClass="menu">
          Voltar para Requisitos
          </html:link>
        </td>
      </tr>
  </table>
</p>
```

6.4.116 buscarusuario.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<logic:equal name="user" value="admin" property="status">
<html:form action="PesquisarUsuario">
  <p align="center" class="textos">
    Buscar Usuário
    <hr align="left" class="textos"/>
  </p>
  <table align="center">
    <tr>
      <th align="left">
        Qualquer parte do Nome ou Username:
      </th>
      <td align="left">
        <html:text property="nomeUsuario"
styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit>
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset>
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.117 buscarvagas.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Vagas para Estágio para o perfil de <bean:write name="user"
property="usuario"/>
</p>
```

```
<logic:present name="vagas">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Inclusao da Vaga</th>
      <th align="center" class="textos">Setor</th>
      <th align="center" class="textos">Carga Horária</th>
      <th align="center" class="textos">Bolsa</th>
      <th align="center" class="textos">Requisitos</th>
      <th align="center" class="textos">Horário</th>
      <th align="center" class="textos">Semestre Mínimo</th>
      <th align="center" class="textos">Vale Transporte</th>
      <th align="center" class="textos">Outros Benefícios</th>
      <th align="center" class="textos">Empresa</th>
      <th align="center" class="textos">Endereço</th>
      <th align="center" class="textos">Contato</th>
    </tr>
    <logic:iterate name="vagas" id="vaga">
      <tr>
        <td align="center" class="textos">
          <bean:write name="vaga" property="inclusao"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="vaga" property="setor"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="vaga" property="horas"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="vaga" property="salario"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga"
property="habilidades"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga" property="turnos"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga" property="semestre"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga"
property="valeTransporte"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga"
property="outrosBeneficios"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga"
property="razaoSocial"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga" property="endereco"/>
        </td>
        <td align="center" class="textos">
            <bean:write name="vaga" property="contato"/>
        </td>
      </tr>
    </logic:iterate>
  </table>
</logic:present>
<logic:notPresent name="vagas">
  <p align="center" class="textos">NÃO HÁ VAGAS NO
MOMENTO.</p>
</logic:notPresent>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.118 consultarcursosvags.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="ConsultarCursosVagas">
  <logic:equal name="user" value="admin" property="status">
    <p align="left" class="textos">
      Consultar Vagas
      <hr align="left" class="textos"/>
    </p>
    <table align="center">
      <tr>
        <th align="left" class="textos">
          Curso:
        </th>
        <td align="left" class="textos">
          <html:select property="codCurso"
onchange="form.submit()" styleClass="formulario">
            <html:options collection="cursos"
property="value" labelProperty="label"/>
          </html:select>
        </td>
      </tr>
    </table>
    <hr align="left" class="textos"/>
    <logic:present name="vagas">
      <table border="1" align="center">
        <tr>
          <th align="center" class="textos">Empresa</th>
          <th align="center" class="textos">Setor</th>
          <th align="center" class="textos">Carga
Horária</th>
          <th align="center" class="textos">Bolsa</th>
          <th align="center" class="textos">Sexo</th>
          <th align="center" class="textos">Desde</th>
          <th align="center" class="textos">Requisitos</th>
          <th align="center" class="textos">Horário</th>
          <th align="center" class="textos">Situação</th>
        </tr>
        <logic:iterate name="vagas" id="vaga">
          <tr>
            <td align="center" class="textos">
              <bean:write name="vaga"
property="empresa"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="setor"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="horas"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="salario"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="sexo"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga"
property="inclusao"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga"
property="habilidades"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="turnos"/>
```

```
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="situacao"/>
            </td>
          </tr>
        </logic:iterate>
      </table>
    </logic:present>
    <logic:notPresent name="vagas">
      <logic:notPresent name="time">
        <p align="center" class="textos">NÃO HÁ VAGAS
NO MOMENTO.</p>
      </logic:notPresent>
      <logic:present name="time">
        <p align="center" class="textos">SELECIONE O
CURSO.</p>
      </logic:present>
    </logic:notPresent>
  </logic:equal>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
        Pagina Inicial
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.119 dadospessoais.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Dados Pessoais
</p>
<html:form action="/SalvarCadastro" focus="usuario">
  <html:hidden property="action"/>
  <table align="center">
    <tr>
      <td align="left" class="textos">
        Nome de Usuario:
      </td>
      <td align="left">
        <logic:equal name="DadosPessoaisForm"
property="action"
          scope="request" value="Create">
          <logic:notEqual name="logado" scope="session"
value="true">
            <html:text property="usuario" size="16"
styleClass="formulario"/>
          </logic:notEqual>
          <logic:equal name="logado" scope="session"
value="true">
            <html:hidden property="usuario" write="true"
styleClass="formulario"/>
          </logic:equal>
        </logic:equal>
        <logic:equal name="DadosPessoaisForm"
property="action"
          scope="request" value="Edit">
          <html:hidden property="usuario" write="true"
styleClass="formulario"/>
        </logic:equal>
      </td>
    </tr>
    <logic:equal name="DadosPessoaisForm" property="action"
      scope="request" value="Create">
      <logic:notPresent name="logado" scope="session">
        <tr>
          <td align="left" class="textos">
            Senha:
          </td>
          <td align="left">
            <html:password property="senha" size="16"
maxlength="16" styleClass="formulario"/>
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Confirme a Senha:
          </th>
          <td align="left">
            <html:password property="senha2" size="16"
maxlength="16" styleClass="formulario"/>
          </td>
        </tr>
      </logic:notPresent>
    </logic:equal>
    <logic:equal name="DadosPessoaisForm" property="action"
scope="request" value="Edit">
      <html:hidden property="senha" />
      <html:hidden property="senha2" />
      <tr>
        <th align="left" class="textos">
          <html:link action="AlterarSenha"
paramId="codalunoestagio"
            paramName="codalunoestagio"
styleClass="menu">
          Alterar Senha
          </html:link>
        </th>
      </tr>
    </logic:equal>
    <tr>
      <th align="left" class="textos">
        Nome
      </th>
      <td align="left">
        <logic:notEqual name="logado" scope="session"
value="true">
          <html:text property="nome_pessoa" size="42,5"
styleClass="formulario"/>
        </logic:notEqual>
        <logic:equal name="logado" scope="session"
value="true">
          <html:hidden property="nome_pessoa" write="true"
styleClass="formulario"/>
        </logic:equal>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Data de Nascimento:
      </th>
      <td align="left" class="textos">
        <logic:notEqual name="logado" scope="session"
value="true">
          <html:text property="dataNascimento" size="16"
styleClass="formulario"/> <em>exemplo 01/01/2006</em>
        </logic:notEqual>
        <logic:equal name="logado" scope="session"
value="true">
          <html:hidden property="dataNascimento"
write="true" styleClass="formulario"/>
        </logic:equal>
      </td>
      <th align="left" class="textos">
        Sexo:
      </th>
```

```
        <td align="left">
          <html:select property="sexo" styleClass="formulario">
            <html:options collection="sexos" property="value"
labelProperty="label"/>
          </html:select>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          E-mail:
        </th>
        <td align="left">
          <html:text property="email" size="20"
styleClass="formulario"/>
        </td>
        <th align="left" class="textos">
          Curso:
        </th>
        <td align="left">
          <html:select property="codCurso"
styleClass="formulario">
            <html:options collection="cursos" property="value"
labelProperty="label"/>
          </html:select>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Semetre:
        </th>
        <td align="left">
          <html:select property="semestre"
styleClass="formulario">
            <html:options name="semestres"/>
          </html:select>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Fone:
        </th>
        <td align="left" class="textos">
          <html:text property="fone" size="16"
styleClass="formulario"/> <em> DDD(2 digitos)+numero
exemplo: 4135653565</em>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Rg:
        </th>
        <td align="left" class="textos">
          <html:text property="rg" size="16"
styleClass="formulario"/>
        </td>
        <th align="left" class="textos">
          Origem:
        </th>
        <td align="left">
          <html:select property="rg_naturalidade"
styleClass="formulario">
            <html:options name="estados" />
          </html:select>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          CPF nº:
        </th>
        <td align="left" class="textos">
          <html:text property="cpf" size="16"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Matrícula:
        </th>
        <td align="left" class="textos">
          <html:text property="matricula" size="16"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">Rua:</th>
        <td align="left">
          <html:text property="rua" size="42,5"
styleClass="formulario"/>
        </td>
        <th align="left" class="textos">N&ordm;:</th>
        <th align="left">
        <html:text property="numero" size="8"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Bairro:
        </th>
        <td align="left">
          <html:text property="bairro" styleClass="formulario"/>
        </td>
        <th align="left" class="textos">Complemento:</th>
        <td align="left">
          <html:text property="complemento"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">CEP:</th>
        <td align="left">
          <html:text property="cep" styleClass="formulario"/>
        </td>
        <th align="left" class="textos">Cidade:</th>
        <td align="left">
          <html:text property="municipio"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">Estado:</th>
        <td align="left">
          <html:select property="estado"
styleClass="formulario">
            <html:options name="estados" />
          </html:select>
        </td>
        <th align="left" class="textos">Pa&iacute;s:</th>
        <td align="left">
          <html:text property="pais" styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Locomoção:
        </th>
        <td align="left">
          <html:select property="locomocao"
styleClass="formulario">
            <html:options collection="locomocao"
property="value" labelProperty="label"/>
          </html:select>
        </td>
      </tr>
```

```
    <tr>
      <th class="textos" align="left">
        Fumante ?
      </th>
      <td align="left">
        <html:select property="fumante"
styleClass="formulario">
          <html:options collection="simNao" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Deficiente ?
      </th>
      <td align="left">
        <html:select property="deficiente"
styleClass="formulario">
          <html:options collection="simNao" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Disponível nos Finais de Semana ?
      </th>
      <td align="left">
        <html:select property="disponivelFds"
styleClass="formulario">
          <html:options collection="simNao" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <html:hidden property="status"/>
    <logic:present name="empresa">
      <tr>
        <th align="left" class="textos">Empresa relacionada ao
usuário:</th>
      </tr>
      <tr>
        <th align="right">
          <html:radio property="empresa" idName="empresa"
value="cnpj" styleClass="formulario"/>
        </th>
        <td align="left" class="textos">
          <bean:write name="empresa"
property="razaoSocial"/>
        </td>
      </tr>
    </logic:present>
    <logic:notPresent name="empresa">
      <html:hidden property="empresa"
styleClass="formulario"/>
    </logic:notPresent>
    <logic:equal name="action" scope="request" value="Edit">
      <logic:present name="habilidades">
        <tr>
          <th align="left" class="textos">
            <html:link action="ListarHabilidadesAluno"
paramId="codalunoestagio"
              paramName="codalunoestagio"
styleClass="menu">
              Habilidades
              </html:link>
            </th>
            <td align="left" class="textos">
              <bean:write name="habilidades"/>
            </td>
          </tr>
        </logic:present>
        <logic:notPresent name="habilidades">
          <tr>
            <th align="left" class="textos">
              <html:link action="ListarHabilidadesAluno"
paramId="codalunoestagio"
                paramName="codalunoestagio"
styleClass="menu">
                Inserir Habilidades
                </html:link>
              </th>
            </tr>
          </logic:notPresent>

          <logic:present name="turnos">
            <tr>
              <th align="left" class="textos">
                <html:link action="ListarTurnosDisponiveis"
paramId="codalunoestagio"
                  paramName="codalunoestagio"
styleClass="menu">
                  Horários Disponíveis
                  </html:link>
                </th>
                <td align="left" class="textos">
                  <bean:write name="turnos"/>
                </td>
              </tr>
            </logic:present>
            <logic:notPresent name="turnos">
              <tr>
                <th align="left" class="textos">
                  <html:link action="ListarTurnosDisponiveis"
paramId="codalunoestagio"
                    paramName="codalunoestagio"
styleClass="menu">
                    Inserir Horários Disponíveis
                    </html:link>
                  </th>
                </tr>
              </logic:notPresent>
            </logic:equal>
          </table>
          <p align="left" >
            <hr align="left" />
          </p>
          <div class="button" id="botao">
            <div align="center">
              <html:submit styleClass="formulario">
              Enviar
              </html:submit>
              <html:reset styleClass="formulario">
                <bean:message key="button.reset"/>
              </html:reset>
              <input type=button value=Voltar onclick="history.go(-1)"
class="formulario">
            </div>
          </div>
</html:form>
```

6.4.120 erro.jsp

```
<%--%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %--%>
<hr align="left" class="textos"/>
<p align="center" class="textos">
ERRO
<hr align="left" class="textos"/>
</p>
```

6.4.121 estilo.css

```
/* Documento CSS da Intranet Acadêmica - UFPR */
.fundo_azul_escuro {
        background-color: #006699;
        overflow: visible;
}
.fundo_azul_claro {
        background-color: #6699CC;
}
.fundo_cinza {
        background-color: #CCCCCC;
}
.fundo_form     { /* Fundo dos formularios */
        background-color: #CCCCCC;
}
a { /* fonte e cor dos links */
        font-family: Verdana, Arial, Helvetica, sans-serif;
        text-decoration: none;
        color: #006699;
        font-size: 12px;
}
a:hover { /* links hover mouse */
        color: #6699CC;
}
input { /* fonte e cor dos botoes e campos */
        font-family: Verdana, Arial, Helvetica, sans-serif;
        color: #006699;
}
.input { /* fonte e cor dos botoes e campos */
        font-family: Verdana, Arial, Helvetica, sans-serif;
        color: #006699;
}
.ufpr { /* fonte e cor do UFPR do titulo */
        font: normal 30px/100px Georgia, "Times New
Roman", Times, serif;
        color: #0C547E;
}
.titulo_menu { /* fonte e cor do titulo do menu */
        font-family: Verdana, Arial, Helvetica, sans-serif;
        color: #FFFFFF;
        font-size: 12px;
        font-weight: bold;
}
.titulo_form { /* fonte do titulo do formulario */
        margin-left: 15px;
        margin-right:15px;
        font-family: Arial, Helvetica, sans-serif; font-size: 12pt;
font-weight:700; color: #336699 ;
}
.letra_form     { /* fonte do formulario */
        font-family: Arial, Helvetica, sans-serif; font-size: 8pt;
font-weight: 800; color: #000000;

        margin-right: 20px;
}
.edit { /* alinhamento dos campos */
        font-family: Verdana, Arial, Helvetica, sans-serif;
        font-size: 10px;
        margin: -10px auto -5px 15px;
}
.path { /* fonte dos direitos */
        font-family: Verdana, Arial, Helvetica, sans-serif;
        font-size: 10px;
        font-weight: 300;
        color: #666666
}
.form { /* alinhamento dos formularios */
        text-align: left;
        width: 520px;
        margin-left: auto;
        margin-right: auto;
}
.branco { /* cor do divisor | do menu */
        color: #FFFFFF;
}
```

6.4.122 footer.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
  <table width="100%" border="0" cellspacing="0"
cellpadding="0">
    <tr>
      <td><html:img src="struts-power.gif" alt="Powered By
Struts" align="right"/></td>
      <td width="30" align="right"><img src="images/canto.gif"
width="30" height="30"></td>
      <td width="320" align="right" class="fundoazul">&copy;
Universidade Federal do Paran&aacute; 2005 - Politica de
Privacidade </td>
    </tr>
  </table>
```

6.4.123 index.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>
<html:html>
  <head>
    <title>Seja bem vindo ao Sistema de Controle de
Est&aacute;gios da UFPR</title>
    <meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
    <html:base/>
    <link href="images/tcc.css" rel="stylesheet"
type="text/css">
  </head>
  <script language="JavaScript">
    function formatar(src, mask)
    {
    var i = src.value.length;
    var saida = mask.substring(0,1);
    var texto = mask.substring(i)
    if (texto.substring(0,1) != saida)
    {
    src.value += texto.substring(0,1);
    }
    }
  </script>
  <body leftmargin="0" topmargin="0" marginwidth="0"
marginheight="0">
    <table width="100%" height="161" border="0"
cellpadding="0" cellspacing="0">
      <tr>
        <td width="7" height="28"
class="fundoazul"> </td>
        <td width="129" rowspan="2" align="center"
valign="bottom" class="fundoazul"><img src="images/logo.gif"
width="129" height="85"></td>
        <td width="713" class="fundoazul"> </td>
        <td width="288" class="fundoazul"><img
src="images/tit_sist_control_estagio.gif" width="288"
height="28"></td>
      </tr>
      <tr>
        <td height="65"> </td>
        <td> </td>
        <td rowspan="5"><img src="images/img_federal.jpg"
width="288" height="131"></td>
      </tr>
      <tr>
        <td height="20"> </td>
```

```
        <td align="center" valign="bottom"> </td>
        <td> </td>
      </tr>
      <tr>
        <td height="19" rowspan="3"> </td>
        <td colspan="2" align="left" valign="top"
class="tcc">Seja bem vindo ao Sistema de Controle de
Est&aacute;gios
        da UFPR</td>
      </tr>
      <logic:present name="user">
      <tr>
        <td height="18" colspan="2" align="left"
valign="middle" class="tcc">Usu&aacute;rio:
        <span class="textoCinza"><bean:write name="user"
property="usuario"/></span></td>
      </tr>
      <tr>
        <td height="14" colspan="2" align="left"
valign="middle" class="tcc">Status: <span
class="textoCinza"><bean:write name="user"
property="status"/></span></td>
      </tr>
      </logic:present>
    </table>
    <table width="100%" height="1" border="0"
cellpadding="0" cellspacing="0">
      <tr>
        <td class="fundoazul"> </td>
      </tr>
    </table>
    <table width="100%" height="5" border="0"
cellpadding="0" cellspacing="0">
      <tr>
        <td class="fundoCinza">                    </td>
      </tr>
    </table>
    <tiles:insert attribute="menu"/>
    <tiles:insert attribute="body"/>
    <tiles:insert attribute="footer"/>
  </body>
</html:html>
```

6.4.124 inicial.jsp

6.4.125 listaragentes.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Agentes de Integração
  <hr align="center" class="textos"/>
</p>
<logic:equal name="user" value="admin" property="status">
<logic:present name="agentes">
  <table border="1" align="center">
    <tr>
      <th align="center">Agente</th>
    </tr>
    <logic:iterate name="agentes" id="agente">
      <tr>
        <td align="center" class="textos">
          <bean:write name="agente" property="agente"/>
        </td>
        <td>
          <html:link action="ExcluirAgente" paramId="cnpj"
            paramName="agente" paramProperty="cnpj"
styleClass="menu">
          Excluir
          </html:link>
        </td>
      </tr>
    </logic:iterate>
  </table>
</logic:present>
<logic:notPresent name="agentes">
  <p align="center" class="textos">NENHUM AGENTE DE
INTEGRAÇÃO CADASTRADO.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.126 listarconvenios.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Convenios
  <hr align="center" class="textos"/>
</p>
<logic:equal name="user" value="admin" property="status">
<logic:present name="convenios">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Codigo</th>
      <th align="center" class="textos">Empresa</th>
      <th align="center" class="textos">Inicio</th>
      <th align="center" class="textos">Validade</th>
      <th align="center" class="textos">Descricao</th>
      <th align="center" class="textos">Situação</th>
    </tr>
    <logic:iterate name="convenios" id="convenio">
      <tr>
        <td align="center" class="textos">
          <bean:write name="convenio"
property="codConvenio"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="convenio" property="empresa"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="convenio"
property="dataInicio"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="convenio" property="dataFim"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="convenio"
property="descricao"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="convenio" property="situacao"/>
        </td>
        <td>
          <html:link action="ExcluirConvenio"
paramId="codConvenio"
```

```
                paramName="convenio"
paramProperty="codConvenio" styleClass="menu">
              Excluir
              </html:link>
            </td>
          </tr>
        </logic:iterate>
      </table>
</logic:present>
<logic:notPresent name="convenios">
  <p align="center" class="textos">NENHUM CONVENIO
CADASTRADO.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center" class="textos">
  <table align="center" cellspacing="10" >
    <tr>
      <td>
        <html:link action="BuscarEmpresaSemConvenio"
styleClass="menu">Novo Convenio</html:link>
      </td>
      <td>
        <html:link action="NovaEmpresa"
styleClass="menu">Nova Empresa</html:link>
      </td>
      <td>
        <html:link action="BuscarConvenios"
styleClass="menu">Buscar Novamente</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.127 listarcursos.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Cursos da UFPR
  <hr align="center" class="textos"/>
</p>
<logic:equal name="user" value="admin" property="status">
<logic:present name="cursos">
  <table border="1" align="center">
    <tr>
      <th align="left" class="textos">Curso</th>
      <th align="left" class="textos">Duração</th>
    </tr>
    <logic:iterate name="cursos" id="curso">
      <tr>
        <td align="left" class="textos">
          <bean:write name="curso" property="nomeCurso"/>
        </td>
        <td align="left" class="textos">
          <bean:write name="curso" property="duracao"/> anos
        </td>
        <td align="left">
          <html:link action="ExcluirCurso"
paramId="codCurso"
              paramName="curso" paramProperty="codCurso"
styleClass="menu">
            Excluir
            </html:link>
          </td>
        </tr>
      </logic:iterate>
      <tr>
        <th align="left" >
          <html:link action="NovoCurso" styleClass="menu">
          Inserir Curso
          </html:link>
        </th>
      </tr>
    </table>
</logic:present>
<logic:notPresent name="cursos">
  <p align="center" class="textos">NENHUM CURSO
CADASTRADO.</p>
  <table>
    <tr>
      <td >
        <html:link action="NovoCurso" styleClass="menu">
        Inserir Curso
        </html:link>
      </td>
    </tr>
  </table>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10" >
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.128 listarempresas.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
Empresas
<hr align="center" class="textos"/>
</p>
<logic:equal name="user" value="admin" property="status">
<logic:present name="empresas">
<table border="1" align="center">
  <tr>
    <th align="center" class="textos">Razão Social</th>
    <th align="center" class="textos">Nome Fantasia</th>
    <th align="center" class="textos">Ramo</th>
    <th align="center" class="textos">CNPJ</th>
    <th align="center" class="textos">E-mail</th>
    <th align="center" class="textos">Site</th>
    <th align="center" class="textos">Convênio</th>
  </tr>
  <logic:iterate name="empresas" id="empresa">
  <tr>
    <td align="center" class="textos">
      <bean:write name="empresa" property="razaoSocial"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="nomeFantasia"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="ramo"/>
```

```
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="cnpj"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="email"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="site"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa"
property="cargoRepresentante"/>
    </td>
    <td align="center" class="textos">
      <html:link action="EditarEmpresa?action=Edit"
paramId="cnpj"
            paramName="empresa" paramProperty="cnpj">
      Editar
      </html:link>
    </td>
    <td align="center" class="textos">
      <%--html:link action="ExcluirEmpresa" paramId="cnpj"
            paramName="empresa" paramProperty="cnpj"--
%>
      Excluir
      <%--/html:link--%>
    </td>
  </tr>
  </logic:iterate>
</table>
<p class="textos" align="center">ATENÇÃO: Ao excluir uma
empresa serão também excluídos os usuarios, termos de
compromisso, vagas e convênios a ela associados.</p>
</logic:present>
<logic:notPresent name="empresas">
  <p align="center" class="textos">NENHUMA EMPRESA
CADASTRADA.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.129 listarempresassemconvenio.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
Empresas
<hr align="center" class="textos"/>
</p>
<logic:equal name="user" value="admin" property="status">
<logic:present name="empresas">
<table border="1" align="center">
  <tr>
    <th align="center">Razão Social</th>
    <th align="center">Nome Fantasia</th>
    <th align="center">Ramo</th>
    <th align="center">CNPJ</th>
    <th align="center">E-mail</th>
    <th align="center">Site</th>
    <th align="center">Convênio</th>
  </tr>
  <logic:iterate name="empresas" id="empresa">
  <tr>
    <td align="center" class="textos">
      <bean:write name="empresa" property="razaoSocial"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="nomeFantasia"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="ramo"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="cnpj"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="email"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa" property="site"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="empresa"
property="cargoRepresentante"/>
    </td>
    <td align="center">
      <html:link action="NovoConvenio" paramId="cnpj"
paramName="empresa" paramProperty="cnpj"
styleClass="menu">
      Criar Convênio
      </html:link>
    </td>
  </tr>
  </logic:iterate>
</table>
</logic:present>
<logic:notPresent name="empresas">
  <p align="center" class="textos">NENHUMA EMPRESA
ENCONTRADA.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="BuscarEmpresaSemConvenio"
styleClass="menu">Buscar Novamente</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.130 listarhabilidades.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Gerenciador de Habilidades / Requisitos
  <hr align="center" class="textos"/>
</p>
```

```
<logic:equal name="user" value="admin" property="status">
<logic:present name="habilidades">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Codigo</th>
      <th align="center" class="textos">Nome</th>
    </tr>
    <logic:iterate name="habilidades" id="habilidade">
      <tr>
        <td align="center" class="textos">
          <bean:write name="habilidade"
property="codHabilidade"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="habilidade"
property="descricaoHabilidade"/>
        </td>
        <td>
          <html:link action="ExcluirHabilidade"
paramId="codHabilidade"
            paramName="habilidade"
paramProperty="codHabilidade" styleClass="menu">
          Excluir
          </html:link>
        </td>
      </tr>
    </logic:iterate>
  </table>
  <p align="center" class="textos">ATENÇÃO: Ao excluir uma
habilidade ou requisito, será também excluído dos perfís de
usuário ou vagas que utilizam.</p>
</logic:present>
<logic:notPresent name="habilidades">
  <p align="center"  class="textos">NÃO HÁ HABILIDADES
CADASTRADAS.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.131 listarhabilidadesaluno.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Habilidades<br>
  Usuário: <bean:write name="user" property="usuario"
scope="session"/>
  <hr align="center" class="textos"/>
</p>
<logic:present name="habilidades">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Codigo</th>
      <th align="center" class="textos">Nome</th>
    </tr>
    <logic:iterate name="habilidades" id="habilidade">
      <tr>
        <td align="center" class="textos">
          <bean:write name="habilidade"
property="codHabilidade"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="habilidade"
property="descricaoHabilidade"/>
        </td>
        <td>
          <html:link action="ExcluirHabilidadeAluno"
paramId="codHabilidadeAluno"
            paramName="habilidade"
paramProperty="codHabilidadeAluno" styleClass="menu">
          Excluir
          </html:link>
        </td>
      </tr>
    </logic:iterate>
    <tr>
      <th>
        <html:link action="NovaHabilidadeAlunoGeral"
paramId="codalunoestagio" paramName="user"
paramProperty="codAlunoEstagio" paramScope="session"
styleClass="menu">
          Inserir Habilidade
          </html:link>
        </th>
      </tr>
  </table>
</logic:present>
<logic:notPresent name="habilidades">
  <p align="center" class="textos">NÃO HÁ HABILIDADES
CADASTRADAS PARA ESSE ALUNO.</p>
  <table>
    <tr>
      <th>
        <html:link action="NovaHabilidadeAlunoGeral"
paramId="codalunoestagio" paramName="user"
paramProperty="codAlunoEstagio" paramScope="session"
styleClass="menu">
          Inserir Habilidade
          </html:link>
        </th>
      </tr>
  </table>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="EditarCadastro?action=Edit"
paramId="usuario" paramName="user" paramProperty="usuario"
paramScope="session" styleClass="menu">Voltar para
Pessoal</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.132 listarhabilidadesvaga.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  Requisitos<br>
  Código da Vaga:<bean:write name="codvaga"/><br>
  Empresa:<bean:write name="empresa"/>
  <hr align="center" class="textos"/>
</p>
<logic:present name="habilidades">
```

```
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Codigo</th>
      <th align="center" class="textos">Nome</th>
    </tr>
    <logic:iterate name="habilidades" id="habilidade">
      <tr>
        <td align="center" class="textos">
          <bean:write name="habilidade"
property="codHabilidade"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="habilidade"
property="descricaoHabilidade"/>
        </td>
        <td>
          <html:link action="ExcluirHabilidadeVaga"
paramId="codHabilidadeVaga"
            paramName="habilidade"
paramProperty="codHabilidadeVaga" styleClass="menu">
            Excluir
          </html:link>
        </td>
      </tr>
    </logic:iterate>
    <tr>
      <th>
        <html:link action="NovaHabilidadeVagaGeral"
paramId="codvaga" paramName="codvaga" styleClass="menu">
        Inserir Requisito
        </html:link>
      </th>
    </tr>
  </table>
</logic:present>
<logic:notPresent name="habilidades">
  <p align="center" class="textos">NÃO HÁ REQUISITOS
CADASTRADOS PARA ESSA VAGA.</p>
  <table>
    <tr>
      <th>
        <html:link action="NovaHabilidadeVagaGeral"
paramId="codvaga" paramName="codvaga" styleClass="menu">
        Inserir Requisito
        </html:link>
      </th>
    </tr>
  </table>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10" >
    <tr>
      <td>
      <logic:equal name="user" property="status"
value="empresa">
        <html:link action="ListarVagas"
styleClass="menu">Voltar para Vagas</html:link>
      </logic:equal>
      <logic:notEqual name="user" property="status"
value="empresa">
        <html:link action="ListarVagas?estagio=interno"
styleClass="menu">Voltar para Vagas</html:link>
      </logic:notEqual>
      </td>
    </tr>
  </table>
</p>
```

6.4.133 listartermos.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
Termos de Compromisso em <bean:write name="empresa"/>
<hr align="center" class="textos"/>
</p>

<logic:present name="termos">
<table border="1" align="center">
  <tr>
    <th align="center" class="textos">Estagiário</th>
    <th align="center" class="textos">Inicio</th>
    <th align="center" class="textos">Validade</th>
    <th align="center" class="textos">Bolsa</th>
    <th align="center" class="textos">Horário</th>
    <th align="center" class="textos">Curso</th>
    <th align="center" class="textos">Setor</th>
    <th align="center" class="textos">Requisitos</th>
    <th align="center" class="textos">Agente de
Integracao</th>
  </tr>
  <logic:iterate name="termos" id="termo">
  <tr>
    <td align="center" class="textos">
      <bean:write name="termo" property="estagiario"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="dataInicio"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="dataFim"/>
    </td>

    <td align="center" class="textos">
      <bean:write name="termo" property="salario"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="horario"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="curso"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="setor"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="requisitos"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="agenteIntegracao"/>
    </td>

    <td>
      <html:link action="EditarTermo?action=Edit"
paramId="codTermo"
        paramName="termo" paramProperty="codTermo"
styleClass="menu">
        Editar
      </html:link>
    </td>
    <td>
      <html:link action="ExcluirTermo" paramId="codTermo"
        paramName="termo" paramProperty="codTermo"
styleClass="menu">
        Excluir
      </html:link>
    </td>
  </tr>
  </logic:iterate>
</table>
</logic:present>
<logic:notPresent name="termos">
```

```
  <p align="center" class="textos">NENHUM TERMO DE
COMPROMISSO CADASTRADO.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.134 listartermoscompromisso.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
Termos de Compromisso
<hr align="center" class="textos"/>
</p>
<logic:equal name="user" value="admin" property="status">
<logic:present name="termos">
<table border="1" align="center" class="textos">
  <tr>
    <th align="center" class="textos">Estagiário</th>
    <th align="center" class="textos">Empresa</th>
    <th align="center" class="textos">Inicio</th>
    <th align="center" class="textos">Validade</th>
    <th align="center" class="textos">Bolsa</th>
    <th align="center" class="textos">Horário</th>
    <th align="center" class="textos">Curso</th>
    <th align="center" class="textos">Setor</th>
    <th align="center" class="textos">Requisitos</th>
    <th align="center" class="textos">Agente de
Integracao</th>
  </tr>
  <logic:iterate name="termos" id="termo">
  <tr>
    <td align="center" class="textos">
      <bean:write name="termo" property="estagiario"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="empresa"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="dataInicio"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="dataFim"/>
    </td>

    <td align="center" class="textos">
      <bean:write name="termo" property="salario"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="horario"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="curso"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="setor"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="requisitos"/>
    </td>
    <td align="center" class="textos">
      <bean:write name="termo" property="agenteIntegracao"/>
    </td>
    <logic:equal name="termo" property="estagioInterno"
value="S">
    <td align="center" class="textos">
      <html:link action="EditarTermo?action=Edit"
paramId="codTermo"
          paramName="termo" paramProperty="codTermo"
styleClass="menu">
        Editar
      </html:link>
    </td>
    </logic:equal>
    <td>
      <html:link action="ExcluirTermoCompromisso"
paramId="codTermo"
          paramName="termo" paramProperty="codTermo"
styleClass="menu">
        Excluir
      </html:link>
    </td>
  </tr>
  </logic:iterate>
</table>
</logic:present>
<logic:notPresent name="termos">
  <p align="center" class="textos">NENHUM TERMO DE
COMPROMISSO CADASTRADO.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="BuscarTermosCompromisso"
styleClass="menu">Buscar Novamente</html:link>
      </td>
    </tr>
  </table>
</p>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.135 listarturnos.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
  Horários<br>
  Código da Vaga: <bean:write name="codvaga"/>
  <hr align="center" class="textos"/>
</p>

<logic:present name="turnos">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Codigo</th>
      <th align="center" class="textos">Nome</th>
    </tr>
    <logic:iterate name="turnosDto" id="turno">
      <tr>
        <td align="center" class="textos">
          <bean:write name="turno" property="codTurno"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="turno" property="turno"/>
        </td>
```

```
          <td>
            <html:link action="ExcluirTurno"
paramId="codTurno"
              paramName="turno" paramProperty="codTurno"
styleClass="menu">
              Excluir
            </html:link>
          </td>
        </tr>
      </logic:iterate>
      <tr>
        <th>
          <html:link action="NovoTurno" paramId="codvaga"
paramName="codvaga" styleClass="menu">
          Inserir Horário
          </html:link>
        </th>
      </tr>
    </table>
</logic:present>
<logic:notPresent name="turnos">
  <p align="center" class="textos">NÃO HÁ HORÁRIOS
CADASTRADOS PARA ESSA VAGA.</p>
  <table>
    <tr>
      <th>
        <html:link action="NovoTurno" paramId="codvaga"
paramName="codvaga" styleClass="menu">
        Inserir Horário
        </html:link>
      </th>
    </tr>
  </table>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <logic:equal name="user" property="status"
value="empresa">
          <html:link action="ListarVagas"
styleClass="menu">Voltar para Vagas</html:link>
        </logic:equal>
        <logic:notEqual name="user" property="status"
value="empresa">
          <html:link action="ListarVagas?estagio=interno"
styleClass="menu">Voltar para Vagas</html:link>
        </logic:notEqual>
      </td>
    </tr>
  </table>
</p>
```

6.4.136 listarturnosdisponiveis.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
  Horários<br>
  Usuário: <bean:write name="user" property="usuario"
scope="session"/>
  <hr align="center" class="textos"/>
</p>

<logic:present name="turnos">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Codigo</th>
      <th align="center" class="textos">Nome</th>
    </tr>
      <logic:iterate name="turnosDto" id="turno">
        <tr>
          <td align="center" class="textos">
            <bean:write name="turno" property="codTurno"/>
          </td>
          <td align="center" class="textos">
            <bean:write name="turno" property="turno"/>
          </td>
          <td>
            <html:link action="ExcluirTurnoDisponivel"
paramId="codTurno"
              paramName="turno" paramProperty="codTurno"
styleClass="menu">
              Excluir
            </html:link>
          </td>
        </tr>
      </logic:iterate>
      <tr>
        <th>
          <html:link action="NovoTurnoDisponivel"
paramId="codalunoestagio" paramName="user"
paramProperty="codAlunoEstagio" paramScope="session"
styleClass="menu">
          Inserir Horário
          </html:link>
        </th>
      </tr>
    </table>
</logic:present>
<logic:notPresent name="turnos">
  <p align="center" class="textos">NÃO HÁ HORÁRIOS
CADASTRADOS PARA ESSE USUÁRIO.</p>
  <table>
    <tr>
      <th>
        <html:link action="NovoTurnoDisponivel"
paramId="codalunoestagio" paramName="user"
paramProperty="codAlunoEstagio" paramScope="session"
styleClass="menu">
          Inserir Horário
          </html:link>
        </th>
      </tr>
    </table>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="EditarCadastro?action=Edit"
paramId="usuario" paramName="user" paramProperty="usuario"
paramScope="session" styleClass="menu">Voltar para
Pessoal</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.137 listarunidade.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
  Selecione a Unidade Concedente do Estágio Interno
  <hr align="center" class="textos"/>
</p>
<logic:present name="empresas">
  <html:form action="NovaVagaEstagioInterno">
    <table border="0" align="center">
```

```
        <tr>
          <td align="right" class="textos">

          </td>
          <td align="left" class="textos">
            Razão Social
          </td>
          <td align="left" class="textos">
            CNPJ
          </td>
        </tr>
        <logic:iterate name="empresas" id="empresa">
          <tr>
            <td align="right" class="textos">
              <html:radio idName="empresa" value="cnpj"
property="cnpj" styleClass="formulario"/>
            </td>
            <td align="left" class="textos">
              <bean:write name="empresa"
property="razaoSocial"/>
            </td>
            <td align="left" class="textos">
              <bean:write name="empresa" property="cnpj"/>
            </td>
          </tr>
        </logic:iterate>
        <tr>
          <td align="right" class="textos">

          </td>
          <td align="left" class="textos">

          </td>
          <td align="center" class="textos">
            <html:submit styleClass="formulario">
            Criar Vaga
            </html:submit>
          </td>
        </tr>
      </table>
    </html:form>
</logic:present>
<logic:notPresent name="empresas">
  <p align="center" class="textos">NENHUMA UNIDADE
ENCONTRADA.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.138 listarusuarios.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>

<p align="center" class="textos">
  Usuários:
  <hr align="center" class="textos"/>
</p>

<logic:present name="usuarios">
  <table border="1" align="center">
    <tr>
      <th align="center" class="textos">Usuário</th>
      <th align="center" class="textos">Nome</th>
      <th align="center" class="textos">E-mail</th>
    </tr>
    <logic:iterate name="usuarios" id="usuario">
      <tr>
        <td align="center">
          <html:link action="EditarCadastro?action=Edit"
paramId="usuario"
            paramName="usuario" paramProperty="usuario"
styleClass="menu">
            <bean:write name="usuario" property="usuario"/>
          </html:link>
        </td>
        <td align="center" class="textos">
          <bean:write name="usuario"
property="nomePessoa"/>
        </td>
        <td align="center" class="textos">
          <bean:write name="usuario" property="email"/>
        </td>
        <td align="center">
          <html:link action="ExcluirCadastro"
paramId="codAlunoEstagio"
            paramName="usuario"
paramProperty="codAlunoEstagio" styleClass="menu">
            Excluir
          </html:link>
        </td>
      </tr>
    </logic:iterate>
  </table>
</logic:present>
<logic:notPresent name="usuarios">
  <p align="center" class="textos">USUÁRIO NÃO
ENCONTRADO.</p>
</logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="BuscarUsuario"
styleClass="menu">
          Buscar Novamente
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.139 listarvagas.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
  <p align="center" class="textos">
    Vagas para Estágio em <bean:write name="empresa"/>
    <hr align="center" class="textos"/>
  </p>
  <logic:present name="vagas">
    <table border="1" align="center">
      <tr>
        <th align="center" class="textos">Cursos</th>
        <th align="center" class="textos">Setor</th>
        <th align="center" class="textos">Carga Horária</th>
        <th align="center" class="textos">Bolsa</th>
        <th align="center" class="textos">Sexo</th>
        <th align="center" class="textos">Desde</th>
        <th align="center" class="textos">Requisitos</th>
        <th align="center" class="textos">Horário</th>
      </tr>
      <logic:iterate name="vagas" id="vaga">
        <tr>
```

```
            <td align="center" class="textos">
              <bean:write name="vaga" property="cursos"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="setor"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="horas"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="salario"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="sexo"/>
            </td>
            <td align="center" class="textos">
              <bean:write name="vaga" property="inclusao"/>
            </td>
            <td align="center">
              <html:link action="ListarHabilidadesVaga"
paramId="codvaga"
                paramName="vaga" paramProperty="codVaga"
styleClass="menu">
                <bean:write name="vaga"
property="habilidades"/>
              </html:link>
            </td>
            <td align="center">
              <html:link action="ListarTurnos"
paramId="codvaga"
                paramName="vaga" paramProperty="codVaga"
styleClass="menu">
                <bean:write name="vaga" property="turnos"/>
              </html:link>
            </td>
            <td align="center">
              <html:link action="EditarVaga?action=Edit"
paramId="codvaga"
                paramName="vaga" paramProperty="codVaga"
styleClass="menu">
              Editar
              </html:link>
            </td>
            <td align="center">
              <html:link action="ExcluirVaga"
paramId="codvaga"
                paramName="vaga" paramProperty="codVaga"
styleClass="menu">
              Excluir
              </html:link>
            </td>
            <td align="center">
              <html:link action="CriarTermoCompromisso"
paramId="codvaga"
                paramName="vaga" paramProperty="codVaga"
styleClass="menu">
              Preencher
              </html:link>
            </td>
          </tr>
        </logic:iterate>
        <tr>
          <th>
            <html:link action="NovaVaga" paramId="cnpj"
paramName="cnpj" styleClass="menu">
            Inserir Vaga
            </html:link>
          </th>
        </tr>
      </table>
    </logic:present>
    <logic:notPresent name="vagas">
      <p align="center" class="textos">NÃO HÁ VAGAS NO
MOMENTO.</p>
      <table>
        <tr>
          <th>
            <html:link action="NovaVaga" paramId="cnpj"
paramName="cnpj" styleClass="menu">
              Inserir Vaga
              </html:link>
            </th>
          </tr>
        </table>
      </logic:notPresent>
<hr align="center" class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.140 login.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/WEB-INF/struts-tiles.tld" prefix="tiles" %>
<html:html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />
<title><bean:message key="index.heading"/></title>
<html:base/>
<link href="LayPic/estilo.css" rel="stylesheet" type="text/css" />
</head>
<body class="fundo_form">
<html:errors />
  <tiles:insert attribute="body"/>
</body>
</html:html>
```

6.4.141 mainMenu.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<logic:present name="user">
  <table height="30" border="0" cellpadding="0"
cellspacing="0">
    <tr align="center" valign="middle">
      <td width="10"> </td>
      <logic:equal name="user" value="admin"
property="status">
        <td align="center" >
          <html:link action="NovoUsuario"
styleClass="menu">Inserir Usuário</html:link>
        </td>
        <td width="15" class="menu">|</td>
        <td align="center" >
          <html:link action="BuscarConvenios"
styleClass="menu">Convenios</html:link>
        </td>
        <td width="15" class="menu">|</td>
        <td align="center" >
          <html:link action="ListarAgentesIntegracao"
styleClass="menu">Agentes de Integracao</html:link>
        </td>
        <td width="15" class="menu">|</td>
        <%--td align="center" >
```

```
            <html:link
action="ListarHabilidades">Habilidades</html:link>
            </td--%>
            <td align="center" >
              <html:link action="ListarCursos"
styleClass="menu">Cursos</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="BuscarEmpresas"
styleClass="menu">Empresas</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link
action="ConsultarCursosVagas?time=first"
styleClass="menu">Vagas</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="BuscarUsuario"
styleClass="menu">Usuários</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="BuscarTermosCompromisso"
styleClass="menu">Termos de Compromisso</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="ListarVagas?estagio=interno"
styleClass="menu">Estágio Interno</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link
action="SelecionarRelatorioSeguro?estagio=interno"
styleClass="menu">Relatório de Seguro</html:link>
            </td>
            <td width="15" class="menu">|</td>
          </logic:equal>
          <logic:equal name="user" value="empresa"
property="status">
            <td align="center" >
              <html:link action="ListarVagas"
styleClass="menu">Vagas</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="ListarTermos"
styleClass="menu">Termos de Compromisso</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="SelecionarRelatorioSeguro"
paramId="cnpj" paramName="cnpj" paramScope="session"
styleClass="menu">Relatório de Seguro</html:link>
            </td>
            <td width="15" class="menu">|</td>
          </logic:equal>
          <logic:equal name="user" value="aluno"
property="status">
            <td align="center" >
              <html:link action="BuscarVagas"
styleClass="menu">Buscar Vagas</html:link>
            </td>
            <td width="15" class="menu">|</td>
            <td align="center" >
              <html:link action="EditarCadastro?action=Edit"
paramId="usuario" paramName="user" paramProperty="usuario"
paramScope="session" styleClass="menu">Pessoal</html:link>
            </td>
            <td width="15" class="menu">|</td>
          </logic:equal>
          <td align="center" >
            <html:link forward="logoff"
styleClass="menu"><strong>Sair</strong></html:link>
          </td>
        </tr>
      </table>
      <table width="100%" height="4" border="0" cellpadding="5"
cellspacing="0">
        <tr>
          <td class="fundoazul"> </td>
        </tr>
      </table>
    </logic:present>
    <html:errors/>
```

6.4.142 novaempresa.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  <logic:equal name="NovaEmpresaForm" property="action"
value="Create">
  Inserir Empresa
  </logic:equal>
  <logic:equal name="NovaEmpresaForm" property="action"
value="Editar">
  Editar Empresa
  </logic:equal>
  <hr align="left" class="textos"/>
</p>
<html:form action="/SalvarEmpresa" focus="cnpj">
  <html:hidden property="action"/>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        CNPJ:
      </th>
      <td align="left">
        <logic:equal name="NovaEmpresaForm"
property="action" value="Create">
          <html:text property="cnpj"
styleClass="formulario"/>
        </logic:equal>
        <logic:equal name="NovaEmpresaForm"
property="action" value="Edit">
          <html:hidden property="cnpj" write="true"
styleClass="formulario"/>
        </logic:equal>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Número do Processo:
      </th>
      <td align="left">
        <html:text property="numProcesso"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Nome Fantasia:
      </th>
      <td align="left">
        <html:text property="nomeFantasia"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
```

```
        Razão Social:
      </th>
      <td align="left">
        <html:text property="razaoSocial"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Representante:
      </th>
      <td align="left" class="textos">
        <html:text property="representante"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left"  class="textos">
        Ramo:
      </th>
      <td align="left" class="textos">
        <html:text property="ramo" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left"  class="textos">
        Telefone:
      </th>
      <td align="left" class="textos">
        <html:text property="fone" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Fax:
      </th>
      <td align="left" class="textos">
        <html:text property="fax" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        E-mail:
      </th>
      <td align="left" class="textos">
        <html:text property="email" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Site:
      </th>
      <td align="left" class="textos">
        <html:text property="site" styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">Rua:</th>
      <td align="left">
        <html:text property="rua" size="42,5"
styleClass="formulario"/>
      </td>
      <th align="left" class="textos">N&ordm;:</th>
      <th align="left">
      <html:text property="numero" size="8"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Bairro:
      </th>
      <td align="left">
        <html:text property="bairro" styleClass="formulario"/>
      </td>
      <th align="left" class="textos">Complemento:</th>
      <td align="left">
        <html:text property="complemento"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">CEP:</th>
      <td align="left">
        <html:text property="cep" styleClass="formulario"/>
      </td>
      <th align="left" class="textos">Cidade:</th>
      <td align="left">
        <html:text property="municipio"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">Estado:</th>
      <td align="left">
        <html:select property="estado"
styleClass="formulario">
          <html:options name="estados" />
        </html:select>
      </td>
      <th align="left" class="textos">Pa&iacute;s:</th>
      <td align="left">
        <html:text property="pais" styleClass="formulario"/>
      </td>
    </tr>
    <logic:equal name="NovaEmpresaForm" property="action"
value="Create">
      <tr>
        <th align="left" class="textos">
          Convenio:
        </th>
      </tr>
      <tr>
        <th align="left" class="textos">
          Inicio*:
        </th>
        <td align="left" class="textos">
          <html:text property="dataInicio"
styleClass="formulario" onkeypress="formatar(this,
'##/##/####')"/> <em>dd/mm/aaaa</em>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Final*:
        </th>
        <td align="left" class="textos">
          <html:text property="dataFim"
styleClass="formulario" onkeypress="formatar(this,
'##/##/####')"/> <em>dd/mm/aaaa</em>
        </td>
      </tr>
    </logic:equal>
    <tr>
      <th align="right" class="textos">
        <html:multibox property="agenteIntegrador"
styleClass="formulario" value="S"/>
      </th>
      <td align="left" class="textos">
        Agente Integrador
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
```

```
    <div align="center">
      <html:submit styleClass="formulario">
        <bean:message key="button.save"/>
      </html:submit>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <logic:equal name="NovaEmpresaForm"
property="action" value="Edit">
          <html:link action="BuscarEmpresas"
styleClass="menu">
          Buscar Novamente
          </html:link>
        </logic:equal>
        <logic:equal name="NovaEmpresaForm"
property="action" value="Create">
          <html:link action="BuscarConvenios"
styleClass="menu">
          Voltar para Convênios
          </html:link>
        </logic:equal>
      </td>
    </tr>
  </table>
</p>
```

6.4.143 novahabilidadealuno.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarHabilidadeAluno">
  <p align="center" class="textos">
    Inserir Habilidade
    <hr align="left" class="textos"/>
  </p>
    <table align="center">
    <tr>
      <th align="left" class="textos">
        Codigo do Aluno:
      </th>
      <td align="left" class="textos">
        <html:hidden property="codAlunoEstagio"
write="true"/>
      </td>
    </tr>
    <logic:present name="todashabilidades">
      <tr>
        <th align="left" class="textos">
          Habilidades:
        </th>
      </tr>
      <logic:iterate id="habilidade" name="todashabilidades">
        <tr>
          <td align="right" class="textos">
            <html:multibox property="codHabilidade"
styleClass="formulario">
              <bean:write name="habilidade"
property="codHabilidade" />
            </html:multibox>
          </td>
          <td align="left" class="textos">
            <bean:write name="habilidade"
property="descricaoHabilidade"/>
          </td>
        </tr>
      </logic:iterate>
    </logic:present>
    <logic:notPresent name="todashabilidades">
      <tr>
        <th align="left" class="textos">
          Habilidades:
        </th>
        <td align="left" class="textos">
          Não há habilidades para serem incluídas.
        </td>
      </tr>
    </logic:notPresent>
    <tr>
      <th align="left">
        <html:link action="NovaHabilidadeAlunoGeral"
paramName="codalunoestagio" paramId="codalunoestagio"
          styleClass="menu">
        Nova Habilidade
        </html:link>
      </th>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset styleClass="formulario">
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="ListarHabilidadesAluno"
paramId="codalunoestagio"
          paramName="codalunoestagio" styleClass="menu">
        Voltar para Habilidades
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.144 novahabilidadealunogeral.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarHabilidadeAlunoGeral">
  <p align="center" class="textos">
    <bean:write name="user" property="usuario"
scope="session"/><br>
    Nova Habilidade
    <hr align="left" class="textos"/>
  </p>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        Codigo do Aluno:
      </th>
      <td align="left" class="textos">
        <html:hidden property="codAlunoEstagio"
write="true"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Habilidade:
      </th>
      <td align="left" class="textos">
```

```
          <html:text property="descricaoHabilidade"
styleClass="formulario"/>
        </td>
        <td align="left">
          <html:link action="BuscarHabilidadeAluno"
paramId="codAlunoEstagio" paramName="codalunoestagio"
styleClass="menu">
          Procurar Habilidades
          </html:link>
        </td>
      </tr>
      <tr>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
          <bean:message key="button.save"/>
        </html:submit>
        <html:reset styleClass="formulario">
          <bean:message key="button.reset"/>
        </html:reset>
      </div>
    </div>
</html:form>
<hr class="textos"/>
<p align="center">
    <table align="center" cellspacing="10">
      <tr>
        <td>
          <html:link action="ListarHabilidadesAluno"
paramId="codalunoestagio"
            paramName="codalunoestagio" styleClass="menu">
          Voltar para Habilidades
          </html:link>
        </td>
      </tr>
    </table>
</p>
```

6.4.145 novahabilidadevaga.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarHabilidadeVaga">
    <p align="center" class="textos">
      <bean:write name="empresa" /><br>
      Inserir Requisito
      <hr align="left" class="textos"/>
    </p>
      <table align="center">
      <tr>
        <th align="left">
          Codigo da Vaga:
        </th>
        <td align="left" class="textos">
          <html:hidden property="codVaga" write="true"/>
        </td>
      </tr>
      <logic:present name="todashabilidades">
        <tr>
        <th align="left" class="textos">
          Requisito:
        </th>
        <logic:iterate id="habilidade" name="todashabilidades">
          <tr>
            <td align="right" class="textos">
              <html:multibox property="codHabilidade"
styleClass="formulario">
                <bean:write name="habilidade"
property="codHabilidade" />
              </html:multibox>
            </td>
            <td align="left" class="textos">
              <bean:write name="habilidade"
property="descricaoHabilidade"/>
            </td>
          </tr>
        </logic:iterate>
      </logic:present>
      <logic:notPresent name="todashabilidades">
        <tr>
          <th align="left" class="textos">
            Requisito:
          </th>
          <td align="left" class="textos">
            Não há requisitos para serem incluídos na vaga.
          </td>
        </tr>
      </logic:notPresent>
      <tr>
        <th align="left">
          <html:link action="NovaHabilidadeVagaGeral"
paramName="codvaga" paramId="codvaga" styleClass="menu">
          Novo Requisito
          </html:link>
        </th>
      </tr>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
          <bean:message key="button.save"/>
        </html:submit>
      </div>
    </div>
</html:form>

<hr class="textos"/>
<p align="center">
    <table align="center" cellspacing="10">
      <tr>
        <td>
          <html:link action="ListarHabilidadesVaga"
paramId="codvaga" paramName="codvaga" styleClass="menu">
          Listar Requisitos
          </html:link>
        </td>
      </tr>
    </table>
</p>
```

6.4.146 novahabilidadevagageral.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarHabilidadeVagaGeral">
    <p align="center" class="textos">
      <bean:write name="empresa"/><br>
      Novo Requisito
      <hr align="left" class="textos"/>
    </p>
    <table align="center">
      <tr>
        <th align="left" class="textos">
          Codigo da Vaga:
        </th>
        <td align="left" class="textos">
          <html:hidden property="codVaga" write="true"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Requisito:
```

```
        </th>
        <td align="left" class="textos">
          <html:text property="descricaoHabilidade"
styleClass="formulario"/>
        </td>
        </td>
        <td align="left">
          <html:link action="BuscarRequisitos"
paramId="codvaga" paramName="codvaga" styleClass="menu">
          Procurar Requisito
          </html:link>
        </td>
      </tr>
      <tr>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
        Enviar
        </html:submit>
      </div>
    </div>
</html:form>
<hr class="textos"/>
<p align="center">
    <table align="center" cellspacing="10">
      <tr>
        <td>
          <logic:equal name="user" property="status"
value="empresa">
            <html:link action="ListarVagas"
styleClass="menu">Voltar para Vagas</html:link>
          </logic:equal>
          <logic:notEqual name="user" property="status"
value="empresa">
            <html:link action="ListarVagas?estagio=interno"
styleClass="menu">Voltar para Vagas</html:link>
          </logic:notEqual>
        </td>
      </tr>
    </table>
</p>
```

6.4.147 novavaga.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarVaga">
    <p align="center" class="textos">
      <bean:write name="empresa" /><br>
      <logic:equal name="NovaVagaForm" property="action"
value="Create">
      Inserir Vaga
      </logic:equal>
      <logic:equal name="NovaVagaForm" property="action"
value="Editar">
      Editar Vaga
      </logic:equal>
      <hr align="left" class="textos"/>
    </p>
    <html:hidden property="action"/>
    <html:hidden property="cnpj"/>
    <html:hidden property="estagio"/>
    <table align="center">
      <logic:equal name="NovaVagaForm" property="action"
value="Editar">
        <tr>
          <th align="left" class="textos">
            Codigo da Vaga:
          </th>
          <td align="left" class="textos">
            <html:hidden property="codVaga" write="true"
styleClass="formulario"/>
          </td>
        </tr>
      </logic:equal>
      <logic:notEqual name="NovaVagaForm" property="action"
value="Editar">
        <html:hidden property="codVaga"
styleClass="formulario"/>
      </logic:notEqual>
      <tr>
        <th align="left" class="textos">
          Setor:
        </th>
        <td align="left" class="textos">
          <html:text property="setor" styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Cursos:
        </th>
        <td align="left" class="textos">
          <%!int divisor = 3;%>
          <%!int indice = 0;%>
          <table>
            <logic:present name="cursos">
              <tr>
              <logic:iterate name="cursos" id="curso"
indexId="index">
                <td class="textos">
                  <html:multibox property="codCurso"
styleClass="formulario">
                    <bean:write name="curso"
property="codCurso"/>
                  </html:multibox>
                  <bean:write name="curso"
property="nomeCurso"/>
                </td>
                <%Integer indice1 = (Integer)
pageContext.getAttribute("index",PageContext.PAGE_SCOPE);
                indice = indice1.intValue();
                if((indice + 1)%divisor == 0){
                %></tr><tr><%
                }%>
              </logic:iterate>
              </tr>
            </logic:present>
          </table>
          <logic:notPresent name="cursos">
          Não informado.
          </logic:notPresent>

        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Horas/Semana:
        </th>
        <td align="left" class="textos">
          <html:text property="horas" styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Semestre Mínimo:
        </th>
        <td align="left" class="textos">
          <html:text property="semestre"
styleClass="formulario"/>
        </td>
      </tr>
```

```
    <tr>
      <th align="left" class="textos">
        Valor da Bolsa:
      </th>
      <td align="left" class="textos">
        <html:text property="salario"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Vale Transporte:
      </th>
      <td align="left" class="textos">
        <html:multibox property="valeTransporte"
styleClass="formulario" value="S"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Sexo:
      </th>
      <td align="left" class="textos">
        <html:select property="sexo" styleClass="formulario">
          <html:options collection="sexos" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Locomoção:
      </th>
      <td align="left" class="textos">
        <html:select property="locomocao"
styleClass="formulario">
          <html:options collection="locomocao"
property="value" labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <td class="textos">
        Disponível para fumante ?
      </td>
      <th align="left" class="textos">
        <html:select property="fumante"
styleClass="formulario">
          <html:options collection="simNao" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        Vaga para Deficiente ?
      </td>
      <th align="left" class="textos">
        <html:select property="deficiente"
styleClass="formulario">
          <html:options collection="simNao" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        Atividades nos Finais de Semana ?
      </td>
      <th align="left" class="textos">
        <html:select property="atividadesFds"
styleClass="formulario">
          <html:options collection="simNao" property="value"
labelProperty="label"/>
        </html:select>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Outros benefícios:
      </th>
      <td align="left" class="textos">
        <html:textarea property="outrosBenefícios"
styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset styleClass="formulario">
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <logic:equal name="user" property="status"
value="empresa">
          <html:link action="ListarVagas"
styleClass="menu">Voltar para Vagas</html:link>
        </logic:equal>
        <logic:notEqual name="user" property="status"
value="empresa">
          <html:link action="ListarVagas?estagio=interno"
styleClass="menu">Voltar para Vagas</html:link>
        </logic:notEqual>
      </td>
    </tr>
  </table>
</p>
```

6.4.148 novoConvenio.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  <logic:equal name="NovoConvenioForm" property="action"
value="Create">
  Inserir Convenio
  </logic:equal>
  <logic:equal name="NovoConvenioForm" property="action"
value="Editar">
  Editar Convenio
  </logic:equal>
  <hr align="left" class="textos"/>
</p>
<logic:present name="empresas">
  <html:form action="SalvarConvenio" focus="codConvenio">
    <html:hidden property="action" styleClass="formulario"/>
    <table align="center">
      <logic:equal name="NovoConvenioForm"
property="action" value="Editar">
        <tr>
          <th align="left" class="textos">
            Codigo do Convenio:
          </th>
          <td align="left" class="textos">
```

```
                <html:hidden property="codConvenio"
write="true" styleClass="formulario"/>
              </td>
            </tr>
          </logic:equal>
          <logic:notEqual name="NovoConvenioForm"
property="action" value="Editar">
            <html:hidden property="codConvenio"
styleClass="formulario"/>
          </logic:notEqual>
          <tr>
            <th align="left" class="textos">
              Inicio*:
            </th>
            <td align="left" class="textos">
              <html:text property="dataInicio"
styleClass="formulario"/> <em>dd/mm/aaaa</em>
            </td>
          </tr>
          <tr>
            <th align="left" class="textos">
              Final*:
            </th>
            <td align="left" class="textos">
              <html:text property="dataFim"
styleClass="formulario"/> <em>dd/mm/aaaa</em>
            </td>
          </tr>
          <tr>
            <th align="left" class="textos">
              Descrição:
            </th>
            <td align="left" class="textos">
              <html:textarea property="descricao"
styleClass="formulario"/>
            </td>
          </tr>
          <tr>
            <th align="left" class="textos">
              Empresa:
            </th>
            <td align="left" class="textos">
              <html:select property="cnpj"
styleClass="formulario">
                <html:options collection="empresas"
property="value" labelProperty="label"/>
              </html:select>
            </td>
          </tr>
        </table>
        <div class="button" id="botao">
          <div align="center">
            <html:submit styleClass="formulario">
              <bean:message key="button.save"/>
            </html:submit>
            <html:reset styleClass="formulario">
              <bean:message key="button.reset"/>
            </html:reset>
          </div>
        </div>
      </html:form>
    </logic:present>
    <logic:notPresent name="empresas">
      <p align="center" class="textos">TODAS AS EMPRESAS
CADASTRADAS JÁ POSSUEM CONVÊNIOS VÁLIDOS.
PARA CADASTRAR UM NOVO CONVÊNIO VOCÊ PODE:
        <ol class="textos">
          <li>CADASTRAR A NOVA EMPRESA ANTES DE
CRIAR O NOVO CONVÊNIO</li>
          <li>AGUARDAR EXPIRAR A VALIDADE DO
CONVÊNIO DE UMA EMPRESA JÁ CADASTRADA E
ENTÃO RENOVÁ-LO.</li>
        </ol>
      </p>
    </logic:notPresent>
    <hr class="textos"/>
    <p align="center">
      <table align="center" cellspacing="10">
        <tr>
          <td>
            <html:link action="BuscarConvenios"
styleClass="menu">
              Voltar
            </html:link>
          </td>
        </tr>
      </table>
    </p>
```

6.4.149 novocurso.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarCurso">
  <logic:equal name="user" property="status" value="admin">
    <p align="center" class="textos">
      Inserir Curso
      <hr align="left" class="textos"/>
    </p>
    <table align="center">
      <tr>
        <th align="left" class="textos">
          Nome:
        </th>
        <td align="left" class="textos">
          <html:text property="nomeCurso"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Duração:
        </th>
        <td align="left" class="textos">
          <html:text property="duracao"
styleClass="formulario"/>
        </td>
      </tr>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
          <bean:message key="button.save"/>
        </html:submit>
        <html:reset styleClass="formulario">
          <bean:message key="button.reset"/>
        </html:reset>
      </div>
    </div>
  </logic:equal>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10"
class="fundo_azul_escuro">
    <tr>
      <td>
        <html:link action="ListarCursos" styleClass="menu">
        Cursos
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.150 novotermocompromisso.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="/SalvarTermoCompromisso">
  <html:hidden property="action"/>
  <html:hidden property="codTermo"/>
  <p align="center" class="textos">
    Novo Termo de Compromisso
    <hr align="left" class="textos"/>
  </p>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        Codigo do Aluno:
      </th>
      <td align="left" class="textos">
        <html:hidden property="codAlunoEstagio" write="true"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Codigo da Vaga:
      </th>
      <td align="left" class="textos">
        <html:hidden property="codVaga" write="true"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Cnpj:
      </th>
      <td align="left" class="textos">
        <html:hidden property="cnpj" write="true"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Data de Inicio*:
      </th>
      <td align="left" class="textos">
        <html:text property="dataInicio"
onkeypress="formatar(this, '##/##/####')"
styleClass="formulario"/> <em>dd/mm/aaaa</em>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Data de Término*:
      </th>
      <td align="left" class="textos">
        <html:text property="dataFim"
onkeypress="formatar(this, '##/##/####')"
styleClass="formulario"/> <em>dd/mm/aaaa</em>
      </td>
    </tr>
    <tr>
      <logic:present name="agente">
        <th align="left" class="textos">
          Agente de Integração:
        </th>
        <td align="left" class="textos">
          <html:select property="canal"
styleClass="formulario">
            <html:options collection="selectAgentes"
property="value" labelProperty="label"/>
          </html:select>
        </td>
        <logic:equal name="NovoTermoCompromissoForm"
property="action" value="Create">
          <tr>
            <td align="left" class="textos">
              <html:link
action="NovoTermoCompromisso?agente=false"
name="parametros" styleClass="menu">
                Termo de Compromisso Sem Agente Integrador
                </html:link>
              </td>
            </tr>
          </logic:equal>
        </logic:present>
        <logic:notPresent name="agente">
          <logic:equal name="NovoTermoCompromissoForm"
property="action" value="Create">
          <td align="left" class="textos">
            <html:hidden property="canal" />
            <html:link
action="NovoTermoCompromisso?agente=true"
name="parametros" styleClass="menu">
              Adicionar Agente Integrador
              </html:link>
            </td>
            </logic:equal>
          </logic:notPresent>
        </tr>
        <tr>
          <th align="left" class="textos">
            Companhia Seguradora:
          </th>
          <td align="left" class="textos">
            <html:text property="seguroCompanhia"
styleClass="formulario" size="50"/>
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Número da Apólice:
          </th>
          <td align="left" class="textos">
            <html:text property="seguroNumApolice"
styleClass="formulario"/>
          </td>
        </tr>
        <tr>
          <td align="center" class="textos">
            PREENCHER OS CAMPOS ABAIXO SOMENTE
PARA O ESTÁGIO NÃO OBRIGATÓRIO.
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Nome do orientador de estágio na unidade concedente:
          </th>
          <td align="left" class="textos">
            <html:text property="orientador"
styleClass="formulario"/>
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Formação profissional do orientador:
          </th>
          <td align="left" class="textos">
            <html:text property="formacaoOrientador"
styleClass="formulario"/>
          </td>
        </tr>
        <logic:present name="estagio">
          <html:hidden property="estagio"/>
          <tr>
```

```
        <th align="left" class="textos">
          Cargo ou função do orientador:
        </th>
        <td align="left" class="textos">
          <html:text property="cargoOrientador"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Unidade/Departamento:
        </th>
        <td align="left" class="textos">
          <html:text property="departamento"
styleClass="formulario"/>
        </td>
      </tr>
      <tr>
        <th align="left" class="textos">
          Ramal:
        </th>
        <td align="left" class="textos">
          <html:text property="ramal"
styleClass="formulario"/>
        </td>
      </tr>
    </logic:present>
    <tr>
      <th align="left" class="textos">
        Área de atividade do(a) estagiário(a):
      </th>
      <td align="left" class="textos">
        <html:text property="areaAtividadeEstagiario"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Objetivos:
      </th>
      <td align="left" class="textos">
        <html:text property="objetivos"
styleClass="formulario"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Atividades a serem desenvolvidas:
      </th>
      <td align="left" class="textos">
        <html:textarea rows="10" cols="50"
property="atividades" styleClass="formulario"/>
      </td>
    </tr>
    <%--tr>
    <th align="left" class="textos">
    Seguro contra acidentes pessoais providenciado pela:
    </th>
    <td align="left" class="textos">
    <html:text property="seguroProvidenciado"
styleClass="formulario"/>
    </td>
    </tr--%>
    <tr>
      <th align="right" class="textos">
        <html:multibox property="obrigatorio" value="true"
styleClass="formulario"/>
      </th>
      <td align="left" class="textos">
        Estágio Obrigatório
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset styleClass="formulario">
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
<hr class="textos"/>
  <p align="center">
    <table align="center" cellspacing="10">
      <tr>
        <td>
          <logic:equal name="NovoTermoCompromissoForm"
property="action" value="Create">
            <html:link action="CriarTermoCompromisso"
paramId="codvaga"
              paramName="codvaga" styleClass="menu">
            Voltar para Buscar Estagiário
            </html:link>
          </logic:equal>
          <logic:equal name="NovoTermoCompromissoForm"
property="action" value="Edit">
            <html:link action="ListarTermos"
styleClass="menu">
            Voltar para Termos de Compromisso
            </html:link>
          </logic:equal>
        </td>
      </tr>
    </table>
  </p>
```

6.4.151 novoturno.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarTurno">
  <p align="center" class="textos">
    Inserir Horário:
    <hr align="left" class="textos"/>
  </p>
    <table align="center">
    <tr>
      <th align="left" class="textos">
        Codigo da Vaga:
      </th>
      <td align="left" class="textos">
        <html:hidden property="codVaga" write="true"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Das *:
      </th>
      <td align="left" class="textos">
        <html:text property="horaInicio"
styleClass="formulario"/> <em class="textos">00:00</em>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        às *:
      </th>
      <td align="left" class="textos">
        <html:text property="horaFim"
styleClass="formulario"/> <em class="textos">00:00</em>
      </td>
```

```
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset styleClass="formulario">
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="ListarTurnos" paramId="codvaga"
          paramName="codvaga" styleClass="menu">
        Listar Horários
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.152 novoturnodisponivel.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<html:form action="SalvarTurnoDisponivel">
  <p align="center" class="textos">
    Inserir Horário Disponível:
    <hr align="left" class="textos"/>
  </p>
    <table align="center">
    <tr>
      <th align="left" class="textos">
        Codigo do Aluno:
      </th>
      <td align="left" class="textos">
        <html:hidden property="codAlunoEstagio"
write="true"/>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        Das *:
      </th>
      <td align="left" class="textos">
        <html:text property="horaInicio"
styleClass="formulario"/> <em class="textos">00:00</em>
      </td>
    </tr>
    <tr>
      <th align="left" class="textos">
        às *:
      </th>
      <td align="left" class="textos">
        <html:text property="horaFim"
styleClass="formulario"/> <em class="textos">00:00</em>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        <bean:message key="button.save"/>
      </html:submit>
      <html:reset styleClass="formulario">
        <bean:message key="button.reset"/>
      </html:reset>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="ListarTurnosDisponiveis"
paramId="codalunoestagio"
          paramName="codalunoestagio" styleClass="menu">
        Listar Horários Disponíveis
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.153 pesquisarconvenios.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Pesquisar Convenios
  <hr align="left" class="textos"/>
</p>
  <html:form action="PesquisarConvenios">
    <logic:present name="mensagem">
      <p align="center" class="textos"><bean:write
name="mensagem"/></p>
    </logic:present>
    <p align="center">Parâmetros para a busca:</p>
    <table align="center">
      <tr>
        <td align="left" class="textos">
          Parâmetro:
        </td>
      </tr>
      <tr>
        <td align="right" class="textos">
          <html:radio property="parametro"
value="razaoSocial" styleClass="formulario"/>
        </td>
        <td align="left" class="textos">
          Razão Social
        </td>
      </tr>
      <tr>
        <td align="right" class="textos">
          <html:radio property="parametro" value="cnpj"
styleClass="formulario"/>
        </td>
        <td align="left" class="textos">
          CNPJ
        </td>
      </tr>
      <tr>
        <td align="left" class="textos">
          Valor:
        </td>
        <td align="left" class="textos">
          <html:text property="valor"
styleClass="formulario"/>
        </td>
      </tr>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
        Pesquisar
        </html:submit>
```

```
      </div>
    </div>
  </html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10"
class="fundo_azul_escuro">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
        Página Inicial
        </html:link>
      </td>
      <td>
        <html:link action="BuscarEmpresaSemConvenio"
styleClass="menu">Novo Convenio</html:link>
      </td>
      <td>
        <html:link action="NovaEmpresa"
styleClass="menu">Nova Empresa</html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.154 pesquisarempresas.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Cadastro de Empresas
  <hr align="left" class="textos"/>
</p>
  <html:form action="PesquisarEmpresas">
  <logic:present name="mensagem">
    <p align="center" class="textos"><bean:write
name="mensagem"/></p>
  </logic:present>
  <table align="center">
    <tr>
      <th align="left" class="textos">
        Razão Social:
      </th>
      <td align="left" class="textos">
        <html:text property="parteDoNome"
styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        Pesquisar
      </html:submit>
    </div>
  </div>
</html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
        Página Inicial
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.155 pesquisarempresassemconvenio.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Novo Convenio: Selecione a Empresa.
  <hr align="left" class="textos"/>
</p>
  <html:form action="PesquisarEmpresasSemConvenio">
    <logic:present name="mensagem">
    <ul>
      <li>
      <p align="left" class="textos"><bean:write
name="mensagem"/></p>
      </li>
    </ul>
    </logic:present>
    <p align="center">Parâmetros para a busca:</p>
    <table align="center">
      <tr>
        <td align="left" class="textos">
          Parâmetro:
        </td>
      </tr>
      <tr>
        <td align="right" class="textos">
          <html:radio property="parametro"
value="razaoSocial" styleClass="formulario"/>
        </td>
        <td align="left" class="textos">
          Razão Social
        </td>
      </tr>
      <tr>
        <td align="right" class="textos">
          <html:radio property="parametro" value="cnpj"
styleClass="formulario"/>
        </td>
        <td align="left" class="textos">
          CNPJ
        </td>
      </tr>
      <tr>
        <td align="left" class="textos">
          Valor:
        </td>
        <td align="left" class="textos">
          <html:text property="valor"
styleClass="formulario"/>
        </td>
      </tr>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
        Pesquisar
        </html:submit>
      </div>
    </div>
  </html:form>
<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
        Página Inicial
        </html:link>
      </td>
      <td>
        <html:link action="NovaEmpresa"
styleClass="menu">Nova Empresa</html:link>
```

```
      </td>
    </tr>
  </table>
</p>
```

6.4.156 pesquisarhabilidadealuno.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Busca por Habilidades
  <hr align="left" class="textos"/>
</p>
  <html:form action="SalvarHabilidadeAluno">
    <html:hidden property="codAlunoEstagio"/>
    <table align="center">
    <logic:present name="todashabilidades">
    <p align="center" class="textos">Quais são as habilidades
necessárias para encontrar uma vaga ?<br>
    Atualmente as vagas de estágio abertas para <bean:write
name="curso"/> possuem os seguintes requisitos:</p>
          <tr>
            <th align="left" class="textos">
              Habilidades:
            </th>
          </tr>
          <tr>
          <%! int multiplicando = 8;%>
          <%! int multiplicador = 1;%>
          <%! int multiploTres = 0;%>
          <%
          java.util.ArrayList habilidades = new
java.util.ArrayList();
          habilidades = (java.util.ArrayList)
request.getAttribute("todashabilidades");
          for (int i=0;i<habilidades.size();++i){
              estagio.bean.Habilidade habilidade =
(estagio.bean.Habilidade) habilidades.get(i);
          %>
          <%--logic:iterate id="habilidade"
name="todashabilidades" indexId="indice"--%>
            <%multiploTres = multiplicando*multiplicador;%>
            <td align="right" class="textos">
              <%--html:multibox property="codHabilidade">
                <bean:write name="habilidade"
property="codHabilidade" />
              </html:multibox--%>
              <input type="checkbox" name="codHabilidade"
value="<%=habilidade.getCodHabilidade()%>"
class="formulario">
            </td>
            <td align="left" class="textos">
              <%--bean:write name="habilidade"
property="descricaoHabilidade"/--%>
              <%=habilidade.getDescricaoHabilidade()%>
            </td>
                        <%if(i == multiploTres){
              ++multiplicador;%>
              </tr>
              <tr>
            <%}%>
          <%--/logic:iterate--%>
          <%}%>
          </tr>
        </logic:present>
        <logic:notPresent name="todashabilidades">
          <tr>
            <td align="left" class="textos">
              Não foram encontradas habilidades.<br>Talvez
todas já estejam incluídas em seu perfil.
            </td>
          </tr>
        </logic:notPresent>
      </table>
      <logic:present name="todashabilidades">
        <div class="button" id="botao">
          <div align="center">
            <html:submit styleClass="formulario">
              Incluir no perfil de <bean:write name="user"
property="usuario"/>
            </html:submit>
          </div>
        </div>
      </logic:present>
    </html:form>

<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="ListarHabilidadesAluno"
paramId="codalunoestagio"
          paramName="codAlunoEstagio"
styleClass="menu">
          Voltar para Habilidades
          </html:link>
        </td>
      </tr>
    </table>
</p>
```

6.4.157 pesquisartermoscompromisso.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<p align="center" class="textos">
  Pesquisar Termos de Compromisso
  <hr align="left" class="textos"/>
</p>
  <html:form action="PesquisarTermosCompromisso">
  <logic:present name="mensagem">
    <p align="center"><bean:write name="mensagem"/></p>
  </logic:present>
  <p align="center" class="textos">Parâmetros para a busca:</p>
  <table align="center" class="textos">
    <tr>
      <td align="left" class="textos">
        Parâmetro:
      </td>
    </tr>
    <tr>
      <td align="right">
        <html:radio property="parametro" value="nome"
styleClass="formulario"/>
      </td>
      <td align="left" class="textos">
        Nome do aluno
      </td>
    </tr>
    <tr>
      <td align="right">
        <html:radio property="parametro" value="cnpj"
styleClass="formulario"/>
      </td>
      <td align="left" class="textos">
        CNPJ
      </td>
    </tr>
    <tr>
      <td align="left" class="textos">
        Valor:
      </td>
      <td align="left">
```

```
        <html:text property="valor" styleClass="formulario"/>
      </td>
    </tr>
  </table>
  <div class="button" id="botao">
    <div align="center">
      <html:submit styleClass="formulario">
        Pesquisar
      </html:submit>
    </div>
  </div>
</html:form>

<hr class="textos"/>
<p align="center">
  <table align="center" cellspacing="10">
    <tr>
      <td>
        <html:link action="Inicial" styleClass="menu">
        Página Inicial
        </html:link>
      </td>
    </tr>
  </table>
</p>
```

6.4.158 selecionarestagiario.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<logic:present name="alunos">
  <html:form action="/NovoTermoCompromisso">
    <p align="center" class="textos">
      Selecione o Estagiário:
      <hr align="center" class="textos"/>
    </p>
    <table border="3" align="center">
      <tr>
        <th align="right" class="textos">
        </th>
        <td align="left" class="textos">
          Nome de Usuário
        </td>
        <td align="left" class="textos">
          Nome
        </td>
        <td align="left" class="textos">
          E-mail
        </td>
      </tr>
      <logic:iterate id="aluno" name="alunos">
        <tr>
          <th align="right" class="textos">
            <html:radio property="codAlunoEstagio"
idName="aluno" value="codAlunoEstagio"
styleClass="formulario"/>
          </th>
          <td align="left" class="textos">
            <bean:write name="aluno" property="usuario"/>
          </td>
          <td align="left" class="textos">
            <bean:write name="aluno"
property="nomePessoa"/>
          </td>
          <td align="left" class="textos">
            <bean:write name="aluno" property="email"/>
          </td>
        </tr>
      </logic:iterate>
    </table>
    <html:hidden property="codVaga"/>
    <div class="button" id="botao">
      <div align="center">
        <html:submit styleClass="formulario">
          <bean:message key="button.save"/>
        </html:submit>
        <html:reset styleClass="formulario">
          <bean:message key="button.reset"/>
        </html:reset>
      </div>
    </div>
  </html:form>
</logic:present>
<logic:notPresent name="alunos">
  <p align="center" class="textos"> Aluno não encontrado !</p>
  <table cellspacing="10" align="center">
    <tr>
      <td>
        <html:link action="CriarTermoCompromisso"
paramId="codvaga"
          paramName="codVaga" styleClass="menu">
        Buscar Novamente
        </html:link>
      </td>
      <td>
        <html:link action="Inicial" styleClass="menu">Pagina
Inicial</html:link>
      </td>
    </tr>
  </table>
</logic:notPresent>
```

6.4.159 selecionarrelatorio.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
  <html:form action="RelatorioSeguro">
    <html:hidden property="cnpj"/>
    <html:hidden property="estagio"/>
    <p align="center" class="textos">
      Relatório mensal de estagiários para benefício de
Seguro<br>
    </p>
    <hr align="center" class="textos"/>
    <p align="center" class="textos">
      Selecione o Relatório Desejado:
    </p>
    <table align="center">
      <tr>
        <th align="right">
          <html:radio property="relatorio" value="cancelar"
styleClass="formulario"/>
        </th>
        <td align="left" class="textos" >
          Cancelar o Seguro
        </td>
      </tr>
      <tr>
        <th align="right">
          <html:radio property="relatorio" value="manter"
styleClass="formulario"/>
        </th>
        <td align="left" class="textos">
          Manter o Seguro
        </td>
      </tr>
      <tr>
        <th align="right">
          <html:radio property="relatorio" value="incluir"
styleClass="formulario"/>
        </th>
        <td align="left" class="textos">
          Incluir no Seguro
        </td>
```

```
        </tr>
        <tr>
          <th align="right">
            <html:radio property="relatorio" value="geral"
styleClass="formulario"/>
          </th>
          <td align="left" class="textos" class="textos">
            Relatório Geral
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Mês de referência:
          </th>
          <td align="left">
            <html:select property="mes"
styleClass="formulario">
              <html:option value="1">Janeiro</html:option>
              <html:option value="2">Fevereiro</html:option>
              <html:option value="3">Março</html:option>
              <html:option value="4">Abril</html:option>
              <html:option value="5">Maio</html:option>
              <html:option value="6">Junho</html:option>
              <html:option value="7">Julho</html:option>
              <html:option value="8">Agosto</html:option>
              <html:option value="9">Setembro</html:option>
              <html:option value="10">Outubro</html:option>
              <html:option
value="11">Novembro</html:option>
              <html:option
value="12">Dezembro</html:option>
            </html:select>
          </td>
        </tr>
        <tr>
          <th align="left" class="textos">
            Ano:
          </th>
          <td align="left">
            <html:select property="ano"
styleClass="formulario">
              <html:options name="anos" labelName="anos" />
            </html:select>
          </td>
        </tr>
      </table>
      <div class="button" id="botao">
        <div align="center">
          <html:submit styleClass="formulario">
          Enviar
          </html:submit>
        </div>
      </div>
    </html:form>
```

6.4.160 sucesso.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
OPERAÇÃO REALIZADA COM SUCESSO !
<hr align="left" class="textos"/>
<p class="textos">
Usuario: <bean:write name="user" property="usuario"/>
</P>
<p class="textos">
Status: <bean:write name="user" property="status"/>
</p>
```

6.4.161 tcc.css

```
/* CSS Document */

.tcc {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #084F98;
        font-style: normal;
        font-weight: bold;
        text-decoration: none;
        background-color: #FFFFFF;
        border: none;
        background-position: 0px 0px;
        height: 0px;
        width: 0px;
        margin: 0px;
        padding: 0px;
        letter-spacing: normal;
        text-align: justify;
        word-spacing: normal;
        list-style-type: square;
}
.fundoazul {
        background-color: #084F98;
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #FFFFFF;
}
.textoCinza {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #666666;
        font-weight: normal;
}
.fundoCinza {
        background-color: #EFEFEF;
}
.menu {
        font-family: Arial, Helvetica, sans-serif;
        color: #084F98;
        font-size: 12px;
        background-color: #FFFFFF;
        text-decoration: none;
}
.menu:Hover {
        font-family: Arial, Helvetica, sans-serif;
        color: #084F98;
        font-size: 12px;
        background-color: #FFFFFF;
        text-decoration: underline;
}
.textos {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #084F98;
}
.formulario {
        font-family: Arial, Helvetica, sans-serif;
        font-size: 11px;
        color: #333333;
        border: 1px outset #EFEFEF;
}
```

6.4.162 termocompromisso.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<p align="center" class="textos">
  <logic:notPresent name="termo">
  Termo de Compromisso Alterado
  </logic:notPresent>
</p>
```

6.4.163 tipousuario.jsp

```
<%@ taglib uri="/tags/struts-html" prefix="html" %>
```

```
<%@ taglib uri="/tags/struts-logic" prefix="logic" %>
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<logic:equal name="user" value="admin" property="status">
  <html:form action="TipoUsuario">
    <p align="center" class="textos">
      Perfil do Usuário
      <hr align="center" class="textos"/>
    </p>
    <table>
      <logic:iterate id="usuario" name="usuarios">
        <tr>
          <th align="right">
            <html:radio property="status" idName="usuario"
value="tipo" styleClass="formulario"/>
          </th>
          <td align="left" class="textos">
            <bean:write name="usuario"
property="mensagem"/>
          </td>
        </tr>
      </logic:iterate>
    </table>
    <div class="button" id="botao">
      <div align="center">
        <html:submit  styleClass="formulario">
          Enviar
        </html:submit>
      </div>
    </div>
  </html:form>
</logic:equal>
<logic:notEqual name="user" value="admin" property="status">
  <p align="center" class="textos">
    VOCÊ NÃO TEM PERMISSÃO PARA VER ESSA
PÁGINA !
  </p>
</logic:notEqual>
```

6.4.164 Welcome.jsp

```
<%@ taglib uri="/tags/struts-bean" prefix="bean" %>
<%@ taglib uri="/tags/struts-html" prefix="html" %>
<html:form action="/LogonSubmit" focus="username">
  <table border="0" width="100%">
    <tr>
      <th align="right">
        username:
      </th>
      <td align="left">
        <html:text property="username" size="16"
maxlength="18"/>
      </td>
    </tr>
    <tr>
      <th align="right">
        senha:
      </th>
      <td align="left">
        <html:password property="password" size="16"
maxlength="18"/>
      </td>
    </tr>
    <tr>
      <td align="right">
        <html:submit property="Submit" value="Submit"/>
      </td>
      <td align="left">
        <html:reset/>
      </td>
    </tr>
    <tr>
      <th align="right">
        Novo Usuário ?
      </th>
      <td align="left">
        <P><html:link action="/TipoUsuario?status=aluno"
styleClass="menu">Registre-se.</html:link></P>
      </td>
    </tr>
  </table>

</html:form>
```

6.4.165 package estagio.action

6.4.166 AlterarSenhaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.dao.AlunoEstagioDao;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class AlterarSenhaAction  extends Action{

   public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
   }

}
```

6.4.167 BuscarEmpresaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.form.BuscarEmpresaForm;
import java.sql.SQLException;
```

```
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

public class BuscarEmpresaAction extends Action{

   public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      EmpresaDao empresaDao = new EmpresaDao();
      BuscarEmpresaForm buscarEmpresaForm =
(BuscarEmpresaForm) form;
      Empresa empresa = null;
      try{
        empresa =
empresaDao.buscarEmpresa(buscarEmpresaForm.getCnpj());
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
      }
      if(empresa != null){
        request.setAttribute("empresa", empresa);
        request.setAttribute("cnpj", empresa.getCnpj());
      } else {
        request.setAttribute("mensagem", "CNPJ
"+buscarEmpresaForm.getCnpj()+" não encontrado.");
        return mapping.findForward(Constants.FAILURE);
      }
      //parametros para dadospessoais.jsp
      String status = request.getParameter("status");
      CursoDao cursoDao = new CursoDao();
      List<Curso> cursos = new ArrayList<Curso>();
      List<LabelValueBean> selectCursos = new
ArrayList<LabelValueBean>();
      try{
        cursos = cursoDao.listarCursos();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
      }
      for(Curso c: cursos){
        selectCursos.add(new
LabelValueBean(c.getNomeCurso(),
c.getCodCurso().toString()));
      }
      if(cursos.size()>0){
        request.setAttribute("cursos", selectCursos);
      } else {
        request.setAttribute("cursos", new
LabelValueBean("0","Não Informado"));
      }

      ArrayList sexos = new ArrayList();
      sexos.add(new LabelValueBean("masculino", "M"));
      sexos.add(new LabelValueBean("feminino", "F"));
      request.setAttribute("sexos", sexos);
      ArrayList simNao = new ArrayList();
      simNao.add(new LabelValueBean("Sim", "S"));
      simNao.add(new LabelValueBean("Não", "N"));
      request.setAttribute("simNao", simNao);
      ArrayList locomotion = new ArrayList();
      locomotion.add(new LabelValueBean("Carro", "C"));
      locomotion.add(new LabelValueBean("Moto", "M"));
      locomotion.add(new LabelValueBean("Não Possui", "N"));
      request.setAttribute("locomocao", locomotion);
      request.setAttribute("usuario", user.getUsuario());
      request.setAttribute("status", status);
      Integer[] semestres = new Integer[12];
      for(int i=0; i<12; i++){
        semestres[i] = Integer.valueOf(i);
      }
      request.setAttribute("semestres", semestres);
      String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
      request.setAttribute("estados", estados);

      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
   }
}
```

6.4.168 BuscarEstagiarioAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.TermoCompromisso;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.TermoCompromissoDao;
import estagio.form.BuscarEstagiarioForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class BuscarEstagiarioAction extends Action{

   public ActionForward execute(
```

```
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    BuscarEstagiarioForm buscarEstagiarioForm =
(BuscarEstagiarioForm) form;
    List<AlunoEstagio> estagiarios = new
ArrayList<AlunoEstagio>();
    try{
      estagiarios =
alunoEstagioDao.buscarEstagiario(buscarEstagiarioForm.getNom
eEstagiario());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.estagiarios"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.estagiarios"));
    }
    /*TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
    List<TermoCompromisso> termoCompromissos = new
ArrayList<TermoCompromisso>();
    try{
      termoCompromissos =
termoCompromissoDao.listarTermosCompromisso();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.termocompromisso"));
    }
    List<AlunoEstagio> estagiariosComTermo = new
ArrayList<AlunoEstagio>();
    for(TermoCompromisso t: termoCompromissos){
      AlunoEstagio estagiario =
alunoEstagioDao.buscarAlunoEstagio(t.getAlunoEstagio().getCod
AlunoEstagio());
      for(AlunoEstagio alunoestagio : estagiarios){

if(alunoestagio.getCodAlunoEstagio().equals(estagiario.getCodAl
unoEstagio())){
          estagiariosComTermo.add(alunoestagio);
        }
      }
    }
    estagiarios.removeAll(estagiariosComTermo);*/
    if(estagiarios.size() > 0){
      request.setAttribute("alunos", estagiarios);
      request.setAttribute("codAlunoEstagio",
estagiarios.get(0).getCodAlunoEstagio());
    }
    String codVaga = (String) request.getParameter("codVaga");
    request.setAttribute("codVaga", codVaga);
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.169 BuscarHabilidadeAlunoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants; 6.4.16
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.bean.HabilidadeVaga;
import estagio.bean.Vaga;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.HabilidadesDao;
import estagio.dao.HabilidadesVagaDao;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class BuscarHabilidadeAlunoAction extends Action {

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {

    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }

    String codAlunoEstagio =
request.getParameter("codAlunoEstagio");
    request.setAttribute("codAlunoEstagio", codAlunoEstagio);
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    AlunoEstagio aluno =
alunoEstagioDao.buscarAlunoEstagio(Integer.valueOf(codAluno
Estagio));
    String curso = aluno.getCurso().getNomeCurso();
    request.setAttribute("curso", curso);
```

```
    List<Habilidade> habilidades = new
ArrayList<Habilidade>();
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
    VagaDao vagaDao = new VagaDao();
    List<Vaga> vagas = new ArrayList<Vaga>();
    List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
    try{
      vagas =
vagaDao.listarVagasAbertas(aluno.getCurso().getCodCurso());
      for(Vaga vaga: vagas){
        for(HabilidadeVaga h: vaga.getHabilidades()){
          boolean jaExiste = false;
          for(HabilidadeVaga hVaga: habilidadesVaga){

if(h.getCodHabilidade().equals(hVaga.getCodHabilidade())){
              jaExiste = true;
            }
          }
          if(!jaExiste){
            habilidadesVaga.add(h);
          }
        }
//habilidadesVaga.addAll(habilidadesVagaDao.listarHabilidadeV
aga(vaga.getCodVaga()));
      }
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidade"));
    }
    for(Habilidade h: habilidadesVaga){
      habilidades.add(h);
      for(HabilidadeAluno hAluno: habilidadesAluno){
        if(habilidades.contains(hAluno) &&
hAluno.getCodHabilidade().intValue() ==
h.getCodHabilidade().intValue()){
          habilidades.remove(h);
        }
      }
    }
    if(habilidades.size() > 0){
      request.setAttribute("todashabilidades", habilidades);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.170 BuscarRequisitosAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeVaga;
import estagio.bean.Vaga;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class BuscarRequisitosAction extends Action {
public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }

      String codVaga = request.getParameter("codvaga");
      request.setAttribute("codvaga", codVaga);
      //HabilidadesDao habilidadesDao = new HabilidadesDao();
      //HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
      VagaDao vagaDao = new VagaDao();
      Vaga vaga =
vagaDao.buscarVaga(Integer.valueOf(codVaga));
      String curso = "";
      for(int i = 0;i<vaga.getCursos().size();i++){
        Curso c = vaga.getCursos().get(i);
        if(vaga.getCursos().size() - 1 != i){
          curso += c.getNomeCurso()+", ";
        } else {
          curso += c.getNomeCurso()+".";
        }
      }
      request.setAttribute("curso", curso);
      List<Vaga> vagas = new ArrayList<Vaga>();
      List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
      List<Habilidade> habilidades = new
ArrayList<Habilidade>();
      try{
        for(Curso cursoVaga: vaga.getCursos()){

vagas.addAll(vagaDao.listarVagas(cursoVaga.getCodCurso()));
        }
        for(Vaga v: vagas){
          for(HabilidadeVaga h: v.getHabilidades()){
            boolean jaExiste = false;
            for(HabilidadeVaga hVaga: habilidadesVaga ){

if(h.getCodHabilidade().equals(hVaga.getCodHabilidade())){
                jaExiste = true;
              }
            }
            if(!jaExiste){
              habilidadesVaga.add(h);
            }
          }
```

```
        }
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidade"));
      }
      for(Habilidade h: habilidadesVaga){
        habilidades.add(h);
        for(Habilidade hVaga: vaga.getHabilidades()){
          if(habilidades.contains(hVaga) &&
              hVaga.getCodHabilidade().intValue() ==
h.getCodHabilidade().intValue()){
            habilidades.remove(hVaga);
          }
        }
      }
      if(habilidades.size() > 0){
        request.setAttribute("requisitos", habilidades);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.171 BuscarVagasAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.bean.Turno;
import estagio.bean.TurnoDisponivel;
import estagio.bean.Vaga;
import estagio.bean.VagaDTO;
import estagio.bean.VagaSelecionadaDTO;
import estagio.dao.EmpresaDao;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.TurnoDisponivelDao;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class BuscarVagasAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    EmpresaDao empresaDao = new EmpresaDao();
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      VagaDao vagaDao = new VagaDao();
      List<Vaga> vagasInicio = new ArrayList<Vaga>();
      try{
        vagasInicio =
vagaDao.buscarVagasAbertas(user.getDisponivelFds().charAt(0),
user.getFumante().charAt(0));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.vaga"));
      }
      //busca turnos e habilidades do aluno
      HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
      List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(user.getCodAlunoEs
tagio());
      TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
      List<TurnoDisponivel> turnosDisponiveis = new
ArrayList<TurnoDisponivel>();
      turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(user.getCodAlunoE
stagio());
      //turnos e habilidades do aluno encontrados

      //compara locomocao, sexo, deficiente do aluno com as
vagas e turnos e habilidades
      //HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
      //TurnoDao turnoDao = new TurnoDao();
      List<Vaga> vagasSelecionadas = new ArrayList<Vaga>();
      for(Vaga v : vagasInicio){
        if(v.getCursos().contains(user.getCurso())){
          if("A".equals(v.getSexo()) ||
user.getSexo().equals(v.getSexo())){
            if("A".equals(v.getLocomocao()) ||
user.getLocomocao().equalsIgnoreCase(v.getLocomocao())){
              if("N".equals(v.getDeficiente()) ||
v.getDeficiente().equals(user.getDeficiente())){
                if(user.getSemestre() >= v.getSemestre()){
                  int qtdTurnos = v.getTurnos().size();
                  int qtdTurnosOk = 0;
                  for(Turno horario: v.getTurnos()){
                    for(Turno disponivel: turnosDisponiveis){

if((horario.getHoraInicio().after(disponivel.getHoraInicio()) &&
horario.getHoraFim().before(disponivel.getHoraFim())) ||

(horario.getHoraInicio().equals(disponivel.getHoraInicio()) &&
horario.getHoraFim().equals(disponivel.getHoraFim())) ||

(horario.getHoraInicio().after(disponivel.getHoraInicio()) &&
horario.getHoraFim().equals(disponivel.getHoraFim())) ||

(horario.getHoraInicio().equals(disponivel.getHoraInicio()) &&
horario.getHoraFim().before(disponivel.getHoraFim()))){
                        ++qtdTurnosOk;
```

```
                        }
                    }
                }
                if(qtdTurnosOk == qtdTurnos){
                    int habilidadesVaga =
v.getHabilidades().size();
                    int habilidadesOk = 0;
                    for(Habilidade habilidadeVaga:
v.getHabilidades()){
                        for(Habilidade habilidadeAluno:
habilidadesAluno){
                            boolean ehSubstring = false;

if((habilidadeVaga.getDescricaoHabilidade().trim().toUpperCase(
).contains(habilidadeAluno.getDescricaoHabilidade().trim().toUp
perCase())) ||

(habilidadeAluno.getDescricaoHabilidade().trim().contains(habili
dadeVaga.getDescricaoHabilidade().trim()))){
                                ehSubstring = true;
                            }

if((habilidadeAluno.getDescricaoHabilidade().trim().equalsIgnore
Case(habilidadeVaga.getDescricaoHabilidade().trim())) ||
                                    (ehSubstring)){
                                ++habilidadesOk;
                            }
                        }
                    }
                    if(habilidadesOk >= habilidadesVaga){
                        vagasSelecionadas.add(v);
                    }
                }
            }
        }
    }
    List<VagaSelecionadaDTO> vagaDtos = new
ArrayList<VagaSelecionadaDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(Vaga vaga:vagasSelecionadas){
        VagaSelecionadaDTO vagaDto = new
VagaSelecionadaDTO();
        vagaDto.setCodVaga(vaga.getCodVaga());
        vagaDto.setHoras(vaga.getHoras().toString());

vagaDto.setInclusao(sdf.format(vaga.getInclusao().getTime()));
        vagaDto.setSalario(vaga.getSalario());
        vagaDto.setSetor(vaga.getSetor());
        vagaDto.setSemestre(vaga.getSemestre());
        char vt= vaga.getValeTransporte().charAt(0);
        switch (vt){
            case 'S':
                vagaDto.setValeTransporte("Sim");
                break;
            case 'N':
                vagaDto.setValeTransporte("Não");
                break;
            default:
                vagaDto.setValeTransporte("");
        }

vagaDto.setOutrosBeneficios(!"".equals(vaga.getOutrosBeneficio
s())?vaga.getOutrosBeneficios():"Não");
        String turnos = "";
        for(int i = 0;i<vaga.getTurnos().size();i++){
            Turno turno= vaga.getTurnos().get(i);
            if( vaga.getTurnos().size() - 1 != i){
                turnos += turno.toString()+", ";
            } else {
                turnos += turno.toString()+".";
            }
        }
        vagaDto.setTurnos("".equals(turnos)?"Não
Informado":turnos);
        String descricaoHabilidade = "";
        for(int i = 0;i<vaga.getHabilidades().size();i++){
            Habilidade habilidade = vaga.getHabilidades().get(i);
            if(vaga.getHabilidades().size() - 1 != i){
                descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
            } else {
                descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
            }
        }

vagaDto.setHabilidades("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
        Empresa empresa = new Empresa();
        empresa = empresaDao.buscarEmpresa(vaga.getCnpj());
        vagaDto.setRazaoSocial(empresa.getRazaoSocial());

vagaDto.setEndereco(empresa.getEnderecoEmpresa().toString());
        vagaDto.setContato(empresa.getFone());
        vagaDtos.add(vagaDto);
    }
    if(vagasSelecionadas.size()>0){
        request.setAttribute("vagas", vagaDtos);
    }
    if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
    } else {
        return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.172 ConsultarCursosVagasAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.Turno;
import estagio.bean.Vaga;
import estagio.bean.VagaDTO;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

public class ConsultarCursosVagasAction extends Action{
    public ActionForward execute(
        ActionMapping mapping,
```

```
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    CursoDao cursoDao = new CursoDao();
    List<Curso> cursos = new ArrayList<Curso>();
    List<LabelValueBean> selectCursos = new
ArrayList<LabelValueBean>();
    try{
      cursos = cursoDao.listarCursos();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
    }
    for(Curso c: cursos){
      selectCursos.add(new
LabelValueBean(c.getNomeCurso(),
c.getCodCurso().toString()));
    }
    if(cursos.size()>0){
      request.setAttribute("cursos", selectCursos);
    } else {
      request.setAttribute("cursos", new
LabelValueBean("0","Não Informado"));
    }
    String codCurso = (String)
request.getParameter("codCurso");
    String time = (String) request.getParameter("time");
    if(time != null){
      request.setAttribute("time", time);
    }
    VagaDao vagaDao = new VagaDao();
    List<Vaga> vagas = new ArrayList<Vaga>();
    if(codCurso != null){
      try{
        vagas =
vagaDao.listarVagasAbertas(Integer.valueOf(codCurso));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.vaga"));
      }
    }
    List<VagaDTO> vagaDtos = new ArrayList<VagaDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    DecimalFormat twoDigits = new
DecimalFormat("R$'0.00");
    EmpresaDao empresaDao = new EmpresaDao();
    for(Vaga vaga:vagas){
      VagaDTO vagaDto = new VagaDTO();
      if(vaga.getPreenchimento() == null ||
"null".equals(vaga.getPreenchimento())){
        vagaDto.setSituacao("Aberta.");
      } else {
        vagaDto.setSituacao("Preenchida.");
      }
      vagaDto.setCodVaga(vaga.getCodVaga());
      Empresa empresa =
empresaDao.buscarEmpresa(vaga.getCnpj());
      vagaDto.setEmpresa(empresa != null ?
empresa.getRazaoSocial():"Desconhecida");
      vagaDto.setHoras(vaga.getHoras().toString());

vagaDto.setInclusao(sdf.format(vaga.getInclusao().getTime()));
      vagaDto.setSalario(vaga.getSalario());
      vagaDto.setSetor(vaga.getSetor());
      vagaDto.setSemestre(vaga.getSemestre());
      char vt= vaga.getValeTransporte().charAt(0);
      switch (vt){
        case 'S':
          vagaDto.setValeTransporte("Sim");
          break;
        case 'N':
          vagaDto.setValeTransporte("Não");
          break;
        default:
          vagaDto.setValeTransporte("");
      }


vagaDto.setOutrosBeneficios(vaga.getOutrosBeneficios());
      char sexo = vaga.getSexo().charAt(0);
      switch (sexo){
        case 'M':
          vagaDto.setSexo("Masculino");
          break;
        case 'F':
          vagaDto.setSexo("Feminino");
          break;
        case 'A':
          vagaDto.setSexo("Qualquer");
          break;
        default:
          vagaDto.setSexo("");
      }
      String turnos = "";
      for(int i = 0;i<vaga.getTurnos().size();i++){
        Turno turno= vaga.getTurnos().get(i);
        if( vaga.getTurnos().size() - 1 != i){
          turnos += turno.toString()+", ";
        } else {
          turnos += turno.toString()+".";
        }
      }
      vagaDto.setTurnos("".equals(turnos)?"Não
Informado":turnos);
      String descricaoHabilidade = "";
      for(int i = 0;i<vaga.getHabilidades().size();i++){
        Habilidade habilidade = vaga.getHabilidades().get(i);
        if(vaga.getHabilidades().size() - 1 != i){
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
        } else {
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
        }
      }

vagaDto.setHabilidades("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
      //vagaDto.setCurso(vaga.getCurso().getNomeCurso());
      vagaDtos.add(vagaDto);
    }
    if(vagas.size()>0){
      request.setAttribute("vagas", vagaDtos);
```

```
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.173 CriarTermoCompromissoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class CriarTermoCompromissoAction extends Action {

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    /*AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    List<AlunoEstagio> alunos = new
ArrayList<AlunoEstagio>();
    try{
      alunos = alunoEstagioDao.listarAlunoEstagio("aluno");
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.alunos"));
    }
    if(alunos.size() > 0){
      request.setAttribute("alunos", alunos);
    }*/

    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.174 EditarCadastroAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.bean.Turno;
import estagio.bean.TurnoDTO;
import estagio.bean.TurnoDisponivel;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.CursoDao;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.TurnoDisponivelDao;
import estagio.form.DadosPessoaisForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

public class EditarCadastroAction extends Action {

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {

    HttpSession session = request.getSession();
    DadosPessoaisForm regform = (DadosPessoaisForm) form;
    AlunoEstagio userCheck = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    String action = request.getParameter("action");
    request.setAttribute("action", action);
    CursoDao cursoDao = new CursoDao();
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    ActionMessages errors = new ActionMessages();
    List<Curso> cursos = new ArrayList<Curso>();
    List<LabelValueBean> selectCursos = new
ArrayList<LabelValueBean>();
    try{
      cursos = cursoDao.listarCursos();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
    }
    for(Curso c: cursos){
      selectCursos.add(new
LabelValueBean(c.getNomeCurso(),
c.getCodCurso().toString()));
    }
```

```
    if(cursos.size()>0){
      request.setAttribute("cursos", selectCursos);
    } else {
      request.setAttribute("cursos", new
LabelValueBean("0","Não Informado"));
    }
    if (userCheck == null) {
      return (mapping.findForward("logon"));
    }
    ArrayList sexos = new ArrayList();
    sexos.add(new LabelValueBean("masculino", "M"));
    sexos.add(new LabelValueBean("feminino", "F"));
    request.setAttribute("sexos", sexos);
    ArrayList simNao = new ArrayList();
    simNao.add(new LabelValueBean("Sim", "S"));
    simNao.add(new LabelValueBean("Não", "N"));
    request.setAttribute("simNao", simNao);
    ArrayList locomotion = new ArrayList();
    locomotion.add(new LabelValueBean("Carro", "C"));
    locomotion.add(new LabelValueBean("Moto", "M"));
    locomotion.add(new LabelValueBean("Não Possui", "N"));
    request.setAttribute("locomocao", locomotion);
    Integer[] semestres = new Integer[12];
    for(int i=0; i<12; i++){
      if(i == 0) continue;
      semestres[i] = Integer.valueOf(i);
    }
    request.setAttribute("semestres", semestres);
    String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
    request.setAttribute("estados", estados);
    String usuario = request.getParameter("usuario");
    try {
      AlunoEstagio alunoEstagio =
alunoEstagioDao.buscarAlunoEstagio(usuario);
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      regform.setAction(action);

regform.setDataNascimento(sdf.format(alunoEstagio.getDataNasc
imento().getTime()));
      regform.setDeficiente(alunoEstagio.getDeficiente());

regform.setDisponivelFds(alunoEstagio.getDisponivelFds());
      regform.setEmail(alunoEstagio.getEmail());
      regform.setFone(alunoEstagio.getFone());
      regform.setFumante(alunoEstagio.getFumante());
      regform.setLocomocao(alunoEstagio.getLocomocao());

regform.setNome_pessoa(alunoEstagio.getNomePessoa());
      //regform.setSenha(alunoEstagio.getSenha());
      //regform.setSenha2(alunoEstagio.getSenha());
      regform.setSexo(alunoEstagio.getSexo());
      regform.setUsuario(alunoEstagio.getUsuario());
      regform.setStatus(alunoEstagio.getStatus());

regform.setCodCurso(alunoEstagio.getCurso().getCodCurso());
      regform.setSemestre(alunoEstagio.getSemestre());
      regform.setRg(alunoEstagio.getRg());

regform.setRg_naturalidade(alunoEstagio.getRg_naturalidade());
      regform.setCpf(alunoEstagio.getCpf().toString());
      regform.setMatricula(alunoEstagio.getMatricula());
      regform.setCep(alunoEstagio.getCep());
      regform.setRua(alunoEstagio.getRua());
      regform.setNumero(alunoEstagio.getNumero());

regform.setComplemento(alunoEstagio.getComplemento());
      regform.setBairro(alunoEstagio.getBairro());
      regform.setEstado(alunoEstagio.getEstado());
      regform.setMunicipio(alunoEstagio.getMunicipio());
      regform.setPais(alunoEstagio.getPais());
      request.setAttribute("codalunoestagio",
alunoEstagio.getCodAlunoEstagio());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.recuperar.dados"));
    }
    //parametros para listar habilidades
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    Integer codAlunoEstagio =
userCheck.getCodAlunoEstagio();
    try{
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(codAlunoEstagio);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadealuno"));
    }
    String descricaoHabilidade = "";
    for(int i = 0;i<habilidadesAluno.size();i++){
      Habilidade habilidade = habilidadesAluno.get(i);
      if(habilidadesAluno.size() - 1 != i){
        descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
      } else {
        descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
      }
    }
    if(habilidadesAluno.size()>0){
      request.setAttribute("habilidades",
"".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
    }
    //fim dos parametros para listar habilidades

    //parametros para listar turnos
    TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
    List<TurnoDisponivel> turnosDisponiveis = new
ArrayList<TurnoDisponivel>();
    try{
      turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(codAlunoEstagio);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
    String turnos = "";
    for(int i = 0;i < turnosDisponiveis.size();i++){
      Turno turno = turnosDisponiveis.get(i);
      TurnoDTO turnoDto = new TurnoDTO();
      turnoDto.setCodTurno(turno.getCodTurno());
      turnoDto.setTurno(turno.toString());
      turnosDTO.add(turnoDto);
      if( turnosDisponiveis.size() - 1 != i){
        turnos += turno.toString()+", ";
```

```
        } else {
          turnos += turno.toString()+".";
        }
      }
      if(turnosDisponiveis.size()>0){
        request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      }
      //fim dos parametros para listar turnos
      if (errors.isEmpty()){
        return mapping.findForward("success");
      } else {
        return mapping.findForward("error");
      }
    }

}
```

6.4.175 EditarEmailAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.ServidorSmtp;
import estagio.dao.Servidor_smtpDao;
import estagio.form.ServidorSmtpForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class EditarEmailAction extends Action {

  public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {

      HttpSession session = request.getSession();
      ServidorSmtpForm serverform = (ServidorSmtpForm) form;
      AlunoEstagio userCheck = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      if (userCheck == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      Servidor_smtpDao servidorDao = new Servidor_smtpDao();
      ServidorSmtp smtpServer = new ServidorSmtp();
      List<ServidorSmtp> servidores = new
ArrayList<ServidorSmtp>();
      ServidorSmtp servidor = new ServidorSmtp();
      try {
        servidores = servidorDao.listarServidor_smtp();
        if(servidores.isEmpty()){
          servidor.setCod_servidor_smtp(Integer.valueOf(1));
          servidor.setEndereco_email("");
          servidor.setPorta_smtp(Integer.valueOf(110));

servidor.setRequer_autenticacao(Boolean.valueOf(false));
          servidor.setSenha("");
          servidor.setUrl("");
          servidor.setUsername("");
          try{
            servidorDao.incluirServidorSmtp(servidor);
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.incluir.servidor"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.incluir.servidor"));
          }
        } else {
          servidor =
servidorDao.buscarServidorSmtp(Integer.valueOf(1));
        }

serverform.setCod_servidor_smtp(servidor.getCod_servidor_smtp
());

serverform.setEndereco_email(servidor.getEndereco_email());
        serverform.setPorta_smtp(servidor.getPorta_smtp());

serverform.setRequer_autenticacao(servidor.getRequer_autentica
cao());
        serverform.setSenha(servidor.getSenha());
        serverform.setUrl(servidor.getUrl());
        serverform.setUsername(servidor.getUsername());
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.recuperar.dados"));
      }

      if (errors.isEmpty()){
        return mapping.findForward("success");
      } else {
        return mapping.findForward("error");
      }
    }
    }
```

6.4.176 EditarEmpresaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.dao.EmpresaDao;
import estagio.form.NovaEmpresaForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class EditarEmpresaAction extends Action {
  public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {

      HttpSession session = request.getSession();
      NovaEmpresaForm empresaform = (NovaEmpresaForm)
form;
      AlunoEstagio userCheck = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
```

```
        String action = request.getParameter("action");
        //request.setAttribute("action", action);
        String cnpj = request.getParameter("cnpj");
        EmpresaDao empresaDao = new EmpresaDao();
        ActionMessages errors = new ActionMessages();
        if (userCheck == null) {
          return (mapping.findForward("logon"));
        }
        try {
          Empresa empresa= empresaDao.buscarEmpresa(cnpj);
          empresaform.setAction(action);
empresaform.setBairro(empresa.getEnderecoEmpresa().getBairro(
));
empresaform.setCargoRepresentante(empresa.getCargoRepresent
ante());
empresaform.setCep(empresa.getEnderecoEmpresa().getCep());
          empresaform.setCnpj(cnpj);
empresaform.setComplemento(empresa.getEnderecoEmpresa().ge
tComplemento());
          empresaform.setEmail(empresa.getEmail());
empresaform.setEstado(empresa.getEnderecoEmpresa().getEstado
());
  empresaform.setFax(empresa.getFax());
          empresaform.setFone(empresa.getFone());
empresaform.setMunicipio(empresa.getEnderecoEmpresa().getM
unicipio());
empresaform.setNomeFantasia(empresa.getNomeFantasia());
empresaform.setNumProcesso(empresa.getNumProcesso());
empresaform.setNumero(empresa.getEnderecoEmpresa().getNum
ero());
empresaform.setPais(empresa.getEnderecoEmpresa().getPais());
          empresaform.setRamo(empresa.getRamo());
          empresaform.setRazaoSocial(empresa.getRazaoSocial());
empresaform.setRepresentante(empresa.getRepresentante());
empresaform.setRua(empresa.getEnderecoEmpresa().getRua());
          empresaform.setSite(empresa.getSite());
empresaform.setAgenteIntegrador(empresa.getAgenteIntegracao()
);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.recuperar.dados"));
        }
        String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
        request.setAttribute("estados", estados);
        if (errors.isEmpty()){
          return mapping.findForward("success");
        } else {
          return mapping.findForward("error");
        }
    }

}
```

6.4.177 EditarTermoAction.java

```
package estagio.action;
import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.TermoCompromisso;
import estagio.dao.AgentesIntegracaoDao;
import estagio.dao.TermoCompromissoDao;
import estagio.form.NovoTermoCompromissoForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

public class EditarTermoAction extends Action {
    public ActionForward execute(
          ActionMapping mapping,
          ActionForm form,
          HttpServletRequest request,
          HttpServletResponse response)
          throws Exception {
            HttpSession session = request.getSession();
        NovoTermoCompromissoForm termoCompromissoform =
(NovoTermoCompromissoForm) form;
        AlunoEstagio userCheck = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        String action = request.getParameter("action");
        if (userCheck == null) {
          return (mapping.findForward("logon"));
        }
        ActionMessages errors = new ActionMessages();
        TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
        SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
        TermoCompromisso termoCompromisso = null;
        try {
          termoCompromisso =
termoCompromissoDao.buscarTermoCompromisso(Integer.value
Of(request.getParameter("codTermo")));
          termoCompromissoform.setAction(action);

termoCompromissoform.setCanal(termoCompromisso.getAgenteI
ntegracao()!=null?termoCompromisso.getAgenteIntegracao().get
Agente():"nenhum");
termoCompromissoform.setCnpj(termoCompromisso.getEmpresa
().getCnpj());
termoCompromissoform.setCodAlunoEstagio(termoCompromiss
o.getAlunoEstagio().getCodAlunoEstagio());
termoCompromissoform.setCodVaga(termoCompromisso.getVag
a().getCodVaga());
termoCompromissoform.setDataFim(sdf.format(termoCompromis
so.getDataFim().getTime()));
termoCompromissoform.setDataInicio(sdf.format(termoCompro
misso.getDataInicio().getTime()));
termoCompromissoform.setObrigatorio("S".equalsIgnoreCase(ter
moCompromisso.getObrigatorio())?"true":"");
termoCompromissoform.setOrientador(termoCompromisso.getPla
noEstagio().getOrientador());
termoCompromissoform.setFormacaoOrientador(termoCompromi
sso.getPlanoEstagio().getFormacaoOrientador());
termoCompromissoform.setObjetivos(termoCompromisso.getPla
noEstagio().getObjetivos());
termoCompromissoform.setAtividades(termoCompromisso.getPla
noEstagio().getAtividades());
termoCompromissoform.setAreaAtividadeEstagiario(termoCompr
omisso.getPlanoEstagio().getAreaAtividadeEstagiario());
termoCompromissoform.setSeguroCompanhia(termoCompromiss
o.getSeguroCompanhia());
termoCompromissoform.setSeguroNumApolice(termoCompromi
sso.getSeguroNumApolice());
termoCompromissoform.setCargoOrientador(termoCompromisso.
getPlanoEstagio().getCargoOrientador());
```

```
termoCompromissoform.setDepartamento(termoCompromisso.ge
tPlanoEstagio().getDepartamento());
termoCompromissoform.setRamal(termoCompromisso.getPlanoE
stagio().getRamal());
//termoCompromissoform.setSeguroProvidenciado(termoCompro
misso.getSeguroProvidenciado());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.recuperar.dados"));
    }
    //seta parametros para NovoTermoCompromisso
    String empresa = termoCompromisso !=
null?termoCompromisso.getEmpresa().getCnpj():"";
    String cnpj = termoCompromisso !=
null?termoCompromisso.getEmpresa().getRazaoSocial():"";
    /*Empresa emp = new Empresa();
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    if(userCheck.getStatus() != null){
      if("empresa".equals(userCheck.getStatus())){
        emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(userCheck.getCodAl
unoEstagio());
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }*/
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    AgentesIntegracaoDao agentesIntegracaoDao = new
AgentesIntegracaoDao();
    List<LabelValueBean> selectAgentes = new
ArrayList<LabelValueBean>();
    List<AgenteIntegracao> agentes = new
ArrayList<AgenteIntegracao>();
    try{
      agentes =
agentesIntegracaoDao.listarAgentesIntegracao();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.agente"));
    }
    for(AgenteIntegracao agente: agentes){
      selectAgentes.add(new
LabelValueBean(agente.getAgente(), agente.getCnpj()));
    }
    if(selectAgentes.size()>0){
      request.setAttribute("selectAgentes", selectAgentes);
    } else {
      request.setAttribute("selectAgentes", new
LabelValueBean("Nao informado", "0"));
    }
    if(termoCompromisso.getAgenteIntegracao()!= null){
      request.setAttribute("agente", "true");
    }

if(Character.valueOf('S').equals(termoCompromisso.getVaga().ge
tEstagioInterno())){
      request.setAttribute("estagio", "interno");
    }
    Map parametros = new HashMap();
    parametros.put("codVaga",
termoCompromisso.getVaga().getCodVaga());
    parametros.put("codAlunoEstagio",
termoCompromisso.getAlunoEstagio().getCodAlunoEstagio());
    request.setAttribute("parametros", parametros);
    //setados parametros para NovoTermoCompromisso
    if (errors.isEmpty()){
      return mapping.findForward("success");
    } else {
      return mapping.findForward("error");
    }
  }

}
```

6.4.178 EditarVagaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Vaga;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.dao.VagaDao;
import estagio.form.NovaVagaForm;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
public class EditarVagaAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
        HttpSession session = request.getSession();
    NovaVagaForm vagaform = (NovaVagaForm) form;
    AlunoEstagio userCheck = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    String action = request.getParameter("action");
    if (userCheck == null) {
      return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    ArrayList sexos = new ArrayList();
    VagaDao vagaDao = new VagaDao();
    DecimalFormat twoDigits = new DecimalFormat();
    Vaga vaga = new Vaga();
    try {
      vaga =
vagaDao.buscarVaga(Integer.valueOf(request.getParameter("codv
aga")));
      vagaform.setAction(action);
      vagaform.setAtividadesFds(vaga.getAtividadesFds());
      vagaform.setCnpj(vaga.getCnpj());
      vagaform.setCodVaga(vaga.getCodVaga().toString());
      vagaform.setDeficiente(vaga.getDeficiente());
      vagaform.setFumante(vaga.getFumante());
      vagaform.setHoras(vaga.getHoras().toString());
      vagaform.setLocomocao(vaga.getLocomocao());
      vagaform.setSalario(vaga.getSalario());
```

```
        vagaform.setSetor(vaga.getSetor());
        vagaform.setSexo(vaga.getSexo());
        vagaform.setSemestre(vaga.getSemestre());
        vagaform.setValeTransporte(vaga.getValeTransporte());
vagaform.setOutrosBenefícios(vaga.getOutrosBeneficios());
        vagaform.setEstagio("");
if(Character.valueOf('S').equals(vaga.getEstagioInterno())){
          vagaform.setEstagio("interno");          }
        String[] codCursos = new String[vaga.getCursos().size()];
        for(Curso c: vaga.getCursos()){
          codCursos[vaga.getCursos().indexOf(c)] =
c.getCodCurso().toString();
        }
        //request.setAttribute("codCurso", codCursos);
        vagaform.setCodCurso(codCursos);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.recuperar.dados"));
      }
      //seta parametros para NovaVaga
      //lista os cursos
      CursoDao cursoDao = new CursoDao();
      List<Curso> cursos = new ArrayList<Curso>();
      try{
        cursos = cursoDao.listarCursos();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
      }
      if(cursos.size()>0){
        request.setAttribute("cursos", cursos);
      }
      //cursos listados
      ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
      options.add(new LabelValueBean("Masculino", "M"));
      options.add(new LabelValueBean("Feminino", "F"));
      options.add(new LabelValueBean("Qualquer", "A"));
      request.setAttribute("sexos", options);
      ArrayList simNaoAll = new ArrayList();
      simNaoAll.add(new LabelValueBean("Sim", "S"));
      simNaoAll.add(new LabelValueBean("Não", "N"));
      simNaoAll.add(new LabelValueBean("Não Importa", "A"));
      request.setAttribute("simNaoAll", simNaoAll);
      ArrayList simNao = new ArrayList();
      simNao.add(new LabelValueBean("Sim", "S"));
      simNao.add(new LabelValueBean("Não", "N"));
      request.setAttribute("simNao", simNao);
      ArrayList locomotion = new ArrayList();
      locomotion.add(new LabelValueBean("Carro", "C"));
      locomotion.add(new LabelValueBean("Moto", "M"));
      locomotion.add(new LabelValueBean("Não Importa", "A"));
      request.setAttribute("locomocao", locomotion);
      String empresa = "";
      String cnpj = "";
      Empresa emp = new Empresa();
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      if(userCheck.getStatus() != null){
        if("empresa".equals(userCheck.getStatus())){
          emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(userCheck.getCodAl
unoEstagio());
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        }
      }
      if(Character.valueOf('S').equals(vaga.getEstagioInterno())){
        cnpj = vaga.getCnpj();
        EmpresaDao empresaDao = new EmpresaDao();
        emp = empresaDao.buscarEmpresa(cnpj);
        if(emp != null){
          empresa = emp.getRazaoSocial();
        }
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      //setados parametros para NovaVaga
      request.setAttribute("action", action);
      if (errors.isEmpty()){
        return mapping.findForward("success");
      } else {
        return mapping.findForward("error");
      }
    }
}
```

6.4.179 EnviaSenhaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.ServidorSmtp;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.Servidor_smtpDao;
import estagio.mail.MailManager;
import estagio.mail.MailServer;
import java.sql.SQLException;
import java.text.DecimalFormat;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.commons.beanutils.PropertyUtils;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class EnviaSenhaAction extends Action{
    /** Creates a new instance of LogonAction */
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    String userName = null;
    String email = null;
    userName = (String)
PropertyUtils.getSimpleProperty(form,"username");
    email = (String)
PropertyUtils.getSimpleProperty(form,"email");
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    Servidor_smtpDao servidorDao = new Servidor_smtpDao();
    AlunoEstagio user = new AlunoEstagio();
    try {
      user = alunoEstagioDao.buscarAlunoEstagio(userName);
    } catch (SQLException e){
      errors.add(
          ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.recuperar.dados"));
    } catch (Exception e){
      errors.add(
          ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
```

```
    }
    if (user != null) {
      if(!email.equals(user.getEmail())) {
        errors.add(
            ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.email.incorreto"));
      }
    } else {
      errors.add(
          ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.usuario.naoexiste"));
    }
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward("error"));
    }else {
      ServidorSmtp servidorSmtp =
servidorDao.listarServidor_smtp().get(0);
      if(servidorSmtp != null){
        try{
          MailServer server = new
MailServer(servidorSmtp.getUrl(),servidorSmtp.getUsername(),se
rvidorSmtp.getSenha(),servidorSmtp.getRequer_autenticacao().bo
oleanValue());
          MailManager m = new MailManager();
          //altera senha e armazenha o hashCode da nova senha
          String novasenha =
String.valueOf(user.getUsuario().hashCode());
          if(user.getUsuario().hashCode() < 0){
            novasenha =
String.valueOf(user.getUsuario().hashCode()*-1);
          }
user.setSenha(String.valueOf(novasenha.hashCode()));
          try{
            alunoEstagioDao.alterarAlunoEstagio(user);
          }catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.alterar.senha"));
          }catch (Exception e) {
errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.senha"));
          }
m.send(server,user.getEmail(),"","",servidorSmtp.getEndereco_e
mail(),"Sistema de Controle de Estágios - UFPR","Recuperação
de senha.\nSua senha foi alterada. Para acessar o sistema utiliza a
senha:\n"+novasenha, servidorSmtp);
          request.setAttribute("mensagem", "Sua senha foi
enviada. Verifique seu e-mail.");
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.enviar.senha"));
        }
      } else {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.servidorsmtp.naoexiste"));
      }
    }
    if(errors.isEmpty()){
      return (mapping.findForward(Constants.SUCCESS));
    } else {
      saveErrors(request, errors );
      return (mapping.findForward(Constants.FAILURE));
    }
  }
}
```

6.4.180 ExcluirAgenteAction.java

```
package estagio.action;
import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.dao.AgentesIntegracaoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirAgenteAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String cnpj = "";
    try{
      AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      cnpj = (String) request.getParameter("cnpj");
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    AgentesIntegracaoDao agentesIntegracaoDao = new
AgentesIntegracaoDao();
    try {
      agentesIntegracaoDao.excluirAgenteIntegracao(cnpj);
    } catch (SQLException e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.delete.agente"));
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.agente"));
    }
    List<AgenteIntegracao> agentesIntegracao = new
ArrayList<AgenteIntegracao>();
    try{
      agentesIntegracao =
agentesIntegracaoDao.listarAgentesIntegracao();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.agente"));
    }
    if(agentesIntegracao.size()>0){
      request.setAttribute("agentes", agentesIntegracao);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.181 ExcluirCadastroAction.java

```
package estagio.action;
```

```
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.dao.AlunoEstagioDao;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirCadastroAction extends Action{

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codAlunoEstagio = "";
      try{
         AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
         codAlunoEstagio = (String)
request.getParameter("codAlunoEstagio");
         if (user == null){
            return mapping.findForward("logon");
         }
      } catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE,
               new ActionMessage("error.generico"));
      }
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
      AlunoEstagio alunoEstagio =
alunoEstagioDao.buscarAlunoEstagio(Integer.valueOf(codAluno
Estagio));
      try {
         alunoEstagioDao.excluirAlunoEstagio(alunoEstagio);
      } catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.alunoestagio"));
      }
      if(!errors.isEmpty()){
         saveErrors(request, errors);
         return mapping.findForward(Constants.FAILURE);
      } else {
         return mapping.findForward(Constants.SUCCESS);
      }
   }
}
```

6.4.182 ExcluirConvenioAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.ConvenioDTO;
import estagio.dao.ConvenioDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirConvenioAction extends Action{

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codConvenio = "";
      try{
         AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
         codConvenio = (String)
request.getParameter("codConvenio");
         if (user == null){
            return mapping.findForward("logon");
         }
      } catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE,
               new ActionMessage("error.generico"));
      }
      ConvenioDao cDao = new ConvenioDao();
      Convenio c = new Convenio();
      c.setCodConvenio(Integer.valueOf(codConvenio));
      c = cDao.buscarConvenio(c);
      try {
         cDao.excluirConvenio(c);
      } catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.convenio"));
      }
      if(errors.isEmpty()){
         List<Convenio> convenios = new
ArrayList<Convenio>();
         try{
            convenios = cDao.listarConvenios();
         } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
         }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
         }
         List<ConvenioDTO> cDtos = new
ArrayList<ConvenioDTO>();
         SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
         GregorianCalendar hoje = new GregorianCalendar();
         for(Convenio conv: convenios){
            ConvenioDTO cDto = new ConvenioDTO();
            cDto.setCodConvenio(conv.getCodConvenio());
cDto.setDataFim(sdf.format(conv.getDataFim().getTime()));
cDto.setDataInicio(sdf.format(conv.getDataInicio().getTime()));
            cDto.setEmpresa(conv.getEmpresa().getRazaoSocial());
            cDto.setDescricao(conv.getDescricao());
            if(hoje.after(conv.getDataFim())){
               cDto.setSituacao("Expirado.");
            } else {
               cDto.setSituacao("Válido.");
            }
            cDtos.add(cDto);
         }
         if(convenios.size()>0){
```

```
            request.setAttribute("convenios", cDtos);
        }
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.183 ExcluirCursoAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.dao.CursoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirCursoAction extends Action{

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codCurso = "";
      try{
        AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        codCurso = (String) request.getParameter("codCurso");
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      CursoDao cursoDao = new CursoDao();
      Curso curso = new Curso();
      curso.setCodCurso(Integer.valueOf(codCurso));
      try {
        cursoDao.excluirCurso(curso);
      } catch (SQLException e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.delete.curso"));
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.curso"));
      }
      //parametros para listarcursos.jsp
      List<Curso> cursos = new ArrayList<Curso>();
      try{
        cursos = cursoDao.listarCursos();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
      }
      if(cursos.size()>0){
        request.setAttribute("cursos", cursos);
      }
      //fim dos parametros para listar cursos.jsp
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.184 ExcluirEmpresaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.Empresa;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirEmpresaAction extends Action{

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String cnpj = "";
      try{
        AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        cnpj = (String) request.getParameter("cnpj");
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      EmpresaDao empresaDao = new EmpresaDao();
      Empresa empresa = new Empresa();
      try {
        empresa = empresaDao.buscarEmpresa(cnpj);
        empresaDao.excluirEmpresa(empresa);
      } catch (SQLException e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.delete.empresa"));
      } catch (Exception e) {
```

```
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.empresa"));
      }
      //parametros para listarempresas.jsp
      List<Empresa> empresas = new ArrayList<Empresa>();
      try{
        empresas = empresaDao.listarEmpresas();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.empresa"));
      }
      ConvenioDao convenioDao = new ConvenioDao();
      List<Convenio> convenios = new ArrayList<Convenio>();
      try{
        convenios = convenioDao.listarConvenios();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
      }
      for(Empresa emp:empresas){
        emp.setCargoRepresentante("Não possui.");
        for(Convenio c: convenios){
          if(c.getEmpresa().getCnpj().equals(emp.getCnpj())){
            GregorianCalendar hoje = new GregorianCalendar();
            if(c.getDataFim().after(hoje)){

emp.setCargoRepresentante(c.getCodConvenio().toString());
            } else {
              emp.setCargoRepresentante("Expirado");
            }
          }
        }
      }
      if(empresas.size()>0){
        request.setAttribute("empresas", empresas);
      }
      //fim dos parametros para listar empresas.jsp
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.185 ExcluirHabilidadeAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.bean.HabilidadeVaga;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.HabilidadesDao;
import estagio.dao.HabilidadesVagaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ExcluirHabilidadeAction extends Action{

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codHabilidade= "";
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        codHabilidade = (String)
request.getParameter("codHabilidade");
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      HabilidadesDao habilidadesDao = new HabilidadesDao();
      HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
      HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
      Habilidade habilidade = null;
      List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
      try{
        habilidadesAluno =
habilidadesAlunoDao.listarHabilidadesAluno(Integer.valueOf(co
dHabilidade));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadealuno"));
      }
      List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
      try{
        habilidadesVaga =
habilidadesVagaDao.listarHabilidadesVaga(Integer.valueOf(codH
abilidade));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadevaga"));
      }
      for(HabilidadeAluno habilidadeAluno: habilidadesAluno){

habilidadesAlunoDao.excluirHabilidadeAluno(habilidadeAluno);
      }
      for(HabilidadeVaga habilidadeVaga: habilidadesVaga){

habilidadesVagaDao.excluirHabilidadeVaga(habilidadeVaga);
      }
      try {
        habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(codHabilidade)
);
        if(habilidade != null){
```

```
habilidadesDao.excluirHabilidade(Integer.valueOf(codHabilidade
));
        }
      } catch (SQLException e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.habilidade"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
      }

      //seta parametros para caso de sucesso(listarhabilidades.jsp)
      List<Habilidade> habilidades = new
ArrayList<Habilidade>();
      try{
        habilidades = habilidadesDao.listarHabilidades();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidades"));
      }
      if(habilidades.size()>0){
        request.setAttribute("habilidades", habilidades);
      }
      //setados parametros para caso de
sucesso(listarhabilidadesaluno.jsp)
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.186 ExcluirHabilidadeAlunoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.HabilidadeAluno;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirHabilidadeAlunoAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codHabilidadeAluno= "";
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      codHabilidadeAluno = (String)
request.getParameter("codHabilidadeAluno");
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    HabilidadeAluno habilidade = null;
    try {
      habilidade =
habilidadesAlunoDao.buscarHabilidadeAluno(Integer.valueOf(co
dHabilidadeAluno));
      if(habilidade != null){

habilidadesAlunoDao.excluirHabilidadeAluno(habilidade);
      }
    } catch (SQLException e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.habilidade"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
    }

    //seta parametros para caso de
sucesso(listarhabilidadesaluno.jsp)
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    Integer codAlunoEstagio = user.getCodAlunoEstagio();
    try{
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(codAlunoEstagio);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadealuno"));
    }
    if(habilidadesAluno.size()>0){
      request.setAttribute("habilidades", habilidadesAluno);
    }
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    //setados parametros para caso de
sucesso(listarhabilidadesaluno.jsp)
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.187 ExcluirHabilidadeVagaAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.HabilidadeVaga;
import estagio.dao.HabilidadesVagaDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
```

```
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirHabilidadeVagaAction extends Action{

   public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codHabilidadeVaga= "";
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        codHabilidadeVaga = (String)
request.getParameter("codHabilidadeVaga");
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
             new ActionMessage("error.generico"));
      }
      HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
      HabilidadeVaga habilidade = null;
      try {
        habilidade =
habilidadesVagaDao.buscarHabilidadeVaga(Integer.valueOf(cod
HabilidadeVaga));
        if(habilidade != null){

habilidadesVagaDao.excluirHabilidadeVaga(habilidade);
        }
      } catch (SQLException e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.requisito"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
      }
      String empresa = "";
      String cnpj = "";
      Empresa emp = new Empresa();
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      if(user.getStatus() != null){
        if("empresa".equals(user.getStatus())){
          try {
            emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
          }
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        }
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      HabilidadesVagaDao habilidadeVagaDao = new
HabilidadesVagaDao();
      List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
      try{
        habilidadesVaga =
habilidadeVagaDao.listarHabilidadeVaga(habilidade.getCodVaga
());
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadevaga"));
      }
      if(habilidadesVaga.size()>0){
        request.setAttribute("habilidades", habilidadesVaga);
      }
      request.setAttribute("codvaga", habilidade.getCodVaga());
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
   }
}
```

6.4.188 ExcluirTermoAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.TermoCompromisso;
import estagio.bean.TermoCompromissoDTO;
import estagio.bean.Turno;
import estagio.dao.TermoCompromissoDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ExcluirTermoAction extends Action{

   public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codTermoCompromisso = "";
```

```
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      codTermoCompromisso = (String)
request.getParameter("codTermo");
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    TermoCompromissoDao tDao = new
TermoCompromissoDao();
    TermoCompromisso termo = new TermoCompromisso();
    try {
      termo =
tDao.buscarTermoCompromisso(Integer.valueOf(codTermoComp
romisso));
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.termoCompromisso"));
    }
    try {
      tDao.excluirTermoCompromisso(termo);
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.termoCompromisso"));
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      String empresa = "";
      String cnpj = "";
      Empresa emp = new Empresa();
      if(user.getStatus() != null){
        if("empresa".equals(user.getStatus())){
          try {
            emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
          } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarioempresa"));
          }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarioempresa"));
          }
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        }
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      List<TermoCompromisso> termoCompromissos = new
ArrayList<TermoCompromisso>();
      try{
        termoCompromissos = tDao.listarTermos(cnpj);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.termocompromisso"));
      }
      List<TermoCompromissoDTO> tDtos = new
ArrayList<TermoCompromissoDTO>();
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      for(TermoCompromisso t:termoCompromissos){
        TermoCompromissoDTO tDto = new
TermoCompromissoDTO();
        tDto.setCodTermo(t.getCodTermo());

tDto.setEstagiario(t.getAlunoEstagio().getNomePessoa());

tDto.setDataInicio(sdf.format(t.getDataInicio().getTime()));
        tDto.setDataFim(sdf.format(t.getDataFim().getTime()));
        tDto.setEmpresa(t.getEmpresa().getRazaoSocial());

tDto.setAgenteIntegracao(t.getAgenteIntegracao()!=null?t.getAge
nteIntegracao().getAgente():"nenhum");
        tDto.setSalario(t.getVaga().getSalario());
        tDto.setSetor(t.getVaga().getSetor());

tDto.setCurso(t.getAlunoEstagio().getCurso().getNomeCurso());
        String turnos = "";
        for(int i = 0;i<t.getVaga().getTurnos().size();i++){
          Turno turno= t.getVaga().getTurnos().get(i);
          if( t.getVaga().getTurnos().size() - 1 != i){
            turnos += turno.toString()+", ";
          } else {
            turnos += turno.toString()+".";
          }
        }
        tDto.setHorario("".equals(turnos)?"Não
Informado":turnos);
        String descricaoHabilidade = "";
        for(int i = 0;i<t.getVaga().getHabilidades().size();i++){
          Habilidade habilidade =
t.getVaga().getHabilidades().get(i);
          if(t.getVaga().getHabilidades().size() - 1 != i){
            descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
          } else {
            descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
          }
        }

tDto.setRequisitos("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);

        tDtos.add(tDto);
      }
      if(termoCompromissos.size()>0){
        request.setAttribute("termos", tDtos);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
  }
}
```

6.4.189 ExcluirTermoCompromissoAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.TermoCompromisso;
```

```
import estagio.dao.TermoCompromissoDao;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ExcluirTermoCompromissoAction extends Action{

   public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codTermoCompromisso = "";
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        codTermoCompromisso = (String)
request.getParameter("codTermo");
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      TermoCompromissoDao tDao = new
TermoCompromissoDao();
      TermoCompromisso termo = new TermoCompromisso();
      try {
        termo =
tDao.buscarTermoCompromisso(Integer.valueOf(codTermoComp
romisso));
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.termoCompromisso"));
      }
      try {
        tDao.excluirTermoCompromisso(termo);
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.termoCompromisso"));
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
   }
}
```

6.4.190 ExcluirTurnoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Turno;
import estagio.bean.TurnoDTO;
import estagio.dao.TurnoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirTurnoAction extends Action{
      public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      String codTurno= "";
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        codTurno = (String) request.getParameter("codTurno");
        if (user == null){
          return mapping.findForward("logon");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      TurnoDao turnoDao = new TurnoDao();
      Turno turno = new Turno();
      try {
        turno =
turnoDao.buscarTurno(Integer.valueOf(codTurno));
        if(turno != null){
          turnoDao.excluirTurno(turno.getCodTurno());
        }
      } catch (SQLException e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.turno"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
      }
      //parametros para listar turnos
      List<Turno> listaTurnos = new ArrayList<Turno>();
      List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
      try{
        listaTurnos =
turnoDao.listarTurnos(Integer.valueOf(turno.getCodVaga()));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
      }
      String turnos = "";
      for(int i = 0;i < listaTurnos.size();i++){
        Turno t = listaTurnos.get(i);
        TurnoDTO turnoDto = new TurnoDTO();
        turnoDto.setCodTurno(t.getCodTurno());
        turnoDto.setTurno(t.toString());
        turnosDTO.add(turnoDto);
        if( listaTurnos.size() - 1 != i){
          turnos += turno.toString()+", ";
        } else {
          turnos += turno.toString()+".";
        }
      }
```

```
    if(listaTurnos.size()>0){
      request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      request.setAttribute("turnosDto", turnosDTO);
    }
    request.setAttribute("codvaga", turno.getCodVaga());
    //fim dos parametros para listar turnos
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.191 ExcluirTurnoDisponivelAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.TurnoDTO;
import estagio.bean.TurnoDisponivel;
import estagio.dao.TurnoDisponivelDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ExcluirTurnoDisponivelAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codTurno= "";
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      codTurno = (String) request.getParameter("codTurno");
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
    TurnoDisponivel turnoDisponivel = new TurnoDisponivel();
    try {
      turnoDisponivel =
turnoDisponivelDao.buscarTurnoDisponivel(Integer.valueOf(cod
Turno));
      if(turnoDisponivel != null){

turnoDisponivelDao.excluirTurnoDisponivel(turnoDisponivel.get
CodTurnoDisponivel());
      }
    } catch (SQLException e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.turno"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
    }

    //parametros para listar turnos
    List<TurnoDisponivel> turnosDisponiveis = new
ArrayList<TurnoDisponivel>();
    List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
    try{
      turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(user.getCodAlunoE
stagio());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    String turnos = "";
    for(int i = 0;i < turnosDisponiveis.size();i++){
      TurnoDisponivel turno = turnosDisponiveis.get(i);
      TurnoDTO turnoDto = new TurnoDTO();
      turnoDto.setCodTurno(turno.getCodTurnoDisponivel());
      turnoDto.setTurno(turno.toString());
      turnosDTO.add(turnoDto);
      if( turnosDisponiveis.size() - 1 != i){
        turnos += turno.toString()+", ";
      } else {
        turnos += turno.toString()+".";
      }
    }
    if(turnosDisponiveis.size()>0){
      request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      request.setAttribute("turnosDto", turnosDTO);
    }
    //fim dos parametros para listar turnos
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.192 ExcluirVagaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.Turno;
import estagio.bean.Vaga;
import estagio.bean.VagaDTO;
import estagio.dao.EmpresaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
```

```
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ExcluirVagaAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codVaga= "";
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      codVaga = (String) request.getParameter("codvaga");
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    VagaDao vagaDao = new VagaDao();
    Vaga v = null;
    try {
      v = vagaDao.buscarVaga(Integer.valueOf(codVaga));
      if(v != null){
        vagaDao.excluirVaga(Integer.valueOf(codVaga));
      }
    } catch (SQLException e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.delete.vaga"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.generico"));
    }
    String cnpj = v.getCnpj();
    Empresa emp = new Empresa();
    try{
      EmpresaDao empresaDao = new EmpresaDao();
      emp = empresaDao.buscarEmpresa(cnpj);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
    }
    String empresa = "";
    if(emp != null){
      empresa = emp.getRazaoSocial();
    }
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    if(user.getStatus() != null){
      if("empresa".equals(user.getStatus())){
        try {
          emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
        }
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }
    List<Vaga> vagas = new ArrayList<Vaga>();
    try{
      vagas = vagaDao.listarVagasAbertas(cnpj);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.vaga"));
    }
    List<VagaDTO> vagaDtos = new ArrayList<VagaDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(Vaga vaga:vagas){
      VagaDTO vagaDto = new VagaDTO();
      vagaDto.setCodVaga(vaga.getCodVaga());
      vagaDto.setHoras(vaga.getHoras().toString());

vagaDto.setInclusao(sdf.format(vaga.getInclusao().getTime()));
      vagaDto.setSalario(vaga.getSalario());
      vagaDto.setSetor(vaga.getSetor());
      char sexo = vaga.getSexo().charAt(0);
      switch (sexo){
        case 'M':
          vagaDto.setSexo("Masculino");
          break;
        case 'F':
          vagaDto.setSexo("Feminino");
          break;
        case 'A':
          vagaDto.setSexo("Qualquer");
          break;
        default:
          vagaDto.setSexo("");
      }
      String turnos = "";
      for(int i = 0;i<vaga.getTurnos().size();i++){
        Turno turno= vaga.getTurnos().get(i);
        if( vaga.getTurnos().size() - 1 != i){
          turnos += turno.toString()+", ";
        } else {
          turnos += turno.toString()+".";
        }
      }
      vagaDto.setTurnos("".equals(turnos)?"Não
Informado":turnos);
      String descricaoHabilidade = "";
      for(int i = 0;i<vaga.getHabilidades().size();i++){
        Habilidade habilidade = vaga.getHabilidades().get(i);
        if(vaga.getHabilidades().size() - 1 != i){
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
        } else {
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
        }
      }

vagaDto.setHabilidades("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
      String cursos = "";
      if(vaga.getCursos().isEmpty()){
        cursos += "Nenhum";
```

```
        } else {
          for(int i = 0;i<vaga.getCursos().size();i++){
            Curso curso = vaga.getCursos().get(i);
            if(vaga.getCursos().size() - 1 != i){
              cursos += curso.getNomeCurso()+", ";
            } else {
              cursos += curso.getNomeCurso()+".";
            }
          }
        }
        vagaDto.setCursos(cursos);
        vagaDto.setSemestre(v.getSemestre());

vagaDto.setValeTransporte(v.getValeTransporte().equals("S")?"S
im":"Não");
        vagaDto.setOutrosBeneficios(v.getOutrosBeneficios());
        vagaDtos.add(vagaDto);
      }
      if(vagas.size()>0){
        request.setAttribute("vagas", vagaDtos);
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);

      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.193 ListarAgentesIntegraçãoAction.java

```
package estagio.action;
import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.dao.AgentesIntegracaoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarAgentesIntegracaoAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    AgentesIntegracaoDao agentesIntegracaoDao = new
AgentesIntegracaoDao();
    List<AgenteIntegracao> agentesIntegracao = new
ArrayList<AgenteIntegracao>();
    try{
      agentesIntegracao =
agentesIntegracaoDao.listarAgentesIntegracao();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.agente"));
    }
    if(agentesIntegracao.size()>0){
      request.setAttribute("agentes", agentesIntegracao);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.194 ListarConveniosAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.ConvenioDTO;
import estagio.dao.ConvenioDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ListarConveniosAction extends Action{
    public ActionForward execute(
     ActionMapping mapping,
     ActionForm form,
     HttpServletRequest request,
     HttpServletResponse response)
     throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
      new ActionMessage("error.generico"));
    }
    ConvenioDao convenioDao = new ConvenioDao();
    List<Convenio> convenios = new ArrayList<Convenio>();
```

```
    try{
      convenios = convenioDao.listarConvenios();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
    }
    List<ConvenioDTO> cDtos = new
ArrayList<ConvenioDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    GregorianCalendar hoje = new GregorianCalendar();
    for(Convenio c:convenios){
      ConvenioDTO cDto = new ConvenioDTO();
      cDto.setCodConvenio(c.getCodConvenio());
      cDto.setDataFim(sdf.format(c.getDataFim().getTime()));

cDto.setDataInicio(sdf.format(c.getDataInicio().getTime()));
      cDto.setEmpresa(c.getEmpresa().getRazaoSocial());
      cDto.setDescricao(c.getDescricao());
      if(hoje.after(c.getDataFim())){
        cDto.setSituacao("Expirado.");
      } else {
        cDto.setSituacao("Válido.");
      }
      cDtos.add(cDto);
    }
    if(convenios.size()>0){
      request.setAttribute("convenios", cDtos);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.195 ListarCursosAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.dao.CursoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ListarCursosAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    CursoDao cursoDao = new CursoDao();
    List<Curso> cursos = new ArrayList<Curso>();
    try{
      cursos = cursoDao.listarCursos();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
    }
    if(cursos.size()>0){
      request.setAttribute("cursos", cursos);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.196 ListarEmpresasAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.Empresa;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarEmpresasAction extends Action{

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
```

```
            new ActionMessage("error.generico"));
      }
      EmpresaDao empresaDao = new EmpresaDao();
      List<Empresa> empresas = new ArrayList<Empresa>();
      try{
        empresas = empresaDao.listarEmpresas();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.empresa"));
      }
      ConvenioDao convenioDao = new ConvenioDao();
      List<Convenio> convenios = new ArrayList<Convenio>();
      try{
        convenios = convenioDao.listarConvenios();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
      }
      for(Empresa emp:empresas){
        emp.setCargoRepresentante("Não possui.");
        for(Convenio c: convenios){
          if(c.getEmpresa().getCnpj().equals(emp.getCnpj())){
            GregorianCalendar hoje = new GregorianCalendar();
            if(c.getDataFim().after(hoje)){

emp.setCargoRepresentante(c.getCodConvenio().toString());
            } else {
              emp.setCargoRepresentante("Expirado");
            }
          }
        }
      }
      if(empresas.size()>0){
        request.setAttribute("empresas", empresas);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.197 ListarHabilidadesAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.dao.HabilidadesDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ListarHabilidadesAction extends Action{

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      HabilidadesDao habilidadesDao = new HabilidadesDao();
      List<Habilidade> habilidades = new
ArrayList<Habilidade>();
      try{
        habilidades = habilidadesDao.listarHabilidades();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidades"));
      }
      if(habilidades.size()>0){
        request.setAttribute("habilidades", habilidades);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.198 ListarHabilidadesAlunoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.HabilidadeAluno;
import estagio.dao.HabilidadesAlunoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class ListarHabilidadesAlunoAction extends Action{
      public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
```

```
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    Integer codAlunoEstagio = user.getCodAlunoEstagio();
    try{
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(codAlunoEstagio);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadealuno"));
    }
    if(habilidadesAluno.size()>0){
      request.setAttribute("habilidades", habilidadesAluno);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.199 ListarHabilidadesVagaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeVaga;
import estagio.bean.Turno;
import estagio.dao.HabilidadesDao;
import estagio.dao.HabilidadesVagaDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarHabilidadesVagaAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      String codVaga = "";
      try{
        codVaga = request.getParameter("codvaga");
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      String empresa = "";
      String cnpj = "";
      Empresa emp = new Empresa();
      if(user.getStatus() != null){
        if("empresa".equals(user.getStatus())){
          try {
            emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
          }
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        }
      }
      HabilidadesVagaDao habilidadeVagaDao = new
HabilidadesVagaDao();
      List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
      try{
        habilidadesVaga =
habilidadeVagaDao.listarHabilidadeVaga(Integer.valueOf(codVa
ga));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadevaga"));
      }
      if(habilidadesVaga.size()>0){
        request.setAttribute("habilidades", habilidadesVaga);
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      request.setAttribute("codvaga", codVaga);
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.200 ListarTermosAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
```

```
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.TermoCompromisso;
import estagio.bean.TermoCompromissoDTO;
import estagio.bean.Turno;
import estagio.dao.TermoCompromissoDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;


public class ListarTermosAction  extends Action{
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    String empresa = "";
    String cnpj = "";
    Empresa emp = new Empresa();
    if(user.getStatus() != null){
      if("empresa".equals(user.getStatus())){
        try {
          emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
        }
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
    List<TermoCompromisso> termoCompromissos = new
ArrayList<TermoCompromisso>();
    try{
      termoCompromissos =
termoCompromissoDao.listarTermos(cnpj);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.termocompromisso"));
    }
    List<TermoCompromissoDTO> tDtos = new
ArrayList<TermoCompromissoDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(TermoCompromisso t:termoCompromissos){
      TermoCompromissoDTO tDto = new
TermoCompromissoDTO();
      tDto.setCodTermo(t.getCodTermo());
      tDto.setEstagiario(t.getAlunoEstagio().getNomePessoa());

tDto.setDataInicio(sdf.format(t.getDataInicio().getTime()));
      tDto.setDataFim(sdf.format(t.getDataFim().getTime()));
      tDto.setEmpresa(t.getEmpresa().getRazaoSocial());
tDto.setAgenteIntegracao(t.getAgenteIntegracao()!=null?t.getAge
nteIntegracao().getAgente():"nenhum");
      tDto.setSalario(t.getVaga().getSalario());
      tDto.setSetor(t.getVaga().getSetor());
tDto.setCurso(t.getAlunoEstagio().getCurso().getNomeCurso());
      String turnos = "";
      for(int i = 0;i<t.getVaga().getTurnos().size();i++){
        Turno turno= t.getVaga().getTurnos().get(i);
        if( t.getVaga().getTurnos().size() - 1 != i){
          turnos += turno.toString()+", ";
        } else {
          turnos += turno.toString()+".";
        }
      }
      tDto.setHorario("".equals(turnos)?"Não
Informado":turnos);
      String descricaoHabilidade = "";
      for(int i = 0;i<t.getVaga().getHabilidades().size();i++){
        Habilidade habilidade =
t.getVaga().getHabilidades().get(i);
        if(t.getVaga().getHabilidades().size() - 1 != i){
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
        } else {
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
        }
      }
      tDto.setRequisitos("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);

tDto.setNomeDocumento("TermoCompromisso"+t.getCodTermo
()+".pdf");
      tDtos.add(tDto);
    }
    if(termoCompromissos.size()>0){
      request.setAttribute("termos", tDtos);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.201 ListarTermosCompromissoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.bean.TermoCompromisso;
import estagio.bean.TermoCompromissoDTO;
import estagio.bean.Turno;
import estagio.dao.TermoCompromissoDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarTermosCompromissoAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
    List<TermoCompromisso> termoCompromissos = new
ArrayList<TermoCompromisso>();
    try{
      termoCompromissos =
termoCompromissoDao.listarTermosCompromisso();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.termocompromisso"));
    }
    List<TermoCompromissoDTO> tDtos = new
ArrayList<TermoCompromissoDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(TermoCompromisso t:termoCompromissos){
      TermoCompromissoDTO tDto = new
TermoCompromissoDTO();
      tDto.setEmpresa(t.getEmpresa().getRazaoSocial());

tDto.setAgenteIntegracao(t.getAgenteIntegracao()!=null?t.getAge
nteIntegracao().getAgente():"nenhum");
      tDto.setCodTermo(t.getCodTermo());
      tDto.setEstagiario(t.getAlunoEstagio().getNomePessoa());
      tDto.setDataInicio(sdf.format(t.getDataInicio().getTime()));
      tDto.setDataFim(sdf.format(t.getDataFim().getTime()));
      tDto.setSalario(t.getVaga().getSalario());
      tDto.setSetor(t.getVaga().getSetor());

tDto.setCurso(t.getAlunoEstagio().getCurso().getNomeCurso());
      String turnos = "";
      for(int i = 0;i<t.getVaga().getTurnos().size();i++){
        Turno turno= t.getVaga().getTurnos().get(i);
        if( t.getVaga().getTurnos().size() - 1 != i){
          turnos += turno.toString()+", ";
        } else {
          turnos += turno.toString()+".";
        }
      }
      tDto.setHorario("".equals(turnos)?"Não
Informado":turnos);
      String descricaoHabilidade = "";
      for(int i = 0;i<t.getVaga().getHabilidades().size();i++){
        Habilidade habilidade =
t.getVaga().getHabilidades().get(i);
        if(t.getVaga().getHabilidades().size() - 1 != i){
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
        } else {
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
        }
      }
      tDto.setRequisitos("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
      tDtos.add(tDto);
    }
    if(termoCompromissos.size()>0){
      request.setAttribute("termos", tDtos);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.202 ListarTurnosAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Turno;
import estagio.bean.TurnoDTO;
import estagio.dao.TurnoDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarTurnosAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
```

```
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    String codVaga = (String) request.getParameter("codvaga");
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    //parametros para listar turnos
    TurnoDao turnoDao = new TurnoDao();
    List<Turno> listaTurnos = new ArrayList<Turno>();
    List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
    try{
      listaTurnos =
turnoDao.listarTurnos(Integer.valueOf(codVaga));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    String turnos = "";
    for(int i = 0;i < listaTurnos.size();i++){
      Turno turno = listaTurnos.get(i);
      TurnoDTO turnoDto = new TurnoDTO();
      turnoDto.setCodTurno(turno.getCodTurno());
      turnoDto.setTurno(turno.toString());
      turnosDTO.add(turnoDto);
      if( listaTurnos.size() - 1 != i){
        turnos += turno.toString()+", ";
      } else {
        turnos += turno.toString()+".";
      }
    }
    if(listaTurnos.size()>0){
      request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      request.setAttribute("turnosDto", turnosDTO);
    }
    //fim dos parametros para listar turnos
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      request.setAttribute("codvaga", codVaga);
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.203 ListarTurnosDisponiveisAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Turno;
import estagio.bean.TurnoDTO;
import estagio.bean.TurnoDisponivel;
import estagio.dao.TurnoDisponivelDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarTurnosDisponiveisAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    String codAlunoEstagio = (String)
request.getParameter("codalunoestagio");
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    //parametros para listar turnos
    TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
    List<TurnoDisponivel> turnosDisponiveis = new
ArrayList<TurnoDisponivel>();
    List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
    try{
      turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    String turnos = "";
    for(int i = 0;i < turnosDisponiveis.size();i++){
      TurnoDisponivel turno = turnosDisponiveis.get(i);
      TurnoDTO turnoDto = new TurnoDTO();
      turnoDto.setCodTurno(turno.getCodTurnoDisponivel());
      turnoDto.setTurno(turno.toString());
      turnosDTO.add(turnoDto);
      if( turnosDisponiveis.size() - 1 != i){
        turnos += turno.toString()+", ";
      } else {
        turnos += turno.toString()+".";
      }
    }
    if(turnosDisponiveis.size()>0){
      request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      request.setAttribute("turnosDto", turnosDTO);
    }
    //fim dos parametros para listar turnos
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
```

```
    }
  }
}
```

6.4.204 ListarVagasAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.Turno;
import estagio.bean.Vaga;
import estagio.bean.VagaDTO;
import estagio.dao.EmpresaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class ListarVagasAction extends Action{
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
            return mapping.findForward("welcome");
        }
    } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
    }
    String empresa = "";
    String cnpj = "";
    Empresa emp = new Empresa();
    if(user.getStatus() != null){
        if("empresa".equals(user.getStatus())){
            try {
                emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
            } catch (SQLException e){
                errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
            }catch (Exception e) {
                errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
            }
            if(emp != null){
                empresa = emp.getRazaoSocial();
                cnpj = emp.getCnpj();
            }
        }
        String estagio = request.getParameter("estagio");
        VagaDao vagaDao = new VagaDao();
        List<Vaga> vagas = new ArrayList<Vaga>();
        try{
            if("interno".equals(estagio)){
                vagas = vagaDao.listarVagasAbertasEstagioInterno();
                if(!vagas.isEmpty()){
                    cnpj = vagas.get(0).getCnpj();
                    EmpresaDao empresaDao = new EmpresaDao();
                    emp = empresaDao.buscarEmpresa(cnpj);
                    if(emp != null){
                        empresa = emp.getRazaoSocial();
                    }
                }
            } else {
                vagas = vagaDao.listarVagasAbertas(cnpj);
            }
        } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.vaga"));
        }
        List<VagaDTO> vagaDtos = new ArrayList<VagaDTO>();
        SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
        for(Vaga vaga:vagas){
            VagaDTO vagaDto = new VagaDTO();
            vagaDto.setCodVaga(vaga.getCodVaga());
            vagaDto.setHoras(vaga.getHoras().toString());

vagaDto.setInclusao(sdf.format(vaga.getInclusao().getTime()));
            vagaDto.setSalario(vaga.getSalario());
            vagaDto.setSetor(vaga.getSetor());
            char sexo = vaga.getSexo().charAt(0);
            switch (sexo){
                case 'M':
                    vagaDto.setSexo("Masculino");
                    break;
                case 'F':
                    vagaDto.setSexo("Feminino");
                    break;
                case 'A':
                    vagaDto.setSexo("Qualquer");
                    break;
                default:
                    vagaDto.setSexo("");
            }
            String turnos = "";
            for(int i = 0;i<vaga.getTurnos().size();i++){
                Turno turno= vaga.getTurnos().get(i);
                if( vaga.getTurnos().size() - 1 != i){
                    turnos += turno.toString()+", ";
                } else {
                    turnos += turno.toString()+".";
                }
            }
            vagaDto.setTurnos("".equals(turnos)?"Não
Informado":turnos);
            String descricaoHabilidade = "";
            for(int i = 0;i<vaga.getHabilidades().size();i++){
                Habilidade habilidade = vaga.getHabilidades().get(i);
                if(vaga.getHabilidades().size() - 1 != i){
                    descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
                } else {
                    descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
                }
```

```
        }
vagaDto.setHabilidades("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
        String cursos = "";
        if(vaga.getCursos().isEmpty()){
          cursos += "Nenhum";
        } else {
          for(int i = 0;i<vaga.getCursos().size();i++){
            Curso curso = vaga.getCursos().get(i);
            if(vaga.getCursos().size() - 1 != i){
              cursos += curso.getNomeCurso()+", ";
            } else {
              cursos += curso.getNomeCurso()+".";
            }
          }
        }
        vagaDto.setCursos(cursos);
        vagaDtos.add(vagaDto);
      }
      if(vagas.size()>0){
        request.setAttribute("vagas", vagaDtos);
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.205 LogoffAction.java

```
package estagio.action;
import estagio.bean.Constants;
import estagio.bean.Pessoa;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

public class LogoffAction extends Action{
    public ActionForward execute(
     ActionMapping mapping,
     ActionForm form,
     HttpServletRequest request,
     HttpServletResponse response)
     throws Exception {
    HttpSession session = request.getSession();
    Pessoa user =
(Pessoa)session.getAttribute(Constants.USER_KEY);
    if(user != null){
      session.removeAttribute("user");
      session.removeAttribute("cnpj");
            session.removeAttribute("logado");
      session.removeAttribute("codPessoa");
      Cookie logado = new Cookie("logado", "false");
      // we want the cookie to be returned to any app below root
      logado.setPath("/");
      response.addCookie(logado);
    }
    return (mapping.findForward(Constants.SUCCESS));
  }
}
```

6.4.206 LogonAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Pessoa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.PessoaDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.commons.beanutils.PropertyUtils;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class LogonAction extends Action{
  /** Creates a new instance of LogonAction */
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    String userName = null;
    String password = null;
    userName = (String)
PropertyUtils.getSimpleProperty(form,"username");
    password = (String)
PropertyUtils.getSimpleProperty(form,"password");
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    AlunoEstagio user = new AlunoEstagio();
    try {
      user = alunoEstagioDao.buscarAlunoEstagio(userName);
    } catch (SQLException e){
      errors.add(
          ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.recuperar.dados"));
    } catch (Exception e){
      errors.add(
          ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    if (user != null) {

if(!String.valueOf(password.hashCode()).equals(user.getSenha()))
{
        errors.add(
            ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.password.incorreto"));
        }
    } else {
      errors.add(
          ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.usuario.naoexiste"));
    }
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward("error"));
    }else {
      HttpSession session = request.getSession();
      if(session.getAttribute(Constants.USER_KEY) != null){
        session.removeAttribute(Constants.USER_KEY);
      }
      session.setAttribute(Constants.USER_KEY, user);
      Cookie codPessoaCookie = new Cookie("codPessoa",
user.getCodPessoa().toString());
```

```
        Cookie logado = new Cookie("logado", "true");
        // we want the cookie to be returned to any app below root
        codPessoaCookie.setPath("/");
        logado.setPath("/");
        response.addCookie(codPessoaCookie);
        response.addCookie(logado);
        if("empresa".equalsIgnoreCase(user.getStatus())){
          String cnpj = "";
          Empresa emp = null;
            try{
              UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
              emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
            } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarioempresa"));
            }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarioempresa"));
            }
            if(emp != null){
              cnpj = emp.getCnpj();
            }
          session.setAttribute("cnpj", cnpj);
        }
        return (mapping.findForward("success"));
    }
  }
}
```

6.4.207 NovaHabilidadeAlunoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.HabilidadesDao;
import estagio.form.NovaHabilidadeAlunoForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class NovaHabilidadeAlunoAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
        ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    String codAlunoEstagio =
request.getParameter("codalunoestagio");
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    List<Habilidade> todasHabilidades = new
ArrayList<Habilidade>();
    List<Habilidade> habilidades = new
ArrayList<Habilidade>();
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    try{
      todasHabilidades = habilidadesDao.listarHabilidades();
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidade"));
    }
    for(Habilidade h: todasHabilidades){
      habilidades.add(h);
      for(HabilidadeAluno hAluno: habilidadesAluno){
        if(hAluno.getCodHabilidade().intValue() ==
h.getCodHabilidade().intValue()){
          habilidades.remove(h);
        }
      }
    }
    /*Integer[] codHabilidades = new
Integer[habilidadesAluno.size()];
    for(int i = 0; i < habilidadesAluno.size();i++){
      HabilidadeAluno h = habilidadesAluno.get(i);
      codHabilidades[i] = h.getCodHabilidade();
    }
    NovaHabilidadeAlunoForm habilidadeAlunoForm =
(NovaHabilidadeAlunoForm) form;
    habilidadeAlunoForm.setCodHabilidade(codHabilidades);*/
    if(habilidades.size() > 0){
      request.setAttribute("todashabilidades", habilidades);
    }
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.208 NovaHabilidadeAlunoGeralAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
```

```
public class NovaHabilidadeAlunoGeralAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codAlunoEstagio =
request.getParameter("codalunoestagio");
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.209 NovaHabilidadeVagaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeVaga;
import estagio.dao.HabilidadesDao;
import estagio.dao.HabilidadesVagaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.form.NovaHabilidadeVagaForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

public class NovaHabilidadeVagaAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    String codVaga = request.getParameter("codvaga");
    request.setAttribute("codvaga", codVaga);
    List<Habilidade> todasHabilidades = new
ArrayList<Habilidade>();
    List<Habilidade> habilidades = new
ArrayList<Habilidade>();
    List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
    HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
    try{
      todasHabilidades = habilidadesDao.listarHabilidades();
      habilidadesVaga =
habilidadesVagaDao.listarHabilidadeVaga(Integer.valueOf(codV
aga));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidade"));
    }
    for(Habilidade h: todasHabilidades){
      habilidades.add(h);
      for(HabilidadeVaga hVaga: habilidadesVaga){
        if(hVaga.getCodHabilidade().intValue() ==
h.getCodHabilidade().intValue()){
          habilidades.remove(h);
        }
      }
    }
    if(habilidades.size() > 0){
      request.setAttribute("todashabilidades", habilidades);
    }
    AlunoEstagio user = null;
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    String empresa = "";
    String cnpj = "";
    Empresa emp = new Empresa();
    if(user.getStatus() != null){
      if("empresa".equals(user.getStatus())){
        try {
          emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
        }
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
```

```
    }
  }
}
```

6.4.210 NovaHabilidadeVagaGeralAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class NovaHabilidadeVagaGeralAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {

    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codVaga = request.getParameter("codvaga");
    request.setAttribute("codvaga", codVaga);
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    String empresa = "";
    String cnpj = "";
    Empresa emp = new Empresa();
    if(user.getStatus() != null){
      if("empresa".equals(user.getStatus())){
        try {
          emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
        }
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.211 NovaVagaAction.java

```
package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

public class NovaVagaAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    EmpresaDao empresaDao = new EmpresaDao();
    ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
    options.add(new LabelValueBean("Masculino", "M"));
    options.add(new LabelValueBean("Feminino", "F"));
    options.add(new LabelValueBean("Qualquer", "A"));
    request.setAttribute("sexos", options);
    ArrayList simNaoAll = new ArrayList();
    simNaoAll.add(new LabelValueBean("Sim", "S"));
    simNaoAll.add(new LabelValueBean("Não", "N"));
    simNaoAll.add(new LabelValueBean("Não Importa", "A"));
    request.setAttribute("simNaoAll", simNaoAll);
    ArrayList simNao = new ArrayList();
    simNao.add(new LabelValueBean("Sim", "S"));
    simNao.add(new LabelValueBean("Não", "N"));
    request.setAttribute("simNao", simNao);
     ArrayList locomotion = new ArrayList();
    locomotion.add(new LabelValueBean("Carro", "C"));
    locomotion.add(new LabelValueBean("Moto", "M"));
    locomotion.add(new LabelValueBean("Não Importa", "A"));
    request.setAttribute("locomocao", locomotion);
    AlunoEstagio user = null;
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
```

```
            errors.add(ActionMessages.GLOBAL_MESSAGE,
                new ActionMessage("error.generico"));
        }
        String empresa = "";
        String cnpj = "";
        Empresa emp = new Empresa();
        if(user.getStatus() != null){
          if("empresa".equals(user.getStatus())){
            try {
              emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
            } catch (SQLException e){
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
            }catch (Exception e) {
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
            }
            if(emp != null){
              empresa = emp.getRazaoSocial();
              cnpj = emp.getCnpj();
            }
          }
        }
        if("".equals(cnpj)){
          return mapping.findForward("buscarcnpj");
        }
        request.setAttribute("empresa", empresa);
        request.setAttribute("cnpj", cnpj);
        //lista os cursos
        CursoDao cursoDao = new CursoDao();
        List<Curso> cursos = new ArrayList<Curso>();
        try{
          cursos = cursoDao.listarCursos();
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
        }
        if(cursos.size()>0){
          request.setAttribute("cursos", cursos);
        }
        //cursos listados
        if(!errors.isEmpty()){
          saveErrors(request, errors);
          return mapping.findForward(Constants.FAILURE);
        } else {
          return mapping.findForward(Constants.SUCCESS);
        }
    }
}
```

6.4.212 NovaVagaEstagioInternoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
public class NovaVagaEstagioInternoAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    EmpresaDao empresaDao = new EmpresaDao();
    ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
    options.add(new LabelValueBean("Masculino", "M"));
    options.add(new LabelValueBean("Feminino", "F"));
    options.add(new LabelValueBean("Qualquer", "A"));
    request.setAttribute("sexos", options);
    ArrayList simNaoAll = new ArrayList();
    simNaoAll.add(new LabelValueBean("Sim", "S"));
    simNaoAll.add(new LabelValueBean("Não", "N"));
    simNaoAll.add(new LabelValueBean("Não Importa", "A"));
    request.setAttribute("simNaoAll", simNaoAll);
    ArrayList simNao = new ArrayList();
    simNao.add(new LabelValueBean("Sim", "S"));
    simNao.add(new LabelValueBean("Não", "N"));
    request.setAttribute("simNao", simNao);

    ArrayList locomotion = new ArrayList();
    locomotion.add(new LabelValueBean("Carro", "C"));
    locomotion.add(new LabelValueBean("Moto", "M"));
    locomotion.add(new LabelValueBean("Não Importa", "A"));
    request.setAttribute("locomocao", locomotion);
    request.setAttribute("estagio", "interno");
    AlunoEstagio user = null;
    //UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    String empresa = "";
    String cnpj = request.getParameter("cnpj");
    Empresa emp = new Empresa();
    try {
      emp = empresaDao.buscarEmpresa(cnpj);
      if(emp != null){
        empresa = emp.getRazaoSocial();
      }
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
    }
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    //lista os cursos
    CursoDao cursoDao = new CursoDao();
    List<Curso> cursos = new ArrayList<Curso>();
    try{
```

```
            cursos = cursoDao.listarCursos();
        } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
        }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
        }
        if(cursos.size()>0){
            request.setAttribute("cursos", cursos);
            //request.setAttribute("qtdcursos", cursos.size()/3);
        }
        //cursos listados
        if(!errors.isEmpty()){
            saveErrors(request, errors);
            return mapping.findForward(Constants.FAILURE);
        } else {
            return mapping.findForward(Constants.SUCCESS);
        }
    }
}
```

6.4.213 NovoConvenioAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Pessoa;

import estagio.dao.EmpresaDao;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
public class NovoConvenioAction extends Action {
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    EmpresaDao empresaDao = new EmpresaDao();
    String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
    String cnpj = (String) request.getParameter("cnpj");
    List<Empresa> empresas = new ArrayList<Empresa>();
    if(cnpj != null && !"".equals(cnpj)){
        empresas.add(empresaDao.buscarEmpresa(cnpj));
        ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
        for(Empresa e: empresas){
            options.add(new LabelValueBean(e.getRazaoSocial(),
e.getCnpj()));
        }
        request.setAttribute("empresas", options);
    }
    request.setAttribute("estados", estados);
    try{
        AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
            return mapping.findForward("logon");
        }
    } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
    }

    if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
    } else {
        return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.214 NovoTermoCompromissoAction.java

```
package estagio.action;
import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Vaga;
import estagio.dao.AgentesIntegracaoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.VagaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
public class NovoTermoCompromissoAction extends Action {
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {

    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    EmpresaDao empresaDao = new EmpresaDao();

    if (user == null) {
        return (mapping.findForward("logon"));
    }
    String codVaga = request.getParameter("codVaga");
    String codAlunoEstagio =
request.getParameter("codAlunoEstagio");
    request.setAttribute("codvaga", codVaga);
    request.setAttribute("codAlunoEstagio", codAlunoEstagio);
    //UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    String empresa = "";
    String cnpj = "";
    Empresa emp = new Empresa();
    Vaga vaga = new Vaga();
```

```
    try {
      VagaDao vagaDao = new VagaDao();
      vaga = vagaDao.buscarVaga(Integer.valueOf(codVaga));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.vaga"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.vaga"));
    }
    cnpj = vaga.getCnpj();
    String agenteInt = request.getParameter("agente");
    if(agenteInt != null && agenteInt.equals("true")){
      request.setAttribute("agente",  agenteInt);
    }
    if(Character.valueOf('S').equals(vaga.getEstagioInterno())){
      request.setAttribute("estagio",  "interno");
    }
    try{
      empresaDao = new EmpresaDao();
      emp = empresaDao.buscarEmpresa(cnpj);
      if(emp != null){
        empresa = emp.getRazaoSocial();
      }

    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
    }
    /*if(user.getStatus() != null){
      if("empresa".equals(user.getStatus())){
        emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }*/
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);

    AgentesIntegracaoDao agentesIntegracaoDao = new
AgentesIntegracaoDao();
    List<LabelValueBean> selectAgentes = new
ArrayList<LabelValueBean>();
    List<AgenteIntegracao> agentes = new
ArrayList<AgenteIntegracao>();
    try{
      agentes =
agentesIntegracaoDao.listarAgentesIntegracao();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.agente"));
    }
    for(AgenteIntegracao agente: agentes){
      selectAgentes.add(new
LabelValueBean(agente.getAgente(), agente.getCnpj()));
    }
    if(selectAgentes.size()>0){
      request.setAttribute("selectAgentes", selectAgentes);
    } else {
      request.setAttribute("selectAgentes", new
LabelValueBean("Nao informado", "0"));
    }
    Map parametros = new HashMap();
    parametros.put("codVaga", codVaga);
    parametros.put("codAlunoEstagio", codAlunoEstagio);
    request.setAttribute("parametros", parametros);
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.215 NovoTurnoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class NovoTurnoAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codAlunoEstagio = request.getParameter("codvaga");
    request.setAttribute("codvaga", codAlunoEstagio);
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.216 NovoTurnoDisponivelAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class NovoTurnoDisponivelAction extends Action {
```

```
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    String codAlunoEstagio =
request.getParameter("codalunoestagio");
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.217 NovoUsuarioAction.java

```
package estagio.action;
import estagio.bean.Constants;
import estagio.bean.Pessoa;
import estagio.bean.Usuario;
import estagio.form.TipoUsuarioForm;
import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessages;
public class NovoUsuarioAction extends Action {

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    HttpSession session = request.getSession();
    TipoUsuarioForm tipoform = (TipoUsuarioForm) form;
    Pessoa userCheck = (Pessoa)
session.getAttribute(Constants.USER_KEY);
    String action = request.getParameter("action");
    if (userCheck == null) {
      return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    //Usuarios
    ArrayList<Usuario> usuarios = new ArrayList<Usuario>();
    Usuario admin = new Usuario();
    admin.setTipo("admin");
    admin.setMensagem("Membro do grupo de
Administradores");
    Usuario empresa = new Usuario();
    empresa.setTipo("empresa");
    empresa.setMensagem("Membro de Empresa");
    Usuario aluno = new Usuario();
    aluno.setTipo("aluno");
    aluno.setMensagem("Aluno");
    usuarios.add(admin);
    usuarios.add(empresa);
    usuarios.add(aluno);
    request.setAttribute("usuarios", usuarios);
    //fim de Usuarios
    if (errors.isEmpty()){
      return mapping.findForward("success");
    } else {
      return mapping.findForward("error");
    } } }
```

6.4.218 PesquisarConveniosAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.ConvenioDTO;
import estagio.bean.Empresa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import estagio.form.PesquisarConveniosForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarConveniosAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    EmpresaDao empresaDao = new EmpresaDao();
    ConvenioDao convenioDao = new ConvenioDao();
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    PesquisarConveniosForm pesquisarConveniosForm =
(PesquisarConveniosForm) form;
    List<Convenio> convenios = new ArrayList<Convenio>();
    String parametro = (String)
request.getParameter("parametro");
    String valor = (String) request.getParameter("valor");
    if("cnpj".equals(parametro)){
      try{
```

```
            convenios = convenioDao.listarConvenios(valor);
        } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.convenio"));
        }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
        }
      } else {
        try{
            List<Empresa> empresas =
empresaDao.pesquisarEmpresa(valor);
            try{
              for(Empresa empresa: empresas){

convenios.addAll(convenioDao.listarConvenios(empresa.getCnpj(
)));
              }
            } catch (SQLException e){
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.listar.convenio"));
            }catch (Exception e) {
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.listar.convenio"));
            }
        } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
        }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
        }
      }
      List<ConvenioDTO> cDtos = new
ArrayList<ConvenioDTO>();
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      GregorianCalendar hoje = new GregorianCalendar();
      for(Convenio c:convenios){
        ConvenioDTO cDto = new ConvenioDTO();
        cDto.setCodConvenio(c.getCodConvenio());
        cDto.setDataFim(sdf.format(c.getDataFim().getTime()));
cDto.setDataInicio(sdf.format(c.getDataInicio().getTime()));
        cDto.setEmpresa(c.getEmpresa().getRazaoSocial());
        cDto.setDescricao(c.getDescricao());
        if(hoje.after(c.getDataFim())){
          cDto.setSituacao("Expirado.");
        } else {
          cDto.setSituacao("Válido.");
        }
        cDtos.add(cDto);
      }
      if(convenios.size()>0){
        request.setAttribute("convenios", cDtos);
      } else {
        request.setAttribute("mensagem", "Convênio não
encontrado.");
        return mapping.findForward(Constants.FAILURE);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.219 PesquisarEmpresasAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.Empresa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import estagio.form.PesquisarEmpresasForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarEmpresasAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    EmpresaDao empresaDao = new EmpresaDao();
    PesquisarEmpresasForm pesquisarEmpresaForm =
(PesquisarEmpresasForm) form;
    List<Empresa> empresas = new ArrayList<Empresa>();
    try{
      empresas =
empresaDao.pesquisarEmpresa(pesquisarEmpresaForm.getParteD
oNome());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
    }
    ConvenioDao convenioDao = new ConvenioDao();
    List<Convenio> convenios = new ArrayList<Convenio>();
    try{
      convenios = convenioDao.listarConvenios();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
    }
    for(Empresa emp:empresas){
      emp.setCargoRepresentante("Não possui.");
      for(Convenio c: convenios){
        if(c.getEmpresa().getCnpj().equals(emp.getCnpj())){
          GregorianCalendar hoje = new GregorianCalendar();
```

```
            if(c.getDataFim().after(hoje)){

emp.setCargoRepresentante(c.getCodConvenio().toString());
            } else {
              emp.setCargoRepresentante("Expirado");
            }
          }
        }
      }
      if(!empresas.isEmpty()){
        request.setAttribute("empresas", empresas);
      } else {
        request.setAttribute("mensagem", "Empresa não
encontrada.");
        return mapping.findForward(Constants.FAILURE);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.220 PesquisarEmpresasSemConvenioAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.Empresa;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import estagio.form.PesquisarConveniosForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarEmpresasSemConvenioAction extends
Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    EmpresaDao empresaDao = new EmpresaDao();
    ConvenioDao convenioDao = new ConvenioDao();
    PesquisarConveniosForm pesquisarConveniosForm =
(PesquisarConveniosForm) form;
    List<Empresa> empresas = new ArrayList<Empresa>();
    String parametro = (String)
request.getParameter("parametro");
    String valor = (String) request.getParameter("valor");
    if("cnpj".equals(parametro)){
      try{
        Empresa empresa =
empresaDao.buscarEmpresaSemConvenio(valor);
        if(empresa != null){
          empresas.add(empresa);
        }
        empresa =
empresaDao.buscarEmpresaConvenioVencido(valor);
        if(empresa != null){
          empresas.add(empresa);
        }
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.convenio"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
      }
    } else {
      try{
        List<Empresa> todasEmpresas =
empresaDao.pesquisarEmpresa(valor);
        for(Empresa empresa: todasEmpresas){
          Empresa emp =
empresaDao.buscarEmpresaConvenioVencido(empresa.getCnpj()
);
          if(emp != null){
            empresas.add(emp);
          }
          emp =
empresaDao.buscarEmpresaSemConvenio(empresa.getCnpj());
          if(emp != null){
            empresas.add(emp);
          }
        }
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
      }
    }
    List<Convenio> convenios = new ArrayList<Convenio>();
    try{
      convenios = convenioDao.listarConvenios();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
    }
    GregorianCalendar hoje = new GregorianCalendar();
    for(Empresa emp:empresas){
      emp.setCargoRepresentante("Não possui.");
      for(Convenio c: convenios){
        if(c.getEmpresa().getCnpj().equals(emp.getCnpj())){
          if(c.getDataFim().after(hoje)){

emp.setCargoRepresentante(c.getCodConvenio().toString());
          } else {
            emp.setCargoRepresentante("Expirado");
          }
        }
```

```
        }
      }
      if(empresas.size()>0){
        request.setAttribute("empresas", empresas);
      } else {
        request.setAttribute("mensagem", "Empresa não
encontrada. Verifique se a empresa já possui um convênio
válido.");
        return mapping.findForward(Constants.FAILURE);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.221 PesquisarHabilidadeAlunoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.HabilidadesDao;
import estagio.form.NovaHabilidadeAlunoForm;
import estagio.form.PesquisarHabilidadeAlunoForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarHabilidadeAlunoAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    PesquisarHabilidadeAlunoForm
pesquisarHabilidadeAlunoForm =
(PesquisarHabilidadeAlunoForm) form;
    String codAlunoEstagio =
request.getParameter("codAlunoEstagio");
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    String parteDoNome = (String)
request.getParameter("nomeHabilidade");
    //Salva as habilidades selecionadas na tela de pesquisa
    if(pesquisarHabilidadeAlunoForm.getCodHabilidade() !=
null){
      for(int i = 0; i <
pesquisarHabilidadeAlunoForm.getCodHabilidade().length; i++){
        Habilidade habilidade = new Habilidade();
        Integer codHabilidadeAluno =
pesquisarHabilidadeAlunoForm.getCodHabilidade()[i];
        try{
          habilidade =
habilidadesDao.buscarHabilidade(codHabilidadeAluno);
          HabilidadeAluno habilidadeAluno = new
HabilidadeAluno();

habilidadeAluno.setCodHabilidade(habilidade.getCodHabilidade(
));

habilidadeAluno.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeAluno.setCodAlunoEstagio(user.getCodAlunoEstagio()
);
habilidadesAlunoDao.incluirHabilidadeAluno(habilidadeAluno);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.salvar.habilidadevaga"));
        }
      }
    }
    //as habilidades selecionadas na tela de pesquisa estao salvas
    List<Habilidade> habilidadesEncontradas = new
ArrayList<Habilidade>();
    List<Habilidade> habilidades = new
ArrayList<Habilidade>();
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    try{
      habilidadesEncontradas =
habilidadesDao.buscarHabilidade(parteDoNome);
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidade"));
    }
    for(Habilidade h: habilidadesEncontradas){
      habilidades.add(h);
      for(HabilidadeAluno hAluno: habilidadesAluno){
        if(hAluno.getCodHabilidade().intValue() ==
h.getCodHabilidade().intValue()){
          habilidades.remove(h);
        }
      }
    }
    /*Integer[] codHabilidades = new
Integer[habilidadesEncontradas.size()];
    for(int i = 0; i < habilidadesEncontradas.size();i++){
      Habilidade h = habilidadesEncontradas.get(i);
      codHabilidades[i] = h.getCodHabilidade();
    }
    request.setAttribute("preenchidas", codHabilidades);*/
    if(habilidades.size() > 0){
      request.setAttribute("todashabilidades", habilidades);
```

```
    }

    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.222 PesquisarTermosCompromissoAction.java

```
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.bean.TermoCompromisso;
import estagio.bean.TermoCompromissoDTO;
import estagio.bean.Turno;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.TermoCompromissoDao;
import estagio.form.PesquisarTermosCompromissoForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarTermosCompromissoAction extends
Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    PesquisarTermosCompromissoForm
pesquisarTermosCompromissoForm =
(PesquisarTermosCompromissoForm) form;
    List<TermoCompromisso> termosCompromisso = new
ArrayList<TermoCompromisso>();
    String parametro = (String)
request.getParameter("parametro");
    String valor = (String) request.getParameter("valor");
    if("cnpj".equals(parametro)){
      try{
        termosCompromisso =
termoCompromissoDao.listarTermos(valor);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.termocompromisso"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.termocompromisso"));
      }
    } else {
      try{
        List<AlunoEstagio> estagiarios =
alunoEstagioDao.buscarEstagiario(valor);
        try{
          for(AlunoEstagio estagiario: estagiarios){

termosCompromisso.addAll(termoCompromissoDao.listarTermos
Compromisso(estagiario.getCodAlunoEstagio()));
          }
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.listar.termocompromisso"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.listar.termocompromisso"));
        }
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.estagiarios"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.estagiarios"));
      }
    }
    List<TermoCompromissoDTO> tDtos = new
ArrayList<TermoCompromissoDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(TermoCompromisso t:termosCompromisso){
      TermoCompromissoDTO tDto = new
TermoCompromissoDTO();
      tDto.setEmpresa(t.getEmpresa().getRazaoSocial());

tDto.setAgenteIntegracao(t.getAgenteIntegracao()!=null?t.getAge
nteIntegracao().getAgente():"nenhum");
      tDto.setCodTermo(t.getCodTermo());
      tDto.setEstagiario(t.getAlunoEstagio().getNomePessoa());

tDto.setDataInicio(sdf.format(t.getDataInicio().getTime()));
      tDto.setDataFim(sdf.format(t.getDataFim().getTime()));
      tDto.setSalario(t.getVaga().getSalario());
      tDto.setSetor(t.getVaga().getSetor());

tDto.setCurso(t.getAlunoEstagio().getCurso().getNomeCurso());
      String turnos = "";
      for(int i = 0;i<t.getVaga().getTurnos().size();i++){
        Turno turno= t.getVaga().getTurnos().get(i);
        if( t.getVaga().getTurnos().size() - 1 != i){
          turnos += turno.toString()+", ";
        } else {
          turnos += turno.toString()+".";
        }
      }
      tDto.setHorario("".equals(turnos)?"Não
Informado":turnos);
      String descricaoHabilidade = "";
      for(int i = 0;i<t.getVaga().getHabilidades().size();i++){
        Habilidade habilidade =
t.getVaga().getHabilidades().get(i);
        if(t.getVaga().getHabilidades().size() - 1 != i){
          descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
```

```
            } else {
              descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
            }
          }
          tDto.setRequisitos("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
          tDto.setEstagioInterno(t.getVaga().getEstagioInterno());
          tDtos.add(tDto);
        }
        if(!termosCompromisso.isEmpty()){
          request.setAttribute("termos", tDtos);
        } else {
          request.setAttribute("mensagem", "Termo de
Compromisso não encontrado.");
          return mapping.findForward(Constants.FAILURE);
        }
        if(!errors.isEmpty()){
          saveErrors(request, errors);
          return mapping.findForward(Constants.FAILURE);
        } else {
          return mapping.findForward(Constants.SUCCESS);
        }
      }
}
```

6.4.223 PesquisarUnidadeAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.dao.EmpresaDao;
import estagio.form.PesquisarEmpresasForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarUnidadeAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    PesquisarEmpresasForm pesquisarEmpresaForm =
(PesquisarEmpresasForm) form;
    List<Empresa> empresas = new ArrayList<Empresa>();
    try{
      EmpresaDao empresaDao = new EmpresaDao();
      empresas =
empresaDao.pesquisarEmpresa(pesquisarEmpresaForm.getParteD
oNome());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.empresa"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.empresa"));
    }
    if(!empresas.isEmpty()){
      request.setAttribute("empresas", empresas);
      request.setAttribute("cnpj", empresas.get(0).getCnpj());
    } else {
      request.setAttribute("mensagem", "Empresa não
encontrada.");
      return mapping.findForward(Constants.FAILURE);
    }
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.224 PesquisarUsuarioAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.dao.AlunoEstagioDao;
import estagio.form.BuscarUsuarioForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class PesquisarUsuarioAction extends Action{
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("welcome");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    BuscarUsuarioForm buscarUsuarioForm =
(BuscarUsuarioForm) form;
    List<AlunoEstagio> usuarios = new
ArrayList<AlunoEstagio>();
    try{
```

```
        usuarios =
alunoEstagioDao.buscarUsuario(buscarUsuarioForm.getNomeUs
uario());
        AlunoEstagio aluno =
alunoEstagioDao.buscarAlunoEstagio(buscarUsuarioForm.getNo
meUsuario());
        if(aluno != null){
          boolean contains = false;
          for(AlunoEstagio a: usuarios){

if(a.getCodAlunoEstagio().equals(aluno.getCodAlunoEstagio())){
              contains = true;
            }
          }
          if(!contains){
            usuarios.add(aluno);
          }
        }
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarios"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarios"));
      }
      if(usuarios.size() > 0){
        request.setAttribute("usuarios", usuarios);
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }
}
```

6.4.225 RelatorioSeguroAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.RelatorioSeguroDTO;
import estagio.dao.RelatorioSeguroDao;
import estagio.form.SelecionarRelatorioForm;
import java.sql.ResultSet;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import net.sf.jasperreports.engine.JRException;
import net.sf.jasperreports.engine.JRResultSetDataSource;
import net.sf.jasperreports.engine.JasperPrint;
import net.sf.jasperreports.engine.JasperRunManager;
import
net.sf.jasperreports.engine.data.JRBeanCollectionDataSource;
import net.sf.jasperreports.view.JasperViewer;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class RelatorioSeguroAction extends Action{
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      RelatorioSeguroDao relatorioSeguroDao = new
RelatorioSeguroDao();
      SelecionarRelatorioForm selecionarRelatorioForm =
(SelecionarRelatorioForm) form;
      String cnpj = selecionarRelatorioForm.getCnpj();
      String mes = selecionarRelatorioForm.getMes();
      String ano = selecionarRelatorioForm.getAno();
      /* implementação da interface JRDataSource para
DataSource ResultSet */
      boolean estagio =
"interno".equals(selecionarRelatorioForm.getEstagio())?true:false
;
      List<RelatorioSeguroDTO> dadosRelatorio =
relatorioSeguroDao.queryRelatorioCancelar(cnpj,
Integer.valueOf(mes), Integer.valueOf(ano), estagio);
      JRBeanCollectionDataSource jrBc = new
JRBeanCollectionDataSource(dadosRelatorio);
      if("incluir".equals(selecionarRelatorioForm.getRelatorio()))
{
        dadosRelatorio =
relatorioSeguroDao.queryRelatorioIncluir(cnpj,
Integer.valueOf(mes), Integer.valueOf(ano), estagio);
      } else
if("manter".equals(selecionarRelatorioForm.getRelatorio())) {
        dadosRelatorio =
relatorioSeguroDao.queryRelatorioManter(cnpj,
Integer.valueOf(mes), Integer.valueOf(ano), estagio);
      } else
if("geral".equals(selecionarRelatorioForm.getRelatorio())) {
        dadosRelatorio =
relatorioSeguroDao.queryRelatorioGeral(cnpj,
Integer.valueOf(mes), Integer.valueOf(ano), estagio);
      }
      jrBc = new JRBeanCollectionDataSource(dadosRelatorio);
      /* HashMap de parametros utilizados no relatório. Sempre
instanciados */
      Map parameters = new HashMap();
      //parameters.put("PROCESSO", "CANCELAR");
      JasperPrint jasperPrint;
      java.net.URL resource =
this.getClass().getClassLoader().getResource("jasper/teste_seguro
.jasper");
      String arq = resource.getFile();
      try {
        //jasperReport =
JasperCompileManager.compileReport("D://TCC//TccEstagio//te
ste_seguro.jrxml");
        byte[] retorno = JasperRunManager.runReportToPdf(arq,
parameters, jrBc);
        //jasperPrint = JasperFillManager.fillReport(is,
parameters, jrRS);
        //JasperViewer.viewReport(arq, false);
        response.setContentType("application/pdf");
        response.setHeader("Content-disposition",
"filename=relatorio.pdf");
        response.setContentLength(retorno.length);
        ServletOutputStream saidaResponse =
response.getOutputStream();
```

```
            saidaResponse.write(retorno, 0, retorno.length);
            saidaResponse.flush();
            saidaResponse.close();
        } catch (JRException e) {
            e.printStackTrace();
        }
        if(!errors.isEmpty()){
            saveErrors(request, errors);
            return mapping.findForward(Constants.FAILURE);
        } else {
            return mapping.findForward(Constants.SUCCESS);
        }
    }
}
```

6.4.226 SalvarCadastroAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Pessoa;
import estagio.bean.UsuarioEmpresa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.PessoaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.form.DadosPessoaisForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarCadastroAction extends Action {
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
    HttpSession session = request.getSession();
    DadosPessoaisForm regform = (DadosPessoaisForm) form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    String action = request.getParameter("action");
    //verifica se o usuario fez login na intranet academica
    String logou = (String) session.getAttribute("logado");
    boolean logado = false;
    String codPessoa = null;
    if("true".equals(logou)){
        logado = true;
    }
    if (action == null) {
        action = "Create";
    }
    if (user == null && !"aluno".equals(regform.getStatus())) {
        return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    if("Create".equals(action)){
        String value = null;
        value = regform.getUsuario();
        try {
            PessoaDao pessoaDao = new PessoaDao();
            Pessoa testAluno =
pessoaDao.buscarPessoa(regform.getUsuario());
            //verifica se o nome de usuario informado já está sendo
utilizado
            if (testAluno != null){
                //se o nome de usuario já estiver sendo utilizado,
verifica se é o próprio usuário q está logado
                if(logado == true){
                    codPessoa = (String)
session.getAttribute("codPessoa");
                    Pessoa pessoaLogada =
pessoaDao.buscarPessoa(Integer.valueOf(codPessoa));
                    //se não for o usuário logado que está tentando
utilizar o username emite um erro e não insere nada no banco
                    if(!pessoaLogada.equals(testAluno)){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.usuario.unico"));
                    }
                    /*se for o mesmo usuario logado que está tentando
cadastrar-se no modulo estagios,
                        prossegue normalmente. O método
incluirAlunoEstagio em AlunoEstagioDao previne
                        a tentativa de nova inserção da mesma pessoa
na tabela pessoa.*/
                }
            }
        } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarios"));
        } catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarios"));
        }
    }
    //carrega todos os campos do formulario, exceto a senha, que
depende de action Create ou Edit
    CursoDao cursoDao = new CursoDao();
    AlunoEstagio novoAluno = new AlunoEstagio();

novoAluno.setDataNascimento(regform.getDataNascimento());
    novoAluno.setDeficiente(regform.getDeficiente());
    novoAluno.setDisponivelFds(regform.getDisponivelFds());
    novoAluno.setEmail(regform.getEmail());
    novoAluno.setFone(regform.getFone());
    novoAluno.setFumante(regform.getFumante());
    novoAluno.setLocomocao(regform.getLocomocao());
    novoAluno.setNomePessoa(regform.getNome_pessoa());
    novoAluno.setSexo(regform.getSexo());
    novoAluno.setUsuario(regform.getUsuario());
    novoAluno.setSemestre(regform.getSemestre());
    novoAluno.setRg(regform.getRg());
    novoAluno.setCpf(Long.valueOf(regform.getCpf()));
novoAluno.setRg_naturalidade(regform.getRg_naturalidade());
    novoAluno.setMatricula(regform.getMatricula());
    novoAluno.setCep(regform.getCep());
    novoAluno.setRua(regform.getRua());
    novoAluno.setNumero(regform.getNumero());
    novoAluno.setComplemento(regform.getComplemento());
    novoAluno.setBairro(regform.getBairro());
    novoAluno.setMunicipio(regform.getMunicipio());
    novoAluno.setEstado(regform.getEstado());
    novoAluno.setPais(regform.getPais());
novoAluno.setCurso(cursoDao.buscarCurso(regform.getCodCurs
o()));
    if (regform.getStatus() == null ||
"".equals(regform.getStatus())){
        novoAluno.setStatus("aluno");
    } else {
        novoAluno.setStatus(regform.getStatus());
    }
    String cnpj = regform.getEmpresa();
    if("Create".equals(action)) {
        if(errors.isEmpty()){
```

```
            try {
              //armazena hashCode da nova senha informada
novoAluno.setSenha(String.valueOf(regform.getSenha().hashCod
e()));
              EmpresaDao empresaDao = new EmpresaDao();
              if ("empresa".equals(novoAluno.getStatus()) &&
empresaDao.buscarEmpresa(cnpj) != null){

alunoEstagioDao.incluirAlunoEstagio(novoAluno);
                novoAluno =
alunoEstagioDao.buscarAlunoEstagio(regform.getUsuario());
                UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
                UsuarioEmpresa usuarioEmpresa = new
UsuarioEmpresa();
                usuarioEmpresa.setCnpj(cnpj);

usuarioEmpresa.setCodAlunoEstagio(novoAluno.getCodAlunoEs
tagio());

usuarioEmpresaDao.incluirUsuarioEmpresa(usuarioEmpresa);
              } else {

alunoEstagioDao.incluirAlunoEstagio(novoAluno);
              }
            } catch (SQLException e){
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
            } catch (Exception e) {
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.incluir.usuario"));
            }
          }
        } else {
          AlunoEstagio alunoAlterar = null;
          try {
            alunoAlterar =
alunoEstagioDao.buscarAlunoEstagio(novoAluno.getUsuario());
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
          } catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.usuario"));
          }
          if(alunoAlterar != null){

novoAluno.setCodAlunoEstagio(alunoAlterar.getCodAlunoEstagi
o());
            //por aqui nao eh possivel alterar a senha, entao mantem
a anterior
            novoAluno.setSenha(alunoAlterar.getSenha());
            try {
              alunoEstagioDao.alterarAlunoEstagio(novoAluno);
            } catch (SQLException e){
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
            } catch (Exception e) {
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.alterar.usuario"));
            }
          }
          if(novoAluno.getUsuario().equals(user.getUsuario())){
            session.removeAttribute(Constants.USER_KEY);
            session.setAttribute(Constants.USER_KEY,
novoAluno);
          }
        }
        if(user == null){
          try{
            novoAluno =
alunoEstagioDao.buscarAlunoEstagio(regform.getUsuario());
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarios"));
          } catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarios"));
          }
          session.removeAttribute(Constants.USER_KEY);
          session.setAttribute(Constants.USER_KEY, novoAluno);
        }
        if (!errors.isEmpty()) {
          saveErrors(request, errors);
          return (mapping.findForward(Constants.FAILURE));
        } else {
          return (mapping.findForward(Constants.SUCCESS));
        }
      }
    }
```

6.4.227 SalvarConvenioAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.ConvenioDTO;
import estagio.bean.Empresa;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import estagio.form.NovoConvenioForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarConvenioAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      HttpSession session = request.getSession();
    NovoConvenioForm convenioform = (NovoConvenioForm)
form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    String action = request.getParameter("action");
    ConvenioDao cDao = new ConvenioDao();
    if (action == null) {
      action = "Create";
    }
    if (user == null) {
      return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    Convenio convenio = new Convenio();
    EmpresaDao empresaDao = new EmpresaDao();
    Empresa empresa = new Empresa();
    empresa =
empresaDao.buscarEmpresa(convenioform.getCnpj());
    GregorianCalendar hoje = new GregorianCalendar();
    convenio.setEmpresa(empresa);
    convenio.setDataFim(convenioform.getDataFim());
    convenio.setDataInicio(convenioform.getDataInicio());
```

```
    convenio.setDescricao(convenioform.getDescricao());
    if("Create".equals(action)) {
      try{
        cDao.incluirConvenio(convenio);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.convenio"));
      }
    } else {
convenio.setCodConvenio(Integer.valueOf(convenioform.getCod
Convenio()));
      cDao.alterarConvenio(convenio);
    }
    List<Convenio> convenios = new ArrayList<Convenio>();
    try{
      convenios = cDao.listarConvenios();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
    }
    List<ConvenioDTO> cDtos = new
ArrayList<ConvenioDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(Convenio c:convenios){
      ConvenioDTO cDto = new ConvenioDTO();
      cDto.setCodConvenio(c.getCodConvenio());
      cDto.setDataFim(sdf.format(c.getDataFim().getTime()));

cDto.setDataInicio(sdf.format(c.getDataInicio().getTime()));
      cDto.setEmpresa(c.getEmpresa().getRazaoSocial());
      cDto.setDescricao(c.getDescricao());
      if(hoje.after(c.getDataFim())){
        cDto.setSituacao("Expirado.");
      } else {
        cDto.setSituacao("Válido.");
      }
      cDtos.add(cDto);
    }
    if(convenios.size()>0){
      request.setAttribute("convenios", cDtos);
    }
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }

  }
}
```

6.4.228 SalvarCursoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.dao.CursoDao;
import estagio.form.NovoCursoForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarCursoAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
      HttpSession session = request.getSession();
    NovoCursoForm cursoForm = (NovoCursoForm) form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    CursoDao cursoDao = new CursoDao();
    if (user == null) {
      return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    Curso curso = new Curso();
    curso.setDuracao(cursoForm.getDuracao());
    curso.setNomeCurso(cursoForm.getNomeCurso());
    try{
      cursoDao.incluirCurso(curso);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.agente"));
    }
    //parametros para listarcursos.jsp
    List<Curso> cursos = new ArrayList<Curso>();
    try{
      cursos = cursoDao.listarCursos();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
    }
    if(cursos.size()>0){
      request.setAttribute("cursos", cursos);
    }
    //fim dos parametros para listar cursos.jsp
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }

  }
}
```

6.4.229 SalvarEmpresaAction.java

```
package estagio.action;
import estagio.bean.AgenteIntegracao;
import estagio.bean.Constants;
import estagio.bean.Convenio;
import estagio.bean.ConvenioDTO;
import estagio.bean.Empresa;
import estagio.bean.Endereco;
import estagio.bean.Pessoa;
import estagio.dao.AgentesIntegracaoDao;
import estagio.dao.ConvenioDao;
import estagio.dao.EmpresaDao;
import estagio.form.NovaEmpresaForm;
import java.sql.SQLException;
```

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarEmpresaAction extends Action {
   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {
          HttpSession session = request.getSession();
      ActionMessages errors = new ActionMessages();
      NovaEmpresaForm empresaform = (NovaEmpresaForm)
form;
      Pessoa user = (Pessoa)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      EmpresaDao eDao = new EmpresaDao();
      if (action == null) {
        action = "Create";
      }
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      String cnpj = empresaform.getCnpj();
      Empresa empresa = new Empresa();

empresa.setCargoRepresentante(empresaform.getCargoRepresent
ante());
      empresa.setCnpj(empresaform.getCnpj());
      empresa.setEmail(empresaform.getEmail());
      empresa.setFax(empresaform.getFax());
      empresa.setFone(empresaform.getFone());
      empresa.setNomeFantasia(empresaform.getNomeFantasia());
      empresa.setRamo(empresaform.getRamo());
      empresa.setRazaoSocial(empresaform.getRazaoSocial());
      empresa.setRepresentante(empresaform.getRepresentante());
      empresa.setSite(empresaform.getSite());
      empresa.setNumProcesso(empresaform.getNumProcesso());
      Endereco endereco = new Endereco();
      endereco.setBairro(empresaform.getBairro());
      endereco.setCep(empresaform.getCep());

endereco.setComplemento(empresaform.getComplemento());
      endereco.setEstado(empresaform.getEstado());
      endereco.setMunicipio(empresaform.getMunicipio());
      endereco.setNumero(empresaform.getNumero());
      endereco.setOwner(empresaform.getCnpj());
      endereco.setPais(empresaform.getPais());
      endereco.setRua(empresaform.getRua());
      empresa.setEnderecoEmpresa(endereco);
      ConvenioDao cDao = new ConvenioDao();
      Convenio convenio = new Convenio();
      convenio.setEmpresa(empresa);
      convenio.setDataFim(empresaform.getDataFim());
      convenio.setDataInicio(empresaform.getDataInicio());
      convenio.setDescricao("");
      if("S".equals(empresaform.getAgenteIntegrador())){
        empresa.setAgenteIntegracao("S");
      } else {
        empresa.setAgenteIntegracao("N");
      }
      if("Create".equals(action)) {
        Empresa eTest = eDao.buscarEmpresa(cnpj);
        if(eTest != null){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.cnpj.unico"));
        }
        if(errors.isEmpty()){
          try{
            eDao.incluirEmpresa(empresa);
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.salvar.empresa"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.salvar.empresa"));
          }
          try{
            cDao.incluirConvenio(convenio);
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.salvar.convenio"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.salvar.convenio"));
          }
        }
      } else {
        try{
          Empresa empresaRegistrada =
eDao.buscarEmpresa(cnpj);

empresa.getEnderecoEmpresa().setCodEndereco(empresaRegistra
da.getEnderecoEmpresa().getCodEndereco());
          eDao.alterarEmpresa(empresa);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.empresa"));
        }
      }
      /*ConvenioDao convenioDao = new ConvenioDao();
      List<Convenio> convenios = new ArrayList<Convenio>();
      try{
        convenios = convenioDao.listarConvenios();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.convenio"));
      }
      List<ConvenioDTO> cDtos = new
ArrayList<ConvenioDTO>();
      SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
      for(Convenio c:convenios){
        ConvenioDTO cDto = new ConvenioDTO();
        cDto.setCodConvenio(c.getCodConvenio());
        cDto.setDataFim(sdf.format(c.getDataFim().getTime()));
cDto.setDataInicio(sdf.format(c.getDataInicio().getTime()));
        cDto.setEmpresa(c.getEmpresa().getRazaoSocial());
        cDto.setDescricao(c.getDescricao());
        GregorianCalendar hoje = new GregorianCalendar();
        if(hoje.after(c.getDataFim())){
          cDto.setSituacao("Expirado.");
        } else {
          cDto.setSituacao("Válido.");
        }
        cDtos.add(cDto);
      }
```

```
    if(convenios.size()>0){
      request.setAttribute("convenios", cDtos);
    }
    */
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
          }
  }
}
```

6.4.230 SalvarHabilidadeAlunoAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.HabilidadesDao;
import estagio.form.PesquisarHabilidadeAlunoForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;


/**
 *
 *
 */
public class SalvarHabilidadeAlunoAction extends Action {

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {


    ActionMessages errors = new ActionMessages();
    HttpSession session = request.getSession();
    AlunoEstagio user = null;
    try{
      user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      if (user == null){
        return mapping.findForward("logon");
      }
    } catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE,
          new ActionMessage("error.generico"));
    }
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    PesquisarHabilidadeAlunoForm
pesquisarHabilidadeAlunoForm =
(PesquisarHabilidadeAlunoForm) form;
    String codAlunoEstagio =
request.getParameter("codAlunoEstagio");
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    //Salva as habilidades selecionadas na tela de pesquisa
    if(pesquisarHabilidadeAlunoForm.getCodHabilidade() !=
null){
      for(int i = 0; i <
pesquisarHabilidadeAlunoForm.getCodHabilidade().length; i++){
        Habilidade habilidade = new Habilidade();
        Integer codHabilidadeAluno =
pesquisarHabilidadeAlunoForm.getCodHabilidade()[i];
        try{
          habilidade =
habilidadesDao.buscarHabilidade(codHabilidadeAluno);
          HabilidadeAluno habilidadeAluno = new
HabilidadeAluno();

habilidadeAluno.setCodHabilidade(habilidade.getCodHabilidade(
));

habilidadeAluno.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());
habilidadeAluno.setCodAlunoEstagio(user.getCodAlunoEstagio()
);
habilidadesAlunoDao.incluirHabilidadeAluno(habilidadeAluno);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.salvar.habilidadevaga"));
        }
      }
    }
    //as habilidades seleccionadas na tela de pesquisa estao salvas
    //parametros para listarhabilidadesaluno.jsp
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    try{
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadealuno"));
    }
    if(habilidadesAluno.size()>0){
      request.setAttribute("habilidades", habilidadesAluno);
    }
    //parametros Ok
    if(!errors.isEmpty()){
      saveErrors(request, errors);
      return mapping.findForward(Constants.FAILURE);
    } else {
      return mapping.findForward(Constants.SUCCESS);
    }
  }
}
```

6.4.231 SalvarHabilidadeAlunoGeralAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import estagio.dao.HabilidadesAlunoDao;
import estagio.dao.HabilidadesDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.form.NovaHabilidadeForm;
```

```
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarHabilidadeAlunoGeralAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {

    HttpSession session = request.getSession();
    NovaHabilidadeForm habilidadeform =
(NovaHabilidadeForm) form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    HabilidadesAlunoDao habilidadesAlunoDao = new
HabilidadesAlunoDao();
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    if (user == null) {
      return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    HabilidadeAluno habilidadeAluno = new HabilidadeAluno();
habilidadeAluno.setDescricaoHabilidade(habilidadeform.getDesc
ricaoHabilidade());
habilidadeAluno.setCodAlunoEstagio(habilidadeform.getCodAlu
noEstagio());
    try{
      List<Habilidade> habilidades = new
ArrayList<Habilidade>();
    /*  habilidades = habilidadesDao.listarHabilidades();
      boolean jaExiste = false;
      for(Habilidade h:habilidades){

if(h.getDescricaoHabilidade().equalsIgnoreCase(habilidadeAluno.
getDescricaoHabilidade())){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.habilidade.unica"));
          jaExiste = true;
          break;
        }
      }
      if(!jaExiste){*/
        habilidadesDao.incluirHabilidade(habilidadeAluno);
        habilidades = habilidadesDao.listarHabilidades();
        for(Habilidade hab:habilidades){

if(hab.getDescricaoHabilidade().equalsIgnoreCase(habilidadeAlu
no.getDescricaoHabilidade())){
habilidadeAluno.setCodHabilidade(hab.getCodHabilidade());
habilidadesAlunoDao.incluirHabilidadeAluno(habilidadeAluno);
            break;
          }
        }
      //}
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.habilidadealuno"));
    }
    //seta parametros para caso de
sucesso(listarhabilidadesaluno.jsp)
    List<HabilidadeAluno> habilidadesAluno = new
ArrayList<HabilidadeAluno>();
    Integer codAlunoEstagio = user.getCodAlunoEstagio();
    try{
      habilidadesAluno =
habilidadesAlunoDao.listarHabilidadeAluno(codAlunoEstagio);
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadealuno"));
    }
    if(habilidadesAluno.size()>0){
      request.setAttribute("habilidades", habilidadesAluno);
    }
    //parametros setados para caso de
sucesso(listarhabilidadesaluno.jsp)
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.232 SalvarHabilidadeVagaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeVaga;
import estagio.dao.HabilidadesDao;
import estagio.dao.HabilidadesVagaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.form.NovaHabilidadeVagaForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
public class SalvarHabilidadeVagaAction extends Action {
  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    HttpSession session = request.getSession();
    ActionMessages errors = new ActionMessages();
    NovaHabilidadeVagaForm habilidadevagaform =
(NovaHabilidadeVagaForm) form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
    HabilidadesDao habilidadesDao = new HabilidadesDao();
    if (user == null) {
      return (mapping.findForward("logon"));
```

```
    }
    if(habilidadevagaform.getCodHabilidade() != null){
      for(int i = 0; i <
habilidadevagaform.getCodHabilidade().length; i++){
        Habilidade habilidade = new Habilidade();
        Integer codHabilidadeVaga =
habilidadevagaform.getCodHabilidade()[i];
        try{
          habilidade =
habilidadesDao.buscarHabilidade(codHabilidadeVaga);
          HabilidadeVaga habilidadeVaga = new
HabilidadeVaga();
habilidadeVaga.setCodHabilidade(habilidade.getCodHabilidade()
);
habilidadeVaga.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());
habilidadeVaga.setCodVaga(habilidadevagaform.getCodVaga());
habilidadesVagaDao.incluirHabilidadeVaga(habilidadeVaga);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.salvar.habilidadevaga"));
        }
      }
    }
    //seta parametros para o caso de
Erro(novahabilidadevaga.jsp)
    List<Habilidade> todasHabilidades = new
ArrayList<Habilidade>();
    List<Habilidade> habilidades = new
ArrayList<Habilidade>();
    List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
    try{
      todasHabilidades = habilidadesDao.listarHabilidades();
      habilidadesVaga =
habilidadesVagaDao.listarHabilidadeVaga(habilidadevagaform.ge
tCodVaga());
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidade"));
    }
    for(Habilidade h: todasHabilidades){
      habilidades.add(h);
      for(HabilidadeVaga hVaga: habilidadesVaga){
        if(hVaga.getCodHabilidade().intValue() ==
h.getCodHabilidade().intValue()){
          habilidades.remove(h);
        }
      }
    }
    if(habilidades.size() > 0){
      request.setAttribute("todashabilidades", habilidades);
    }
    //parametros setados para o caso de erro
    //parametro abaixo para ambos os casos
    String codVaga =
habilidadevagaform.getCodVaga().toString();
    //seta parametros para caso de
sucesso(listarhabilidadesvaga.jsp)
    String empresa = "";
    String cnpj = "";
    Empresa emp = new Empresa();
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    if(user.getStatus() != null){
      if("empresa".equals(user.getStatus())){
        try{
          emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
        }
        if(emp != null){
          empresa = emp.getRazaoSocial();
          cnpj = emp.getCnpj();
        }
      }
    }
    if(habilidadesVaga.size()>0){
      request.setAttribute("habilidades", habilidadesVaga);
    }
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    request.setAttribute("codvaga", codVaga);
    //parametros setados para caso de
sucesso(listarhabilidadesvaga.jsp)
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.233 SalvarHabilidadeVagaGeralAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeVaga;
import estagio.bean.Vaga;
import estagio.dao.EmpresaDao;
import estagio.dao.HabilidadesDao;
import estagio.dao.HabilidadesVagaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.dao.VagaDao;
import estagio.form.NovaHabilidadeForm;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarHabilidadeVagaGeralAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    HttpSession session = request.getSession();
    NovaHabilidadeForm habilidadeform =
(NovaHabilidadeForm) form;
```

```
      AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      HabilidadesVagaDao habilidadesVagaDao = new
HabilidadesVagaDao();
      HabilidadesDao habilidadesDao = new HabilidadesDao();
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      HabilidadeVaga habilidadeVaga = new HabilidadeVaga();

habilidadeVaga.setDescricaoHabilidade(habilidadeform.getDescri
caoHabilidade());
      habilidadeVaga.setCodVaga(habilidadeform.getCodVaga());
      try{
        List<Habilidade> habilidades = new
ArrayList<Habilidade>();
        /*habilidades = habilidadesDao.listarHabilidades();
        boolean jaExiste = false;
        for(Habilidade h:habilidades){

if(h.getDescricaoHabilidade().equalsIgnoreCase(habilidadeVaga.
getDescricaoHabilidade())){
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.habilidade.unica"));
              jaExiste = true;
              break;
            }
        }
        if(!jaExiste){*/
        habilidadesDao.incluirHabilidade(habilidadeVaga);
        habilidades = habilidadesDao.listarHabilidades();
        for(Habilidade hab:habilidades){
if(hab.getDescricaoHabilidade().equalsIgnoreCase(habilidadeVag
a.getDescricaoHabilidade())){
habilidadeVaga.setCodHabilidade(hab.getCodHabilidade());
habilidadesVagaDao.incluirHabilidadeVaga(habilidadeVaga);
              break;
            }
        }
        //}
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.habilidadevaga"));
      }
      //seta parametros para caso de
sucesso(listarhabilidadesvaga.jsp)
      String codVaga = habilidadeform.getCodVaga().toString();
      String empresa = "";
      String cnpj = "";
      Empresa emp = new Empresa();
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      if(user.getStatus() != null){
        if("empresa".equals(user.getStatus())){
          try {
            emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
          }
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        }
      }
      Vaga vaga = new Vaga();
      try {
        VagaDao vagaDao = new VagaDao();
        vaga = vagaDao.buscarVaga(Integer.valueOf(codVaga));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.vaga"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.vaga"));
      }
      if(Character.valueOf('S').equals(vaga.getEstagioInterno())){
        cnpj = vaga.getCnpj();
        EmpresaDao empresaDao = new EmpresaDao();
        emp = empresaDao.buscarEmpresa(cnpj);
        if(emp != null){
          empresa = emp.getRazaoSocial();
        }
      }
      HabilidadesVagaDao habilidadeVagaDao = new
HabilidadesVagaDao();
      List<HabilidadeVaga> habilidadesVaga = new
ArrayList<HabilidadeVaga>();
      try{
        habilidadesVaga =
habilidadeVagaDao.listarHabilidadeVaga(Integer.valueOf(codVa
ga));
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.habilidadevaga"));
      }
      if(habilidadesVaga.size()>0){
        request.setAttribute("habilidades", habilidadesVaga);
      }
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      request.setAttribute("codvaga", codVaga);
      //parametros setados para caso de
sucesso(listarhabilidadesvaga.jsp)
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return (mapping.findForward(Constants.FAILURE));
      } else {
        return (mapping.findForward(Constants.SUCCESS));
      }
    }
}
```

6.4.234 SalvarSenhaAlteradaAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.dao.AlunoEstagioDao;
import estagio.form.AlterarSenhaForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarSenhaAlteradaAction  extends Action{
```

```
  public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
      try{
        user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        if (user == null){
          return mapping.findForward("welcome");
        }
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE,
            new ActionMessage("error.generico"));
      }
      AlterarSenhaForm alterarSenhaForm = (AlterarSenhaForm)
form;
      AlunoEstagio alunoEstagio = null;
      try{
        alunoEstagio =
alunoEstagioDao.buscarAlunoEstagio(Integer.valueOf(alterarSen
haForm.getCodAlunoEstagio()));
        //armazenha o hashCode da nova senha

alunoEstagio.setSenha(String.valueOf(alterarSenhaForm.getSenh
a().hashCode()));
        try{
          alunoEstagioDao.alterarAlunoEstagio(alunoEstagio);
        }catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.alterar.senha"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.senha"));
        }
      }catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarios"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarios"));
      }
      if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
      } else {
        return mapping.findForward(Constants.SUCCESS);
      }
    }

}
```

6.4.235 SalvarServidorSmtpAction.java

```
package estagio.action;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.ServidorSmtp;
import estagio.dao.Servidor_smtpDao;
import estagio.form.ServidorSmtpForm;
import java.sql.SQLException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarServidorSmtpAction extends Action {
      public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      HttpSession session = request.getSession();
      ServidorSmtpForm serverForm = (ServidorSmtpForm) form;
      AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      Servidor_smtpDao serverDao = new Servidor_smtpDao();
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      ServidorSmtp servidorSmtp = new ServidorSmtp();

servidorSmtp.setCod_servidor_smtp(serverForm.getCod_servidor
_smtp());

servidorSmtp.setEndereco_email(serverForm.getEndereco_email(
));
      servidorSmtp.setPorta_smtp(serverForm.getPorta_smtp());

servidorSmtp.setRequer_autenticacao(Boolean.valueOf(serverFor
m.getRequer_autenticacao() != null?true:false));
      servidorSmtp.setSenha(serverForm.getSenha());
      servidorSmtp.setUrl(serverForm.getUrl());
      servidorSmtp.setUsername(serverForm.getUsername());

      try{
        serverDao.alterarServidorSmtp(servidorSmtp);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.editar.servidorsmtp"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.editar.servidorsmtp"));
      }

      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return (mapping.findForward(Constants.FAILURE));
      } else {
        return (mapping.findForward(Constants.SUCCESS));
      }

    }
}
```

6.4.236 SalvarTermoCompromissoAction.java

```
package estagio.action;
import com.lowagie.text.Document;
import com.lowagie.text.DocumentException;
import com.lowagie.text.Font;
import com.lowagie.text.PageSize;
import com.lowagie.text.Paragraph;
import com.lowagie.text.pdf.PdfWriter;
import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Endereco;
import estagio.bean.Habilidade;
import estagio.bean.PlanoEstagio;
import estagio.bean.TermoCompromisso;
```

```
import estagio.bean.TermoCompromissoDTO;
import estagio.bean.Turno;
import estagio.bean.Vaga;
import estagio.dao.AgentesIntegracaoDao;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.EmpresaDao;
import estagio.dao.PlanosEstagioDao;
import estagio.dao.TermoCompromissoDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.dao.VagaDao;
import estagio.form.NovoTermoCompromissoForm;
import java.io.FileOutputStream;
import java.io.IOException;
import java.sql.SQLException;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.List;
import java.util.SimpleTimeZone;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
public class SalvarTermoCompromissoAction extends Action {
    public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {
    HttpSession session = request.getSession();
    ActionMessages errors = new ActionMessages();
    NovoTermoCompromissoForm novoTermoForm =
(NovoTermoCompromissoForm) form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    if (user == null) {
      return (mapping.findForward("logon"));
    }
    VagaDao vagaDao = new VagaDao();
    AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
    EmpresaDao empresaDao = new EmpresaDao();
    TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
    PlanosEstagioDao planosEstagioDao = new
PlanosEstagioDao();
    TermoCompromisso termoCompromisso = new
TermoCompromisso();
    GregorianCalendar dataInicio = new GregorianCalendar();
    GregorianCalendar dataFim = new GregorianCalendar();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    try {
dataInicio.setTime(sdf.parse(novoTermoForm.getDataInicio()));

dataFim.setTime(sdf.parse(novoTermoForm.getDataFim()));
    } catch (ParseException p){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.data"));
    }
    termoCompromisso.setDataInicio(dataInicio);
    termoCompromisso.setDataFim(dataFim);
termoCompromisso.setAlunoEstagio(alunoEstagioDao.buscarAlu
noEstagio(novoTermoForm.getCodAlunoEstagio()));
termoCompromisso.setEmpresa(empresaDao.buscarEmpresa(nov
oTermoForm.getCnpj()));
    AgentesIntegracaoDao agentesIntegracaoDao = new
AgentesIntegracaoDao();
    AgenteIntegracao agenteIntegracao =
agentesIntegracaoDao.buscarAgenteIntegracao(novoTermoForm.
getCanal());
    if(agenteIntegracao == null){
      agenteIntegracao = new AgenteIntegracao();
    }
    termoCompromisso.setAgenteIntegracao(agenteIntegracao);
    if("true".equals(novoTermoForm.getObrigatorio())){
      termoCompromisso.setObrigatorio("S");
    } else {
      termoCompromisso.setObrigatorio("N");
    }
termoCompromisso.setSeguroCompanhia(novoTermoForm.getSe
guroCompanhia());
termoCompromisso.setSeguroNumApolice(novoTermoForm.getS
eguroNumApolice());

termoCompromisso.setSeguroProvidenciado(termoCompromisso.
getEmpresa().getRazaoSocial());
    PlanoEstagio planoEstagio = new PlanoEstagio();
planoEstagio.setAreaAtividadeEstagiario(novoTermoForm.getAr
eaAtividadeEstagiario());
planoEstagio.setAtividades(novoTermoForm.getAtividades());
planoEstagio.setFormacaoOrientador(novoTermoForm.getFormac
aoOrientador());
planoEstagio.setObjetivos(novoTermoForm.getObjetivos());
planoEstagio.setOrientador(novoTermoForm.getOrientador());
planoEstagio.setCargoOrientador(novoTermoForm.getCargoOrie
ntador());
    planoEstagio.setRamal(novoTermoForm.getRamal());
planoEstagio.setDepartamento(novoTermoForm.getDepartamento
());
    String action = (String) request.getParameter("action");
    boolean interno = false;
    Vaga v = null;
    try{
      v =
vagaDao.buscarVaga(novoTermoForm.getCodVaga());
      termoCompromisso.setVaga(v);
      interno =
Character.valueOf('S').equals(v.getEstagioInterno())?true:false;
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.vaga"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.vaga"));
    }
    if("Create".equalsIgnoreCase(action)){
      try{
        v.setPreenchimento(new GregorianCalendar());
        termoCompromisso.setVaga(v);
termoCompromissoDao.incluirTermoCompromisso(termoCompr
omisso);
        TermoCompromisso termoSalvo =
termoCompromissoDao.buscarTermoCompromisso(termoCompr
omisso.getAlunoEstagio().getCodAlunoEstagio(),
termoCompromisso.getEmpresa().getCnpj(),
termoCompromisso.getVaga().getCodVaga());
planoEstagio.setCodTermo(termoSalvo.getCodTermo());
        planosEstagioDao.incluirPlanoEstagio(planoEstagio);
        termoCompromisso.setPlanoEstagio(planoEstagio);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.termo"));
```

```
        }
      } else {
        try{
          //TermoCompromisso termoAntigo =
termoCompromissoDao.buscarTermoCompromisso(novoTermoF
orm.getCodAlunoEstagio(), novoTermoForm.getCnpj(),
novoTermoForm.getCodVaga());
          String codTermo = request.getParameter("codTermo");
          TermoCompromisso termoAntigo =
termoCompromissoDao.buscarTermoCompromisso(Integer.value
Of(codTermo));

termoCompromisso.setCodTermo(Integer.valueOf(codTermo));

planoEstagio.setCodTermo(Integer.valueOf(codTermo));

planoEstagio.setCodPlanoEstagio(termoAntigo.getPlanoEstagio()
.getCodPlanoEstagio());
          termoCompromisso.setPlanoEstagio(planoEstagio);

termoCompromissoDao.alterarTermoCompromisso(termoCompr
omisso);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.termo"));
        }
      }
      Document document = new Document(PageSize.A4, 30, 30,
40, 40);
      try {
        Font smallFont = new Font(Font.TIMES_ROMAN,9,
Font.NORMAL);
        response.setContentType("application/pdf");
        response.setHeader("Content-disposition",
"filename=termocompromisso.pdf");
        //PdfWriter.getInstance(document, new
FileOutputStream("D://TCC//TccEstagio//build//web//pages//Ter
moCompromisso"+termoSalvo.getCodTermo()+".pdf"));
        PdfWriter.getInstance(document,
response.getOutputStream());
        document.addAuthor("UFPR");
        document.addSubject("Contrato de Estágio");
        document.open();
        // adicionando o titulo ao documento
        String titulo = "TERMO DE COMPROMISSO DE
ESTÁGIO\n VINCULADO AO CONVÊNIO CELEBRADO\n
ENTRE A UNIVERSIDADE FEDERAL DO\n PARANÁ E "
+termoCompromisso.getEmpresa().getRazaoSocial().toUpperCas
e();
        if(interno){
          titulo = "TERMO DE COMPROMISSO PARA
ESTÁGIO\nPARA ALUNOS DA UNIVERSIDADE FEDERAL
DO\nPARANÁ (INSTRUÇÃO NORMATIVA Nº 01/92-CEP)";
        }
        Paragraph para = new Paragraph(titulo, smallFont);
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        para = new Paragraph(" ");
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        // adicionando o primeiro parágrafo ao documento
        String ano = "";
        int anoInt;
        if(termoCompromisso.getAlunoEstagio().getSemestre()/2
!= 0){
if(termoCompromisso.getAlunoEstagio().getSemestre()%2 != 0){
          anoInt =
(termoCompromisso.getAlunoEstagio().getSemestre()/2)+1;
        } else {
          anoInt =
termoCompromisso.getAlunoEstagio().getSemestre()/2;
        }
        } else {
          anoInt = 1;
        }
if(anoInt>termoCompromisso.getAlunoEstagio().getCurso().getD
uracao().intValue()){
          anoInt =
termoCompromisso.getAlunoEstagio().getCurso().getDuracao().i
ntValue();
        }
        ano = Integer.valueOf(anoInt).toString();
        String primeiroParagrafo = "    A
"+termoCompromisso.getEmpresa().getRazaoSocial()
        +", sediada à Rua
"+termoCompromisso.getEmpresa().getEnderecoEmpresa().getRu
a()
        +",
n°"+termoCompromisso.getEmpresa().getEnderecoEmpresa().get
Numero()
        +", Cidade
"+termoCompromisso.getEmpresa().getEnderecoEmpresa().getM
unicipio()
        +", CEP
"+termoCompromisso.getEmpresa().getEnderecoEmpresa().getCe
p()
        +", CGC "+termoCompromisso.getEmpresa().getCnpj()
        +", Fone "+termoCompromisso.getEmpresa().getFone()
        +" doravante denominada Unidade Concedente por seu
representante
"+termoCompromisso.getEmpresa().getRepresentante()
        +" e de outro lado,
"+termoCompromisso.getAlunoEstagio().getNomePessoa()
        +", RG n°
"+termoCompromisso.getAlunoEstagio().getRg()
        +", CPF
"+termoCompromisso.getAlunoEstagio().getCpf()
        +", estudante do "+ano
          +"º ano do Curso de
"+termoCompromisso.getAlunoEstagio().getCurso().getNomeCur
so()
          +", Matrícula n°
"+termoCompromisso.getAlunoEstagio().getMatricula()
          +", residente à Rua
"+termoCompromisso.getAlunoEstagio().getRua()
          +" ,
n°"+termoCompromisso.getAlunoEstagio().getNumero()

+(!"".equals(termoCompromisso.getAlunoEstagio().getCompleme
nto())?",
"+termoCompromisso.getAlunoEstagio().getComplemento():"")
          +", na Cidade de
"+termoCompromisso.getAlunoEstagio().getMunicipio()
          +", Estado
"+termoCompromisso.getAlunoEstagio().getEstado()
          +", CEP
"+termoCompromisso.getAlunoEstagio().getCep()
          +", Fone
"+termoCompromisso.getAlunoEstagio().getFone()
          +" , Data de Nascimento
"+sdf.format(termoCompromisso.getAlunoEstagio().getDataNasc
imento().getTime())
          +", doravante
denominad"+("M".equalsIgnoreCase(termoCompromisso.getAlun
oEstagio().getSexo())?"o":"a")
          +"
Estagiári"+("M".equalsIgnoreCase(termoCompromisso.getAluno
Estagio().getSexo())?"o":"a")
          +", com interviniência da Instituição de Ensino,
celebram o presente Termo de Compromisso, mediante as
seguintes cláusulas e condições:";
```

```
        if(interno){
          primeiroParagrafo = "A UNIVERSIDADE FEDERAL
DO PARANÁ, sediada à Rua XV de Novembro n.º 1299 Curitiba
CEP 80020-300 PR CGC 75095679/0001-49 Fone 310-2656 ou
310-2675, doravante denominada UNIDADE CONCEDENTE
representada neste ato por seu Reitor e de outro lado, "

+termoCompromisso.getAlunoEstagio().getNomePessoa()
                +", RG nº
"+termoCompromisso.getAlunoEstagio().getRg()
                +", CPF
"+termoCompromisso.getAlunoEstagio().getCpf()
                +", estudante do "+ano
                +"º ano do Curso de
"+termoCompromisso.getAlunoEstagio().getCurso().getNomeCur
so()
                +", Matrícula nº
"+termoCompromisso.getAlunoEstagio().getMatricula()
                +", residente à Rua
"+termoCompromisso.getAlunoEstagio().getRua()
                +" ,
nº"+termoCompromisso.getAlunoEstagio().getNumero()

+(!"".equals(termoCompromisso.getAlunoEstagio().getCompleme
nto())?",
"+termoCompromisso.getAlunoEstagio().getComplemento():"")
                +", na Cidade de
"+termoCompromisso.getAlunoEstagio().getMunicipio()
                +", Estado
"+termoCompromisso.getAlunoEstagio().getEstado()
                +", CEP
"+termoCompromisso.getAlunoEstagio().getCep()
                +", Fone
"+termoCompromisso.getAlunoEstagio().getFone()
                +" , Data de Nascimento
"+sdf.format(termoCompromisso.getAlunoEstagio().getDataNasc
imento().getTime())
                +", doravante
denominad"+("M".equalsIgnoreCase(termoCompromisso.getAlun
oEstagio().getSexo())?"o":"a")
                +"
Estagiári"+("M".equalsIgnoreCase(termoCompromisso.getAluno
Estagio().getSexo())?"o":"a")
                +" com interveniência da Instituição de Ensino,
celebram o presente Termo de Compromisso, mediante as
seguintes cláusulas e condições:";
        }
        para = new Paragraph(primeiroParagrafo, smallFont);
        para.setAlignment(para.ALIGN_JUSTIFIED);
        document.add(para);
        para = new Paragraph(" ");
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        // adicionando clausula primeira ao documento
        String clausulaPrimeira = "CLÁUSULA PRIMEIRA -
As atividades a serem desenvolvidas durante o Estágio constam
de programação acordada entre as partes – Plano de Estágio no
verso – e terão por finalidade propiciar ao Estagiário uma
experiência acadêmica-profissional em um campo de trabalho
determinado, visando:";
        para = new Paragraph(clausulaPrimeira, smallFont);
        para.setAlignment(para.ALIGN_JUSTIFIED);
        document.add(para);
        String alineaA = "a) o aprimoramento técnico-científico
em sua formação;";
        para = new Paragraph(alineaA, smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
        String alineaB = "b) a maior proximidade do aluno, com
as condições reais de trabalho, por intermédio de práticas afins
com a natureza e especificidade de cada curso.";
        para = new Paragraph(alineaB, smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
        // adicionando clausula segunda ao documento
        String clausulaSegunda = "CLÁUSULA SEGUNDA -
O estágio será desenvolvido no período de "

+sdf.format(termoCompromisso.getDataInicio().getTime())
                +" a "

+sdf.format(termoCompromisso.getDataFim().getTime())
                +", (correspondente, mínimo, a um semestre letivo) "
                +", no horário ";
        for(int i = 0;i <
termoCompromisso.getVaga().getTurnos().size();i++){
          Turno turno =
termoCompromisso.getVaga().getTurnos().get(i);
          if( termoCompromisso.getVaga().getTurnos().size() - 1
!= i){
            clausulaSegunda += turno.toString()+" e ";
          } else {
            clausulaSegunda += turno.toString()+"";
          }
        }
        clausulaSegunda+=", num total de "+
            +termoCompromisso.getVaga().getHoras()
            +" hs semanais, compatíveis com o horário escolar
podendo ser denunciado a qualquer tempo, unilateralmente e
mediante comunicação escrita, ou ser prorrogado, através de
emissão de Termo Aditivo;";
        if(interno){
          clausulaSegunda = "CLÁUSULA SEGUNDA - As
atividades principais a serem desenvolvidas pelo Estagiário,
compatíveis com o contexto básico da profissão da qual o curso se
refere, são as seguintes:\n"
              +planoEstagio.getAtividades()
              +" tendo como orientador "
              +planoEstagio.getOrientador()
              +", "+planoEstagio.getFormacaoOrientador()
              +", "+planoEstagio.getCargoOrientador()+".";
        }
        para = new Paragraph(clausulaSegunda, smallFont);
        para.setAlignment(para.ALIGN_JUSTIFIED);
        document.add(para);
        // adicionando clausula terceira ao documento
        String clausulaTerceira = "CLÁUSULA TERCEIRA -
Na vigência deste Termo de Compromisso o Estagiário será
protegido contra Acidentes Pessoais, providenciado pela "

+termoCompromisso.getEmpresa().getRazaoSocial()+" e
representado pela Apólice nº "
            +termoCompromisso.getSeguroNumApolice()+" da
Companhia "
            +termoCompromisso.getSeguroCompanhia()+" em
conformidade com o que preceitua o Artigo 4º da Lei nº 6494/77,
de 07 de dezembro de 1977.";
        if(interno){
          clausulaTerceira = "CLÁUSULA TERCEIRA - O
estágio será desenvolvido no período de "

+sdf.format(termoCompromisso.getDataInicio().getTime())
                +" a "

+sdf.format(termoCompromisso.getDataFim().getTime())
                +", (correspondente, mínimo, a um semestre letivo)
"
                +", no horário ";
          for(int i = 0;i <
termoCompromisso.getVaga().getTurnos().size();i++){
            Turno turno =
termoCompromisso.getVaga().getTurnos().get(i);
            if( termoCompromisso.getVaga().getTurnos().size() -
1 != i){
```

```
            clausulaTerceira += turno.toString()+" e ";
          } else {
            clausulaTerceira += turno.toString()+"";
          }
        }
        if(termoCompromisso.getVaga().getTurnos().size() ==
0){
          clausulaTerceira+=" não informado ";
        }
        clausulaTerceira+=", num total de "+
            +termoCompromisso.getVaga().getHoras()
            +" hs semanais, compatíveis com o horário escolar
podendo ser denunciado a qualquer tempo, unilateralmente e
mediante comunicação escrita, ou ser prorrogado por uma única
vez, através de emissão de Termo Aditivo;";
      }
      String paragrafoUnico = "Parágrafo Único - Em caso
do presente estágio ser prorrogado, o preenchimento e a assinatura
do Termo Aditivo deverão ser providenciados antes da data de
encerramento, contida na Cláusula Terceira neste Termo de
Compromisso;";
      para = new Paragraph(clausulaTerceira, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      if(interno){
        para = new Paragraph(paragrafoUnico, smallFont);
        para.setAlignment(para.ALIGN_JUSTIFIED);
        document.add(para);
      }
      // adicionando clausula quarta ao documento
      String clausulaQuarta = "CLÁUSULA QUARTA -
Durante o período de Estágio, o estudante poderá receber
uma Bolsa Auxílio, no valor de R$"
          +termoCompromisso.getVaga().getSalario()
          +", paga mensalmente pela Unidade Concedente;";
      if(interno){
        clausulaQuarta = "CLÁUSULA QUARTA - Na
vigência deste Termo de Compromisso o Estagiário será
protegido contra Acidentes Pessoais, providenciado pela "

+termoCompromisso.getEmpresa().getRazaoSocial()+" e
representado pela Apólice nº "
          +termoCompromisso.getSeguroNumApolice()+"
da Companhia "
          +termoCompromisso.getSeguroCompanhia()+" em
conformidade com o que preceitua o Artigo 4º da Lei nº 6494/77,
de 07 de dezembro de 1977.";
      }
      para = new Paragraph(clausulaQuarta, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      // adicionando clausula quinta ao documento
      String clausulaQuinta = "CLÁUSULA QUINTA -
Caberá ao Estagiário cumprir a programação estabelecida,
observando as normas internas da Unidade Concedente, bem
como, elaborar relatório referente ao Estágio quando solicitado
pela Unidade Concedente ou pela Instituição de Ensino;";
      if(interno){
        clausulaQuinta = "CLÁUSULA QUINTA - Durante
o período de Estágio, o estudante poderá receber uma Bolsa
Auxílio, no valor de R$"
          +termoCompromisso.getVaga().getSalario()
          +", paga mensalmente pela Unidade Concedente,
desde que cumpra o mínimo de 20 horas semanais, de acordo com
o item 3 da instituição Normativa -SAF.";
      }
      para = new Paragraph(clausulaQuinta, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      // adicionando clausula sexta ao documento
      String clausulaSexta = "CLÁUSULA SEXTA - O
Estagiário responderá pelas perdas e danos decorrentes da
inobservância das normas internas ou das constantes no presente
contrato;";
      if(interno){
        clausulaSexta = "CLÁUSULA SEXTA - Caberá ao
Estagiário cumprir a programação estabelecida, observando as
normas internas da Unidade Concedente, bem como, elaborar
relatório referente ao Estágio quando solicitado pela Unidade
Concedente ou pela Instituição de Ensino;";
      }
      para = new Paragraph(clausulaSexta, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      // adicionando clausula setima ao documento
      String clausulaSetima = "CLÁUSULA SÉTIMA - Nos
termos do Artigo 4º da Lei nº 6496/77 e do Artigo 6º do Decreto
nº 87.494/82, o Estagiário não terá, para quaisquer efeitos,
vínculo empregatício com a Unidade Concedente;";
      if(interno){
        clausulaSetima = "CLÁUSULA SÉTIMA - O
Estagiário responderá pelas perdas e danos decorrentes da
inobservância das normas internas ou das constantes no presente
contrato;";
      }
      para = new Paragraph(clausulaSetima, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      // adicionando clausula oitava ao documento
      String clausulaOitava= "CLÁULULA OITAVA -
Constituem motivo para interrupção automática da vigência do
presente Termo de Compromisso de Estágio;";
      if(interno){
        clausulaOitava = "CLÁUSULA OITAVA - Nos
termos do Artigo 4º da Lei nº 6496/77 e do Artigo 6º do Decreto
nº 87.494/82, o Estagiário não terá, para quaisquer efeitos,
vínculo empregatício com a Unidade Concedente;";
      }
      para = new Paragraph(clausulaOitava, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      // adicionando clausula nona ao documento - para estagio
interno
      String clausulaNona = "CLÁULULA NONA -
Constituem motivo para interrupção automática da vigência do
presente Termo de Compromisso de Estágio;";
      if(interno){
        para = new Paragraph(clausulaNona, smallFont);
        para.setAlignment(para.ALIGN_JUSTIFIED);
        document.add(para);
      }
      alineaA = "a) a conclusão ou abandono do curso e o
trancamento de matrícula;";
      para = new Paragraph(alineaA, smallFont);
      para.setAlignment(para.ALIGN_LEFT);
      document.add(para);
      alineaB = "b) a pedido do estagiário;";
      para = new Paragraph(alineaB, smallFont);
      para.setAlignment(para.ALIGN_LEFT);
      document.add(para);
      String alineaC = "c) o não cumprimento do convencionado
neste Termo de Compromisso.";
      if(interno){
        alineaC = "c) a pedido da Instituição de Ensino";
      }
      para = new Paragraph(alineaC, smallFont);
      para.setAlignment(para.ALIGN_LEFT);
      document.add(para);
      if(interno){
        String alineaD = "d) a pedido da Unidade Concedente;";
        para = new Paragraph(alineaD, smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
```

```
        String alineaE = "e) o não cumprimento do
convencionado neste Termo de Compromisso.";
        para = new Paragraph(alineaE, smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
      }
      // adicionando ultimo paragrafo ao documento
      String ultimoParagrafo = "E, por estar de inteiro e comum
acordo com as condições deste Termo de Compromisso , as partes
assinam em 04 (quatro) vias de igual teor.";
      para = new Paragraph(ultimoParagrafo, smallFont);
      para.setAlignment(para.ALIGN_JUSTIFIED);
      document.add(para);
      // adicionando a data ao documento
      // Cria uma TIME ZONE correspondente ao horário de
Brasília
      SimpleTimeZone pdt = new
          SimpleTimeZone(-3 * 60 * 60 * 1000,"GMT-3:00");
      // Seta as regras para o horário de verão Brasileiro
      // Começando no primeiro domingo após o dia primeiro
      pdt.setStartRule(Calendar.NOVEMBER, 1,
Calendar.SUNDAY,0);
      // Terminando no último domingo do mês de Fevereiro
      pdt.setEndRule(Calendar.FEBRUARY, -1,
Calendar.SUNDAY,0);
      // Instanciando um GregorianCalendar com com a time
zone de BSB
      // e levando em consideração as regras do horário de
verão.
      Calendar dataHoje = new GregorianCalendar(pdt);
      String mes;
      switch (dataHoje.get(Calendar.MONTH)){
        case 0:
          mes = "janeiro";
          break;
        case 1:
          mes = "fevereiro";
          break;
        case 2:
          mes = "março";
          break;
        case 3:
          mes = "abril";
          break;
        case 4:
          mes = "maio";
          break;
        case 5:
          mes = "junho";
          break;
        case 6:
          mes = "julho";
          break;
        case 7:
          mes = "agosto";
          break;
        case 8:
          mes = "setembro";
          break;
        case 9:
          mes = "outubro";
          break;
        case 10:
          mes = "novembro";
          break;
        case 11:
          mes = "dezembro";
          break;
        default:
          mes = "";
      }
      para = new Paragraph("Curitiba,
"+dataHoje.get(Calendar.DAY_OF_MONTH)+" de "+mes+" de
"+dataHoje.get(Calendar.YEAR)+".", smallFont);
      para.setAlignment(para.ALIGN_CENTER);
      document.add(para);
      para = new Paragraph(" ");
      para.setAlignment(para.ALIGN_CENTER);
      document.add(para);
      //adiciona campos para assinaturas
      para = new
Paragraph("_____
_____",
smallFont);
      para.setAlignment(para.ALIGN_LEFT);
      document.add(para);
      String assinatura = "UNIDADE CONCEDENTE
ESTAGIÁRIO";
      if(interno){
        assinatura += "\nCOORDENAÇÃO GERAL DE
ESTÁGIO";
      }
      para = new Paragraph(assinatura, smallFont);
      para.setAlignment(para.ALIGN_LEFT);
      document.add(para);
      para = new Paragraph("(assinatura e carimbo)
(assinatura)", smallFont);
      para.setAlignment(para.ALIGN_LEFT);
      document.add(para);
      para = new Paragraph(" ");
      para.setAlignment(para.ALIGN_CENTER);
      document.add(para);
      para = new Paragraph(" ");
      para.setAlignment(para.ALIGN_CENTER);
      document.add(para);
      if(!interno){
        para = new
Paragraph("_____
_____",
smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
        para = new Paragraph("COORDENADOR DO
CURSO – UFPR
COORDENAÇÃO GERAL DE ESTÁGIOS", smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
        para = new Paragraph("(assinatura e carimbo)
(assinatura e carimbo)", smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
      } else {
        para = new
Paragraph("_____",
smallFont);
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        para = new Paragraph("COORDENADOR DO
CURSO", smallFont);
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        para = new Paragraph("(assinatura e carimbo)",
smallFont);
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
      }
      // Adiciona mais uma página
      document.newPage();
      titulo = "PLANO DE ESTÁGIO\nINSTRUÇÃO
NORMATIVA N° 01/03-CEPE";
      if(interno){
        titulo = "Ficha nº 1 – PLANO DE
ESTÁGIO\n(Instrução Normativa nº 01/92 CEP)";
```

```
        }
        para = new Paragraph(titulo, smallFont);
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        para = new Paragraph(" ");
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        para = new Paragraph(" ");
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        //Adiciona o primeiro parágrafo
        if(interno){
          para = new Paragraph("1. IDENTIFICAÇÃO DO
ESTÁGIO:", smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
        }
        String obrigatorio = request.getParameter("obrigatorio");
        if(!"true".equals(obrigatorio)){
          primeiroParagrafo = "(      ) ESTÁGIO
OBRIGATÓRIO              ( X ) ESTÁGIO NÃO
OBRIGATÓRIO";
        } else {
          primeiroParagrafo = "( X ) ESTÁGIO
OBRIGATÓRIO              (      ) ESTÁGIO NÃO
OBRIGATÓRIO";
        }
        para = new Paragraph(primeiroParagrafo, smallFont);
        para.setAlignment(para.ALIGN_CENTER);
        document.add(para);
        if(!interno){
          para = new Paragraph(" ");
          para.setAlignment(para.ALIGN_CENTER);
          document.add(para);
          para = new Paragraph(" ");
          para.setAlignment(para.ALIGN_CENTER);
          document.add(para);
        }
        //Adiciona campo 1
        String campo = "";
        if(!interno){
          campo += "01.  ";
        }
        campo = "Nome do aluno (a):
"+termoCompromisso.getAlunoEstagio().getNomePessoa();
        para = new Paragraph(campo, smallFont);
        para.setAlignment(para.ALIGN_LEFT);
        document.add(para);
        if(interno){
          Endereco end = new Endereco();

end.setBairro(termoCompromisso.getAlunoEstagio().getBairro());

end.setCep(termoCompromisso.getAlunoEstagio().getCep());

end.setComplemento(termoCompromisso.getAlunoEstagio().getC
omplemento());

end.setEstado(termoCompromisso.getAlunoEstagio().getEstado())
;

end.setMunicipio(termoCompromisso.getAlunoEstagio().getMuni
cipio());

end.setNumero(Integer.valueOf(termoCompromisso.getAlunoEst
agio().getNumero()).intValue());

end.setPais(termoCompromisso.getAlunoEstagio().getPais());

end.setRua(termoCompromisso.getAlunoEstagio().getRua());
          campo = "Endereço: "+end.toString();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Fone:
"+termoCompromisso.getAlunoEstagio().getFone();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Nome do curso:
"+termoCompromisso.getAlunoEstagio().getCurso().getNomeCur
so();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Nº de Matrícula:
"+termoCompromisso.getAlunoEstagio().getMatricula();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Ano/período que está cursando:
"+termoCompromisso.getAlunoEstagio().getSemestre();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Disciplina concomitante ao estágio:
_____"
;
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
        }
        //Adiciona campo 2
        if(interno){
          para = new Paragraph(" ");
          para.setAlignment(para.ALIGN_CENTER);
          document.add(para);
          campo = "2. DADOS REFERENTES AO LOCAL DE
ESTÁGIO:";
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Unidade/Departamento:
"+planoEstagio.getDepartamento();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Ramal: "+planoEstagio.getRamal();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Nome do Orientador:
"+planoEstagio.getOrientador();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Cargo ou função:
"+planoEstagio.getCargoOrientador();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
          campo = "Formação Profissional:
"+planoEstagio.getFormacaoOrientador();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
        } else {
          campo = "02.  Nome do orientador de estágio na
unidade concedente: "+planoEstagio.getOrientador();
          para = new Paragraph(campo, smallFont);
          para.setAlignment(para.ALIGN_LEFT);
          document.add(para);
        }
        //Adiciona campo 3
```

```
        if(interno){
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            campo = "03. DESENVOLVIMENTO";
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            campo = "3.1 - Objetivo:
"+planoEstagio.getObjetivos();
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            campo = "3.2 - Atividades previstas:
"+planoEstagio.getAtividades();
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            campo = "3.3 - Cronograma:";
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            campo = "Data de início:
"+sdf.format(termoCompromisso.getDataInicio().getTime());
            campo += "     Previsão de término:
"+sdf.format(termoCompromisso.getDataFim().getTime());;
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            String turnos = "";
            for(int i = 0;i <
termoCompromisso.getVaga().getTurnos().size();i++){
                Turno turno =
termoCompromisso.getVaga().getTurnos().get(i);
                if( termoCompromisso.getVaga().getTurnos().size() -
1 != i){
                    turnos += turno.toString()+", ";
                } else {
                    turnos += turno.toString()+".";
                }
            }
            campo = "Horário: "+turnos;
            campo+= "Dias da Semana: ";
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            campo = "Obs.:
_____
_____";
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
        } else {
            campo = "03.  Formação profissional do orientador:
"+planoEstagio.getFormacaoOrientador();
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
        }
        if(!interno){
            //Adiciona campo 4
            campo = "04.  Ramo de atividade da Unidade
Concedente:
"+(!"true".equals(obrigatorio)?termoCompromisso.getEmpresa().
getRamo():"");
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            //Adiciona campo 5
            campo = "05.  Área de atividade do(a) estagiário(a):
"+planoEstagio.getAreaAtividadeEstagiario();
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            //Adiciona campo 6
            campo = "06.  Objetivos:
"+planoEstagio.getObjetivos();
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            //Adiciona campo 7
            campo = "07.  Atividades a serem desenvolvidas:
"+planoEstagio.getAtividades();
            para = new Paragraph(campo, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            //Adiciona outro paragrafo
            para = new Paragraph("A SER PREENCHIDA PELA
COE", smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            //Adiciona campo 8
            para = new Paragraph("08.  Professor supervisor –
UFPR (Para emissão de certificado): ", smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            alineaA = "a) Modalidade da supervisão:    [ ] Direta
[ ] Semi-Direta    [ ] Indireta";
            para = new Paragraph(alineaA, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            alineaB = "b) Número de horas da supervisão no
período: _____";
            para = new Paragraph(alineaB, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            alineaC = "c) Número de estagiários concomitantes com
esta supervisão: _____";
            para = new Paragraph(alineaC, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            //adiciona campos para assinaturas
            para = new
Paragraph("_____
_____",
smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("          Estagiário
Orientador de estágio na unidade concedente", smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("(assinatura)
(assinatura e carimbo)", smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
```

```
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new
Paragraph("_____",
smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Professor Supervisor – UFPR",
smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("(assinatura)", smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new
Paragraph("_____",
smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Comissão Orientadora de
Estágio (COE) do Curso", smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("(assinatura)", smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
          } else {
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Curitiba,
"+dataHoje.get(Calendar.DAY_OF_MONTH)+" de "+mes+" de
"+dataHoje.get(Calendar.YEAR)+".", smallFont);
            para.setAlignment(para.ALIGN_RIGHT);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Assinatura do Aluno:
_____", smallFont);
            para.setAlignment(para.ALIGN_RIGHT);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Visto do orientador:
_____",
smallFont);
            para.setAlignment(para.ALIGN_RIGHT);
            document.add(para);
            //Adiciona outro paragrafo
            para = new Paragraph("A SER PREENCHIDA PELA
COE", smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            //Adiciona campo 8
            para = new Paragraph("08.  Professor supervisor –
UFPR (Para emissão de certificado): ", smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            alineaA = "a) Modalidade da supervisão:    [ ] Direta
[ ] Semi-Direta    [ ] Indireta";
            para = new Paragraph(alineaA, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            alineaB = "b) Número de horas da supervisão no
período: _____";
            para = new Paragraph(alineaB, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            alineaC = "c) Número de estagiários concomitantes com
esta supervisão: _____";
            para = new Paragraph(alineaC, smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Parecer da Comissão
Orientadora de Estágio (COE), em se tratando de aluno da UFPR ,
em ESTÁGIO NÃO
OBRIGATÓRIO_____
_____
_____
_____", smallFont);
            para.setAlignment(para.ALIGN_LEFT);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph(" ");
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
            para = new Paragraph("Curitiba,
"+sdf.format(dataHoje.getTime())+"      "+"Assinatura:
_____",
smallFont);
            para.setAlignment(para.ALIGN_CENTER);
            document.add(para);
          }
        } catch(DocumentException de) {
          System.err.println(de.getMessage());
        } catch(IOException ioe) {
          System.err.println(ioe.getMessage());
        }
        document.close();
        //seta parametros para o caso de
Erro(novotermocompromisso.jsp)
        request.setAttribute("codvaga",
novoTermoForm.getCodVaga());
        request.setAttribute("codalunoestagio",
novoTermoForm.getCodAlunoEstagio());
        UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
        String empresa = "";
        String cnpj = "";
        /*Empresa emp = new Empresa();
        if(user.getStatus() != null){
          if("empresa".equals(user.getStatus())){
            try {
              emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
            } catch (SQLException e){
              errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
            }catch (Exception e) {
```

```
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
          }
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        } else {*/
      empresa =
termoCompromisso.getEmpresa().getRazaoSocial();
      cnpj = termoCompromisso.getEmpresa().getCnpj();
      //}
      //}
      request.setAttribute("empresa", empresa);
      request.setAttribute("cnpj", cnpj);
      //parametros setados para o caso de erro
      //parametros para o caso de sucesso(listartermos.jsp)

      List<TermoCompromisso> termoCompromissos = new
ArrayList<TermoCompromisso>();
      try{
        termoCompromissos =
termoCompromissoDao.listarTermos(cnpj);
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.termocompromisso"));
      }
      List<TermoCompromissoDTO> tDtos = new
ArrayList<TermoCompromissoDTO>();
      for(TermoCompromisso t:termoCompromissos){
        TermoCompromissoDTO tDto = new
TermoCompromissoDTO();
        tDto.setCodTermo(t.getCodTermo());
        tDto.setEstagiario(t.getAlunoEstagio().getNomePessoa());

tDto.setDataInicio(sdf.format(t.getDataInicio().getTime()));
        tDto.setDataFim(sdf.format(t.getDataFim().getTime()));
        tDto.setEmpresa(t.getEmpresa().getRazaoSocial());

tDto.setAgenteIntegracao(t.getAgenteIntegracao()!=null?t.getAge
nteIntegracao().getAgente():"nenhum");
        tDto.setSalario(t.getVaga().getSalario());
        tDto.setSetor(t.getVaga().getSetor());

tDto.setCurso(t.getAlunoEstagio().getCurso().getNomeCurso());
        String turnos = "";
        for(int i = 0;i<t.getVaga().getTurnos().size();i++){
          Turno turno= t.getVaga().getTurnos().get(i);
          if( t.getVaga().getTurnos().size() - 1 != i){
            turnos += turno.toString()+", ";
          } else {
            turnos += turno.toString()+".";
          }
        }
        tDto.setHorario("".equals(turnos)?"Não
Informado":turnos);
        String descricaoHabilidade = "";
        for(int i = 0;i<t.getVaga().getHabilidades().size();i++){
          Habilidade habilidade =
t.getVaga().getHabilidades().get(i);
          if(t.getVaga().getHabilidades().size() - 1 != i){
            descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
          } else {
            descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
          }
        }
        tDto.setRequisitos("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);

        tDtos.add(tDto);
      }
      if(termoCompromissos.size()>0){
        request.setAttribute("termos", tDtos);
      }
      if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return (mapping.findForward(Constants.FAILURE));
      } else {
        return (mapping.findForward(Constants.SUCCESS));
      }
    }
}
```

6.4.237 SalvarTurnoAction.java

```
/*
 * SalvarTurnoAction.java
 * Created on 16 de Fevereiro de 2006, 17:31
 */

package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Turno;
import estagio.bean.TurnoDTO;
import estagio.dao.TurnoDao;
import estagio.form.NovoTurnoForm;
import java.sql.SQLException;
import java.sql.Time;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarTurnoAction extends Action {

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {


      HttpSession session = request.getSession();
      NovoTurnoForm turnoform = (NovoTurnoForm) form;
      AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      TurnoDao turnoDao = new TurnoDao();
      String codVaga = turnoform.getCodVaga().toString();
      request.setAttribute("codvaga", codVaga);
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      Turno novoTurno = new Turno();
      novoTurno.setCodVaga(turnoform.getCodVaga());
```

```
    long horasInicio =
Long.valueOf(turnoform.getHoraInicio().substring(0,2));
    long minutosInicio =
Long.valueOf(turnoform.getHoraInicio().substring(3));
    long timeInMillis = 3*60*60*1000 +
horasInicio*60*60*1000 + minutosInicio*60*1000;
    Time horaInicio = new Time(timeInMillis);
    novoTurno.setHoraInicio(horaInicio);
    long horasFim =
Long.valueOf(turnoform.getHoraFim().substring(0,2));
    long minutosFim =
Long.valueOf(turnoform.getHoraFim().substring(3));
    timeInMillis = 3*60*60*1000 + horasFim*60*60*1000 +
minutosFim*60*1000;
    Time horaFim = new Time(timeInMillis);
    novoTurno.setHoraFim(horaFim);
    List<Turno> listaTurnos = new ArrayList<Turno>();
    try{
      listaTurnos =
turnoDao.listarTurnos(Integer.valueOf(codVaga));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    try{
      boolean turnoInvalido = false;
      for(Turno t:listaTurnos){
        if(novoTurno.getHoraInicio().after(t.getHoraInicio())
&& novoTurno.getHoraInicio().before(t.getHoraFim()) ||
            (novoTurno.getHoraFim().after(t.getHoraInicio())
&& novoTurno.getHoraFim().before(t.getHoraFim()))){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.turno.invalido"));
          turnoInvalido = true;
          break;
        }
      }
      if(!turnoInvalido){
        turnoDao.incluirTurno(novoTurno);
      }
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.turno"));
    }
    //parametros para listar turnos
    try{
      listaTurnos =
turnoDao.listarTurnos(Integer.valueOf(codVaga));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
    String turnos = "";
    for(int i = 0;i < listaTurnos.size();i++){
      Turno turno = listaTurnos.get(i);
      TurnoDTO turnoDto = new TurnoDTO();
      turnoDto.setCodTurno(turno.getCodTurno());
      turnoDto.setTurno(turno.toString());
      turnosDTO.add(turnoDto);
      if( listaTurnos.size() - 1 != i){
        turnos += turno.toString()+", ";
      } else {
        turnos += turno.toString()+".";
      }
    }
    if(listaTurnos.size()>0){
      request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      request.setAttribute("turnosDto", turnosDTO);
    }
    //fim dos parametros para listar turnos
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.238 SalvarTurnoDisponivelAction.java

```
/*
 * SalvarTurnoDisponivelAction.java
 * Created on 14 de Fevereiro de 2006, 15:35
 */

package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Turno;
import estagio.bean.TurnoDTO;
import estagio.bean.TurnoDisponivel;
import estagio.dao.TurnoDisponivelDao;
import estagio.form.TurnoDisponivelForm;
import java.sql.SQLException;
import java.sql.Time;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
 */
public class SalvarTurnoDisponivelAction extends Action {

  public ActionForward execute(
      ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response)
      throws Exception {


    HttpSession session = request.getSession();
    TurnoDisponivelForm turnodisponivelform =
(TurnoDisponivelForm) form;
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
    String codAlunoEstagio =
user.getCodAlunoEstagio().toString();
```

```
    request.setAttribute("codalunoestagio", codAlunoEstagio);
    if (user == null) {
      return (mapping.findForward("logon"));
    }
    ActionMessages errors = new ActionMessages();
    TurnoDisponivel turnoDisponivel = new TurnoDisponivel();

turnoDisponivel.setCodAlunoEstagio(turnodisponivelform.getCo
dAlunoEstagio());
    long horasInicio =
Long.valueOf(turnodisponivelform.getHoraInicio().substring(0,2)
);
    long minutosInicio =
Long.valueOf(turnodisponivelform.getHoraInicio().substring(3));
    long timeInMillis = 3*60*60*1000 +
horasInicio*60*60*1000 + minutosInicio*60*1000;
    Time horaInicio = new Time(timeInMillis);
    turnoDisponivel.setHoraInicio(horaInicio);
    long horasFim =
Long.valueOf(turnodisponivelform.getHoraFim().substring(0,2));
    long minutosFim =
Long.valueOf(turnodisponivelform.getHoraFim().substring(3));
    timeInMillis = 3*60*60*1000 + horasFim*60*60*1000 +
minutosFim*60*1000;
    Time horaFim = new Time(timeInMillis);
    turnoDisponivel.setHoraFim(horaFim);
    List<TurnoDisponivel> turnosDisponiveis = new
ArrayList<TurnoDisponivel>();
    try{
      turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    try{
      boolean turnoInvalido = false;
      for(TurnoDisponivel t:turnosDisponiveis){

if((turnoDisponivel.getHoraInicio().after(t.getHoraInicio()) &&
turnoDisponivel.getHoraInicio().before(t.getHoraFim())) ||

(turnoDisponivel.getHoraFim().after(t.getHoraInicio()) &&
turnoDisponivel.getHoraFim().before(t.getHoraFim()))){
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.turno.invalido"));
          turnoInvalido = true;
          break;
        }
      }
      if(!turnoInvalido){

turnoDisponivelDao.incluirTurnoDisponivel(turnoDisponivel);
      }
    }catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.turno"));
    }
    try{
      turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(Integer.valueOf(cod
AlunoEstagio));
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.turnos"));
    }
    //parametros para listar turnos
    List<TurnoDTO> turnosDTO = new
ArrayList<TurnoDTO>();
    String turnos = "";
    for(int i = 0;i < turnosDisponiveis.size();i++){
      TurnoDisponivel turno = turnosDisponiveis.get(i);
      TurnoDTO turnoDto = new TurnoDTO();
      turnoDto.setCodTurno(turno.getCodTurnoDisponivel());
      turnoDto.setTurno(turno.toString());
      turnosDTO.add(turnoDto);
      if( turnosDisponiveis.size() - 1 != i){
        turnos += turno.toString()+", ";
      } else {
        turnos += turno.toString()+".";
      }
    }
    if(turnosDisponiveis.size()>0){
      request.setAttribute("turnos","".equals(turnos)?"Não
Informado":turnos);
      request.setAttribute("turnosDto", turnosDTO);
    }
    //fim dos parametros para listar turnos
    if (!errors.isEmpty()) {
      saveErrors(request, errors);
      return (mapping.findForward(Constants.FAILURE));
    } else {
      return (mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.239 SalvarVagaAction.java

```
/*
 * SalvarVagaAction.java
 * Created on 3 de Fevereiro de 2006, 11:10
 */

package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Habilidade;
import estagio.bean.Turno;
import estagio.bean.Vaga;
import estagio.bean.VagaCurso;
import estagio.bean.VagaDTO;
import estagio.dao.EmpresaDao;
import estagio.dao.UsuarioEmpresaDao;
import estagio.dao.VagaDao;
import estagio.dao.VagasCursosDao;
import estagio.form.NovaVagaForm;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
```

```
/**
 *
 *
 */
public class SalvarVagaAction extends Action {

   public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {


      HttpSession session = request.getSession();
      NovaVagaForm vagaform = (NovaVagaForm) form;
      AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      String action = request.getParameter("action");
      VagaDao vDao = new VagaDao();
      if (action == null) {
        action = "Create";
      }
      if (user == null) {
        return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      Vaga vaga = new Vaga();
      vaga.setAtividadesFds(vagaform.getAtividadesFds());
      vaga.setCnpj(vagaform.getCnpj());
      vaga.setDeficiente(vagaform.getDeficiente());
      vaga.setFumante(vagaform.getFumante());
      vaga.setHoras(Integer.valueOf(vagaform.getHoras()));
      GregorianCalendar inclusao = new GregorianCalendar();
      vaga.setInclusao(inclusao);
      vaga.setLocomocao(vagaform.getLocomocao());
      vaga.setSalario(vagaform.getSalario());
      vaga.setSetor(vagaform.getSetor());
      vaga.setSexo(vagaform.getSexo());
      vaga.setSemestre(vagaform.getSemestre());

vaga.setValeTransporte(!"".equals(vagaform.getValeTransporte())
&& vagaform.getValeTransporte() != null ?
vagaform.getValeTransporte():"N");
      vaga.setOutrosBeneficios(vagaform.getOutrosBenefícios());
      if("interno".equals(vagaform.getEstagio())){
        vaga.setEstagioInterno(Character.valueOf('S'));
      } else {
        vaga.setEstagioInterno(Character.valueOf('N'));
      }
      //CursoDao cursoDao = new CursoDao();

//vaga.setCurso(cursoDao.buscarCurso(vagaform.getCodCurso()))
;
      Vaga novaVaga = new Vaga();
      VagasCursosDao vagasCursosDao = new
VagasCursosDao();
      if("Create".equals(action)) {
        try{
          vDao.incluirVaga(vaga);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.salvar.vaga"));
        }
        novaVaga = vDao.buscarNovaVaga();
      } else {

vaga.setCodVaga(Integer.valueOf(vagaform.getCodVaga()));
        try{
          vDao.alterarVaga(vaga);
        } catch (SQLException e){
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.alterar.vaga"));
        }catch (Exception e) {
          errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.alterar.vaga"));
        }
        novaVaga.setCodVaga(vaga.getCodVaga());
      }
      //Salva os cursos relacionados à vaga

vagasCursosDao.excluirVagaCurso(novaVaga.getCodVaga());
      if(vagaform.getCodCurso() != null){
        for(int i = 0; i < vagaform.getCodCurso().length; i++){
          VagaCurso vagaCurso = new VagaCurso();
          Integer codCurso =
Integer.valueOf(vagaform.getCodCurso()[i]);
          try{
            vagaCurso.setCodCurso(codCurso);
            vagaCurso.setCodVaga(novaVaga.getCodVaga());
            vagasCursosDao.incluirVagaCurso(vagaCurso);
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.salvar.vagacurso"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.salvar.vagacurso"));
          }
        }
      }
      //os cursos relacionados à vaga estao salvos

      //seta parametros para ListarVagas (caso naum houver erros)
      String empresa = "";
      String cnpj = "";
      Empresa emp = new Empresa();
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      if(user.getStatus() != null){
        if("empresa".equals(user.getStatus())){
          try {
            emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
          } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("sqlerror.buscar.usuarioempresa"));
          }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionMessage("error.buscar.usuarioempresa"));
          }
          if(emp != null){
            empresa = emp.getRazaoSocial();
            cnpj = emp.getCnpj();
          }
        }
      }
      if("interno".equals(vagaform.getEstagio())){
        cnpj = vaga.getCnpj();
        EmpresaDao empresaDao = new EmpresaDao();
        emp = empresaDao.buscarEmpresa(cnpj);
        if(emp != null){
          empresa = emp.getRazaoSocial();
        }
      }
      List<Vaga> vagas = new ArrayList<Vaga>();
      try{

if(Character.valueOf('S').equals(vaga.getEstagioInterno())){
          vagas = vDao.listarVagasAbertasEstagioInterno();
```

```
        } else {
            vagas = vDao.listarVagasAbertas(cnpj);
        }
    } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
    }catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.vaga"));
    }
    List<VagaDTO> vagaDtos = new ArrayList<VagaDTO>();
    SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
    for(Vaga v :vagas){
        VagaDTO vagaDto = new VagaDTO();
        vagaDto.setCodVaga(v.getCodVaga());
        vagaDto.setHoras(v.getHoras().toString());

vagaDto.setInclusao(sdf.format(v.getInclusao().getTime()));
        vagaDto.setSalario(v.getSalario());
        vagaDto.setSetor(v.getSetor());
        char sexo = v.getSexo().charAt(0);
        switch (sexo){
            case 'M':
                vagaDto.setSexo("Masculino");
                break;
            case 'F':
                vagaDto.setSexo("Feminino");
                break;
            case 'A':
                vagaDto.setSexo("Qualquer");
                break;
            default:
                vagaDto.setSexo("");
        }
        String turnos = "";
        for(int i = 0;i<v.getTurnos().size();i++){
            Turno turno= v.getTurnos().get(i);
            if( v.getTurnos().size() - 1 != i){
                turnos += turno.toString()+", ";
            } else {
                turnos += turno.toString()+".";
            }
        }
        vagaDto.setTurnos("".equals(turnos)?"Não
Informado":turnos);
        String descricaoHabilidade = "";
        for(int i = 0;i<v.getHabilidades().size();i++){
            Habilidade habilidade = v.getHabilidades().get(i);
            if(v.getHabilidades().size() - 1 != i){
                descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+", ";
            } else {
                descricaoHabilidade +=
habilidade.getDescricaoHabilidade()+".";
            }
        }

vagaDto.setHabilidades("".equals(descricaoHabilidade)?"Não
Informado":descricaoHabilidade);
        String cursos = "";
        if(v.getCursos().isEmpty()){
            cursos += "Nenhum";
        } else {
            for(int i = 0;i< v.getCursos().size();i++){
                Curso curso = v.getCursos().get(i);
                if(v.getCursos().size() - 1 != i){
                    cursos += curso.getNomeCurso()+", ";
                } else {
                    cursos += curso.getNomeCurso()+".";
                }
            }
        }
        vagaDto.setCursos(cursos);
        vagaDto.setSemestre(v.getSemestre());

vagaDto.setValeTransporte(v.getValeTransporte().equals("S")?"S
im":"Não");
        vagaDto.setOutrosBeneficios(v.getOutrosBeneficios());
        vagaDtos.add(vagaDto);
    }
    if(vagas.size()>0){
        request.setAttribute("vagas", vagaDtos);
    }
    request.setAttribute("empresa", empresa);
    request.setAttribute("cnpj", cnpj);
    //setados parametros para ListarVagas
    //seta parametros para NovaVaga (em caso de erro)
    ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
    options.add(new LabelValueBean("Masculino", "M"));
    options.add(new LabelValueBean("Feminino", "F"));
    options.add(new LabelValueBean("Qualquer", "A"));
    request.setAttribute("sexos", options);
    ArrayList simNaoAll = new ArrayList();
    simNaoAll.add(new LabelValueBean("Sim", "S"));
    simNaoAll.add(new LabelValueBean("Não", "N"));
    simNaoAll.add(new LabelValueBean("Não Importa", "A"));
    request.setAttribute("simNaoAll", simNaoAll);
    ArrayList simNao = new ArrayList();
    simNao.add(new LabelValueBean("Sim", "S"));
    simNao.add(new LabelValueBean("Não", "N"));
    request.setAttribute("simNao", simNao);
    ArrayList locomotion = new ArrayList();
    locomotion.add(new LabelValueBean("Carro", "C"));
    locomotion.add(new LabelValueBean("Moto", "M"));
    locomotion.add(new LabelValueBean("Não Importa", "A"));
    request.setAttribute("locomocao", locomotion);
    //setados parametros para NovaVaga (em caso de erro)
    if (!errors.isEmpty()) {
        saveErrors(request, errors);
        return (mapping.findForward(Constants.FAILURE));
    } else {

        return (mapping.findForward(Constants.SUCCESS));
    }
  }
}
```

6.4.240 SelecionarRelatorioSeguroAction.java

```
/*
 * SelecionarRelatorioSeguroAction.java
 * Created on 1 de Maio de 2006, 15:25
 */

package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 *
```

```
*/
public class SelecionarRelatorioSeguroAction  extends Action{

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      AlunoEstagio user = null;
      String codVaga = (String) request.getParameter("codvaga");
      try{
         user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
         if (user == null){
            return mapping.findForward("welcome");
         }
      } catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE,
               new ActionMessage("error.generico"));
      }
      java.util.GregorianCalendar data = new
java.util.GregorianCalendar();
      java.text.SimpleDateFormat sdf = new
java.text.SimpleDateFormat("yyyy");
      Integer ano = Integer.valueOf(sdf.format(data.getTime()));
      Integer anoAntes = ano - 1;

      String anos[] ={ano.toString(), anoAntes.toString()};
      request.setAttribute("anos", anos);
      return mapping.findForward(Constants.SUCCESS);
   }
}
```

6.4.241 TipoUsuarioAction.java

```
/*
 * TipoUsuarioAction.java
 * Created on 31 de Janeiro de 2006, 22:01
 */

package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Usuario;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;

/**
 *
 *
 */
public class TipoUsuarioAction extends Action {

   public ActionForward execute(
         ActionMapping mapping,
         ActionForm form,
         HttpServletRequest request,
         HttpServletResponse response)
         throws Exception {


      HttpSession session = request.getSession();
      AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
      EmpresaDao empresaDao = new EmpresaDao();
      String status = request.getParameter("status");
      if (user == null && !"aluno".equals(status)) {
         return (mapping.findForward("logon"));
      }
      ActionMessages errors = new ActionMessages();
      if("empresa".equals(status)){
         /*List<Empresa> empresas = new ArrayList<Empresa>();
         try {
            empresas = empresaDao.listarEmpresas();
         } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
         } catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.empresa"));
         }
         request.setAttribute("empresas", empresas );*/
         return (mapping.findForward("buscarempresa"));
      }
      CursoDao cursoDao = new CursoDao();
      List<Curso> cursos = new ArrayList<Curso>();
      List<LabelValueBean> selectCursos = new
ArrayList<LabelValueBean>();
      try{
         cursos = cursoDao.listarCursos();
      } catch (SQLException e){
         errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
      }catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
      }
      for(Curso c: cursos){
         selectCursos.add(new
LabelValueBean(c.getNomeCurso(),
c.getCodCurso().toString()));
      }
      if(cursos.size()>0){
         request.setAttribute("cursos", selectCursos);
      } else {
         request.setAttribute("cursos", new
LabelValueBean("0","Não Informado"));
      }

      ArrayList sexos = new ArrayList();
      sexos.add(new LabelValueBean("masculino", "M"));
      sexos.add(new LabelValueBean("feminino", "F"));
      request.setAttribute("sexos", sexos);
      ArrayList simNao = new ArrayList();
      simNao.add(new LabelValueBean("Sim", "S"));
      simNao.add(new LabelValueBean("Não", "N"));
      request.setAttribute("simNao", simNao);
      ArrayList locomotion = new ArrayList();
      locomotion.add(new LabelValueBean("Carro", "C"));
      locomotion.add(new LabelValueBean("Moto", "M"));
      locomotion.add(new LabelValueBean("Não Possui", "N"));
      request.setAttribute("locomocao", locomotion);
      request.setAttribute("usuario", user !=
null?user.getUsuario():"guest");
      request.setAttribute("status", status);
      Integer[] semestres = new Integer[12];
```

```
        String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
        request.setAttribute("estados", estados);
        for(int i=1; i<12; i++){
          semestres[i] = Integer.valueOf(i);
        }
        request.setAttribute("semestres", semestres);
        //Usuarios (para o caso de erro)
        ArrayList<Usuario> usuarios = new ArrayList<Usuario>();
        Usuario admin = new Usuario();
        admin.setTipo("admin");
        admin.setMensagem("Membro do grupo de
Administradores");
        Usuario empresa = new Usuario();
        empresa.setTipo("empresa");
        empresa.setMensagem("Membro de Empresa");
        Usuario aluno = new Usuario();
        aluno.setTipo("aluno");
        aluno.setMensagem("Aluno");
        usuarios.add(admin);
        usuarios.add(empresa);
        usuarios.add(aluno);
        request.setAttribute("usuarios", usuarios);
        //fim de Usuarios
        if (!errors.isEmpty()) {
          saveErrors(request, errors);
          return (mapping.findForward(Constants.FAILURE));
        } else {
          return (mapping.findForward(Constants.SUCCESS));
        }
    }
}
```

6.4.242 WelcomeAction.java

```
/*
 * WelcomeAction.java
 * Created on 28 de Junho de 2006, 09:21
 */

package estagio.action;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.bean.Pessoa;
import estagio.dao.AlunoEstagioDao;
import estagio.dao.PessoaDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.util.HashMap;
import java.util.Hashtable;
import java.util.Map;
import javax.servlet.ServletContext;
import javax.servlet.http.Cookie;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;

/**
 *
 * @author Ricardo
 */
public class WelcomeAction extends Action{

    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
      ActionMessages errors = new ActionMessages();
      HttpSession session = request.getSession();
      //busca dados de outro contexto - Intranet Academica
      Cookie[] cookies = request.getCookies();
      if (cookies != null) {
        for(int i = 0;i < cookies.length;i++) {
          if("logado".equals(cookies[i].getName()) ||
"codPessoa".equals(cookies[i].getName())){
            session.setAttribute(cookies[i].getName(),
cookies[i].getValue());
          }
        }
      }
      //ServletContext context = session.getServletContext();
      // read the name of the Context storing shared session data
      //String SignonContext =
context.getInitParameter("TestaLoginTcc");
      // get the Context containing the shared session data
      //ServletContext ssocontext =
context.getContext(SignonContext);
      //Map dadosIntranetAcademica = new HashMap();
      // read the shareddata
      //Hashtable shareddata = (Hashtable)
ssocontext.getAttribute("dados_compartilhados");
      //if (shareddata != null) {
        // get the right Map using the ssosessionid
        //dadosIntranetAcademica = (HashMap)
shareddata.get(ssosessionid);
      //}
      AlunoEstagio user = null;
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();

      String logado = (String) session.getAttribute("logado");
      if(logado == null){
        return mapping.findForward(Constants.FAILURE);
      }
      if ("true".equals(logado)){
        String codPessoa = (String)
session.getAttribute("codPessoa");
        try {
          PessoaDao pessoaDao = new PessoaDao();
          Pessoa pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(codPessoa));
          if(pessoa != null){
            user =
alunoEstagioDao.buscarAlunoEstagio(pessoa.getUsuario());
          }
        } catch (SQLException e){
          errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new ActionMessage("error.recuperar.dados"));
        } catch (Exception e){
          errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new ActionMessage("error.generico"));
        }
        if (user == null) {
          errors.add(
              ActionMessages.GLOBAL_MESSAGE,
              new ActionMessage("error.usuario.naoexiste"));
        } else {
          if(session.getAttribute(Constants.USER_KEY) !=
null){
            session.removeAttribute(Constants.USER_KEY);
          }
```

```
            session.setAttribute(Constants.USER_KEY, user);
            if("empresa".equalsIgnoreCase(user.getStatus())){
                String cnpj = "";
                Empresa emp = null;
                try{
                    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
                    emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
                } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarioempresa"));
                }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarioempresa"));
                }
                if(emp != null){
                    cnpj = emp.getCnpj();
                }
                session.setAttribute("cnpj", cnpj);
            }
        }
    }

    if(!errors.isEmpty()){
        saveErrors(request, errors);
        return mapping.findForward(Constants.FAILURE);
    } else {
        return mapping.findForward(Constants.SUCCESS);
    }
  }

}
```

6.4.243 package estagio.bean

6.4.244 AgenteIntegracao.java

```
/*
 * AgenteIntegracao.java
 * Created on 27 de Fevereiro de 2006, 11:32
 */

package estagio.bean;

/**
 *
 *
 */
public class AgenteIntegracao {

    private Integer codAgente;
    private String agente;
    private String cnpj;

  /** Creates a new instance of AgenteIntegracao */
  public AgenteIntegracao() {
  }

  public Integer getCodAgente() {
    return codAgente;
  }

  public void setCodAgente(Integer codAgente) {
    this.codAgente = codAgente;
  }

  public String getAgente() {
    return agente;
  }

  public void setAgente(String agente) {
    this.agente = agente;
  }

  public String getCnpj() {
    return cnpj;
  }

  public void setCnpj(String cnpj) {
    this.cnpj = cnpj;
  }

}
```

6.4.245 AlunoEstagio.java

```
/*
 * AlunoEstagio.java
 * Created on 26 de Outubro de 2005, 00:00
 */

package estagio.bean;
import java.util.List;

/**
 *
 *
 */
public class AlunoEstagio extends Pessoa{

  private Integer codAlunoEstagio;
  private String disponivelFds;
  private String fumante;
  private String deficiente;
  private String locomocao;
  private String sexo;
  private String email;
  private String fone;
  private Integer semestre;
  private List<HabilidadeAluno> habilidades;
  private List<TurnoDisponivel> turnosDisponiveis;
  private Curso curso;
  private String rg;
  private String rg_naturalidade;
  private Long cpf;
  private String matricula;
  private String cep;
  private String rua;
  private String numero;
  private String complemento;
  private String bairro;
  private String municipio;
  private String estado;
  private String pais;

  /** Creates a new instance of AlunoEstagio */
  public AlunoEstagio() {
  }

  public Integer getCodAlunoEstagio() {
    return codAlunoEstagio;
  }

  public void setCodAlunoEstagio(Integer codAlunoEstagio) {
    this.codAlunoEstagio = codAlunoEstagio;
  }

  public String getDisponivelFds() {
    return disponivelFds;
  }

  public void setDisponivelFds(String disponivelFds) {
```

```
    this.disponivelFds = disponivelFds;
  }

  public String getFumante() {
    return fumante;
  }

  public void setFumante(String fumante) {
    this.fumante = fumante;
  }

  public String getDeficiente() {
    return deficiente;
  }

  public void setDeficiente(String deficiente) {
    this.deficiente = deficiente;
  }

  public String getLocomocao() {
    return locomocao;
  }

  public void setLocomocao(String locomocao) {
    this.locomocao = locomocao;
  }

  public String getSexo() {
    return sexo;
  }

  public void setSexo(String sexo) {
    this.sexo = sexo;
  }

  public String getEmail() {
    return email;
  }

  public void setEmail(String email) {
    this.email = email;
  }

  public String getFone() {
    return fone;
  }

  public void setFone(String fone) {
    this.fone = fone;
  }

  public List<HabilidadeAluno> getHabilidades() {
    return habilidades;
  }

  public void setHabilidades(List<HabilidadeAluno>
habilidades) {
    this.habilidades = habilidades;
  }

  public List<TurnoDisponivel> getTurnosDisponiveis() {
    return turnosDisponiveis;
  }

  public void setTurnosDisponiveis(List<TurnoDisponivel>
turnosDisponiveis) {
    this.turnosDisponiveis = turnosDisponiveis;
  }

  public Curso getCurso() {
    return curso;
  }

  public void setCurso(Curso curso) {
    this.curso = curso;
  }

  public Integer getSemestre() {
    return semestre;
  }

  public void setSemestre(Integer semestre) {
    this.semestre = semestre;
  }

  public String getRg() {
    return rg;
  }

  public void setRg(String rg) {
    this.rg = rg;
  }

  public String getRg_naturalidade() {
    return rg_naturalidade;
  }

  public void setRg_naturalidade(String rg_naturalidade) {
    this.rg_naturalidade = rg_naturalidade;
  }

  public Long getCpf() {
    return cpf;
  }

  public void setCpf(Long cpf) {
    this.cpf = cpf;
  }

  public String getMatricula() {
    return matricula;
  }

  public void setMatricula(String matricula) {
    this.matricula = matricula;
  }

  public String getCep() {
    return cep;
  }

  public void setCep(String cep) {
    this.cep = cep;
  }

  public String getRua() {
    return rua;
  }

  public void setRua(String rua) {
    this.rua = rua;
  }

  public String getNumero() {
    return numero;
  }

  public void setNumero(String numero) {
    this.numero = numero;
  }

  public String getComplemento() {
    return complemento;
  }
```

```
    public void setComplemento(String complemento) {
        this.complemento = complemento;
    }

    public String getBairro() {
        return bairro;
    }

    public void setBairro(String bairro) {
        this.bairro = bairro;
    }

    public String getMunicipio() {
        return municipio;
    }

    public void setMunicipio(String municipio) {
        this.municipio = municipio;
    }

    public String getEstado() {
        return estado;
    }

    public void setEstado(String estado) {
        this.estado = estado;
    }

    public String getPais() {
        return pais;
    }

    public void setPais(String pais) {
        this.pais = pais;
    }
}
```

6.4.246 Constants.java

```
/*
 * Constants.java
 * Created on 11 de Novembro de 2005, 22:46
 */
package estagio.bean;

/**
 *
 *
 */
public final class Constants {

    /**
     * The token representing "failure" for this application.
     */
    public static final String FAILURE = "error";

    /**
     * The token representing "success" for this application.
     */
    public static final String SUCCESS = "success";

    /**
     * The session scope attribute under which the User object
     * for the currently logged in user is stored.
     */
    public static final String USER_KEY = "user";

    public static final String OUTRO = "outro";
}
```

6.4.247 Convenio.java

```
/*
 * Convenio.java
 * Created on 17 de Janeiro de 2006, 20:58
 */

package estagio.bean;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.GregorianCalendar;

/**
 *
 *
 */
public class Convenio {

    /** Creates a new instance of Convenio */
    public Convenio() {
    }
    private Integer codConvenio;
    private GregorianCalendar dataInicio;
    private GregorianCalendar dataFim;
    private String descricao;
    private Empresa empresa;

    public Integer getCodConvenio() {
        return codConvenio;
    }

    public void setCodConvenio(Integer codConvenio) {
        this.codConvenio = codConvenio;
    }

    public GregorianCalendar getDataInicio() {
        return dataInicio;
    }

    public void setDataInicio(GregorianCalendar dataInicio) {
        this.dataInicio = dataInicio;
    }

    public void setDataInicio(String dataInicio) {
        GregorianCalendar data = new GregorianCalendar();
        SimpleDateFormat df = new
SimpleDateFormat("dd/MM/yyyy");
        try {
            if (dataInicio != null){
                Date dataTest = df.parse(dataInicio);
                data.setTime(dataTest);
                this.dataInicio = data;
            } else {
                this.dataInicio = new GregorianCalendar();
            }
        } catch (ParseException e){
            this.dataInicio = new GregorianCalendar();
        }
    }

    public GregorianCalendar getDataFim() {
        return dataFim;
    }

    public void setDataFim(GregorianCalendar dataFim) {
        this.dataFim = dataFim;
    }

    public void setDataFim(String dataFim) {
        GregorianCalendar data = new GregorianCalendar();
```

```
    SimpleDateFormat df = new
SimpleDateFormat("dd/MM/yyyy");
    try {
      if (dataFim != null){
        Date dataTest = df.parse(dataFim);
        data.setTime(dataTest);
        this.dataFim = data;
      } else {
        this.dataFim = new GregorianCalendar();
      }
    } catch (ParseException e){
      this.dataFim = new GregorianCalendar();
    }
  }

  public String getDescricao() {
    return descricao;
  }

  public void setDescricao(String descricao) {
    this.descricao = descricao;
  }

  public Empresa getEmpresa() {
    return empresa;
  }

  public void setEmpresa(Empresa empresa) {
    this.empresa = empresa;
  }

}
```

6.4.248 ConvenioDTO.java

```
/*
 * ConvenioDTO.java
 * Created on 17 de Janeiro de 2006, 21:41
 */

package estagio.bean;

/**
 *
 *
 */
public class ConvenioDTO {

  /** Creates a new instance of ConvenioDTO */
  public ConvenioDTO() {
  }

  private Integer codConvenio;
  private String dataInicio;
  private String dataFim;
  private String descricao;
  private String empresa;
  private String situacao;


  public Integer getCodConvenio() {
    return codConvenio;
  }

  public void setCodConvenio(Integer codConvenio) {
    this.codConvenio = codConvenio;
  }

  public String getDataInicio() {
    return dataInicio;
  }

  public void setDataInicio(String dataInicio) {
    this.dataInicio = dataInicio;
  }

  public String getDataFim() {
    return dataFim;
  }

  public void setDataFim(String dataFim) {
    this.dataFim = dataFim;
  }

  public String getDescricao() {
    return descricao;
  }

  public void setDescricao(String descricao) {
    this.descricao = descricao;
  }

  public String getEmpresa() {
    return empresa;
  }

  public void setEmpresa(String empresa) {
    this.empresa = empresa;
  }

  public String getSituacao() {
    return situacao;
  }

  public void setSituacao(String situacao) {
    this.situacao = situacao;
  }

}
```

6.4.249 Curso.java

```
/*
 * Curso.java
 * Created on 2 de Março de 2006, 14:24
 */

package estagio.bean;

/**
 *
 *
 */
public class Curso {

  private Integer codCurso;
  private String nomeCurso;
  private Integer duracao;

  /** Creates a new instance of Curso */
  public Curso() {
  }

  public Integer getCodCurso() {
    return codCurso;
  }

  public void setCodCurso(Integer codCurso) {
    this.codCurso = codCurso;
  }

  public String getNomeCurso() {
    return nomeCurso;
  }

  public void setNomeCurso(String nomeCurso) {
```

```
    this.nomeCurso = nomeCurso;
  }

  public Integer getDuracao() {
    return duracao;
  }

  public void setDuracao(Integer duracao) {
    this.duracao = duracao;
  }

  public boolean equals(Object o) {
    if (!(o instanceof Curso)) return false;
    if (o == this) return true;
    Curso c = (Curso) o;
    // só para simplificar estou supondo que nenhum dos campos
é null
    return codCurso.equals(c.getCodCurso()) &&
nomeCurso.equals(c.getNomeCurso()) &&
duracao.equals(c.getDuracao());
  }
  public int hashCode() {
    int ret = codCurso.hashCode();
    ret = ret * 37 + nomeCurso.hashCode();
    ret = ret * 37 + duracao.hashCode();
    // se houver mais campos, vá repetindo a expressão acima
"ret = ret * 37 + hashCode do proximo campo"
    // por exemplo, ret = ret * 37 + telefone.hashCode();
    return ret;
  }

}
```

6.4.250 Empresa.java

```
package estagio.bean;
import java.util.List;
/**
 *Classe que representa uma empresa.
 *
 *@since 12/10/05
 */

public class Empresa
{
  private String cnpj;
  private String razaoSocial;
  private String nomeFantasia;
  private String ramo;
  private Endereco enderecoEmpresa;
  private String site;
  private String email;
  private String fone;
  private String representante;
  private String cargoRepresentante;
  private String fax;
  private String numProcesso;
  private String agenteIntegracao;

  /**
   * Creates a new instance of Empresa.
   */
  public Empresa()
  {

  }

  public String getCnpj() {
    return cnpj;
  }

  public void setCnpj(String cnpj) {
    this.cnpj = cnpj;
  }

  public String getRazaoSocial() {
    return razaoSocial;
  }

  public void setRazaoSocial(String razaoSocial) {
    this.razaoSocial = razaoSocial;
  }

  public String getNomeFantasia() {
    return nomeFantasia;
  }

  public void setNomeFantasia(String nomeFantasia) {
    this.nomeFantasia = nomeFantasia;
  }

  public String getRamo() {
    return ramo;
  }

  public void setRamo(String ramo) {
    this.ramo = ramo;
  }

  public Endereco getEnderecoEmpresa() {
    return enderecoEmpresa;
  }

  public void setEnderecoEmpresa(Endereco enderecoEmpresa)
{
    this.enderecoEmpresa = enderecoEmpresa;
  }

  public String getSite() {
    return site;
  }

  public void setSite(String site) {
    this.site = site;
  }

  public String getEmail() {
    return email;
  }

  public void setEmail(String email) {
    this.email = email;
  }


  public String getRepresentante() {
    return representante;
  }

  public void setRepresentante(String representante) {
    this.representante = representante;
  }

  public String getCargoRepresentante() {
    return cargoRepresentante;
  }

  public void setCargoRepresentante(String cargoRepresentante)
{
    this.cargoRepresentante = cargoRepresentante;
  }

  public String getFone() {
    return fone;
  }
```

```
  public void setFone(String fone) {
    this.fone = fone;
  }

  public String getFax() {
    return fax;
  }

  public void setFax(String fax) {
    this.fax = fax;
  }

  public String getNumProcesso() {
    return numProcesso;
  }

  public void setNumProcesso(String numProcesso) {
    this.numProcesso = numProcesso;
  }

  public String getAgenteIntegracao() {
    return agenteIntegracao;
  }

  public void setAgenteIntegracao(String agenteIntegracao) {
    this.agenteIntegracao = agenteIntegracao;
  }
}
```

6.4.251 Endereco.java

```
package estagio.bean;
/**
 *Classe que representa o Endereco de uma pessoa.
 *
 *@since 12/10/05
 */

public class Endereco
{
  private Integer codEndereco;
  private String cep;
  private String rua;
  private int numero;
  private String complemento;
  private String bairro;
  private String municipio;
  private String estado;
  private String pais;
  private String owner;

  /**
   * Creates a new instance of Endereco
   */
  public Endereco()   {
  }

  public String getCep() {
    return cep;
  }

  public void setCep(String cep) {
    this.cep = cep;
  }

  public String getRua() {
    return rua;
  }

  public void setRua(String rua) {
    this.rua = rua;
  }

  public int getNumero() {
    return numero;
  }

  public void setNumero(int numero) {
    this.numero = numero;
  }

  public String getComplemento() {
    return complemento;
  }

  public void setComplemento(String complemento) {
    this.complemento = complemento;
  }

  public String getBairro() {
    return bairro;
  }

  public void setBairro(String bairro) {
    this.bairro = bairro;
  }

  public String getMunicipio() {
    return municipio;
  }

  public void setMunicipio(String municipio) {
    this.municipio = municipio;
  }

  public String getEstado() {
    return estado;
  }

  public void setEstado(String estado) {
    this.estado = estado;
  }

  public String getPais() {
    return pais;
  }

  public void setPais(String pais) {
    this.pais = pais;
  }

  public String getOwner() {
    return owner;
  }

  public void setOwner(String owner) {
    this.owner = owner;
  }

  public String toString(){
    return "Rua "+this.getRua()+" nº"+this.getNumero()+",
"+(".equals(this.getComplemento()) ||
"null".equals(this.getComplemento()) || this.getComplemento() ==
null?"":this.getComplemento()+",")+this.getBairro()+",
"+this.getMunicipio()+", "+this.getEstado().toUpperCase()+",
CEP "+this.getCep()+".";
  }

  public Integer getCodEndereco() {
    return codEndereco;
  }

  public void setCodEndereco(Integer codEndereco) {
    this.codEndereco = codEndereco;
```

```
    }

}
```

6.4.252 Habilidade.java

```
/*
 * Habilidade.java
 * Created on 28 de Janeiro de 2006, 11:39
 */

package estagio.bean;

/**
 *
 *
 */
public class Habilidade implements Comparable {

    /**
     * Creates a new instance of Habilidade
     */
    public Habilidade() {
    }

    private Integer codHabilidade;
    private String descricaoHabilidade;

    public Integer getCodHabilidade() {
        return codHabilidade;
    }

    public void setCodHabilidade(Integer codHabilidade) {
        this.codHabilidade = codHabilidade;
    }

    public String getDescricaoHabilidade() {
        return descricaoHabilidade;
    }

    public void setDescricaoHabilidade(String
descricaoHabilidade) {
        this.descricaoHabilidade = descricaoHabilidade;
    }

    public int compareTo(Object h){
        Habilidade outra = (Habilidade) h;
        if(outra.getCodHabilidade().equals(getCodHabilidade())){
            return 0;
        } else {
            return 1;
        }
    }

    public boolean equals(Object obj) {
        Habilidade h = (Habilidade) obj;
        boolean retValue = false;
        if(getCodHabilidade().equals(h.getCodHabilidade())){
            retValue = true;
        }
        return retValue;
    }

}
```

6.4.253 HabilidadeAluno.java

```
/*
 * HabilidadeAluno.java
 * Created on 29 de Janeiro de 2006, 17:02
 */

package estagio.bean;

/**
 *
 *
 */
public class HabilidadeAluno extends Habilidade{

    /** Creates a new instance of HabilidadeAluno */
    public HabilidadeAluno() {
    }

    private Integer codAlunoEstagio;
    private Integer codHabilidadeAluno;

    public Integer getCodAlunoEstagio() {
        return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
        this.codAlunoEstagio = codAlunoEstagio;
    }

    public Integer getCodHabilidadeAluno() {
        return codHabilidadeAluno;
    }

    public void setCodHabilidadeAluno(Integer
codHabilidadeAluno) {
        this.codHabilidadeAluno = codHabilidadeAluno;
    }

}
```

6.4.254 HabilidadeVaga.java

```
/*
 * HabilidadeVaga.java
 * Created on 29 de Janeiro de 2006, 18:34
 */

package estagio.bean;

/**
 *
 *
 */
public class HabilidadeVaga extends Habilidade{

    /** Creates a new instance of HabilidadeVaga */
    public HabilidadeVaga() {
    }

    private Integer codHabilidadeVaga;
    private Integer codVaga;

    public Integer getCodHabilidadeVaga() {
        return codHabilidadeVaga;
    }

    public void setCodHabilidadeVaga(Integer
codHabilidadeVaga) {
        this.codHabilidadeVaga = codHabilidadeVaga;
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public boolean equals(Object obj) {
```

```
    HabilidadeVaga h = (HabilidadeVaga) obj;
    boolean retValue = false;
    if(getCodHabilidade().equals(h.getCodHabilidade())){
      retValue = true;
    }
    return retValue;
  }
}
```

6.4.255 Pessoa.java

```
package estagio.bean;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.GregorianCalendar;
/**
 *Classe que define uma pessoa.
 *@author Ricardo
 *@since 12/10/05
 */

public class Pessoa
{
  private String usuario;
  private String senha;
  private Integer codPessoa;
  private String nomePessoa;
  private GregorianCalendar dataNascimento;
  private String status;

  /**
   * Creates a new instance of Pessoa.
   */
  public Pessoa()
  {

  }

  public String getUsuario() {
    return usuario;
  }

  public void setUsuario(String usuario) {
    this.usuario = usuario;
  }

  public String getSenha() {
    return senha;
  }

  public void setSenha(String senha) {
    this.senha = senha;
  }

  public Integer getCodPessoa() {
    return codPessoa;
  }

  public void setCodPessoa(Integer codPessoa) {
    this.codPessoa = codPessoa;
  }

  public String getNomePessoa() {
    return nomePessoa;
  }

  public void setNomePessoa(String nomePessoa) {
    this.nomePessoa = nomePessoa;
  }

  public GregorianCalendar getDataNascimento() {
    return dataNascimento;
  }

  public void setDataNascimento(GregorianCalendar
dataNascimento) {
    this.dataNascimento = dataNascimento;
  }

  public void setDataNascimento(String dataNascimento) {
    GregorianCalendar data = new GregorianCalendar();
    SimpleDateFormat df = new
SimpleDateFormat("dd/MM/yyyy");
    try {
      if (dataNascimento != null){
        Date dataTest = df.parse(dataNascimento);
        data.setTime(dataTest);
        this.dataNascimento = data;
      } else {
        this.dataNascimento = new GregorianCalendar();
      }
    } catch (ParseException e){
      this.dataNascimento = new GregorianCalendar();
    }
  }

  public String getStatus() {
    return status;
  }

  public void setStatus(String status) {
    this.status = status;
  }

  public int hashCode() {
    int ret = codPessoa.hashCode();
    ret = ret * 37 + nomePessoa.hashCode();
    ret = ret * 37 + usuario.hashCode();
    ret = ret * 37 + senha.hashCode();
    ret = ret * 37 + dataNascimento.hashCode();
    ret = ret * 37 + status.hashCode();
    // se houver mais campos, vá repetindo a expressão acima
"ret = ret * 37 + hashCode do proximo campo"
    // por exemplo, ret = ret * 37 + telefone.hashCode();
    return ret;
  }

  public boolean equals(Object o) {
    if (!(o instanceof Pessoa)) return false;
    if (o == this) return true;
    Pessoa p = (Pessoa) o;
    return codPessoa.equals(p.getCodPessoa()) &&
usuario.equals(p.getUsuario());
  }
}
```

6.4.256 PlanoEstagio.java

```
/*
 * PlanoEstagio.java
 * Created on 25 de Abril de 2006, 08:47
 */

package estagio.bean;

/**
 *
 *
 */
public class PlanoEstagio {

  private Integer codPlanoEstagio;
  private Integer codTermo;
  private String orientador;
  private String formacaoOrientador;
```

```
    private String areaAtividadeEstagiario;
    private String objetivos;
    private String atividades;
    private String departamento;
    private String cargoOrientador;
    private String ramal;

    /** Creates a new instance of PlanoEstagio */
    public PlanoEstagio() {
    }

    public String getOrientador() {
        return orientador;
    }

    public void setOrientador(String orientador) {
        this.orientador = orientador;
    }

    public String getFormacaoOrientador() {
        return formacaoOrientador;
    }

    public void setFormacaoOrientador(String
formacaoOrientador) {
        this.formacaoOrientador = formacaoOrientador;
    }

    public String getAreaAtividadeEstagiario() {
        return areaAtividadeEstagiario;
    }

    public void setAreaAtividadeEstagiario(String
areaAtividadeEstagiario) {
        this.areaAtividadeEstagiario = areaAtividadeEstagiario;
    }

    public String getObjetivos() {
        return objetivos;
    }

    public void setObjetivos(String objetivos) {
        this.objetivos = objetivos;
    }

    public String getAtividades() {
        return atividades;
    }

    public void setAtividades(String atividades) {
        this.atividades = atividades;
    }

    public Integer getCodPlanoEstagio() {
        return codPlanoEstagio;
    }

    public void setCodPlanoEstagio(Integer codPlanoEstagio) {
        this.codPlanoEstagio = codPlanoEstagio;
    }

    public Integer getCodTermo() {
        return codTermo;
    }

    public void setCodTermo(Integer codTermo) {
        this.codTermo = codTermo;
    }

    public String getDepartamento() {
        return departamento;
    }

    public void setDepartamento(String departamento) {
        this.departamento = departamento;
    }

    public String getCargoOrientador() {
        return cargoOrientador;
    }

    public void setCargoOrientador(String cargoOrientador) {
        this.cargoOrientador = cargoOrientador;
    }

    public String getRamal() {
        return ramal;
    }

    public void setRamal(String ramal) {
        this.ramal = ramal;
    }

}
```

6.4.257 RelatorioSeguroDTO.java

```
/*
 * RelatorioSeguroDTO.java
 * Created on 1 de Maio de 2006, 17:31
 */

package estagio.bean;

import java.math.BigDecimal;
import java.sql.Date;

/**
 *
 *
 */
public class RelatorioSeguroDTO {

    private String nome_pessoa;
    private BigDecimal cpf;
    private Date nasc_pessoa;
    private Date datainicio;
    private Date datafim;
    private String empresa_cnpj;
    private String processo;
    private Date data;

    /** Creates a new instance of RelatorioSeguroDTO */
    public RelatorioSeguroDTO() {
    }

    public String getNome_pessoa() {
        return nome_pessoa;
    }

    public void setNome_pessoa(String nome_pessoa) {
        this.nome_pessoa = nome_pessoa;
    }

    public BigDecimal getCpf() {
        return cpf;
    }

    public void setCpf(BigDecimal cpf) {
        this.cpf = cpf;
    }

    public Date getNasc_pessoa() {
        return nasc_pessoa;
    }
```

```
    public void setNasc_pessoa(Date nasc_pessoa) {
        this.nasc_pessoa = nasc_pessoa;
    }

    public Date getDatainicio() {
        return datainicio;
    }

    public void setDatainicio(Date datainicio) {
        this.datainicio = datainicio;
    }

    public Date getDatafim() {
        return datafim;
    }

    public void setDatafim(Date datafim) {
        this.datafim = datafim;
    }

    public String getEmpresa_cnpj() {
        return empresa_cnpj;
    }

    public void setEmpresa_cnpj(String empresa_cnpj) {
        this.empresa_cnpj = empresa_cnpj;
    }

    public String getProcesso() {
        return processo;
    }

    public void setProcesso(String processo) {
        this.processo = processo;
    }

    public Date getData() {
        return data;
    }

    public void setData(Date data) {
        this.data = data;
    }

}
```

6.4.258 ServidorSmtp.java

```
/*
 * ServidorSmtp.java
 * Created on 15 de Maio de 2006, 17:07
 */

package estagio.bean;

/**
 *
 *
 */
public class ServidorSmtp {

    private Integer cod_servidor_smtp;
    private String url;
    private String username;
    private String senha;
    private Integer porta_smtp;
    private String endereco_email;
    private Boolean requer_autenticacao;

    /** Creates a new instance of ServidorSmtp */
    public ServidorSmtp() {
    }

    public Integer getCod_servidor_smtp() {
        return cod_servidor_smtp;
    }

    public void setCod_servidor_smtp(Integer cod_servidor_smtp)
{
        this.cod_servidor_smtp = cod_servidor_smtp;
    }

    public String getUrl() {
        return url;
    }

    public void setUrl(String url) {
        this.url = url;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getSenha() {
        return senha;
    }

    public void setSenha(String senha) {
        this.senha = senha;
    }

    public Integer getPorta_smtp() {
        return porta_smtp;
    }

    public void setPorta_smtp(Integer porta_smtp) {
        this.porta_smtp = porta_smtp;
    }

    public String getEndereco_email() {
        return endereco_email;
    }

    public void setEndereco_email(String endereco_email) {
        this.endereco_email = endereco_email;
    }

    public Boolean getRequer_autenticacao() {
        return requer_autenticacao;
    }

    public void setRequer_autenticacao(Boolean
requer_autenticacao) {
        this.requer_autenticacao = requer_autenticacao;
    }

}
```

6.4.259 TermoCompromisso.java

```
/*
 * TermoCompromisso.java
 * Created on 21 de Fevereiro de 2006, 15:15
 */

package estagio.bean;

import java.util.GregorianCalendar;

/**
```

```
 *
 *
 */
public class TermoCompromisso {

   private Integer codTermo;
   private AlunoEstagio alunoEstagio;
   private Empresa empresa;
   private GregorianCalendar dataInicio;
   private GregorianCalendar dataFim;
   private AgenteIntegracao agenteIntegracao;
   private Vaga vaga;
   private PlanoEstagio planoEstagio;
   private String obrigatorio;
   private String seguroCompanhia;
   private Long seguroNumApolice;
   private String seguroProvidenciado;


   /** Creates a new instance of TermoCompromisso */
   public TermoCompromisso() {
   }

   public Integer getCodTermo() {
      return codTermo;
   }

   public void setCodTermo(Integer codTermo) {
      this.codTermo = codTermo;
   }

   public AlunoEstagio getAlunoEstagio() {
      return alunoEstagio;
   }

   public void setAlunoEstagio(AlunoEstagio alunoEstagio) {
      this.alunoEstagio = alunoEstagio;
   }

   public Empresa getEmpresa() {
      return empresa;
   }

   public void setEmpresa(Empresa empresa) {
      this.empresa = empresa;
   }

   public GregorianCalendar getDataInicio() {
      return dataInicio;
   }

   public void setDataInicio(GregorianCalendar dataInicio) {
      this.dataInicio = dataInicio;
   }

   public GregorianCalendar getDataFim() {
      return dataFim;
   }

   public void setDataFim(GregorianCalendar dataFim) {
      this.dataFim = dataFim;
   }

   public AgenteIntegracao getAgenteIntegracao() {
      return agenteIntegracao;
   }

   public void setAgenteIntegracao(AgenteIntegracao
agenteIntegracao) {
      this.agenteIntegracao = agenteIntegracao;
   }

   public Vaga getVaga() {
      return vaga;
   }

   public void setVaga(Vaga vaga) {
      this.vaga = vaga;
   }

   public PlanoEstagio getPlanoEstagio() {
      return planoEstagio;
   }

   public void setPlanoEstagio(PlanoEstagio planoEstagio) {
      this.planoEstagio = planoEstagio;
   }

   public String getObrigatorio() {
      return obrigatorio;
   }

   public void setObrigatorio(String obrigatorio) {
      this.obrigatorio = obrigatorio;
   }

   public String getSeguroCompanhia() {
      return seguroCompanhia;
   }

   public void setSeguroCompanhia(String seguroCompanhia) {
      this.seguroCompanhia = seguroCompanhia;
   }

   public Long getSeguroNumApolice() {
      return seguroNumApolice;
   }

   public void setSeguroNumApolice(Long seguroNumApolice) {
      this.seguroNumApolice = seguroNumApolice;
   }

   public String getSeguroProvidenciado() {
      return seguroProvidenciado;
   }

   public void setSeguroProvidenciado(String
seguroProvidenciado) {
      this.seguroProvidenciado = seguroProvidenciado;
   }

}
```

6.4.260 TermoCompromissoDTO.java

```
/*
 * TermoCompromissoDTO.java
 * Created on 22 de Fevereiro de 2006, 17:37
 */

package estagio.bean;

import java.util.List;
import org.apache.struts.util.LabelValueBean;

/**
 *
 *
 */
public class TermoCompromissoDTO {

   private Integer codTermo;
   private String estagiario;
   private String dataInicio;
   private String dataFim;
```

```
private String agenteIntegracao;
private String empresa;
private String curso;
private String setor;
private String requisitos;
private String salario;
private String horario;
private String nomeDocumento;
private Character estagioInterno;

/** Creates a new instance of TermoCompromissoDTO */
public TermoCompromissoDTO() {
}

public String getEstagiario() {
    return estagiario;
}

public void setEstagiario(String estagiario) {
    this.estagiario = estagiario;
}

public String getDataInicio() {
    return dataInicio;
}

public void setDataInicio(String dataInicio) {
    this.dataInicio = dataInicio;
}

public String getDataFim() {
    return dataFim;
}

public void setDataFim(String dataFim) {
    this.dataFim = dataFim;
}

public Integer getCodTermo() {
    return codTermo;
}

public void setCodTermo(Integer codTermo) {
    this.codTermo = codTermo;
}

public String getAgenteIntegracao() {
    return agenteIntegracao;
}

public void setAgenteIntegracao(String agenteIntegracao) {
    this.agenteIntegracao = agenteIntegracao;
}

public String getEmpresa() {
    return empresa;
}

public void setEmpresa(String empresa) {
    this.empresa = empresa;
}

public String getCurso() {
    return curso;
}

public void setCurso(String curso) {
    this.curso = curso;
}

public String getSetor() {
    return setor;
}

public void setSetor(String setor) {
    this.setor = setor;
}

public String getRequisitos() {
    return requisitos;
}

public void setRequisitos(String requisitos) {
    this.requisitos = requisitos;
}

public String getSalario() {
    return salario;
}

public void setSalario(String salario) {
    this.salario = salario;
}

public String getHorario() {
    return horario;
}

public void setHorario(String horario) {
    this.horario = horario;
}

public String getNomeDocumento() {
    return nomeDocumento;
}

public void setNomeDocumento(String nomeDocumento) {
    this.nomeDocumento = nomeDocumento;
}

public Character getEstagioInterno() {
    return estagioInterno;
}

public void setEstagioInterno(Character estagioInterno) {
    this.estagioInterno = estagioInterno;
}

}
```

6.4.261 Turno.java

```
/*
 * Turno.java
 * Created on 28 de Janeiro de 2006, 10:44
 */

package estagio.bean;

import java.sql.Time;
import java.text.SimpleDateFormat;

/**
 *
 *
 */
public class Turno {

    /** Creates a new instance of Turno */
    public Turno() {
    }

    private Integer codTurno;
    private Integer codVaga;
    private String turno;
```

```
    private Time horaInicio;
    private Time horaFim;
    private SimpleDateFormat sdf;


    public String toString(){
        sdf = new SimpleDateFormat("HH:mm");
        if(getHoraInicio() == null || getHoraFim() == null){
            return "não informado";
        }
        return "das "+sdf.format(getHoraInicio())+" às
"+sdf.format(getHoraFim());
    }

    public Integer getCodTurno() {
        return codTurno;
    }

    public void setCodTurno(Integer codTurno) {
        this.codTurno = codTurno;
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public String getTurno() {
        return turno;
    }

    public void setTurno(String turno) {
        this.turno = turno;
    }

    public Time getHoraInicio() {
        return horaInicio;
    }

    public void setHoraInicio(Time horaInicio) {
        this.horaInicio = horaInicio;
    }

    public Time getHoraFim() {
        return horaFim;
    }

    public void setHoraFim(Time horaFim) {
        this.horaFim = horaFim;
    }

}
```

6.4.262 TurnoDisponivel.java

```
/*
 * TurnoDisponivel.java
 * Created on 9 de Fevereiro de 2006, 22:18
 */

package estagio.bean;

/**
 *
 *
 */
public class TurnoDisponivel extends Turno{

    private Integer codAlunoEstagio;
    private Integer codTurnoDisponivel;

    /** Creates a new instance of TurnoDisponivel */
    public TurnoDisponivel() {
    }

    public Integer getCodAlunoEstagio() {
        return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
        this.codAlunoEstagio = codAlunoEstagio;
    }

    public Integer getCodTurnoDisponivel() {
        return codTurnoDisponivel;
    }

    public void setCodTurnoDisponivel(Integer
codTurnoDisponivel) {
        this.codTurnoDisponivel = codTurnoDisponivel;
    }

}
```

6.4.263 TurnoDTO.java

```
/*
 * TurnoDTO.java
 * Created on 10 de Fevereiro de 2006, 00:11
 */

package estagio.bean;

/**
 *
 *
 */
public class TurnoDTO {

    private Integer codTurno;
    private String turno;

    /** Creates a new instance of TurnoDTO */
    public TurnoDTO() {
    }

    public Integer getCodTurno() {
        return codTurno;
    }

    public void setCodTurno(Integer codTurno) {
        this.codTurno = codTurno;
    }

    public String getTurno() {
        return turno;
    }

    public void setTurno(String turno) {
        this.turno = turno;
    }

}
```

6.4.264 Usuario.java

```
/*
 * Usuario.java
 * Created on 27 de Janeiro de 2006, 11:38
 */

package estagio.bean;
```

```
/**
 * Classe utlizada em dadospessoais.jsp para fazer o radio
 *
 */
public class Usuario {

    /** Creates a new instance of Usuario */
    public Usuario() {
    }

    private String tipo;
    private String mensagem;

    public String getTipo() {
        return tipo;
    }

    public void setTipo(String tipo) {
        this.tipo = tipo;
    }

    public String getMensagem() {
        return mensagem;
    }

    public void setMensagem(String mensagem) {
        this.mensagem = mensagem;
    }
}
```

6.4.265 UsuarioEmpresa.java

```
/*
 * UsuarioEmpresa.java
 * Created on 30 de Janeiro de 2006, 17:38
 */

package estagio.bean;

/**
 *
 *
 */
public class UsuarioEmpresa {

    /** Creates a new instance of UsuarioEmpresa */
    public UsuarioEmpresa() {
    }

    private String cnpj;
    private Integer codAlunoEstagio;

    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

    public Integer getCodAlunoEstagio() {
        return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
        this.codAlunoEstagio = codAlunoEstagio;
    }

}
```

6.4.266 Vaga.java

```
/*
 * Vaga.java
 * Created on 28 de Janeiro de 2006, 10:07
 */

package estagio.bean;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class Vaga {

    /** Creates a new instance of Vaga */
    public Vaga() {
    }

    private Integer codVaga;
    private String cnpj;
    private String setor;
    private Integer horas;
    private String salario;
    private String sexo;
    private GregorianCalendar inclusao;
    private GregorianCalendar preenchimento;
    private List<Turno> turnos;
    private List<HabilidadeVaga> habilidades;
    private String locomocao;
    private String deficiente;
    private String atividadesFds;
    private String fumante;
    private Integer semestre;
    private String valeTransporte;
    private String outrosBeneficios;
    private Character estagioInterno;
    private List<Curso> cursos;

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

    public String getSetor() {
        return setor;
    }

    public void setSetor(String setor) {
        this.setor = setor;
    }

    public Integer getHoras() {
        return horas;
    }
```

```
    public void setHoras(Integer horas) {
        this.horas = horas;
    }

    public String getSalario() {
        return salario;
    }

    public void setSalario(String salario) {
        this.salario = salario;
    }

    public String getSexo() {
        return sexo;
    }

    public void setSexo(String sexo) {
        this.sexo = sexo;
    }

    public GregorianCalendar getInclusao() {
        return inclusao;
    }

    public void setInclusao(String inclusao) {
        GregorianCalendar data = new GregorianCalendar();
        SimpleDateFormat df = new
SimpleDateFormat("dd/MM/yyyy");
        try {
            if (inclusao != null){
                Date dataTest = df.parse(inclusao);
                data.setTime(dataTest);
                this.inclusao = data;
            } else {
                this.inclusao = new GregorianCalendar();
            }
        } catch (ParseException e){
            this.inclusao = new GregorianCalendar();
        }
    }

    public void setInclusao(GregorianCalendar inclusao) {
        this.inclusao = inclusao;
    }

    public GregorianCalendar getPreenchimento() {
        return preenchimento;
    }

    public void setPreenchimento(GregorianCalendar
preenchimento) {
        this.preenchimento = preenchimento;
    }

    public List<Turno> getTurnos() {
        return turnos;
    }

    public void setTurnos(List<Turno> turnos) {
        this.turnos = turnos;
    }

    public List<HabilidadeVaga> getHabilidades() {
        return habilidades;
    }

    public void setHabilidades(List<HabilidadeVaga> habilidades)
{
        this.habilidades = habilidades;
    }

    public String getLocomocao() {
        return locomocao;
    }

    public void setLocomocao(String locomocao) {
        this.locomocao = locomocao;
    }

    public String getDeficiente() {
        return deficiente;
    }

    public void setDeficiente(String deficiente) {
        this.deficiente = deficiente;
    }

    public String getAtividadesFds() {
        return atividadesFds;
    }

    public void setAtividadesFds(String atividadesFds) {
        this.atividadesFds = atividadesFds;
    }

    public String getFumante() {
        return fumante;
    }

    public void setFumante(String fumante) {
        this.fumante = fumante;
    }

    public Integer getSemestre() {
        return semestre;
    }

    public void setSemestre(Integer semestre) {
        this.semestre = semestre;
    }

    public String getValeTransporte() {
        return valeTransporte;
    }

    public void setValeTransporte(String valeTransporte) {
        this.valeTransporte = valeTransporte;
    }

    public String getOutrosBeneficios() {
        return outrosBeneficios;
    }

    public void setOutrosBeneficios(String outrosBeneficios) {
        this.outrosBeneficios = outrosBeneficios;
    }

    public List<Curso> getCursos() {
        return cursos;
    }

    public void setCursos(List<Curso> cursos) {
        this.cursos = cursos;
    }

    public boolean equals(Object o) {
        if (!(o instanceof Vaga)) return false;
        if (o == this) return true;
        Vaga v = (Vaga) o;
        // só para simplificar estou supondo que nenhum dos campos
é null
        return codVaga.equals(v.getCodVaga());
    }
    public int hashCode() {
```

```
        int ret = codVaga.hashCode();
        ret = ret * 37 + cnpj.hashCode();
        ret = ret * 37 + setor.hashCode();
        ret = ret * 37 + horas.hashCode();
        ret = ret * 37 + salario.hashCode();
        ret = ret * 37 + sexo.hashCode();
        ret = ret * 37 + inclusao.hashCode();
        ret = ret * 37 + locomocao.hashCode();
        ret = ret * 37 + deficiente.hashCode();
        ret = ret * 37 + atividadesFds.hashCode();
        ret = ret * 37 + fumante.hashCode();
        // se houver mais campos, vá repetindo a expressão acima
"ret = ret * 37 + hashCode do proximo campo"
        // por exemplo, ret = ret * 37 + telefone.hashCode();
        return ret;
    }

    public Character getEstagioInterno() {
        return estagioInterno;
    }

    public void setEstagioInterno(Character estagioInterno) {
        this.estagioInterno = estagioInterno;
    }

}
```

6.4.267 VagaCurso.java

```
/*
 * VagaCurso.java
 * Created on 10 de Abril de 2006, 14:30
 */

package estagio.bean;

/**
 *
 *
 */
public class VagaCurso {

    private Integer codVagaCurso;
    private Integer codCurso;
    private Integer codVaga;

    /** Creates a new instance of VagaCurso */
    public VagaCurso() {
    }

    public Integer getCodVagaCurso() {
        return codVagaCurso;
    }

    public void setCodVagaCurso(Integer codVagaCurso) {
        this.codVagaCurso = codVagaCurso;
    }

    public Integer getCodCurso() {
        return codCurso;
    }

    public void setCodCurso(Integer codCurso) {
        this.codCurso = codCurso;
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

}
```

6.4.268 VagaDTO.java

```
/*
 * VagaDTO.java
 * Created on 30 de Janeiro de 2006, 17:11
 */

package estagio.bean;

/**
 *
 *
 */
public class VagaDTO {

    /** Creates a new instance of VagaDTO */
    public VagaDTO() {
    }

    private Integer codVaga;
    private String setor;
    private String horas;
    private String salario;
    private String sexo;
    private String inclusao;
    private String habilidades;
    private String turnos;
    private String cursos;
    private String empresa;
    private String situacao;
    private Integer semestre;
    private String valeTransporte;
    private String outrosBeneficios;

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public String getSetor() {
        return setor;
    }

    public void setSetor(String setor) {
        this.setor = setor;
    }

    public String getHoras() {
        return horas;
    }

    public void setHoras(String horas) {
        this.horas = horas;
    }

    public String getSalario() {
        return salario;
    }

    public void setSalario(String salario) {
        this.salario = salario;
    }

    public String getSexo() {
        return sexo;
```

```
}

public void setSexo(String sexo) {
    this.sexo = sexo;
}

public String getInclusao() {
    return inclusao;
}

public void setInclusao(String inclusao) {
    this.inclusao = inclusao;
}

public String getHabilidades() {
    return habilidades;
}

public void setHabilidades(String habilidades) {
    this.habilidades = habilidades;
}

public String getTurnos() {
    return turnos;
}

public void setTurnos(String turnos) {
    this.turnos = turnos;
}

public String getCursos() {
    return cursos;
}

public void setCursos(String cursos) {
    this.cursos = cursos;
}

public String getEmpresa() {
    return empresa;
}

public void setEmpresa(String empresa) {
    this.empresa = empresa;
}

public String getSituacao() {
    return situacao;
}

public void setSituacao(String situacao) {
    this.situacao = situacao;
}

public Integer getSemestre() {
    return semestre;
}

public void setSemestre(Integer semestre) {
    this.semestre = semestre;
}

public String getValeTransporte() {
    return valeTransporte;
}

public void setValeTransporte(String valeTransporte) {
    this.valeTransporte = valeTransporte;
}

public String getOutrosBeneficios() {
    return outrosBeneficios;
}

public void setOutrosBeneficios(String outrosBeneficios) {
    this.outrosBeneficios = outrosBeneficios;
}
}
```

6.4.269 VagaSelecionadaDTO.java

```
/*
 * VagaSelecionadaDTO.java
 * Created on 19 de Fevereiro de 2006, 15:32
 */

package estagio.bean;

/**
 *
 *
 */
public class VagaSelecionadaDTO {

    private Integer codVaga;
    private String setor;
    private String horas;
    private String salario;
    private String inclusao;
    private String habilidades;
    private String turnos;
    private String razaoSocial;
    private String endereco;
    private String contato;
    private Integer semestre;
    private String valeTransporte;
    private String outrosBeneficios;

    /** Creates a new instance of VagaSelecionadaDTO */
    public VagaSelecionadaDTO() {
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public String getSetor() {
        return setor;
    }

    public void setSetor(String setor) {
        this.setor = setor;
    }

    public String getHoras() {
        return horas;
    }

    public void setHoras(String horas) {
        this.horas = horas;
    }

    public String getSalario() {
        return salario;
    }

    public void setSalario(String salario) {
        this.salario = salario;
    }
```

```
public String getInclusao() {
    return inclusao;
}

public void setInclusao(String inclusao) {
    this.inclusao = inclusao;
}

public String getHabilidades() {
    return habilidades;
}

public void setHabilidades(String habilidades) {
    this.habilidades = habilidades;
}

public String getTurnos() {
    return turnos;
}

public void setTurnos(String turnos) {
    this.turnos = turnos;
}

public String getRazaoSocial() {
    return razaoSocial;
}

public void setRazaoSocial(String razaoSocial) {
    this.razaoSocial = razaoSocial;
}

public String getEndereco() {
    return endereco;
}

public void setEndereco(String endereco) {
    this.endereco = endereco;
}

public String getContato() {
    return contato;
}

public void setContato(String contato) {
    this.contato = contato;
}

public Integer getSemestre() {
    return semestre;
}

public void setSemestre(Integer semestre) {
    this.semestre = semestre;
}

public String getValeTransporte() {
    return valeTransporte;
}

public void setValeTransporte(String valeTransporte) {
    this.valeTransporte = valeTransporte;
}

public String getOutrosBeneficios() {
    return outrosBeneficios;
}

public void setOutrosBeneficios(String outrosBeneficios) {
    this.outrosBeneficios = outrosBeneficios;
}
}
```

6.4.270 package estagio.dao

6.4.271 AgentesIntegracaoDao.java

```
/*
 * AgentesIntegracaoDao.java
 * Created on 27 de Fevereiro de 2006, 11:31
 */

package estagio.dao;

import estagio.bean.AgenteIntegracao;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class AgentesIntegracaoDao extends BaseDao{

    EmpresaDao empresaDao;
    /** Creates a new instance of AgenteIntegracaoDao */

    public AgentesIntegracaoDao() throws Exception{
        empresaDao = new EmpresaDao();
    }

    public List<AgenteIntegracao> listarAgentesIntegracao()
throws SQLException, Exception {
        return empresaDao.listarAgentesIntegracao();
    }

    public AgenteIntegracao buscarAgenteIntegracao(String cnpj)
throws SQLException, Exception {
        return empresaDao.buscarAgenteIntegracao(cnpj);
    }

    public void excluirAgenteIntegracao(String cnpj) throws
SQLException, Exception {
        empresaDao.excluirAgenteIntegracao(cnpj);
    }
}
```

6.4.272 AlunoEstagioDao.java

```
/*
 * AlunoEstagioDao.java
 * Created on 26 de Janeiro de 2006, 17:42
 */

package estagio.dao;

import estagio.bean.AlunoEstagio;
import estagio.bean.HabilidadeAluno;
import estagio.bean.Pessoa;
import estagio.bean.TurnoDisponivel;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author Ricardo
```

```
*/
public class AlunoEstagioDao extends BaseDao{

  PessoaDao pessoaDao;
  HabilidadesAlunoDao habilidadesAlunoDao;
  /** Creates a new instance of AlunoEstagioDao */
  public AlunoEstagioDao() throws Exception{
    pessoaDao = new PessoaDao();
    habilidadesAlunoDao = new HabilidadesAlunoDao();
  }

  public void incluirAlunoEstagio(AlunoEstagio alunoEstagio)
throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Pessoa pessoa = new Pessoa();
    try {
      pessoa =
pessoaDao.buscarPessoa(alunoEstagio.getUsuario());
      if(pessoa == null){
        pessoaDao.incluirPessoa(alunoEstagio);
        pessoa =
pessoaDao.buscarNovaPessoa(alunoEstagio.getUsuario());
      }
      AlunoEstagio alunoEstagioTeste =
buscarAlunoEstagio(alunoEstagio.getUsuario());
      if (pessoa != null && alunoEstagioTeste == null){
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        SimpleDateFormat sdf = new
SimpleDateFormat("yyyy-MM-dd");
        String dataNasc =
sdf.format(alunoEstagio.getDataNascimento().getTime());
        sb.append("insert into alunoestagio (disponivelfds,
fumante, deficiente, locomocao, pessoa_codPessoa, email, fone,
status, sexo, cursos_codcurso, semestre, rg, rg_naturalidade, cpf,
matricula, cep, rua, numero, complemento, bairro, municipio,
estado, pais)");
        sb.append(" values ('");
        sb.append(alunoEstagio.getDisponivelFds());
        sb.append("','");
        sb.append(alunoEstagio.getFumante());
        sb.append("','");
        sb.append(alunoEstagio.getDeficiente());
        sb.append("','");
        sb.append(alunoEstagio.getLocomocao());
        sb.append("','");
        sb.append(pessoa.getCodPessoa());
        sb.append("','");
        sb.append(alunoEstagio.getEmail());
        sb.append("','");
        sb.append(alunoEstagio.getFone());
        sb.append("','");
        sb.append(alunoEstagio.getStatus());
        sb.append("','");
        sb.append(alunoEstagio.getSexo());
        sb.append("','");
        sb.append(alunoEstagio.getCurso().getCodCurso());
        sb.append("','");
        sb.append(alunoEstagio.getSemestre());
        sb.append("','");
        sb.append(alunoEstagio.getRg());
        sb.append("','");
        sb.append(alunoEstagio.getRg_naturalidade());
        sb.append("','");
        sb.append(alunoEstagio.getCpf());
        sb.append("','");
        sb.append(alunoEstagio.getMatricula());
        sb.append("','");
        sb.append(alunoEstagio.getCep());
        sb.append("','");
        sb.append(alunoEstagio.getRua());
        sb.append("','");
        sb.append(alunoEstagio.getNumero());
        sb.append("','");
        sb.append(alunoEstagio.getComplemento());
        sb.append("','");
        sb.append(alunoEstagio.getBairro());
        sb.append("','");
        sb.append(alunoEstagio.getMunicipio());
        sb.append("','");
        sb.append(alunoEstagio.getEstado());
        sb.append("','");
        sb.append(alunoEstagio.getPais());
        sb.append("')");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      }
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirAlunoEstagio(AlunoEstagio alunoEstagio)
throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
    ResultSet rs = null;
    try {

habilidadesAlunoDao.excluirHabilidadesAluno(alunoEstagio.get
CodAlunoEstagio());
      if("empresa".equals(alunoEstagio.getStatus())){

usuarioEmpresaDao.excluirUsuarioEmpresa(alunoEstagio.getCod
AlunoEstagio());
      }
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from AlunoEstagio where
codalunoestagio=");
      sb.append("'");
      sb.append(alunoEstagio.getCodAlunoEstagio());
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
      pessoaDao.excluirPessoa(alunoEstagio.getUsuario());
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void alterarAlunoEstagio(AlunoEstagio alunoEstagio)
throws SQLException, Exception {
    Connection conn = null;
```

```
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            pessoaDao.alterarPessoa(alunoEstagio);
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
            String dataNasc =
sdf.format(alunoEstagio.getDataNascimento().getTime());
            StringBuffer sb = new StringBuffer();
            sb.append("update alunoestagio set
disponivelfds='").append(alunoEstagio.getDisponivelFds())
            .append("',
fumante='").append(alunoEstagio.getFumante())
            .append("',
deficiente='").append(alunoEstagio.getDeficiente())
            .append("',
locomocao='").append(alunoEstagio.getLocomocao())
            .append("', email='").append(alunoEstagio.getEmail())
            .append("', fone='").append(alunoEstagio.getFone())
            .append("', status='").append(alunoEstagio.getStatus())
            .append("', sexo='").append(alunoEstagio.getSexo())
            .append("',
cursos_codcurso='").append(alunoEstagio.getCurso().getCodCurs
o())
            .append("', semestre
='").append(alunoEstagio.getSemestre())
            .append("', rg ='").append(alunoEstagio.getRg())
            .append("', rg_naturalidade
='").append(alunoEstagio.getRg_naturalidade())
            .append("',
cpf='").append(alunoEstagio.getCpf().toString())
            .append("',
matricula='").append(alunoEstagio.getMatricula())
            .append("', cep='").append(alunoEstagio.getCep())
            .append("', rua='").append(alunoEstagio.getRua())
            .append("', numero='").append(alunoEstagio.getNumero())
            .append("',
complemento='").append(alunoEstagio.getComplemento())
            .append("', bairro='").append(alunoEstagio.getBairro())
            .append("',
municipio='").append(alunoEstagio.getMunicipio())
            .append("', estado='").append(alunoEstagio.getEstado())
            .append("', pais='").append(alunoEstagio.getPais())
            .append("' where codalunoestagio
='").append(alunoEstagio.getCodAlunoEstagio()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public AlunoEstagio buscarAlunoEstagio(String usuario)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        AlunoEstagio alunoEstagio = new AlunoEstagio();
        Pessoa pessoa = new Pessoa();
        try {
            pessoa = pessoaDao.buscarPessoa(usuario);
            if(pessoa != null){
                conn = getConnection();
                StringBuffer sb = new StringBuffer();
                sb.append("select codalunoestagio from alunoestagio
where pessoa_codPessoa='")
                .append(pessoa.getCodPessoa())
                .append("'");
                stmt = conn.prepareStatement(sb.toString());
                rs = stmt.executeQuery();
                if(rs.next()){
                    alunoEstagio =
buscarAlunoEstagio(Integer.valueOf(rs.getInt("codalunoestagio")
));
                } else {
                    return null;
                }
            } else {
                return null;
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return alunoEstagio;
    }

    public List<AlunoEstagio> listarAlunoEstagio() throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<AlunoEstagio> lista = new ArrayList<AlunoEstagio>();
        TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
        CursoDao cursoDao = new CursoDao();
        try {
            conn = getConnection();
            String sb = "select codalunoestagio from alunoestagio";
            stmt = conn.prepareStatement(sb);
            rs = stmt.executeQuery();
            while(rs.next()){
                AlunoEstagio alunoEstagio =
buscarAlunoEstagio(Integer.valueOf(rs.getInt("codalunoestagio")
));
                lista.add(alunoEstagio);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return lista;
    }

    public AlunoEstagio buscarAlunoEstagio(Integer
codAlunoEstagio) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        AlunoEstagio alunoEstagio = new AlunoEstagio();
        Pessoa pessoa = new Pessoa();
        TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
        CursoDao cursoDao = new CursoDao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codalunoestagio, sexo, disponivelfds,
fumante, deficiente, locomocao, cursos_codcurso,
pessoa_codPessoa, email, fone, status, semestre, rg,
rg_naturalidade, cpf, matricula, cep, rua, numero, complemento,
```

```
bairro, municipio, estado, pais from alunoestagio where
codalunoestagio='')
        .append(codAlunoEstagio)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(rs.getInt("pessoa_codPe
ssoa")));
          if (pessoa != null){

alunoEstagio.setCodAlunoEstagio(Integer.valueOf(rs.getInt("cod
alunoestagio")));
            alunoEstagio.setCodPessoa(pessoa.getCodPessoa());

alunoEstagio.setDataNascimento(pessoa.getDataNascimento());

alunoEstagio.setDeficiente(rs.getString("deficiente"));

alunoEstagio.setDisponivelFds(rs.getString("disponivelfds"));
            alunoEstagio.setFumante(rs.getString("fumante"));

alunoEstagio.setLocomocao(rs.getString("locomocao"));

alunoEstagio.setNomePessoa(pessoa.getNomePessoa());
            alunoEstagio.setSenha(pessoa.getSenha());
            alunoEstagio.setSexo(rs.getString("sexo"));
            alunoEstagio.setUsuario(pessoa.getUsuario());
            alunoEstagio.setEmail(rs.getString("email"));
            alunoEstagio.setFone(rs.getString("fone"));
            alunoEstagio.setStatus(rs.getString("status"));

alunoEstagio.setSemestre(Integer.valueOf(rs.getInt("semestre")));
            List<HabilidadeAluno> habilidades = new
ArrayList<HabilidadeAluno>();
            habilidades =
habilidadesAlunoDao.listarHabilidadeAluno(alunoEstagio.getCod
AlunoEstagio());
            alunoEstagio.setHabilidades(habilidades);
            List<TurnoDisponivel> turnosDisponiveis = new
ArrayList<TurnoDisponivel>();
            turnosDisponiveis =
turnoDisponivelDao.listarTurnosDisponiveis(alunoEstagio.getCo
dAlunoEstagio());

alunoEstagio.setTurnosDisponiveis(turnosDisponiveis);

alunoEstagio.setCurso(cursoDao.buscarCurso(Integer.valueOf(rs.
getInt("cursos_codcurso"))));

alunoEstagio.setRg(Long.valueOf(rs.getInt("rg")).toString());

alunoEstagio.setRg_naturalidade(rs.getString("rg_naturalidade"));

alunoEstagio.setCpf(Long.valueOf(rs.getLong("cpf")));

alunoEstagio.setMatricula(Long.valueOf(rs.getInt("matricula")).t
oString());
            alunoEstagio.setCep(rs.getString("cep"));
            alunoEstagio.setRua(rs.getString("rua"));
            alunoEstagio.setNumero(rs.getString("numero"));

alunoEstagio.setComplemento(rs.getString("complemento"));
            alunoEstagio.setBairro(rs.getString("bairro"));

alunoEstagio.setMunicipio(rs.getString("municipio"));
            alunoEstagio.setEstado(rs.getString("estado"));
            alunoEstagio.setPais(rs.getString("pais"));
          } else {
            return null;
          }
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } catch (Exception e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return alunoEstagio;
    }
    public List<AlunoEstagio> listarAlunoEstagio(String status)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<AlunoEstagio> lista = new ArrayList<AlunoEstagio>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codalunoestagio from alunoestagio
where upper(status) = upper('")
        .append(status).append("')");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){
          AlunoEstagio alunoEstagio =
buscarAlunoEstagio(Integer.valueOf(rs.getInt("codalunoestagio")
));
          lista.add(alunoEstagio);
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return lista;
    }

    public List<AlunoEstagio> buscarEstagiario(String
parteNome) throws SQLException, Exception {
      TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
      List<AlunoEstagio> estagiarios = new
ArrayList<AlunoEstagio>();
      List<Pessoa> pessoas = new ArrayList<Pessoa>();
      pessoas = pessoaDao.buscarEstagiario(parteNome);
      for(Pessoa pessoa: pessoas){
        AlunoEstagio alunoEstagio =
buscarAlunoEstagio(pessoa.getUsuario());
        if(alunoEstagio != null){
          if("aluno".equals(alunoEstagio.getStatus())){
            estagiarios.add(alunoEstagio);
          }
        }
      }
      return estagiarios;
    }

    public List<AlunoEstagio> buscarUsuario(String parteNome)
throws SQLException, Exception {
      TurnoDisponivelDao turnoDisponivelDao = new
TurnoDisponivelDao();
      List<AlunoEstagio> usuarios = new
ArrayList<AlunoEstagio>();
      List<Pessoa> pessoas = new ArrayList<Pessoa>();
      pessoas = pessoaDao.buscarEstagiario(parteNome);
      for(Pessoa pessoa: pessoas){
        AlunoEstagio alunoEstagio =
buscarAlunoEstagio(pessoa.getUsuario());
        if(alunoEstagio != null){
```

```
            usuarios.add(alunoEstagio);
        }
    }
    return usuarios;
  }

}
```

6.4.273 BaseDao.java

```
/*
 * BaseDao.java
 * Created on 25 de Outubro de 2005, 11:36
 */

package estagio.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;

/**
 *
 *
 */
public class BaseDao {

  protected static DataSource dataSource;

  public BaseDao(){
    if (dataSource == null) {
      try {
        Context initCtx = new InitialContext();
        dataSource = (DataSource)
initCtx.lookup("java:comp/env/jdbc/postgres");
      } catch (NamingException ex) {
        System.out.println(ex.getMessage());
      }
    }
  }

  protected Connection getConnection() throws SQLException {
    Connection conn = null;
    try {
      conn = dataSource.getConnection();
    }
    catch (SQLException e) {
      throw e;
    }
    return conn;
  }


  protected void closeConnection(
    Connection conn,
    PreparedStatement stmt,
    ResultSet rs) {
    if (rs != null) {
      try {
        rs.close();
      } catch (SQLException e) {
      }
    }
    if (stmt != null) {
      try {
        stmt.close();
      } catch (SQLException e) {
      }
    }
    if (conn != null) {
      try {
        conn.close();
      } catch (SQLException e) {
      }
    }
  }

}
```

6.4.274 ConvenioDao.java

```
/*
 * ConvenioDao.java
 * Created on 17 de Janeiro de 2006, 20:57
 */

package estagio.dao;

import estagio.bean.Convenio;
import estagio.bean.Empresa;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class ConvenioDao extends BaseDao{

  /** Creates a new instance of ConvenioDao */
  public ConvenioDao() throws Exception{
  }

  public List<Convenio> listarConvenios() throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    ArrayList<Convenio> convenios  = new
ArrayList<Convenio>();
    EmpresaDao empresaDao = new EmpresaDao();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select codConvenio, datainicio, descricao,
datafim, empresa_cnpj from convenio order by empresa_cnpj");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      while (rs.next()) {
        Convenio convenio = new Convenio();
        GregorianCalendar dataInicio = new
GregorianCalendar();
        GregorianCalendar dataFim = new
GregorianCalendar();

convenio.setCodConvenio(Integer.valueOf(rs.getInt("codConveni
o")));
        dataFim.setTime(rs.getDate("dataFim"));
        dataInicio.setTime(rs.getDate("dataInicio"));
        convenio.setDataFim(dataFim);
        convenio.setDataInicio(dataInicio);
        convenio.setDescricao(rs.getString("descricao"));
        String cnpj = rs.getString("empresa_cnpj");
        Empresa empresa = empresaDao.buscarEmpresa(cnpj);
        convenio.setEmpresa(empresa);
        convenios.add(convenio);
```

```
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return convenios;
    }

    public List<Convenio> listarConvenios(String cnpj) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        ArrayList<Convenio> convenios  = new
ArrayList<Convenio>();
        EmpresaDao empresaDao = new EmpresaDao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codConvenio, datainicio, descricao,
datafim, empresa_cnpj from convenio where empresa_cnpj='")
                .append(cnpj).append("' order by empresa_cnpj");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while (rs.next()) {
                Convenio convenio = new Convenio();
                GregorianCalendar dataInicio = new
GregorianCalendar();
                GregorianCalendar dataFim = new
GregorianCalendar();

convenio.setCodConvenio(Integer.valueOf(rs.getInt("codConveni
o")));
                dataFim.setTime(rs.getDate("dataFim"));
                dataInicio.setTime(rs.getDate("dataInicio"));
                convenio.setDataFim(dataFim);
                convenio.setDataInicio(dataInicio);
                convenio.setDescricao(rs.getString("descricao"));
                Empresa empresa = empresaDao.buscarEmpresa(cnpj);
                convenio.setEmpresa(empresa);
                convenios.add(convenio);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return convenios;
    }


    public Convenio buscarConvenio(String cnpj) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Convenio convenio = new Convenio();
        EmpresaDao empresaDao = new EmpresaDao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codConvenio, datainicio, descricao,
datafim, empresa_cnpj from convenio where empresa_cnpj='")
                .append(cnpj)
                .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()) {
                GregorianCalendar dataInicio = new
GregorianCalendar();
                GregorianCalendar dataFim = new
GregorianCalendar();

convenio.setCodConvenio(Integer.valueOf(rs.getInt("codConveni
o")));
                dataFim.setTime(rs.getDate("dataFim"));
                dataInicio.setTime(rs.getDate("dataInicio"));
                convenio.setDataFim(dataFim);
                convenio.setDataInicio(dataInicio);
                convenio.setDescricao(rs.getString("descricao"));
                Empresa empresa =
empresaDao.buscarEmpresa(rs.getString("empresa_cnpj"));
                convenio.setEmpresa(empresa);
            } else {
                GregorianCalendar dataInicio = new
GregorianCalendar();
                GregorianCalendar dataFim = new
GregorianCalendar();
                convenio.setCodConvenio(Integer.valueOf(0));
                convenio.setDataFim(dataFim);
                convenio.setDataInicio(dataInicio);
                convenio.setDescricao("");
                Empresa empresa = new Empresa();
                convenio.setEmpresa(empresa);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return convenio;
    }

    public Convenio buscarConvenio(Convenio c) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Convenio convenio = new Convenio();
        EmpresaDao empresaDao = new EmpresaDao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codConvenio, datainicio, descricao,
datafim, empresa_cnpj from convenio where codConvenio='")
                .append(c.getCodConvenio())
                .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()) {
                GregorianCalendar dataInicio = new
GregorianCalendar();
                GregorianCalendar dataFim = new
GregorianCalendar();

convenio.setCodConvenio(Integer.valueOf(rs.getInt("codConveni
o")));
                dataFim.setTime(rs.getDate("dataFim"));
                dataInicio.setTime(rs.getDate("dataInicio"));
                convenio.setDataFim(dataFim);
                convenio.setDataInicio(dataInicio);
                convenio.setDescricao(rs.getString("descricao"));
                Empresa empresa =
empresaDao.buscarEmpresa(rs.getString("empresa_cnpj"));
                convenio.setEmpresa(empresa);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return convenio;
```

```
  }

  public void incluirConvenio(Convenio c) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
      String dataInicio =
sdf.format(c.getDataInicio().getTime());
      String dataFim = sdf.format(c.getDataFim().getTime());
      if(c.getEmpresa() == null){
        c.setEmpresa(new Empresa());
      }
      sb.append("insert into Convenio (datainicio, descricao,
datafim, empresa_cnpj)");
      sb.append(" values ('");
      sb.append(dataInicio);
      sb.append("','");
      sb.append(c.getDescricao());
      sb.append("','");
      sb.append(dataFim);
      sb.append("','");
      sb.append(c.getEmpresa().getCnpj());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirConvenio(Convenio convenio) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from Convenio where codConvenio=");
      sb.append("'");
      sb.append(convenio.getCodConvenio());
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirConvenio(String cnpj) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from Convenio where empresa_cnpj=");
      sb.append("'");
      sb.append(cnpj);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void alterarConvenio(Convenio c) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    if(c.getEmpresa() == null){
        c.setEmpresa(new Empresa());
    }
    try {
      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
      String dataInicio =
sdf.format(c.getDataInicio().getTime());
      String dataFim = sdf.format(c.getDataFim().getTime());
      StringBuffer sb = new StringBuffer();
      sb.append("update convenio set empresa_cnpj =
'").append(c.getEmpresa().getCnpj())
      .append("', dataInicio='").append(c.getDataInicio())
      .append("', dataFim='").append(c.getDataFim())
      .append("', descricao='").append(c.getDescricao())
      .append("' where codConvenio
='").append(c.getCodConvenio()).append("'");
      conn = getConnection();
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } catch (Exception e){
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }
}
```

6.4.275 CursoDao.java

```
/*
 * CursoDao.java
 * Created on 2 de Março de 2006, 14:26
```

```
*/

package estagio.dao;

import estagio.bean.Curso;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class CursoDao extends BaseDao {

   /** Creates a new instance of CursoDao */
   public CursoDao() {
   }

   public Curso buscarCurso(Integer codCurso) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Curso curso = new Curso();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select codcurso, duracao, nome from cursos
where codcurso='");
         sb.append(codCurso);
         sb.append("'");
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         if(rs.next()){

curso.setCodCurso(Integer.valueOf(rs.getInt("codcurso")));

curso.setDuracao(Integer.valueOf(rs.getInt("duracao")));
            curso.setNomeCurso(rs.getString("nome"));
         }
      } catch (SQLException e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
      return curso;
   }

   public void incluirCurso(Curso curso) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("insert into cursos ( nome, duracao)");
         sb.append("  values ('");
         sb.append(curso.getNomeCurso());
         sb.append("','");
         sb.append(curso.getDuracao());
         sb.append("')");
         stmt = conn.prepareStatement(sb.toString());
         stmt.executeUpdate();
      } catch (SQLException e) {
         throw e;
      } finally {
         if (rs != null) {
            rs.close();
         }
         if (stmt != null) {
            stmt.close();
         }
         closeConnection(conn, stmt, rs);
      }
   }

   public void excluirCurso(Curso curso) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("delete from cursos where codcurso=");
         sb.append("'");
         sb.append(curso.getCodCurso());
         sb.append("'");
         stmt = conn.prepareStatement(sb.toString());
         stmt.executeUpdate();
      } catch (SQLException e) {
         throw e;
      } finally {
         if (rs != null) {
            rs.close();
         }
         if (stmt != null) {
            stmt.close();
         }
         closeConnection(conn, stmt, rs);
      }
   }

   public void alterarCurso(Curso curso) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         StringBuffer sb = new StringBuffer();
         sb.append("update cursos set
nome='").append(curso.getNomeCurso())
         .append("', duracao='").append(curso.getDuracao())
         .append("' where codcurso
='").append(curso.getCodCurso()).append("'");
         conn = getConnection();
         stmt = conn.prepareStatement(sb.toString());
         stmt.executeUpdate();
      } catch (SQLException e) {
         throw e;
      } catch (Exception e){
         throw e;
      } finally {
         if (rs != null) {
            rs.close();
         }
         if (stmt != null) {
            stmt.close();
         }
         closeConnection(conn, stmt, rs);
      }
   }

   public List<Curso> listarCursos() throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Curso> cursos = new ArrayList<Curso>();
```

```
    try {
      conn = getConnection();
      stmt = conn.prepareStatement("select codcurso, duracao,
nome from cursos order by nome");
      rs = stmt.executeQuery();
      while(rs.next()){
        Curso curso = new Curso();

curso.setCodCurso(Integer.valueOf(rs.getInt("codcurso")));

curso.setDuracao(Integer.valueOf(rs.getInt("duracao")));
        curso.setNomeCurso(rs.getString("nome"));
        cursos.add(curso);
      }
    } catch (SQLException e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return cursos;
  }

}
```

6.4.276 EmpresaDao.java

```
/*
 * EmpresaDao.java
 * Created on 3 de Novembro de 2005, 22:28
 */

package estagio.dao;
import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Convenio;
import estagio.bean.Empresa;
import estagio.bean.Endereco;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class EmpresaDao extends BaseDao{

  /** Creates a new instance of EmpresaDao */
  public EmpresaDao() throws Exception{
  }

  public Empresa buscarEmpresa(String cnpj) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Empresa empresa = new Empresa();
    try {
      Endereco enderecoEmpresa = new Endereco();
      EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select cnpj,
enderecoEmpresa_codEnderecoEmpresa,")
      .append(" razaoSocial, ramo, site, email, nomeFantasia,
fone, fax, representante,")
      .append(" cargoRepresentante, numprocesso,
agente_integracao from Empresa where cnpj='")
      .append(cnpj)
      .append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if(rs.next()){
        enderecoEmpresa =
enderecoEmpresaDao.buscarEnderecoEmpresa(rs.getInt("enderec
oEmpresa_codEnderecoEmpresa"));
        empresa.setCnpj(rs.getString("cnpj"));
        empresa.setRazaoSocial(rs.getString("razaosocial"));
        empresa.setRamo(rs.getString("ramo"));
        empresa.setSite(rs.getString("site"));
        empresa.setEmail(rs.getString("email"));

empresa.setNomeFantasia(rs.getString("nomefantasia"));
        empresa.setFone(rs.getString("fone"));

empresa.setRepresentante(rs.getString("representante"));

empresa.setCargoRepresentante(rs.getString("cargorepresentante"
));
        empresa.setEnderecoEmpresa(enderecoEmpresa);
        empresa.setFax(rs.getString("fax"));
        empresa.setNumProcesso(rs.getString("numprocesso"));

empresa.setAgenteIntegracao(rs.getString("agente_integracao"));

      } else {
        return null;
      }
    } catch (SQLException e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return empresa;
  }

  public List<Empresa> listarEmpresas() throws SQLException,
Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    List<Empresa> empresas = new ArrayList<Empresa>();
    try {
      EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select cnpj,
enderecoEmpresa_codEnderecoEmpresa,")
      .append(" razaoSocial, ramo, site, email, nomeFantasia,
fone, fax, representante,")
      .append(" cargoRepresentante, numprocesso,
agente_integracao from empresa order by razaoSocial");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      while(rs.next()){
        Endereco enderecoEmpresa = new Endereco();
        Empresa empresa = new Empresa();
        enderecoEmpresa =
enderecoEmpresaDao.buscarEnderecoEmpresa(rs.getInt("enderec
oEmpresa_codEnderecoEmpresa"));
        empresa.setCnpj(rs.getString("cnpj"));
        empresa.setRazaoSocial(rs.getString("razaosocial"));
        empresa.setRamo(rs.getString("ramo"));
        empresa.setSite(rs.getString("site"));
        empresa.setEmail(rs.getString("email"));

empresa.setNomeFantasia(rs.getString("nomefantasia"));
        empresa.setFone(rs.getString("fone"));
```

```
empresa.setRepresentante(rs.getString("representante"));

empresa.setCargoRepresentante(rs.getString("cargorepresentante"
));
            empresa.setEnderecoEmpresa(enderecoEmpresa);
            empresa.setFax(rs.getString("fax"));
            empresa.setNumProcesso(rs.getString("numprocesso"));

empresa.setAgenteIntegracao(rs.getString("agente_integracao"));
            empresas.add(empresa);
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return empresas;
      }

      public List<Empresa> listarEmpresasSemConvenio() throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Empresa> empresas = new ArrayList<Empresa>();
        try {
          EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select cnpj from empresa except (select
empresa_cnpj from convenio)");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while(rs.next()){
            Endereco enderecoEmpresa = new Endereco();
            Empresa empresa = new Empresa();
            empresa = buscarEmpresa(rs.getString("cnpj"));
            empresas.add(empresa);
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return empresas;
      }

      public List<Empresa> listarEmpresasConvenioVencido()
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Empresa> empresas = new ArrayList<Empresa>();
        ConvenioDao convenioDao = new ConvenioDao();
        try {
          EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select cnpj,
enderecoEmpresa_codEnderecoEmpresa,")
          .append(" razaoSocial, ramo, site, email, nomeFantasia,
fone, fax, representante,")
          .append(" cargoRepresentante, numprocesso,
agente_integracao from empresa");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while(rs.next()){
            Endereco enderecoEmpresa = new Endereco();
            Empresa empresa = new Empresa();
            Convenio convenio =
convenioDao.buscarConvenio(rs.getString("cnpj"));
            GregorianCalendar data = new GregorianCalendar();
            if(data.after(convenio.getDataFim())){
              enderecoEmpresa =
enderecoEmpresaDao.buscarEnderecoEmpresa(rs.getInt("enderec
oEmpresa_codEnderecoEmpresa"));
              empresa.setCnpj(rs.getString("cnpj"));
              empresa.setRazaoSocial(rs.getString("razaosocial"));
              empresa.setRamo(rs.getString("ramo"));
              empresa.setSite(rs.getString("site"));
              empresa.setEmail(rs.getString("email"));

empresa.setNomeFantasia(rs.getString("nomefantasia"));
              empresa.setFone(rs.getString("fone"));

empresa.setRepresentante(rs.getString("representante"));

empresa.setCargoRepresentante(rs.getString("cargorepresentante"
));
              empresa.setEnderecoEmpresa(enderecoEmpresa);
              empresa.setFax(rs.getString("fax"));

empresa.setNumProcesso(rs.getString("numprocesso"));

empresa.setAgenteIntegracao(rs.getString("agente_integracao"));
              empresas.add(empresa);
            }
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return empresas;
      }


      public void incluirEmpresa(Empresa empresa) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        EnderecoEmpresaDao enderecoDao = new
EnderecoEmpresaDao();

enderecoDao.incluirEnderecoEmpresa(empresa.getEnderecoEmpr
esa());

empresa.setEnderecoEmpresa(enderecoDao.buscarEnderecoEmpr
esa(empresa.getCnpj()));
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("insert into Empresa (cnpj,
enderecoEmpresa_codEnderecoEmpresa, razaoSocial, ramo,
site,")
          .append(" email, nomeFantasia, fone, fax, representante,
cargoRepresentante, numprocesso, agente_integracao)")
          .append("  values ('")
          .append(empresa.getCnpj())
          .append("','")

.append(empresa.getEnderecoEmpresa().getCodEndereco())
          .append("','")
          .append(empresa.getRazaoSocial())
          .append("','")
          .append(empresa.getRamo())
          .append("','")
          .append(empresa.getSite())
          .append("','")
          .append(empresa.getEmail())
```

```
        .append("','")
        .append(empresa.getNomeFantasia())
        .append("','")
        .append(empresa.getFone())
        .append("','")
        .append(empresa.getFax())
        .append("','")
        .append(empresa.getRepresentante())
        .append("','")
        .append(empresa.getCargoRepresentante())
        .append("','")
        .append(empresa.getNumProcesso())
        .append("','")
        .append(empresa.getAgenteIntegracao())
        .append("')");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirEmpresa(Empresa empresa) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
      UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
      VagaDao vagaDao = new VagaDao();
      ConvenioDao convenioDao = new ConvenioDao();
      TermoCompromissoDao termoCompromissoDao = new
TermoCompromissoDao();
      try {
        List<AlunoEstagio> usuariosEmpresa = new
ArrayList<AlunoEstagio>();
        usuariosEmpresa =
usuarioEmpresaDao.listarUsuarioEmpresas(empresa.getCnpj());

usuarioEmpresaDao.excluirUsuarioEmpresa(empresa.getCnpj());
        for(AlunoEstagio usuario: usuariosEmpresa ){
          alunoEstagioDao.excluirAlunoEstagio(usuario);
        }

termoCompromissoDao.excluirTermosCompromisso(empresa.get
Cnpj());
        vagaDao.excluirVagas(empresa.getCnpj());
        convenioDao.excluirConvenio(empresa.getCnpj());
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from Empresa where cnpj=");
        sb.append("'");
        sb.append(empresa.getCnpj());
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();

enderecoEmpresaDao.excluirEndereco(empresa.getEnderecoEmp
resa());
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void alterarEmpresa(Empresa empresa) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
      try {

enderecoEmpresaDao.alterarEndereco(empresa.getEnderecoEmpr
esa());
        StringBuffer sb = new StringBuffer();
        sb.append("update Empresa set
razaoSocial='").append(empresa.getRazaoSocial()).append("',
ramo='")
        .append(empresa.getRamo()).append("',
site='").append(empresa.getSite()).append("', email='")
        .append(empresa.getEmail()).append("',
nomefantasia='").append(empresa.getNomeFantasia())
        .append("', fone='").append(empresa.getFone()).append("',
representante='").append(empresa.getRepresentante())
        .append("', fax='").append(empresa.getFax())
        .append("', cargoRepresentante
='").append(empresa.getCargoRepresentante())
        .append("', numprocesso
='").append(empresa.getNumProcesso())
        .append("',
agente_integracao='").append(empresa.getAgenteIntegracao())
        .append("' where cnpj
='").append(empresa.getCnpj()).append("'");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } catch (Exception e){
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public List<Empresa> pesquisarEmpresa(String
parteDaRazaoSocial) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Empresa> empresas = new ArrayList<Empresa>();
      try {
        EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
```

```
        sb.append("select cnpj,
enderecoEmpresa_codEnderecoEmpresa,")
        .append(" razaoSocial, ramo, site, email, nomeFantasia,
fone, fax, representante,")
        .append(" cargoRepresentante, numprocesso,
agente_integracao from Empresa where upper(razaoSocial) like
upper('%")
        .append(parteDaRazaoSocial)
        .append("%')");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){
          Empresa empresa = new Empresa();
          Endereco enderecoEmpresa = new Endereco();
          enderecoEmpresa =
enderecoEmpresaDao.buscarEnderecoEmpresa(rs.getInt("enderec
oEmpresa_codEnderecoEmpresa"));
          empresa.setCnpj(rs.getString("cnpj"));
          empresa.setRazaoSocial(rs.getString("razaosocial"));
          empresa.setRamo(rs.getString("ramo"));
          empresa.setSite(rs.getString("site"));
          empresa.setEmail(rs.getString("email"));

empresa.setNomeFantasia(rs.getString("nomefantasia"));
          empresa.setFone(rs.getString("fone"));

empresa.setRepresentante(rs.getString("representante"));

empresa.setCargoRepresentante(rs.getString("cargorepresentante"
));
          empresa.setEnderecoEmpresa(enderecoEmpresa);
          empresa.setFax(rs.getString("fax"));
          empresa.setNumProcesso(rs.getString("numprocesso"));

empresa.setAgenteIntegracao(rs.getString("agente_integracao"));
          empresas.add(empresa);
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return empresas;
    }

    public Empresa buscarEmpresaSemConvenio(String cnpj)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Empresa empresa = new Empresa();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select cnpj from empresa where cnpj =
'").append(cnpj)
        .append("' except (select empresa_cnpj from convenio)");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          Endereco enderecoEmpresa = new Endereco();
          empresa = buscarEmpresa(rs.getString("cnpj"));
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return empresa;
    }

    public Empresa buscarEmpresaConvenioVencido(String cnpj)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Empresa empresa = new Empresa();
      ConvenioDao convenioDao = new ConvenioDao();
      try {
        EnderecoEmpresaDao enderecoEmpresaDao = new
EnderecoEmpresaDao();
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select cnpj,
enderecoEmpresa_codEnderecoEmpresa,")
        .append(" razaoSocial, ramo, site, email, nomeFantasia,
fone, fax, representante,")
        .append(" cargoRepresentante, numprocesso,
agente_integracao from empresa where cnpj
='").append(cnpj).append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          Endereco enderecoEmpresa = new Endereco();
          Convenio convenio =
convenioDao.buscarConvenio(rs.getString("cnpj"));
          GregorianCalendar data = new GregorianCalendar();
          if(data.after(convenio.getDataFim())){
            enderecoEmpresa =
enderecoEmpresaDao.buscarEnderecoEmpresa(rs.getInt("enderec
oEmpresa_codEnderecoEmpresa"));
            empresa.setCnpj(rs.getString("cnpj"));
            empresa.setRazaoSocial(rs.getString("razaosocial"));
            empresa.setRamo(rs.getString("ramo"));
            empresa.setSite(rs.getString("site"));
            empresa.setEmail(rs.getString("email"));

empresa.setNomeFantasia(rs.getString("nomefantasia"));
            empresa.setFone(rs.getString("fone"));

empresa.setRepresentante(rs.getString("representante"));

empresa.setCargoRepresentante(rs.getString("cargorepresentante"
));
            empresa.setEnderecoEmpresa(enderecoEmpresa);
            empresa.setFax(rs.getString("fax"));

empresa.setNumProcesso(rs.getString("numprocesso"));

empresa.setAgenteIntegracao(rs.getString("agente_integracao"));
          } else {
            return null;
          }
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return empresa;
    }

    public List<AgenteIntegracao> listarAgentesIntegracao()
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<AgenteIntegracao> agentes = new
ArrayList<AgenteIntegracao>();
      try {
```

```
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select cnpj, razaoSocial from empresa where
agente_integracao='S' order by razaoSocial");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
              AgenteIntegracao agenteIntegracao = new
AgenteIntegracao();
              agenteIntegracao.setCnpj(rs.getString("cnpj"));

agenteIntegracao.setAgente(rs.getString("razaosocial"));
              agentes.add(agenteIntegracao);
            }
          } catch (SQLException e) {
            throw e;
          } finally {
            closeConnection(conn, stmt, rs);
          }
          return agentes;
        }

        public AgenteIntegracao buscarAgenteIntegracao(String cnpj)
throws SQLException, Exception {
          Connection conn = null;
          PreparedStatement stmt = null;
          ResultSet rs = null;
          AgenteIntegracao agenteIntegracao = new
AgenteIntegracao();
          try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select cnpj, razaoSocial from empresa where
agente_integracao='S' and cnpj='")
            .append(cnpj)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()) {

agenteIntegracao.setAgente(rs.getString("razaoSocial"));
              agenteIntegracao.setCnpj(rs.getString("cnpj"));
            } else {
              return null;
            }
          } catch (SQLException e) {
            throw e;
          } finally {
            closeConnection(conn, stmt, rs);
          }
          return agenteIntegracao;
        }

        public void excluirAgenteIntegracao(String cnpj) throws
SQLException, Exception {
          Connection conn = null;
          PreparedStatement stmt = null;
          ResultSet rs = null;
          try {
            StringBuffer sb = new StringBuffer();
            sb.append("update Empresa set agente_integracao='N'
where cnpj ='").append(cnpj).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
          } catch (SQLException e) {
            throw e;
          } catch (Exception e){
            throw e;
          } finally {
            if (rs != null) {
              rs.close();
            }
            if (stmt != null) {
              stmt.close();
            }
            closeConnection(conn, stmt, rs);
          }
        }

}
```

6.4.277 EnderecoEmpresaDao.java

```
/*
 * EnderecoEmpresaDao.java
 * Created on 3 de Novembro de 2005, 23:35
 */

package estagio.dao;
import estagio.bean.Endereco;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 *
 */
public class EnderecoEmpresaDao extends BaseDao{

    /** Creates a new instance of EnderecoEnderecoEmpresaDao
*/
    public EnderecoEmpresaDao() throws Exception{
    }

    public Endereco buscarEnderecoEmpresa(Integer
codEnderecoEmpresa) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Endereco enderecoEmpresa= new Endereco();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codenderecoEmpresa, cep, rua, numero,
complemento, bairro, municipio, estado, pais, cnpj from
EnderecoEmpresa where codEnderecoEmpresa='");
          sb.append(codEnderecoEmpresa);
          sb.append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if(rs.next()){

enderecoEmpresa.setCodEndereco(Integer.valueOf(rs.getInt("cod
enderecoempresa")));
            enderecoEmpresa.setOwner(rs.getString("cnpj"));
            enderecoEmpresa.setCep(rs.getString("cep"));
            enderecoEmpresa.setRua(rs.getString("rua"));
            enderecoEmpresa.setNumero(rs.getInt("numero"));

enderecoEmpresa.setComplemento(rs.getString("complemento"))
;
            enderecoEmpresa.setBairro(rs.getString("bairro"));

enderecoEmpresa.setMunicipio(rs.getString("municipio"));
            enderecoEmpresa.setEstado(rs.getString("estado"));
            enderecoEmpresa.setPais(rs.getString("pais"));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
```

```
    return enderecoEmpresa;
  }

  public void incluirEnderecoEmpresa(Endereco endereco)
throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("insert into EnderecoEmpresa (cep, rua,
numero, complemento, bairro, municipio, estado, pais, cnpj)");
      sb.append("  values ('");
      sb.append(endereco.getCep());
      sb.append("','");
      sb.append(endereco.getRua());
      sb.append("','");
      sb.append(endereco.getNumero());
      sb.append("','");
      sb.append(endereco.getComplemento());
      sb.append("','");
      sb.append(endereco.getBairro());
      sb.append("','");
      sb.append(endereco.getMunicipio());
      sb.append("','");
      sb.append(endereco.getEstado());
      sb.append("','");
      sb.append(endereco.getPais());
      sb.append("','");
      sb.append(endereco.getOwner());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirEndereco(Endereco endereco) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from EnderecoEmpresa where cnpj=");
      sb.append("'");
      sb.append(endereco.getOwner());
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void alterarEndereco(Endereco endereco) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      StringBuffer sb = new StringBuffer();
      sb.append("update EnderecoEmpresa set cep
='").append(endereco.getCep())
      .append("', rua='").append(endereco.getRua()).append("',
numero='")
      .append(endereco.getNumero()).append("',
complemento='").append(endereco.getComplemento())
      .append("',
bairro='").append(endereco.getBairro()).append("',
municipio='").append(endereco.getMunicipio())
      .append("',
estado='").append(endereco.getEstado()).append("',
pais='").append(endereco.getPais())
      .append("' where cnpj
='").append(endereco.getOwner()).append("'");

      conn = getConnection();
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } catch (Exception e){
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public Endereco buscarEnderecoEmpresa(String cnpj) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    Endereco enderecoEmpresa= new Endereco();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select codenderecoempresa, cep, rua, numero,
complemento, bairro, municipio, estado, pais, cnpj from
EnderecoEmpresa where cnpj='");
      sb.append(cnpj);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if(rs.next()){

enderecoEmpresa.setCodEndereco(Integer.valueOf(rs.getInt("cod
enderecoempresa")));
        enderecoEmpresa.setOwner(rs.getString("cnpj"));
        enderecoEmpresa.setCep(rs.getString("cep"));
        enderecoEmpresa.setRua(rs.getString("rua"));
        enderecoEmpresa.setNumero(rs.getInt("numero"));

enderecoEmpresa.setComplemento(rs.getString("complemento"))
;
        enderecoEmpresa.setBairro(rs.getString("bairro"));

enderecoEmpresa.setMunicipio(rs.getString("municipio"));
```

```
            enderecoEmpresa.setEstado(rs.getString("estado"));
            enderecoEmpresa.setPais(rs.getString("pais"));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return enderecoEmpresa;
    }

}
```

6.4.278 EnderecoPessoaDao.java

```
/*
 * EnderecoPessoaDao.java
 * Created on 6 de Novembro de 2005, 20:48
 */

package estagio.dao;

import estagio.bean.Endereco;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

/**
 *
 *
 */
public class EnderecoPessoaDao extends BaseDao{

    /** Creates a new instance of EnderecoEnderecoPessoaDao */
    public EnderecoPessoaDao() throws Exception{
    }

    public Endereco buscarEnderecoPessoa(String cpf) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Endereco enderecoPessoa= new Endereco();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select cep, rua, numero, complemento, bairro,
municipio, estado, pais, cpf from EnderecoPessoa where cpf='");
          sb.append(cpf);
          sb.append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if(rs.next()){
            enderecoPessoa.setCep(rs.getString("cep"));
            enderecoPessoa.setRua(rs.getString("rua"));
            enderecoPessoa.setNumero(rs.getInt("numero"));

enderecoPessoa.setComplemento(rs.getString("complemento"));
            enderecoPessoa.setBairro(rs.getString("bairro"));

enderecoPessoa.setMunicipio(rs.getString("municipio"));
            enderecoPessoa.setEstado(rs.getString("estado"));
            enderecoPessoa.setPais(rs.getString("pais"));
            enderecoPessoa.setOwner(rs.getString("cpf"));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return enderecoPessoa;
    }

    public void incluirEnderecoPessoa(Endereco endereco) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("insert into EnderecoPessoa (cep, rua, numero,
complemento, bairro, municipio, estado, pais, cpf)");
          sb.append("  values ('");
          sb.append(endereco.getCep());
          sb.append("','");
          sb.append(endereco.getRua());
          sb.append("','");
          sb.append(endereco.getNumero());
          sb.append("','");
          sb.append(endereco.getComplemento());
          sb.append("','");
          sb.append(endereco.getBairro());
          sb.append("','");
          sb.append(endereco.getMunicipio());
          sb.append("','");
          sb.append(endereco.getEstado());
          sb.append("','");
          sb.append(endereco.getPais());
          sb.append("','");
          sb.append(endereco.getOwner());
          sb.append("')");
          stmt = conn.prepareStatement(sb.toString());
          stmt.executeUpdate();
        } catch (SQLException e) {
          throw e;
        } finally {
          if (rs != null) {
            rs.close();
          }
          if (stmt != null) {
            stmt.close();
          }
          closeConnection(conn, stmt, rs);
        }
    }

    public void excluirEndereco(Endereco endereco) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("delete from EnderecoPessoa where cpf=");
          sb.append("'");
          sb.append(endereco.getOwner());
          sb.append("'");
          stmt = conn.prepareStatement(sb.toString());
          stmt.executeUpdate();
        } catch (SQLException e) {
          throw e;
        } finally {
          if (rs != null) {
            rs.close();
          }
          if (stmt != null) {
            stmt.close();
          }
          closeConnection(conn, stmt, rs);
        }
    }
```

```
    public void alterarEndereco(Endereco endereco) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update EnderecoPessoa set cep
=").append(endereco.getCep())
            .append(", rua=").append(endereco.getRua()).append(",
numero=")
            .append(endereco.getNumero()).append(",
complemento=").append(endereco.getComplemento())
            .append(",
bairro=").append(endereco.getBairro()).append(",
municipio=").append(endereco.getMunicipio())
            .append(",
estado=").append(endereco.getEstado()).append(",
pais=").append(endereco.getPais())
            .append(" where cpf
=").append(endereco.getOwner()).append("");

            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }
}
```

6.4.279 HabilidadesAlunoDao.java

```
/*
 * HabilidadeAlunosAlunoDao.java
 * Created on 28 de Janeiro de 2006, 11:49
 */

package estagio.dao;

import estagio.bean.AlunoEstagio;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeAluno;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class HabilidadesAlunoDao extends BaseDao{

    /**
     * Creates a new instance of HabilidadeAlunoDao
     */

    private HabilidadesDao habilidadesDao;

    public HabilidadesAlunoDao() throws Exception{
        habilidadesDao = new HabilidadesDao();
    }


    public void incluirHabilidadeAluno(HabilidadeAluno
habilidadeAluno) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into habilidadesaluno
(alunoestagio_codalunoestagio, habilidades_codhabilidade)");
            sb.append("  values ('");
            sb.append(habilidadeAluno.getCodAlunoEstagio());
            sb.append("','");
            sb.append(habilidadeAluno.getCodHabilidade());
            sb.append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirHabilidadeAluno(HabilidadeAluno
habilidadeAluno) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from habilidadesAluno where
codhabilidadealuno=");
            sb.append("'");
            sb.append(habilidadeAluno.getCodHabilidadeAluno());
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirHabilidadesAluno(Integer codAlunoEstagio)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from habilidadesAluno where
alunoestagio_codalunoestagio=");
```

```
            sb.append("");
            sb.append(codAlunoEstagio);
            sb.append("");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public List<HabilidadeAluno> listarHabilidadeAluno(String
usuario) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<HabilidadeAluno> habilidades = new
ArrayList<HabilidadeAluno>();
        AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
        try {
            AlunoEstagio aluno =
alunoEstagioDao.buscarAlunoEstagio(usuario);
            if (aluno != null){
                conn = getConnection();
                StringBuffer sb = new StringBuffer();
                sb.append("select codhabilidadealuno,
alunoestagio_codalunoestagio, habilidades_codhabilidade from
habilidadesaluno where alunoestagio_codalunoestagio='")
                .append(aluno.getCodAlunoEstagio())
                .append("");
                stmt = conn.prepareStatement(sb.toString());
                rs = stmt.executeQuery();
                while(rs.next()){
                    HabilidadeAluno habilidadeAluno = new
HabilidadeAluno();
                    Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));

habilidadeAluno.setCodAlunoEstagio(aluno.getCodAlunoEstagio
());

habilidadeAluno.setCodHabilidadeAluno(Integer.valueOf(rs.getIn
t("codhabilidadealuno")));
                    if(habilidade != null){

habilidadeAluno.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeAluno.setCodHabilidade(habilidade.getCodHabilidade(
));
                    } else {
                        habilidadeAluno.setDescricaoHabilidade("");

habilidadeAluno.setCodHabilidade(Integer.valueOf(0));
                    }
                    habilidades.add(habilidadeAluno);
                }
            }
        } catch (SQLException e) {
            throw e;
        } catch (Exception e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return habilidades;
    }

    public List<HabilidadeAluno> listarHabilidadeAluno(Integer
codAlunoEstagio) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<HabilidadeAluno> habilidades = new
ArrayList<HabilidadeAluno>();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codhabilidadealuno,
alunoestagio_codalunoestagio, habilidades_codhabilidade from
habilidadesaluno where alunoestagio_codalunoestagio='")
            .append(codAlunoEstagio)
            .append("");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
                HabilidadeAluno habilidadeAluno = new
HabilidadeAluno();
                Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));

habilidadeAluno.setCodAlunoEstagio(codAlunoEstagio);

habilidadeAluno.setCodHabilidadeAluno(Integer.valueOf(rs.getIn
t("codhabilidadealuno")));
                if(habilidade != null){

habilidadeAluno.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeAluno.setCodHabilidade(habilidade.getCodHabilidade(
));
                } else {
                    habilidadeAluno.setDescricaoHabilidade("");

habilidadeAluno.setCodHabilidade(Integer.valueOf(0));
                }
                habilidades.add(habilidadeAluno);
            }
        } catch (SQLException e) {
            throw e;
        } catch (Exception e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return habilidades;
    }

    public HabilidadeAluno buscarHabilidadeAluno(Integer
codHabilidadeAluno) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        HabilidadeAluno habilidadeAluno = new HabilidadeAluno();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codhabilidadealuno,
alunoestagio_codalunoestagio, habilidades_codhabilidade from
habilidadesaluno where codhabilidadealuno='")
            .append(codHabilidadeAluno)
            .append("");
            stmt = conn.prepareStatement(sb.toString());
```

```
        rs = stmt.executeQuery();
        if(rs.next()){
          Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));

habilidadeAluno.setCodAlunoEstagio(Integer.valueOf(rs.getInt("
alunoestagio_codalunoestagio")));

habilidadeAluno.setCodHabilidadeAluno(codHabilidadeAluno);
          if(habilidade != null){

habilidadeAluno.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeAluno.setCodHabilidade(habilidade.getCodHabilidade(
));
          } else {
            habilidadeAluno.setDescricaoHabilidade("");

habilidadeAluno.setCodHabilidade(Integer.valueOf(0));
          }
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return habilidadeAluno;
    }

    public List<HabilidadeAluno> listarHabilidadesAluno(Integer
codHabilidade) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<HabilidadeAluno> habilidades = new
ArrayList<HabilidadeAluno>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codhabilidadealuno,
alunoestagio_codalunoestagio, habilidades_codhabilidade from
habilidadesaluno where habilidades_codhabilidade='")
        .append(codHabilidade)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){
          HabilidadeAluno habilidadeAluno = new
HabilidadeAluno();
          Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));

habilidadeAluno.setCodAlunoEstagio(Integer.valueOf(rs.getInt("
alunoestagio_codalunoestagio")));

habilidadeAluno.setCodHabilidadeAluno(Integer.valueOf(rs.getIn
t("codhabilidadealuno")));
          if(habilidade != null){

habilidadeAluno.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeAluno.setCodHabilidade(habilidade.getCodHabilidade(
));
          } else {
            habilidadeAluno.setDescricaoHabilidade("");

habilidadeAluno.setCodHabilidade(Integer.valueOf(0));
          }
          habilidades.add(habilidadeAluno);
        }
      } catch (SQLException e) {
        throw e;
      } catch (Exception e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return habilidades;
    }

}
```

6.4.280 HabilidadesDao.java

```
/*
 * HabilidadesDao.java
 * Created on 29 de Janeiro de 2006, 16:50
 */

package estagio.dao;

import estagio.bean.Habilidade;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class HabilidadesDao extends BaseDao{

  /**
   * Creates a new instance of HabilidadesDao
   */
  public HabilidadesDao() throws Exception{
  }


  public void incluirHabilidade(Habilidade habilidade) throws
SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("insert into habilidades (nomehabilidade)");
      sb.append(" values ('");
      sb.append(habilidade.getDescricaoHabilidade());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
```

```
    }

    public void excluirHabilidade(Integer codHabilidade) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from habilidades where
codhabilidade=");
            sb.append("'");
            sb.append(codHabilidade);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarHabilidade(Habilidade habilidade) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update habilidades set
nomehabilidade='").append(habilidade.getDescricaoHabilidade())
            .append("' where codhabilidade
='").append(habilidade.getCodHabilidade()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public List<Habilidade> listarHabilidades() throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Habilidade> habilidades = new
ArrayList<Habilidade>();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            String query = "select codhabilidade, nomehabilidade
from habilidades order by nomehabilidade";
            stmt = conn.prepareStatement(query);
            rs = stmt.executeQuery();
            while(rs.next()){
                Habilidade habilidade = new Habilidade();

habilidade.setCodHabilidade(Integer.valueOf(rs.getInt("codhabili
dade")));

habilidade.setDescricaoHabilidade(rs.getString("nomehabilidade"
));
                habilidades.add(habilidade);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return habilidades;
    }

    public Habilidade buscarHabilidade(Integer codHabilidade)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Habilidade habilidade = new Habilidade();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codhabilidade, nomehabilidade from
habilidades where codhabilidade='")
            .append(codHabilidade)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if(rs.next()){

habilidade.setCodHabilidade(Integer.valueOf(rs.getInt("codhabili
dade")));

habilidade.setDescricaoHabilidade(rs.getString("nomehabilidade"
));
            } else {
                return null;
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return habilidade;
    }

    public List<Habilidade> buscarHabilidade(String
parteDoNome) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Habilidade> habilidades = new
ArrayList<Habilidade>();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codhabilidade, nomehabilidade from
habilidades where nomehabilidade like '%")
            .append(parteDoNome)
            .append("%'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
                Habilidade habilidade = new Habilidade();
```

```
habilidade.setCodHabilidade(Integer.valueOf(rs.getInt("codhabili
dade")));

habilidade.setDescricaoHabilidade(rs.getString("nomehabilidade"
));
            habilidades.add(habilidade);
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return habilidades;
    }

}
```

6.4.281 HabilidadesVagaDao.java

```
/*
 * HabilidadesVagaDao.java
 * Created on 29 de Janeiro de 2006, 18:35
 */

package estagio.dao;

import estagio.bean.AlunoEstagio;
import estagio.bean.Habilidade;
import estagio.bean.HabilidadeVaga;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class HabilidadesVagaDao extends BaseDao{

    /**
     * Creates a new instance of HabilidadesVagaDao
     */

    private HabilidadesDao habilidadesDao;

    public HabilidadesVagaDao() throws Exception{
        habilidadesDao = new HabilidadesDao();
    }


    public void incluirHabilidadeVaga(HabilidadeVaga
habilidadeVaga) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into habilidadesvaga (vagas_codvaga,
habilidades_codhabilidade)");
            sb.append("  values ('");
            sb.append(habilidadeVaga.getCodVaga());
            sb.append("','");
            sb.append(habilidadeVaga.getCodHabilidade());
            sb.append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirHabilidadeVaga(HabilidadeVaga
habilidadeVaga) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from habilidadesvaga where
codhabilidadevaga='");
            sb.append("'");
            sb.append(habilidadeVaga.getCodHabilidadeVaga());
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirHabilidadesVaga(Integer codVaga) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from habilidadesvaga where
vagas_codvaga='");
            sb.append("'");
            sb.append(codVaga);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public List<HabilidadeVaga> listarHabilidadeVaga(Integer
codVaga) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
```

```
    List<HabilidadeVaga> habilidades = new
ArrayList<HabilidadeVaga>();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select codhabilidadevaga, vagas_codvaga,
habilidades_codhabilidade from habilidadesvaga where
vagas_codvaga='")
      .append(codVaga)
      .append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      while(rs.next()){
        HabilidadeVaga habilidadeVaga = new
HabilidadeVaga();
        Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));
        habilidadeVaga.setCodVaga(codVaga);

habilidadeVaga.setCodHabilidadeVaga(Integer.valueOf(rs.getInt(
"codhabilidadevaga")));
        if(habilidade != null){

habilidadeVaga.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeVaga.setCodHabilidade(habilidade.getCodHabilidade()
);
        } else {
          habilidadeVaga.setDescricaoHabilidade("");

habilidadeVaga.setCodHabilidade(Integer.valueOf(0));
        }
        habilidades.add(habilidadeVaga);
      }
    } catch (SQLException e) {
      throw e;
    } catch (Exception e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return habilidades;
  }

  public HabilidadeVaga buscarHabilidadeVaga(Integer
codHabilidadeVaga) throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    HabilidadeVaga habilidadeVaga = new HabilidadeVaga();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select codhabilidadevaga, vagas_codvaga,
habilidades_codhabilidade from habilidadesvaga where
codhabilidadevaga='")
      .append(codHabilidadeVaga)
      .append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      if(rs.next()){
        Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));

habilidadeVaga.setCodVaga(Integer.valueOf(rs.getInt("vagas_co
dvaga")));

habilidadeVaga.setCodHabilidadeVaga(codHabilidadeVaga);
        if(habilidade != null){

habilidadeVaga.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeVaga.setCodHabilidade(habilidade.getCodHabilidade()
);
        } else {
          habilidadeVaga.setDescricaoHabilidade("");

habilidadeVaga.setCodHabilidade(Integer.valueOf(0));
        }
      } else {
        return null;
      }
    } catch (SQLException e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
    return habilidadeVaga;
  }

  public List<HabilidadeVaga> listarHabilidadesVaga(Integer
codHabilidade) throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    List<HabilidadeVaga> habilidades = new
ArrayList<HabilidadeVaga>();
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("select codhabilidadevaga, vagas_codvaga,
habilidades_codhabilidade from habilidadesvaga where
habilidades_codhabilidade='")
      .append(codHabilidade)
      .append("'");
      stmt = conn.prepareStatement(sb.toString());
      rs = stmt.executeQuery();
      while(rs.next()){
        HabilidadeVaga habilidadeVaga = new
HabilidadeVaga();
        Habilidade habilidade =
habilidadesDao.buscarHabilidade(Integer.valueOf(rs.getInt("habil
idades_codhabilidade")));

habilidadeVaga.setCodVaga(Integer.valueOf(rs.getInt("vagas_co
dvaga")));

habilidadeVaga.setCodHabilidadeVaga(Integer.valueOf(rs.getInt(
"codhabilidadevaga")));
        if(habilidade != null){

habilidadeVaga.setDescricaoHabilidade(habilidade.getDescricao
Habilidade());

habilidadeVaga.setCodHabilidade(habilidade.getCodHabilidade()
);
        } else {
          habilidadeVaga.setDescricaoHabilidade("");

habilidadeVaga.setCodHabilidade(Integer.valueOf(0));
        }
        habilidades.add(habilidadeVaga);
      }
    } catch (SQLException e) {
      throw e;
    } catch (Exception e) {
      throw e;
    } finally {
      closeConnection(conn, stmt, rs);
    }
```

```
        return habilidades;
    }
}
```

6.4.282 PessoaDao.java

```
/*
 * PessoaDao.java
 * Created on 26 de Janeiro de 2006, 17:13
 */

package estagio.dao;

import estagio.bean.Pessoa;
import estagio.bean.Pessoa;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 * @author Ricardo
 */
public class PessoaDao extends BaseDao{

    /** Creates a new instance of PessoaDao */
    public PessoaDao() throws Exception{
    }


    public void incluirPessoa(Pessoa pessoa) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
            String dataNasc =
sdf.format(pessoa.getDataNascimento().getTime());
            sb.append("insert into Pessoa (nome_pessoa, nasc_pessoa,
usuario, senha)");
            sb.append("  values ('");
            sb.append(pessoa.getNomePessoa());
            sb.append("','");
            sb.append(dataNasc);
            sb.append("','");
            sb.append(pessoa.getUsuario());
            sb.append("','");
            sb.append(pessoa.getSenha());
            sb.append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirPessoa(String usuario) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from pessoa where usuario=");
            sb.append("'");
            sb.append(usuario);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarPessoa(Pessoa pessoa) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
            String dataNasc =
sdf.format(pessoa.getDataNascimento().getTime());
            StringBuffer sb = new StringBuffer();
            sb.append("update pessoa set
nome_pessoa='").append(pessoa.getNomePessoa())
            .append("', nasc_pessoa='").append(dataNasc)
            .append("', senha='").append(pessoa.getSenha())
            .append("' where usuario
='").append(pessoa.getUsuario()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public Pessoa buscarPessoa(String usuario) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Pessoa pessoa = new Pessoa();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
```

```
        sb.append("select cod_pessoa, nome_pessoa, nasc_pessoa,
senha, usuario from pessoa where upper(usuario) = upper('")
        .append(usuario)
        .append("')");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){

pessoa.setCodPessoa(Integer.valueOf(rs.getInt("cod_pessoa")));
          GregorianCalendar dataNascimento = new
GregorianCalendar();
          dataNascimento.setTime(rs.getDate("nasc_pessoa"));
          pessoa.setDataNascimento(dataNascimento);
          pessoa.setNomePessoa(rs.getString("nome_pessoa"));
          pessoa.setSenha(rs.getString("senha"));
          pessoa.setUsuario(rs.getString("usuario"));
          pessoa.setStatus("");
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return pessoa;
    }

    public Pessoa buscarNovaPessoa(String usuario) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Pessoa pessoa = new Pessoa();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select max(cod_pessoa) as codpessoa from
pessoa where upper(usuario) = upper('")
        .append(usuario)
        .append("')");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          pessoa =
buscarPessoa(Integer.valueOf(rs.getInt("codpessoa")));
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return pessoa;
    }

    public Pessoa buscarPessoa(Integer codPessoa) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Pessoa pessoa = new Pessoa();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select nome_pessoa, nasc_pessoa, senha,
usuario from pessoa where cod_pessoa='")
        .append(codPessoa)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          pessoa.setCodPessoa(codPessoa);
          GregorianCalendar dataNascimento = new
GregorianCalendar();
          dataNascimento.setTime(rs.getDate("nasc_pessoa"));
          pessoa.setDataNascimento(dataNascimento);
          pessoa.setNomePessoa(rs.getString("nome_pessoa"));
          pessoa.setSenha(rs.getString("senha"));
          pessoa.setUsuario(rs.getString("usuario"));
          pessoa.setStatus("");
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return pessoa;
    }

    public List<Pessoa> buscarEstagiario(String parteNome)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Pessoa> estagiarios = new ArrayList<Pessoa>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select cod_pessoa, nome_pessoa, nasc_pessoa,
senha, usuario from pessoa where upper(nome_pessoa) like
upper('%")
        .append(parteNome)
        .append("%') order by nome_pessoa");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){
          Pessoa pessoa = new Pessoa();

pessoa.setCodPessoa(Integer.valueOf(rs.getInt("cod_pessoa")));
          GregorianCalendar dataNascimento = new
GregorianCalendar();
          dataNascimento.setTime(rs.getDate("nasc_pessoa"));
          pessoa.setDataNascimento(dataNascimento);
          pessoa.setNomePessoa(rs.getString("nome_pessoa"));
          pessoa.setSenha(rs.getString("senha"));
          pessoa.setUsuario(rs.getString("usuario"));
          pessoa.setStatus("");
          estagiarios.add(pessoa);
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return estagiarios;
    }
}
```

6.4.283 PlanosEstagioDao.java

```
/*
 * PlanosEstagioDao.java
 * Created on 25 de Abril de 2006, 09:15
 */

package estagio.dao;

import estagio.bean.PlanoEstagio;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class PlanosEstagioDao extends BaseDao{


   /** Creates a new instance of PlanosEstagioDao */
   public PlanosEstagioDao() throws Exception{

   }

   public List<PlanoEstagio> listarPlanosEstagio() throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      ArrayList<PlanoEstagio> planosEstagio  = new
ArrayList<PlanoEstagio>();
      try {
         conn = getConnection();
         stmt = conn.prepareStatement("select codplanoestagio
from planos_estagio");
         rs = stmt.executeQuery();
         while (rs.next()) {

planosEstagio.add(buscarPlanoEstagio(Integer.valueOf(rs.getInt("
codplanoestagio"))));
         }
      } catch (SQLException e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
      return planosEstagio;
   }

   public PlanoEstagio buscarPlanoEstagio(Integer
codPlanoEstagio) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      PlanoEstagio planoEstagio = new PlanoEstagio();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select codplanoestagio, orientador,
formacao_orientador, area_atividade_estagiario, objetivos,
atividades, codtermo, cargo_orientador, ramal, departamento from
planos_estagio where codplanoestagio ='")
         .append(codPlanoEstagio)
         .append("'");
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         if (rs.next()) {
            planoEstagio.setCodPlanoEstagio(codPlanoEstagio);

planoEstagio.setCodTermo(Integer.valueOf(rs.getInt("codtermo")
));

planoEstagio.setAreaAtividadeEstagiario(rs.getString("area_ativi
dade_estagiario"));
            planoEstagio.setAtividades(rs.getString("atividades"));

planoEstagio.setFormacaoOrientador(rs.getString("formacao_orie
ntador"));
            planoEstagio.setObjetivos(rs.getString("objetivos"));
            planoEstagio.setOrientador(rs.getString("orientador"));
planoEstagio.setCargoOrientador(rs.getString("cargo_orientador"
));

planoEstagio.setDepartamento(rs.getString("departamento"));
               planoEstagio.setRamal(rs.getString("ramal"));
            } else {
               return null;
            }
         } catch (SQLException e) {
            throw e;
         } finally {
            closeConnection(conn, stmt, rs);
         }
         return planoEstagio;
      }

      public PlanoEstagio buscarPlanoEstagio(int codTermo) throws
SQLException, Exception {
         Connection conn = null;
         PreparedStatement stmt = null;
         ResultSet rs = null;
         try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codplanoestagio from planos_estagio
where codtermo = '")
            .append(codTermo)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if (rs.next()) {
               return
buscarPlanoEstagio(Integer.valueOf(rs.getInt("codplanoestagio"))
);
            } else {
               return null;
            }
         } catch (SQLException e) {
            throw e;
         } finally {
            closeConnection(conn, stmt, rs);
         }
      }

      public void incluirPlanoEstagio(PlanoEstagio planoEstagio)
throws SQLException, Exception {
         Connection conn = null;
         PreparedStatement stmt = null;
         ResultSet rs = null;
         try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into planos_estagio (orientador,
formacao_orientador, area_atividade_estagiario, objetivos,
atividades, codtermo, cargo_orientador, ramal, departamento)")
            .append("  values ('")
            .append(planoEstagio.getOrientador())
            .append("','")
            .append(planoEstagio.getFormacaoOrientador())
            .append("','")
            .append(planoEstagio.getAreaAtividadeEstagiario())
            .append("','")
            .append(planoEstagio.getObjetivos())
            .append("','")
            .append(planoEstagio.getAtividades())
            .append("','")
            .append(planoEstagio.getCodTermo())
            .append("','")
            .append(planoEstagio.getCargoOrientador() !=
null?planoEstagio.getCargoOrientador():"")
            .append("','")
```

```
        .append(planoEstagio.getRamal() !=
null?planoEstagio.getRamal():"")
        .append("','")
        .append(planoEstagio.getDepartamento() !=
null?planoEstagio.getDepartamento():"")
        .append("')");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirPlanoEstagio(Integer codPlanoEstagio)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from planos_estagio where
codplanoestagio = '")
        .append(codPlanoEstagio)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirPlanosEstagio(int codTermo) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from planos_estagio where codtermo =
'")
        .append(codTermo)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void alterarPlanoEstagio(PlanoEstagio planoEstagio)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        StringBuffer sb = new StringBuffer();
        sb.append("update planos_estagio set
orientador='").append(planoEstagio.getOrientador())
        .append("',
formacao_orientador='").append(planoEstagio.getFormacaoOrient
ador())
        .append("', area_atividade_estagiario =
'").append(planoEstagio.getAreaAtividadeEstagiario())
        .append("', objetivos =
'").append(planoEstagio.getObjetivos())
        .append("', atividades =
'").append(planoEstagio.getAtividades())
        .append("', cargo_orientador =
'").append(planoEstagio.getCargoOrientador())
        .append("', ramal = '").append(planoEstagio.getRamal())
        .append("', departamento =
'").append(planoEstagio.getDepartamento())
        .append("' where codplanoestagio =
'").append(planoEstagio.getCodPlanoEstagio()).append("'");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } catch (Exception e){
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }
}
```

6.4.284 RelatorioSeguroDao.java

```
/*
 * RelatorioSeguroDao.java
 * Created on 30 de Abril de 2006, 11:28
 */

package estagio.dao;

import estagio.bean.RelatorioSeguroDTO;
import java.math.BigDecimal;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.GregorianCalendar;
import java.util.List;


/**
 *
 *
 */
```

```
public class RelatorioSeguroDao extends BaseDao{

   /** Creates a new instance of RelatorioSeguroDao */
   public RelatorioSeguroDao() throws Exception{
   }

   public List<RelatorioSeguroDTO>
queryRelatorioCancelar(String cnpj, Integer mes, Integer ano,
boolean estagioInterno) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd HH:mm");
      List<RelatorioSeguroDTO> lista = new
ArrayList<RelatorioSeguroDTO>();
      GregorianCalendar data = new GregorianCalendar();
      data.set(Calendar.YEAR, ano.intValue());
      data.set(Calendar.MONTH, mes.intValue() - 1);
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         if(!estagioInterno){
            sb.append("SELECT DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim, t.empresa_cnpj,
varchar 'CANCELAR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t WHERE p.cod_pessoa =
a.pessoa_codpessoa ")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datafim) =
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND t.empresa_cnpj =
'").append(cnpj).append("' ordey by nome_pessoa");
         } else {
            sb.append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim, t.empresa_cnpj,
varchar 'CANCELAR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t, vagas v ")
            .append("WHERE p.cod_pessoa = a.pessoa_codpessoa
")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datafim) =
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND upper(v.estagio_interno) = upper('S') ")
            .append("AND t.vagas_codvaga = v.codvaga ordey by
nome_pessoa");
         }
         //stmt =
conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,R
esultSet.CONCUR_UPDATABLE);
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         GregorianCalendar dataInicio = new GregorianCalendar();
         GregorianCalendar dataFim = new GregorianCalendar();
         //SimpleDateFormat formataData = new
SimpleDateFormat("dd/MM/yyyy");
         while(rs.next()){
            RelatorioSeguroDTO relatorioDto = new
RelatorioSeguroDTO();

relatorioDto.setCpf(BigDecimal.valueOf(rs.getLong("cpf")));

relatorioDto.setEmpresa_cnpj(rs.getString("empresa_cnpj"));

relatorioDto.setNasc_pessoa(rs.getDate("nasc_pessoa"));
            relatorioDto.setDatafim(rs.getDate("datafim"));
            relatorioDto.setDatainicio(rs.getDate("datainicio"));

relatorioDto.setNome_pessoa(rs.getString("nome_pessoa"));
            relatorioDto.setProcesso(rs.getString("processo"));
            relatorioDto.setData(rs.getDate("data"));
            String dataIn = rs.getDate("datainicio").toString();
            String dataFi = rs.getDate("datafim").toString();
            lista.add(relatorioDto);
         }
      } catch (SQLException e) {
         throw e;
      } catch (Exception e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
      return lista;
   }

   public List<RelatorioSeguroDTO>
queryRelatorioIncluir(String cnpj, Integer mes, Integer ano,
boolean estagioInterno) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd HH:mm");
      List<RelatorioSeguroDTO> lista = new
ArrayList<RelatorioSeguroDTO>();
      GregorianCalendar data = new GregorianCalendar();
      data.set(Calendar.YEAR, ano.intValue());
      data.set(Calendar.MONTH, mes.intValue() - 1 );
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         if(!estagioInterno){
            sb.append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim, t.empresa_cnpj,
varchar 'INCLUIR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t WHERE p.cod_pessoa =
a.pessoa_codpessoa ")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datainicio)
= EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND t.empresa_cnpj =
'").append(cnpj).append("' ordey by nome_pessoa");
         } else {
            sb.append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim, t.empresa_cnpj,
varchar 'INCLUIR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t, vagas v ")
            .append("WHERE p.cod_pessoa = a.pessoa_codpessoa
")
```

```
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datainicio)
= EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND upper(v.estagio_interno) = upper('S') ")
            .append("AND t.vagas_codvaga = v.codvaga ordey by
nome_pessoa");
          }
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          GregorianCalendar dataInicio = new GregorianCalendar();
          GregorianCalendar dataFim = new GregorianCalendar();
          //SimpleDateFormat formataData = new
SimpleDateFormat("dd/MM/yyyy");
          while(rs.next()){
            RelatorioSeguroDTO relatorioDto = new
RelatorioSeguroDTO();

relatorioDto.setCpf(BigDecimal.valueOf(rs.getLong("cpf")));

relatorioDto.setEmpresa_cnpj(rs.getString("empresa_cnpj"));

relatorioDto.setNasc_pessoa(rs.getDate("nasc_pessoa"));
            relatorioDto.setDatafim(rs.getDate("datafim"));
            relatorioDto.setDatainicio(rs.getDate("datainicio"));

relatorioDto.setNome_pessoa(rs.getString("nome_pessoa"));
            relatorioDto.setProcesso(rs.getString("processo"));
            relatorioDto.setData(rs.getDate("data"));
            String dataIn = rs.getDate("datainicio").toString();
            String dataFi = rs.getDate("datafim").toString();
            lista.add(relatorioDto);
          }
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        }finally {
          closeConnection(conn, stmt, rs);
        }
        return lista;
    }

    public List<RelatorioSeguroDTO>
queryRelatorioManter(String cnpj, Integer mes, Integer ano,
boolean estagioInterno) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd HH:mm");
        List<RelatorioSeguroDTO> lista = new
ArrayList<RelatorioSeguroDTO>();
        GregorianCalendar data = new GregorianCalendar();
        data.set(Calendar.YEAR, ano.intValue());
        data.set(Calendar.MONTH, mes.intValue() - 1);
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          if(!estagioInterno){
            sb.append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'MANTER' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t WHERE p.cod_pessoa =
a.pessoa_codpessoa ")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND (")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append(") AND t.empresa_cnpj =
'").append(cnpj).append("' ordey by nome_pessoa");
          } else {
            sb.append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'MANTER' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t, vagas v ")
            .append("WHERE p.cod_pessoa = a.pessoa_codpessoa
")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND (")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
```

```
            .append("OR")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("))")
            .append("AND upper(v.estagio_interno) = upper('S') ")
            .append("AND t.vagas_codvaga = v.codvaga ordey by
nome_pessoa");
          }
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          GregorianCalendar dataInicio = new GregorianCalendar();
          GregorianCalendar dataFim = new GregorianCalendar();
          //SimpleDateFormat formataData = new
SimpleDateFormat("dd/MM/yyyy");
          while(rs.next()){
            RelatorioSeguroDTO relatorioDto = new
RelatorioSeguroDTO();

relatorioDto.setCpf(BigDecimal.valueOf(rs.getLong("cpf")));

relatorioDto.setEmpresa_cnpj(rs.getString("empresa_cnpj"));

relatorioDto.setNasc_pessoa(rs.getDate("nasc_pessoa"));
            relatorioDto.setDatafim(rs.getDate("datafim"));
            relatorioDto.setDatainicio(rs.getDate("datainicio"));

relatorioDto.setNome_pessoa(rs.getString("nome_pessoa"));
            relatorioDto.setProcesso(rs.getString("processo"));
            relatorioDto.setData(rs.getDate("data"));
            String dataIn = rs.getDate("datainicio").toString();
            String dataFi = rs.getDate("datafim").toString();
          }
        } catch (SQLException e) {
          throw e;
        } catch (Exception e) {
          throw e;
        }finally {
          closeConnection(conn, stmt, rs);
        }
        return lista;
    }
```

```
    public List<RelatorioSeguroDTO> queryRelatorioGeral(String
cnpj, Integer mes, Integer ano, boolean estagioInterno) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<RelatorioSeguroDTO> lista = new
ArrayList<RelatorioSeguroDTO>();
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd HH:mm");
        GregorianCalendar data = new GregorianCalendar();
        data.set(Calendar.YEAR, ano.intValue());
        data.set(Calendar.MONTH, mes.intValue() - 1 );
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          if(!estagioInterno){
            sb.append("SELECT DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'CANCELAR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t WHERE p.cod_pessoa =
a.pessoa_codpessoa ")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datafim) =
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(" UNION ALL ")
            .append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'INCLUIR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t WHERE p.cod_pessoa =
a.pessoa_codpessoa ")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datainicio)
= EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(" UNION ALL ")
            .append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'MANTER' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t WHERE p.cod_pessoa =
a.pessoa_codpessoa ")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND (")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
```

```
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append(") ")
            .append("AND t.empresa_cnpj =
'").append(cnpj).append("'");
        } else {
            sb.append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'CANCELAR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t, vagas v ")
            .append("WHERE p.cod_pessoa = a.pessoa_codpessoa
")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datafim) =
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND upper(v.estagio_interno) = upper('S') ")
            .append("AND t.vagas_codvaga = v.codvaga")
            .append(" UNION ALL ")
            .append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'INCLUIR' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t, vagas v ")
            .append("WHERE p.cod_pessoa = a.pessoa_codpessoa
")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND EXTRACT(MONTH FROM datainicio)
= EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND upper(v.estagio_interno) = upper('S') ")
            .append("AND t.vagas_codvaga = v.codvaga")
            .append(" UNION ALL ")
            .append("SELECT  DISTINCT ON (cpf) cpf,
nome_pessoa, nasc_pessoa, datainicio, datafim,  t.empresa_cnpj,
varchar 'MANTER' as processo, date
'").append(sdf.format(data.getTime())).append("' as data ")
            .append("FROM pessoa p, alunoestagio a,
termocompromisso t, vagas v ")
            .append("WHERE p.cod_pessoa = a.pessoa_codpessoa
")
            .append("AND a.codalunoestagio =
t.alunoestagio_codalunoestagio ")
            .append("AND (")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(MONTH FROM datainicio) <
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datainicio) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) =
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(MONTH FROM datafim) >
EXTRACT(MONTH FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append(")")
            .append("OR")
            .append("(")
            .append("EXTRACT(YEAR FROM datainicio) <
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("AND EXTRACT(YEAR FROM datafim) >
EXTRACT(YEAR FROM timestamp
'").append(sdf.format(data.getTime())).append("') ")
            .append("))")
            .append("AND upper(v.estagio_interno) = upper('S') ")
            .append("AND t.vagas_codvaga = v.codvaga ordey by
nome_pessoa");
        }
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        GregorianCalendar dataInicio = new GregorianCalendar();
        GregorianCalendar dataFim = new GregorianCalendar();
        //SimpleDateFormat formataData = new
SimpleDateFormat("dd/MM/yyyy");
        while(rs.next()){
          RelatorioSeguroDTO relatorioDto = new
RelatorioSeguroDTO();

relatorioDto.setCpf(BigDecimal.valueOf(rs.getLong("cpf")));
```

```
relatorioDto.setEmpresa_cnpj(rs.getString("empresa_cnpj"));

relatorioDto.setNasc_pessoa(rs.getDate("nasc_pessoa"));
            relatorioDto.setDatafim(rs.getDate("datafim"));
            relatorioDto.setDatainicio(rs.getDate("datainicio"));

relatorioDto.setNome_pessoa(rs.getString("nome_pessoa"));
            relatorioDto.setProcesso(rs.getString("processo"));
            relatorioDto.setData(rs.getDate("data"));
            /*String dataIn = rs.getDate("datainicio").toString();
            String dataFi = rs.getDate("datafim").toString();
            //int testeMes =
Integer.valueOf(dataIn.substring(5,7)).intValue();
            dataInicio.set(Calendar.MONTH,
Integer.valueOf(dataIn.substring(5,7)).intValue() - 1);
            dataFim.set(Calendar.MONTH,
Integer.valueOf(dataFi.substring(5,7)).intValue() - 1);
            //int testeAno =
Integer.valueOf(dataIn.substring(0,4)).intValue();
            dataInicio.set(Calendar.YEAR,
Integer.valueOf(dataIn.substring(0,4)).intValue());
            dataFim.set(Calendar.YEAR,
Integer.valueOf(dataFi.substring(0,4)).intValue());
            if(dataInicio.get(Calendar.MONTH) ==
data.get(Calendar.MONTH) && dataInicio.get(Calendar.YEAR)
== data.get(Calendar.YEAR)){
                relatorioDto.setProcesso("INCLUIR");
            } else if(dataFim.get(Calendar.MONTH) ==
data.get(Calendar.MONTH) && dataFim.get(Calendar.YEAR)
== data.get(Calendar.YEAR)){
                relatorioDto.setProcesso("CANCELAR");
            } else {
                relatorioDto.setProcesso("MANTER");
            }*/
            lista.add(relatorioDto);
        }
      } catch (SQLException e) {
        throw e;
      } catch (Exception e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return lista;
    }
}
```

6.4.285 Servidor_smtpDao.java

```
/*
 * Servidor_smtpDao.java
 * Created on 15 de Maio de 2006, 17:06
 */

package estagio.dao;

import estagio.bean.ServidorSmtp;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class Servidor_smtpDao extends BaseDao{


  /** Creates a new instance of Servidor_smtpDao */
  public Servidor_smtpDao() throws Exception{

  }

  public List<ServidorSmtp> listarServidor_smtp() throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      ArrayList<ServidorSmtp> servidores = new
ArrayList<ServidorSmtp>();
      try {
        conn = getConnection();
        stmt = conn.prepareStatement("select cod_servidor_smtp
from servidor_smtp");
        rs = stmt.executeQuery();
        while (rs.next()) {

servidores.add(buscarServidorSmtp(Integer.valueOf(rs.getInt("co
d_servidor_smtp"))));
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return servidores;
  }

  public ServidorSmtp buscarServidorSmtp(Integer
codServidorSmtp) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      ServidorSmtp servidorSmtp = new ServidorSmtp();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select cod_servidor_smtp, url, username,
senha, porta_smtp, endereco_email, requer_autenticacao from
servidor_smtp where cod_servidor_smtp ='")
        .append(codServidorSmtp)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if (rs.next()) {
          servidorSmtp.setCod_servidor_smtp(codServidorSmtp);

servidorSmtp.setEndereco_email(rs.getString("endereco_email"))
;

servidorSmtp.setPorta_smtp(Integer.valueOf(rs.getInt("porta_smt
p")));

servidorSmtp.setRequer_autenticacao(rs.getBoolean("requer_aute
nticacao"));
          servidorSmtp.setSenha(rs.getString("senha"));
          servidorSmtp.setUrl(rs.getString("url"));
          servidorSmtp.setUsername(rs.getString("username"));
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return servidorSmtp;
  }

  public void incluirServidorSmtp(ServidorSmtp servidorSmtp)
throws SQLException, Exception {
```

```
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into servidor_smtp (cod_servidor_smtp,
url, username, senha, porta_smtp, endereco_email,
requer_autenticacao)")
            .append(" values ('")
            .append(servidorSmtp.getCod_servidor_smtp())
            .append("','")
            .append(servidorSmtp.getUrl())
            .append("','")
            .append(servidorSmtp.getUsername())
            .append("','")
            .append(servidorSmtp.getSenha())
            .append("','")
            .append(servidorSmtp.getPorta_smtp())
            .append("','")
            .append(servidorSmtp.getEndereco_email())
            .append("','")
            .append(servidorSmtp.getRequer_autenticacao())
            .append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirServidorSmtp(Integer codServidorSmtp)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from servidor_smtp where
cod_servidor_smtp = '")
            .append(codServidorSmtp)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarServidorSmtp(ServidorSmtp servidorSmtp)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update servidor_smtp set
url='").append(servidorSmtp.getUrl())
            .append("', username =
'").append(servidorSmtp.getUsername())
            .append("', senha = '").append(servidorSmtp.getSenha())
            .append("', porta_smtp =
'").append(servidorSmtp.getPorta_smtp())
            .append("', endereco_email =
'").append(servidorSmtp.getEndereco_email())
            .append("', requer_autenticacao =
'").append(servidorSmtp.getRequer_autenticacao())
            .append("' where cod_servidor_smtp =
'").append(servidorSmtp.getCod_servidor_smtp()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }
}
```

6.4.286 TermoCompromissoDao.java

```
/*
 * TermoCompromissoDao.java
 * Created on 21 de Fevereiro de 2006, 15:13
 */

package estagio.dao;

import estagio.bean.Empresa;
import estagio.bean.TermoCompromisso;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class TermoCompromissoDao extends BaseDao{

    AgentesIntegracaoDao agentesIntegracaoDao;
    VagaDao vagaDao;
    PlanosEstagioDao planosEstagioDao;

    /** Creates a new instance of TermoCompromissoDao */
    public TermoCompromissoDao() throws Exception{
        agentesIntegracaoDao = new AgentesIntegracaoDao();
        vagaDao = new VagaDao();
        planosEstagioDao = new PlanosEstagioDao();
    }

    public List<TermoCompromisso> listarTermosCompromisso()
throws SQLException, Exception {
        Connection conn = null;
```

```
        PreparedStatement stmt = null;
        ResultSet rs = null;
        ArrayList<TermoCompromisso> termoCompromissos =
new ArrayList<TermoCompromisso>();
        try {
          conn = getConnection();
          stmt = conn.prepareStatement("select codtermo from
termocompromisso order by empresa_cnpj");
          rs = stmt.executeQuery();
          while (rs.next()) {

termoCompromissos.add(buscarTermoCompromisso(Integer.valu
eOf(rs.getInt("codtermo"))));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return termoCompromissos;
    }

    public List<TermoCompromisso>
listarTermosCompromisso(Integer codAlunoEstagio) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        ArrayList<TermoCompromisso> termoCompromissos =
new ArrayList<TermoCompromisso>();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codtermo from termocompromisso
where alunoestagio_codalunoestagio = '")
          .append(codAlunoEstagio).append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while (rs.next()) {

termoCompromissos.add(buscarTermoCompromisso(Integer.valu
eOf(rs.getInt("codtermo"))));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return termoCompromissos;
    }

    public List<TermoCompromisso> listarTermos(String cnpj)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        ArrayList<TermoCompromisso> termoCompromissos =
new ArrayList<TermoCompromisso>();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codtermo from termocompromisso
where empresa_cnpj = '")
          .append(cnpj).append("' order by
alunoestagio_codalunoestagio");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          while (rs.next()) {

termoCompromissos.add(buscarTermoCompromisso(Integer.valu
eOf(rs.getInt("codtermo"))));
          }
        } catch (SQLException e) {
          throw e;
        } finally {
          closeConnection(conn, stmt, rs);
        }
        return termoCompromissos;
    }


    public TermoCompromisso buscarTermoCompromisso(Integer
codTermo) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        TermoCompromisso termoCompromisso = new
TermoCompromisso();
        EmpresaDao empresaDao = new EmpresaDao();
        AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
        try {
          conn = getConnection();
          StringBuffer sb = new StringBuffer();
          sb.append("select codtermo ,
alunoestagio_codalunoestagio, empresa_cnpj,
agente_integracao_cnpj, datainicio, datafim, vagas_codvaga,
obrigatorio, seguro_num_apolice, seguro_companhia,
seguro_providenciado from termocompromisso where codtermo
='")
          .append(codTermo)
          .append("'");
          stmt = conn.prepareStatement(sb.toString());
          rs = stmt.executeQuery();
          if (rs.next()) {
            GregorianCalendar dataInicio = new
GregorianCalendar();
            GregorianCalendar dataFim = new
GregorianCalendar();

termoCompromisso.setCodTermo(Integer.valueOf(rs.getInt("codt
ermo")));
            dataFim.setTime(rs.getDate("dataFim"));
            dataInicio.setTime(rs.getDate("dataInicio"));
            termoCompromisso.setDataFim(dataFim);
            termoCompromisso.setDataInicio(dataInicio);

termoCompromisso.setAgenteIntegracao(agentesIntegracaoDao.b
uscarAgenteIntegracao(rs.getString("agente_integracao_cnpj")));
            String cnpj = rs.getString("empresa_cnpj");
            Empresa empresa = empresaDao.buscarEmpresa(cnpj);
            termoCompromisso.setEmpresa(empresa);
            Integer codAlunoEstagio =
Integer.valueOf(rs.getInt("alunoestagio_codalunoestagio"));

termoCompromisso.setAlunoEstagio(alunoEstagioDao.buscarAlu
noEstagio(codAlunoEstagio));

termoCompromisso.setVaga(vagaDao.buscarVaga(Integer.valueO
f(rs.getInt("vagas_codvaga"))));

termoCompromisso.setPlanoEstagio(planosEstagioDao.buscarPla
noEstagio(rs.getInt("codtermo")));

termoCompromisso.setObrigatorio(rs.getString("obrigatorio"));

termoCompromisso.setSeguroCompanhia(rs.getString("seguro_co
mpanhia"));

termoCompromisso.setSeguroNumApolice(rs.getLong("seguro_n
um_apolice"));

termoCompromisso.setSeguroProvidenciado(rs.getString("seguro
_providenciado"));
```

```
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return termoCompromisso;
    }

    public TermoCompromisso buscarTermoCompromisso(Integer
codAlunoEstagio, String cnpj, Integer codVaga) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      TermoCompromisso termoCompromisso = new
TermoCompromisso();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codtermo from termocompromisso
where alunoestagio_codalunoestagio ='")
        .append(codAlunoEstagio)
        .append("' and empresa_cnpj = '").append(cnpj).append("'
and vagas_codvaga ='").append(codVaga).append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if (rs.next()) {
          termoCompromisso =
buscarTermoCompromisso(Integer.valueOf(rs.getInt("codtermo")
));
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return termoCompromisso;
    }

    public void incluirTermoCompromisso(TermoCompromisso
termoCompromisso) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
        String dataInicio =
sdf.format(termoCompromisso.getDataInicio().getTime());
        String dataFim =
sdf.format(termoCompromisso.getDataFim().getTime());
        if(termoCompromisso.getEmpresa() == null){
          termoCompromisso.setEmpresa(new Empresa());
        }
        sb.append("insert into termocompromisso
(alunoestagio_codalunoestagio, empresa_cnpj, ");
        if(termoCompromisso.getAgenteIntegracao() != null){
          if(termoCompromisso.getAgenteIntegracao().getCnpj()
!= null){
            sb.append("agente_integracao_cnpj,");
          }
        }
        sb.append(" datainicio, datafim, vagas_codvaga,
obrigatorio, seguro_num_apolice, seguro_companhia,
seguro_providenciado)")
        .append(" values ('")

.append(termoCompromisso.getAlunoEstagio().getCodAlunoEsta
gio())
        .append("','")
        .append(termoCompromisso.getEmpresa().getCnpj())
        .append("','");
        if(termoCompromisso.getAgenteIntegracao() != null){
          if(termoCompromisso.getAgenteIntegracao().getCnpj()
!= null){

sb.append(termoCompromisso.getAgenteIntegracao().getCnpj())
            .append("','");
          }
        }
        sb.append(dataInicio)
        .append("','")
        .append(dataFim)
        .append("','")
        .append(termoCompromisso.getVaga().getCodVaga())
        .append("','")
        .append(termoCompromisso.getObrigatorio())
        .append("','")
        .append(termoCompromisso.getSeguroNumApolice())
        .append("','")
        .append(termoCompromisso.getSeguroCompanhia())
        .append("','")
        .append(termoCompromisso.getSeguroProvidenciado())
        .append("')");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
        vagaDao.preencherVaga(termoCompromisso.getVaga());
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void excluirTermoCompromisso(TermoCompromisso
termoCompromisso) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {

planosEstagioDao.excluirPlanosEstagio(termoCompromisso.getC
odTermo().intValue());
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("delete from termocompromisso where
codtermo = '")
        .append(termoCompromisso.getCodTermo())
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
        vagaDao.abrirVaga(termoCompromisso.getVaga());
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }
```

```
    public void excluirTermosCompromisso(String cnpj) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            List<TermoCompromisso> termos = listarTermos(cnpj);
            for(TermoCompromisso termo: termos){

planosEstagioDao.excluirPlanosEstagio(termo.getCodTermo().int
Value());
            }
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from TermoCompromisso where
empresa_cnpj = '")
            .append(cnpj)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarTermoCompromisso(TermoCompromisso
termoCompromisso) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        if(termoCompromisso.getEmpresa() == null){
            termoCompromisso.setEmpresa(new Empresa());
        }
        try {

planosEstagioDao.alterarPlanoEstagio(termoCompromisso.getPla
noEstagio());
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
            String dataInicio =
sdf.format(termoCompromisso.getDataInicio().getTime());
            String dataFim =
sdf.format(termoCompromisso.getDataFim().getTime());
            StringBuffer sb = new StringBuffer();
            sb.append("update termoCompromisso set
dataInicio='").append(sdf.format(termoCompromisso.getDataInici
o().getTime()))
            .append("',
dataFim='").append(sdf.format(termoCompromisso.getDataFim().
getTime()));
            if(termoCompromisso.getAgenteIntegracao() != null){
                if(termoCompromisso.getAgenteIntegracao().getCnpj()
!= null){
                    sb.append("', agente_integracao_cnpj =
'").append(termoCompromisso.getAgenteIntegracao().getCnpj());
                }
            }
            sb.append("', obrigatorio =
'").append(termoCompromisso.getObrigatorio().charAt(0))
            .append("', seguro_num_apolice =
'").append(termoCompromisso.getSeguroNumApolice())
            .append("', seguro_companhia =
'").append(termoCompromisso.getSeguroCompanhia())
            .append("', seguro_providenciado =
'").append(termoCompromisso.getSeguroProvidenciado())
            .append("' where codtermo =
'").append(termoCompromisso.getCodTermo()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }
}
```

6.4.287 TurnoDao.java

```
/*
 * TurnoDao.java
 * Created on 28 de Janeiro de 2006, 10:46
 */

package estagio.dao;

import estagio.bean.Turno;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class TurnoDao extends BaseDao{

    /** Creates a new instance of TurnoDao */
    public TurnoDao() throws Exception{
    }


    public void incluirTurno(Turno turno) throws SQLException,
Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into turno (vagas_codvaga, horainicio,
horafim)")
              .append("  values ('")
              .append(turno.getCodVaga())
              .append("','")
              .append(turno.getHoraInicio().toString())
              .append("','")
              .append(turno.getHoraFim().toString())
              .append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
```

```
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirTurnos(Integer codVaga) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from turno where vagas_codvaga=");
            sb.append("'");
            sb.append(codVaga);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirTurno(Integer codTurno) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from turno where codturno=");
            sb.append("'");
            sb.append(codTurno);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void alterarTurno(Turno turno) throws SQLException,
Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update turno set
vagas_codvaga='").append(turno.getCodVaga())
            .append("', turno='").append(turno.getTurno())
            .append("',
horainicio='").append(turno.getHoraInicio().toString())
            .append("',
horafim='").append(turno.getHoraFim().toString())
            .append("' where codturno
='").append(turno.getCodTurno()).append("'");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public List<Turno> listarTurnos(Integer codVaga) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Turno> turnos = new ArrayList<Turno>();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codturno, vagas_codvaga, turno,
horainicio, horafim from turno where vagas_codvaga='")
            .append(codVaga)
            .append("' order by horainicio");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
                Turno turno = new Turno();

turno.setCodTurno(Integer.valueOf(rs.getInt("codturno")));
                turno.setCodVaga(codVaga);
                turno.setHoraInicio(rs.getTime("horainicio"));
                turno.setHoraFim(rs.getTime("horafim"));
                turno.setTurno(rs.getString("turno"));
                turnos.add(turno);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return turnos;
    }

    public Turno buscarTurno(Integer codTurno) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        Turno turno = new Turno();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
```

```
        sb.append("select codturno, vagas_codvaga, turno,
horainicio, horafim from turno where codturno='")
        .append(codTurno)
        .append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          turno.setCodTurno(codTurno);

turno.setCodVaga(Integer.valueOf(rs.getInt("vagas_codvaga")));
          turno.setHoraInicio(rs.getTime("horainicio"));
          turno.setHoraFim(rs.getTime("horafim"));
          turno.setTurno(rs.getString("turno"));
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return turno;
    }
}
```

6.4.288 TurnoDisponivelDao.java

```
/*
 * TurnoDisponivelDisponivelDao.java
 * Created on 9 de Fevereiro de 2006, 20:46
 */

package estagio.dao;

import estagio.bean.TurnoDisponivel;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class TurnoDisponivelDao extends BaseDao{

  /** Creates a new instance of TurnoDisponivelDao */
  public TurnoDisponivelDao() throws Exception{
  }


  public void incluirTurnoDisponivel(TurnoDisponivel
turnodisponivel) throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("insert into turnodisponivel
(alunoestagio_codalunoestagio, horainicio, horafim)");
      sb.append(" values ('");
      sb.append(turnodisponivel.getCodAlunoEstagio());
      sb.append("','");
      sb.append(turnodisponivel.getHoraInicio().toString());
      sb.append("','");
      sb.append(turnodisponivel.getHoraFim().toString());
      sb.append("')");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirTurnosDisponiveis(Integer
codAlunoEstagio) throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from turnodisponivel where
alunoestagio_codalunoestagio=");
      sb.append("'");
      sb.append(codAlunoEstagio);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void excluirTurnoDisponivel(Integer
codTurnoDisponivel) throws SQLException, Exception {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;
    try {
      conn = getConnection();
      StringBuffer sb = new StringBuffer();
      sb.append("delete from turnodisponivel where
codturnodisponivel=");
      sb.append("'");
      sb.append(codTurnoDisponivel);
      sb.append("'");
      stmt = conn.prepareStatement(sb.toString());
      stmt.executeUpdate();
    } catch (SQLException e) {
      throw e;
    } finally {
      if (rs != null) {
        rs.close();
      }
      if (stmt != null) {
        stmt.close();
      }
      closeConnection(conn, stmt, rs);
    }
  }

  public void alterarTurnoDisponivel(TurnoDisponivel
turnodisponivel) throws SQLException, Exception {
    Connection conn = null;
```

```
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            StringBuffer sb = new StringBuffer();
            sb.append("update turnodisponivel set
alunoestagio_codalunoestagio='").append(turnodisponivel.getCod
AlunoEstagio())
            .append("',
turnodisponivel='").append(turnodisponivel.getTurno())
            .append("',
horainicio='").append(turnodisponivel.getHoraInicio().toString())
            .append("',
horafim='").append(turnodisponivel.getHoraFim().toString())
            .append("' where codturnodisponivel
='").append(turnodisponivel.getCodTurnoDisponivel()).append("'
");
            conn = getConnection();
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e){
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public List<TurnoDisponivel> listarTurnosDisponiveis(Integer
codAlunoEstagio) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<TurnoDisponivel> turnodisponiveis = new
ArrayList<TurnoDisponivel>();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codturnodisponivel,
alunoestagio_codalunoestagio, turnodisponivel, horainicio,
horafim from turnodisponivel where
alunoestagio_codalunoestagio='")
            .append(codAlunoEstagio)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
                TurnoDisponivel turnodisponivel = new
TurnoDisponivel();

turnodisponivel.setCodTurnoDisponivel(Integer.valueOf(rs.getInt
("codturnodisponivel")));

turnodisponivel.setHoraInicio(rs.getTime("horainicio"));
                turnodisponivel.setHoraFim(rs.getTime("horafim"));

turnodisponivel.setTurno(rs.getString("turnodisponivel"));

turnodisponivel.setCodAlunoEstagio(Integer.valueOf(rs.getInt("al
unoestagio_codalunoestagio")));
                turnodisponiveis.add(turnodisponivel);
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return turnodisponiveis;
    }

    public TurnoDisponivel buscarTurnoDisponivel(Integer
codTurnoDisponivel) throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        TurnoDisponivel turnodisponivel = new TurnoDisponivel();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select codturnodisponivel,
alunoestagio_codalunoestagio, turnodisponivel, horainicio,
horafim from turnodisponivel where codturnodisponivel='")
            .append(codTurnoDisponivel)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if(rs.next()){

turnodisponivel.setCodTurnoDisponivel(codTurnoDisponivel);

turnodisponivel.setCodAlunoEstagio(Integer.valueOf(rs.getInt("al
unoestagio_codalunoestagio")));

turnodisponivel.setHoraInicio(rs.getTime("horainicio"));
                turnodisponivel.setHoraFim(rs.getTime("horafim"));

turnodisponivel.setTurno(rs.getString("turnodisponivel"));

turnodisponivel.setCodAlunoEstagio(Integer.valueOf(rs.getInt("al
unoestagio_codalunoestagio")));
            } else {
                return null;
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return turnodisponivel;
    }
}
```

6.4.289 UsuarioEmpresaDao.java

```
/*
 * UsuarioEmpresaDao.java
 * Created on 30 de Janeiro de 2006, 17:29
 */

package estagio.dao;

import estagio.bean.AlunoEstagio;
import estagio.bean.Empresa;
import estagio.bean.Turno;
import estagio.bean.UsuarioEmpresa;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
```

```
public class UsuarioEmpresaDao  extends BaseDao{

   /** Creates a new instance of UsuarioEmpresaDao */
   public UsuarioEmpresaDao() throws Exception{
   }


   public void incluirUsuarioEmpresa(UsuarioEmpresa
usuarioEmpresa) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("insert into usuario_empresa (empresa_cnpj,
alunoestagio_codalunoestagio )");
         sb.append("  values ('");
         sb.append(usuarioEmpresa.getCnpj());
         sb.append("',");
         sb.append(usuarioEmpresa.getCodAlunoEstagio());
         sb.append(")");
         stmt = conn.prepareStatement(sb.toString());
         stmt.executeUpdate();
      } catch (SQLException e) {
         throw e;
      } catch (Exception e) {
         throw e;
      } finally {
         if (rs != null) {
            rs.close();
         }
         if (stmt != null) {
            stmt.close();
         }
         closeConnection(conn, stmt, rs);
      }
   }

   public void excluirUsuarioEmpresa(Integer codAlunoEstagio)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("delete from usuario_empresa where
alunoestagio_codalunoestagio =");
         sb.append("'");
         sb.append(codAlunoEstagio);
         sb.append("'");
         stmt = conn.prepareStatement(sb.toString());
         stmt.executeUpdate();
      } catch (SQLException e) {
         throw e;
      } finally {
         if (rs != null) {
            rs.close();
         }
         if (stmt != null) {
            stmt.close();
         }
         closeConnection(conn, stmt, rs);
      }
   }

   public void excluirUsuarioEmpresa(String cnpj) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("delete from usuario_empresa where
empresa_cnpj =");
         sb.append("'");
         sb.append(cnpj);
         sb.append("'");
         stmt = conn.prepareStatement(sb.toString());
         stmt.executeUpdate();
      } catch (SQLException e) {
         throw e;
      } finally {
         if (rs != null) {
            rs.close();
         }
         if (stmt != null) {
            stmt.close();
         }
         closeConnection(conn, stmt, rs);
      }
   }

   public List<AlunoEstagio> listarUsuarioEmpresas(String cnpj)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<AlunoEstagio> usuarios = new
ArrayList<AlunoEstagio>();
      AlunoEstagioDao alunoEstagioDao = new
AlunoEstagioDao();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select alunoestagio_codalunoestagio from
usuario_empresa where empresa_cnpj='")
         .append(cnpj)
         .append("'");
         stmt = conn.prepareStatement(sb.toString());
         rs = stmt.executeQuery();
         while(rs.next()){
            AlunoEstagio usuario = null;
            Integer codAlunoEstagio =
Integer.valueOf(rs.getInt("alunoestagio_codalunoestagio"));
            usuario =
alunoEstagioDao.buscarAlunoEstagio(codAlunoEstagio);
            if(usuario != null){
               usuarios.add(usuario);
            }

         }
      } catch (SQLException e) {
         throw e;
      } finally {
         closeConnection(conn, stmt, rs);
      }
      return usuarios;
   }

   public Empresa buscarUsuarioEmpresa(Integer
codAlunoEstagio) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      Empresa empresa = new Empresa();
      EmpresaDao empresaDao = new EmpresaDao();
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         sb.append("select empresa_cnpj from usuario_empresa
where alunoestagio_codalunoestagio='")
```

```
            .append(codAlunoEstagio)
            .append("");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if(rs.next()){
               String cnpj = rs.getString("empresa_cnpj");
               empresa = empresaDao.buscarEmpresa(cnpj);
            } else {
               return null;
            }
         } catch (SQLException e) {
            throw e;
         } finally {
            closeConnection(conn, stmt, rs);
         }
         return empresa;
      }
}
```

6.4.290 VagaDao.java

```
/*
 * VagaDao.java
 * Created on 28 de Janeiro de 2006, 10:10
 */

package estagio.dao;

import estagio.bean.HabilidadeVaga;
import estagio.bean.Turno;
import estagio.bean.Vaga;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.GregorianCalendar;
import java.util.List;

/**
 *
 *
 */
public class VagaDao extends BaseDao{

   private TurnoDao turnoDao;
   private HabilidadesVagaDao habilidadesVagaDao;
   private VagasCursosDao vagasCursosDao;

   /** Creates a new instance of VagaDao */
   public VagaDao() throws Exception{
      turnoDao = new TurnoDao();
      habilidadesVagaDao = new HabilidadesVagaDao();
      vagasCursosDao = new VagasCursosDao();
   }


   public void incluirVaga(Vaga vaga) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         conn = getConnection();
         StringBuffer sb = new StringBuffer();
         SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
         String inclusao =
sdf.format(vaga.getInclusao().getTime());
         sb.append("insert into vagas (empresa_cnpj, setor,
horassemana, salario, sexo, inclusao, locomocao, fumante,
deficiente, atividadesfds, semestre, vale_transporte,
outros_beneficios, estagio_interno )")
            .append(" values ('")
            .append(vaga.getCnpj())
            .append("','")
            .append(vaga.getSetor())
            .append("','")
            .append(vaga.getHoras())
            .append("','")
            .append(vaga.getSalario())
            .append("','")
            .append(vaga.getSexo())
            .append("','")
            .append(inclusao)
            .append("','")
            .append(vaga.getLocomocao())
            .append("','")
            .append(vaga.getFumante())
            .append("','")
            .append(vaga.getDeficiente())
            .append("','")
            .append(vaga.getAtividadesFds())
            .append("','")
            .append(vaga.getSemestre())
            .append("','")
            .append(vaga.getValeTransporte().charAt(0))
            .append("','")
            .append(vaga.getOutrosBeneficios())
            .append("','")
            .append(vaga.getEstagioInterno())
            .append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
         } catch (SQLException e) {
            throw e;
         } finally {
            if (rs != null) {
               rs.close();
            }
            if (stmt != null) {
               stmt.close();
            }
            closeConnection(conn, stmt, rs);
         }
      }

   public void excluirVagas(String cnpj) throws SQLException,
Exception {
      try {
         List<Vaga> vagas = listarVagas(cnpj);
         for(Vaga vaga: vagas){
            excluirVaga(vaga.getCodVaga());
         }
      } catch (SQLException e) {
         throw e;
      } catch (Exception e) {
         throw e;
      }
   }

   public void excluirVaga(Integer codVaga) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
         turnoDao.excluirTurnos(codVaga);
         habilidadesVagaDao.excluirHabilidadesVaga(codVaga);
         vagasCursosDao.excluirVagaCurso(codVaga);
         conn = getConnection();
```

```
        StringBuffer sb = new StringBuffer();
        sb.append("delete from vagas where codvaga=");
        sb.append("'");
        sb.append(codVaga);
        sb.append("'");
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void alterarVaga(Vaga vaga) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
        DecimalFormat df = new DecimalFormat("0.00");
        StringBuffer sb = new StringBuffer();
        sb.append("update vagas set
empresa_cnpj='").append(vaga.getCnpj())
        .append("', setor='").append(vaga.getSetor())
        .append("', horassemana='").append(vaga.getHoras())
        .append("', salario='").append(vaga.getSalario())
        .append("', sexo='").append(vaga.getSexo())
        .append("', locomocao='").append(vaga.getLocomocao())
        .append("', fumante='").append(vaga.getFumante())
        .append("', deficiente='").append(vaga.getDeficiente())
        .append("',
atividadesfds='").append(vaga.getAtividadesFds())
        .append("', semestre='").append(vaga.getSemestre())
        .append("',
vale_transporte='").append(vaga.getValeTransporte().charAt(0))
        .append("',
outros_beneficios='").append(vaga.getOutrosBeneficios())
        .append("' where codvaga
='").append(vaga.getCodVaga()).append("'");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } catch (Exception e){
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public void preencherVaga(Vaga vaga) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-
MM-dd");
        String preenchimento =
sdf.format(vaga.getPreenchimento().getTime());
        StringBuffer sb = new StringBuffer();
        sb.append("update vagas set
preenchimento='").append(preenchimento)
        .append("' where codvaga
='").append(vaga.getCodVaga()).append("'");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } catch (Exception e){
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public List<Vaga> listarVagas(String cnpj) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga from vagas where
empresa_cnpj='")
        .append(cnpj)
        .append("' order by inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }

    public List<Vaga> listarVagasAbertas(String cnpj) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga from vagas where
empresa_cnpj='")
        .append(cnpj)
        .append("' and preenchimento is null order by inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
```

```
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }


    public Vaga buscarVaga(Integer codVaga) throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      CursoDao cursoDao = new CursoDao();
      Vaga vaga = new Vaga();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga, empresa_cnpj, setor,
horassemana, salario, sexo, inclusao, preenchimento, locomocao,
fumante, deficiente, atividadesfds, semestre, vale_transporte,
outros_beneficios, estagio_interno from vagas where codVaga='")
          .append(codVaga)
          .append("'");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        if(rs.next()){
          vaga.setCodVaga(codVaga);
          vaga.setCnpj(rs.getString("empresa_cnpj"));
          vaga.setSetor(rs.getString("setor"));

vaga.setHoras(Integer.valueOf(rs.getInt("horassemana")));
          vaga.setSalario(rs.getString("salario"));
          vaga.setSexo(rs.getString("sexo"));
          GregorianCalendar inclusao = new
GregorianCalendar();
          GregorianCalendar preenchimento = new
GregorianCalendar();
          inclusao.setTime(rs.getDate("inclusao"));
          vaga.setInclusao(inclusao);
          Date datePreenchimento =
rs.getDate("preenchimento");
          if(datePreenchimento != null){
            preenchimento.setTime(datePreenchimento);
            vaga.setPreenchimento(preenchimento);
          }
          List<HabilidadeVaga> habilidades = new
ArrayList<HabilidadeVaga>();
          habilidades =
habilidadesVagaDao.listarHabilidadeVaga(vaga.getCodVaga());
          vaga.setHabilidades(habilidades);
          List<Turno> turnos = new ArrayList<Turno>();
          turnos = turnoDao.listarTurnos(vaga.getCodVaga());
          vaga.setTurnos(turnos);
          vaga.setLocomocao(rs.getString("locomocao"));
          vaga.setFumante(rs.getString("fumante"));
          vaga.setDeficiente(rs.getString("deficiente"));
          vaga.setAtividadesFds(rs.getString("atividadesfds"));

vaga.setCursos(vagasCursosDao.listarCursos(codVaga));

vaga.setSemestre(Integer.valueOf(rs.getInt("semestre")));

vaga.setValeTransporte(rs.getString("vale_transporte"));

vaga.setOutrosBeneficios(rs.getString("outros_beneficios"));

vaga.setEstagioInterno(Character.valueOf(rs.getString("estagio_i
nterno").charAt(0)));
        } else {
          return null;
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vaga;
    }

    public List<Vaga> listarVagasAbertas() throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga from vagas where
preenchimento is null order by inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
}
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }

    public List<Vaga> listarVagasAbertasEstagioInterno() throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga from vagas where
preenchimento is null and estagio_interno = 'S' order by
inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
}
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }

    public List<Vaga> listarVagasPreenchidasEstagioInterno()
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
```

```
        sb.append("select codvaga from vagas where
preenchimento is not null and estagio_interno = 'S' order by
inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
}
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }

    public List<Vaga> listarTodasVagasEstagioInterno() throws
SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga from vagas where
estagio_interno = 'S' order by inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
}
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }


    public List<Vaga> buscarVagasAbertas(char disponivelFds,
char fumante) throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagas = new ArrayList<Vaga>();
      CursoDao cursoDao = new CursoDao();
      try {
        conn = getConnection();
        StringBuffer sb = new StringBuffer();
        sb.append("select codvaga from vagas where");
        if (fumante == 'S'){
          sb.append(" fumante = 'S' and");
        }
        if(disponivelFds == 'N'){
          sb.append(" atividadesfds = 'N' and");
        }
        sb.append(" preenchimento is null order by inclusao");
        stmt = conn.prepareStatement(sb.toString());
        rs = stmt.executeQuery();
        while(rs.next()){

vagas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga"))));
        }
      } catch (SQLException e) {
        throw e;
      } finally {
        closeConnection(conn, stmt, rs);
      }
      return vagas;
    }


    public List<Vaga> listarVagas(Integer codCurso) throws
SQLException, Exception {
      try {
        return vagasCursosDao.listarVagas(codCurso);
      }catch (SQLException e) {
        throw e;
      }catch (Exception e) {
        throw e;
      }
    }

    public void abrirVaga(Vaga vaga) throws SQLException,
Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      try {
        StringBuffer sb = new StringBuffer();
        sb.append("update vagas set preenchimento = null where
codvaga = '")
          .append(vaga.getCodVaga()).append("'");
        conn = getConnection();
        stmt = conn.prepareStatement(sb.toString());
        stmt.executeUpdate();
      } catch (SQLException e) {
        throw e;
      } catch (Exception e){
        throw e;
      } finally {
        if (rs != null) {
          rs.close();
        }
        if (stmt != null) {
          stmt.close();
        }
        closeConnection(conn, stmt, rs);
      }
    }

    public List<Vaga> listarVagasAbertas(Integer codCurso)
throws SQLException, Exception {
      Connection conn = null;
      PreparedStatement stmt = null;
      ResultSet rs = null;
      List<Vaga> vagasAbertas = new ArrayList<Vaga>();
      List<Vaga> vagasAbertasCurso = new ArrayList<Vaga>();
      CursoDao cursoDao = new CursoDao();
      try {
        List<Vaga> vagasCurso =
vagasCursosDao.listarVagas(codCurso);
        conn = getConnection();
        String query = "select codvaga from vagas where
preenchimento is null order by empresa_cnpj" ;
        stmt = conn.prepareStatement(query);
        rs = stmt.executeQuery();
        while(rs.next()){

vagasAbertas.add(buscarVaga(Integer.valueOf(rs.getInt("codvaga
")));
        }
        for(Vaga v:vagasAbertas){
          if(vagasCurso.contains(v)){
            vagasAbertasCurso.add(v);
          }
        }
      } catch (SQLException e) {
        throw e;
      } finally {
```

```
            closeConnection(conn, stmt, rs);
        }
        return vagasAbertasCurso;
    }

    public Vaga buscarNovaVaga() throws SQLException,
Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        CursoDao cursoDao = new CursoDao();
        try {
            conn = getConnection();
            stmt = conn.prepareStatement("select max(codvaga) as
codvaga from vagas");
            rs = stmt.executeQuery();
            if(rs.next()){
                return
buscarVaga(Integer.valueOf(rs.getInt("codvaga")));
            } else {
                return null;
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
    }

}
```

6.4.291 VagasCursosDao.java

```
/*
 * VagasCursosDao.java
 * Created on 10 de Abril de 2006, 14:29
 */

package estagio.dao;
import estagio.bean.Curso;
import estagio.bean.Vaga;
import estagio.bean.VagaCurso;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

/**
 *
 *
 */
public class VagasCursosDao extends BaseDao{

    /** Creates a new instance of VagasCursosDao */
    public VagasCursosDao() throws Exception{
    }


    public void incluirVagaCurso(VagaCurso vagaCurso) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("insert into vagas_cursos (vagas_codvaga,
cursos_codcurso)");
            sb.append(" values ('");
            sb.append(vagaCurso.getCodVaga());
            sb.append("','");
            sb.append(vagaCurso.getCodCurso());
            sb.append("')");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } catch (Exception e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirVagaCurso(Integer codVaga) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from vagas_cursos where
vagas_codvaga =");
            sb.append("'");
            sb.append(codVaga);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
            closeConnection(conn, stmt, rs);
        }
    }

    public void excluirVagaCurso(String codCurso) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("delete from vagas_cursos where
cursos_codcurso =");
            sb.append("'");
            sb.append(codCurso);
            sb.append("'");
            stmt = conn.prepareStatement(sb.toString());
            stmt.executeUpdate();
        } catch (SQLException e) {
            throw e;
        } finally {
            if (rs != null) {
                rs.close();
            }
            if (stmt != null) {
                stmt.close();
            }
```

```
            closeConnection(conn, stmt, rs);
        }
    }

    public List<Vaga> listarVagas(Integer codCurso) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Vaga> vagas = new ArrayList<Vaga>();
        VagaDao vagaDao = new VagaDao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select vagas_codvaga from vagas_cursos
where cursos_codcurso='")
            .append(codCurso)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
                Vaga vaga = null;
                Integer codVaga=
Integer.valueOf(rs.getInt("vagas_codvaga"));
                vaga = vagaDao.buscarVaga(codVaga);
                if(vaga != null){
                    vagas.add(vaga);
                }
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return vagas;
    }

    public List<Curso> listarCursos(Integer codVaga) throws
SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        List<Curso> cursos = new ArrayList<Curso>();
        CursoDao cursoDao = new CursoDao();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select cursos_codcurso from vagas_cursos
where vagas_codvaga='")
            .append(codVaga)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            while(rs.next()){
                Curso curso = null;
                Integer codCurso=
Integer.valueOf(rs.getInt("cursos_codcurso"));
                curso = cursoDao.buscarCurso(codCurso);
                if(curso != null){
                    cursos.add(curso);
                }
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return cursos;
    }

    public VagaCurso buscarVagaCurso(Integer codVagaCurso)
throws SQLException, Exception {
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;
        VagaCurso vagaCurso = new VagaCurso();
        try {
            conn = getConnection();
            StringBuffer sb = new StringBuffer();
            sb.append("select vagas_codvaga, cursos_codcurso from
vagas_cursos where codvagas_cursos='")
            .append(codVagaCurso)
            .append("'");
            stmt = conn.prepareStatement(sb.toString());
            rs = stmt.executeQuery();
            if(rs.next()){
                vagaCurso.setCodVagaCurso(codVagaCurso);

vagaCurso.setCodCurso(Integer.valueOf(rs.getInt("cursos_codcur
so")));

vagaCurso.setCodVaga(Integer.valueOf(rs.getInt("vagas_codvaga
")));
            } else {
                return null;
            }
        } catch (SQLException e) {
            throw e;
        } finally {
            closeConnection(conn, stmt, rs);
        }
        return vagaCurso;
    }
}
```

6.4.292 package estagio.form

6.4.293 AlterarSenhaForm.java

```
/*
 * AlterarSenhaForm.java
 * Created on 12 de Junho de 2006, 21:48
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class AlterarSenhaForm extends ValidatorForm{

    private String senha;
    private String senha2;
    private String codAlunoEstagio;


    public void reset(ActionMapping mapping, HttpServletRequest
request) {

setCodAlunoEstagio(request.getParameter("codalunoestagio"));
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
```

```
    if (!senha.equals(senha2)) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionError("prompt.confirmacaosenha"));
    }
    return errors;
  }

  public String getSenha() {
    return senha;
  }

  public void setSenha(String senha) {
    this.senha = senha;
  }

  public String getSenha2() {
    return senha2;
  }

  public void setSenha2(String senha2) {
    this.senha2 = senha2;
  }

  public String getCodAlunoEstagio() {
    return codAlunoEstagio;
  }

  public void setCodAlunoEstagio(String codAlunoEstagio) {
    this.codAlunoEstagio = codAlunoEstagio;
  }

}
```

6.4.294 BuscarEmpresaForm.java

```
/*
 * BuscarEmpresaForm.java
 * Created on 22 de Março de 2006, 20:48
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class BuscarEmpresaForm extends ValidatorForm{

  private String status;
  private String cnpj;


  public void reset(ActionMapping mapping, HttpServletRequest
request) {
    setStatus("empresa");
  }

  public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
    ActionErrors errors = super.validate(mapping, request);
    return errors;
  }

  public String getStatus() {
    return status;
  }

  public void setStatus(String status) {
    this.status = status;
  }

  public String getCnpj() {
    return cnpj;
  }

  public void setCnpj(String cnpj) {
    this.cnpj = cnpj;
  }
}
```

6.4.295 BuscarEstagiarioForm.java

```
/*
 * BuscarEstagiarioForm.java
 * Created on 2 de Março de 2006, 09:13
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class BuscarEstagiarioForm extends ValidatorForm{

  private Integer codVaga;
  private String nomeEstagiario;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
    setCodVaga(request.getParameter("codvaga")!= null ?
Integer.valueOf(request.getParameter("codvaga")):
Integer.valueOf(0));
  }

  public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
    ActionErrors errors = super.validate(mapping, request);
    return errors;
  }

  public Integer getCodVaga() {
    return codVaga;
  }

  public void setCodVaga(Integer codVaga) {
    this.codVaga = codVaga;
  }

  public String getNomeEstagiario() {
    return nomeEstagiario;
  }

  public void setNomeEstagiario(String nomeEstagiario) {
    this.nomeEstagiario = nomeEstagiario;
  }
}
```

6.4.296 BuscarUsuarioForm.java

```
/*
 * BuscarUsuarioForm.java
 * Created on 5 de Março de 2006, 15:59
 */
```

```
package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class BuscarUsuarioForm extends ValidatorForm{

   private String nomeUsuario;

   public void reset(ActionMapping mapping, HttpServletRequest
request) {

   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public String getNomeUsuario() {
      return nomeUsuario;
   }

   public void setNomeUsuario(String nomeUsuario) {
      this.nomeUsuario = nomeUsuario;
   }

}
```

6.4.297 ConsultarCursosVagasForm.java

```
/*
 * ConsultarCursosVagasForm.java
 * Created on 2 de Março de 2006, 21:25
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class ConsultarCursosVagasForm extends ValidatorForm{

   private Integer codCurso;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
   }


   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public Integer getCodCurso() {
      return codCurso;
   }

   public void setCodCurso(Integer codCurso) {
      this.codCurso = codCurso;
   }
}
```

6.4.298 DadosPessoaisForm.java

```
/*
 * DadosPessoaisForm.java
 * Created on 26 de Janeiro de 2006, 19:01
 */

package estagio.form;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.bean.Pessoa;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.PessoaDao;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 * @author Ricardo
 */
public class DadosPessoaisForm extends ValidatorForm  {

   private String action;
   //dados de Pessoa
   private String nome_pessoa;
   private String dataNascimento;
   private String usuario;
   private String senha;
   private String senha2;
   private String status;
   //dados de alunoEstagio
   private String sexo;
   private String email;
   private String fone;
   private String disponivelFds;
   private String fumante;
   private String deficiente;
   private String locomocao;
   private String empresa;
   private Integer codCurso;
   private Integer semestre;
   private String rg;
   private String rg_naturalidade;
   private String cpf;
   private String matricula;
   private String cep;
   private String rua;
   private String numero;
   private String complemento;
   private String bairro;
   private String municipio;
```

```
  private String estado;
  private String pais;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
    setAction("Create");
    setStatus(request.getParameter("status"));
    setEmpresa((String) request.getAttribute("cnpj"));
    HttpSession session = request.getSession();
    String logado = (String) session.getAttribute("logado");
    if(logado != null){
      if ("true".equals(logado)){
        String codPessoa = (String)
session.getAttribute("codPessoa");
        Pessoa pessoa = null;
        try {
          PessoaDao pessoaDao = new PessoaDao();
          pessoa =
pessoaDao.buscarPessoa(Integer.valueOf(codPessoa));
        } catch (SQLException e){
          e.printStackTrace();
        } catch (Exception e){
          e.printStackTrace();
        }
        if(pessoa != null){
          SimpleDateFormat sdf = new
SimpleDateFormat("dd/MM/yyyy");
          setUsuario(pessoa.getUsuario());
          setNome_pessoa(pessoa.getNomePessoa());
          setSenha(pessoa.getSenha());

setDataNascimento(sdf.format(pessoa.getDataNascimento().getTi
me()));
        }
      }
    }
  }

  public ActionErrors validate(ActionMapping mapping,
HttpServletRequest request) {
    // Only need crossfield validations here
    // Perform validator framework validations
    ActionErrors errors = super.validate(mapping, request);
    HttpSession session = request.getSession();
    AlunoEstagio user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
    CursoDao cursoDao = new CursoDao();
    List<Curso> cursos = new ArrayList<Curso>();
    List<LabelValueBean> selectCursos = new
ArrayList<LabelValueBean>();
    try{
      cursos = cursoDao.listarCursos();
    } catch (SQLException e){
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
    }catch (Exception e) {
      errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
    }
    for(Curso c: cursos){
      selectCursos.add(new
LabelValueBean(c.getNomeCurso(),
c.getCodCurso().toString()));
    }
    if(cursos.size()>0){
      request.setAttribute("cursos", selectCursos);
    } else {
      request.setAttribute("cursos", new
LabelValueBean("0","Não Informado"));
    }
    if("empresa".equals(status)){
      List<Empresa> empresas = new ArrayList<Empresa>();
      try {
        EmpresaDao empresaDao = new EmpresaDao();
        empresas = empresaDao.listarEmpresas();
      } catch (SQLException e){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      } catch (Exception e) {
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.empresa"));
      }
      request.setAttribute("empresas", empresas );
    }
    ArrayList sexos = new ArrayList();
    sexos.add(new LabelValueBean("masculino", "M"));
    sexos.add(new LabelValueBean("feminino", "F"));
    request.setAttribute("sexos", sexos);
    ArrayList simNao = new ArrayList();
    simNao.add(new LabelValueBean("Sim", "S"));
    simNao.add(new LabelValueBean("Não", "N"));
    request.setAttribute("simNao", simNao);
    ArrayList locomotion = new ArrayList();
    locomotion.add(new LabelValueBean("Carro", "C"));
    locomotion.add(new LabelValueBean("Moto", "M"));
    locomotion.add(new LabelValueBean("Não Possui", "N"));
    request.setAttribute("locomocao", locomotion);
    request.setAttribute("usuario", user !=
null?user.getUsuario():"guest");
    String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
    request.setAttribute("estados", estados);
    Integer[] semestres = new Integer[12];
    for(int i=0; i<12; i++){
      if(i == 0) continue;
      semestres[i] = Integer.valueOf(i);
    }
    request.setAttribute("semestres", semestres);
    request.setAttribute("status", status);
    if("empresa".equals(getStatus())){
      if(getEmpresa() == null || "".equals(getEmpresa())){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionError("prompt.empresa.required"));
      }
    }
    String logado = (String) session.getAttribute("logado");
    if("Create".equals(getAction()) && logado == null){
      if(getSenha() == null || "".equals(getSenha())){
        errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionError("prompt.senha.required"));
      }
      if(senha != null){
        if (!senha.equals(senha2)) {
          errors.add(ActionMessages.GLOBAL_MESSAGE,
new ActionError("prompt.confirmacaosenha"));
        }
      }

    }
    return errors;
  }

  public String getAction() {
    return action;
  }

  public void setAction(String action) {
    this.action = action;
  }

  public String getNome_pessoa() {
    return nome_pessoa;
```

```
}

public void setNome_pessoa(String nome_pessoa) {
  this.nome_pessoa = nome_pessoa;
}

public String getDataNascimento() {
  return dataNascimento;
}

public void setDataNascimento(String dataNascimento) {
  this.dataNascimento = dataNascimento;
}

public String getUsuario() {
  return usuario;
}

public void setUsuario(String usuario) {
  this.usuario = usuario;
}

public String getSenha() {
  return senha;
}

public void setSenha(String senha) {
  this.senha = senha;
}

public String getSenha2() {
  return senha2;
}

public void setSenha2(String senha2) {
  this.senha2 = senha2;
}

public String getSexo() {
  return sexo;
}

public void setSexo(String sexo) {
  this.sexo = sexo;
}

public String getEmail() {
  return email;
}

public void setEmail(String email) {
  this.email = email;
}

public String getFone() {
  return fone;
}

public void setFone(String fone) {
  this.fone = fone;
}

public String getDisponivelFds() {
  return disponivelFds;
}

public void setDisponivelFds(String disponivelFds) {
  this.disponivelFds = disponivelFds;
}

public String getFumante() {
  return fumante;
}

}

public void setFumante(String fumante) {
  this.fumante = fumante;
}

public String getDeficiente() {
  return deficiente;
}

public void setDeficiente(String deficiente) {
  this.deficiente = deficiente;
}

public String getLocomocao() {
  return locomocao;
}

public void setLocomocao(String locomocao) {
  this.locomocao = locomocao;
}

public String getStatus() {
  return status;
}

public void setStatus(String status) {
  this.status = status;
}

public String getEmpresa() {
  return empresa;
}

public void setEmpresa(String empresa) {
  this.empresa = empresa;
}

public Integer getCodCurso() {
  return codCurso;
}

public void setCodCurso(Integer codCurso) {
  this.codCurso = codCurso;
}

public Integer getSemestre() {
  return semestre;
}

public void setSemestre(Integer semestre) {
  this.semestre = semestre;
}

public String getRg() {
  return rg;
}

public void setRg(String rg) {
  this.rg = rg;
}

public String getRg_naturalidade() {
  return rg_naturalidade;
}

public void setRg_naturalidade(String rg_naturalidade) {
  this.rg_naturalidade = rg_naturalidade;
}

public String getCpf() {
  return cpf;
```

```
}

public void setCpf(String cpf) {
   this.cpf = cpf;
}

public String getMatricula() {
   return matricula;
}

public void setMatricula(String matricula) {
   this.matricula = matricula;
}

public String getCep() {
   return cep;
}

public void setCep(String cep) {
   this.cep = cep;
}

public String getRua() {
   return rua;
}

public void setRua(String rua) {
   this.rua = rua;
}

public String getNumero() {
   return numero;
}

public void setNumero(String numero) {
   this.numero = numero;
}

public String getComplemento() {
   return complemento;
}

public void setComplemento(String complemento) {
   this.complemento = complemento;
}

public String getBairro() {
   return bairro;
}

public void setBairro(String bairro) {
   this.bairro = bairro;
}

public String getMunicipio() {
   return municipio;
}

public void setMunicipio(String municipio) {
   this.municipio = municipio;
}

public String getEstado() {
   return estado;
}

public void setEstado(String estado) {
   this.estado = estado;
}

public String getPais() {
   return pais;
   }

   public void setPais(String pais) {
      this.pais = pais;
   }

}
```

6.4.299 InformaUserNameForm.java

```
/*
 * InformaUserNameForm.java
 * Created on 15 de Maio de 2006, 14:38
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class InformaUserNameForm extends ValidatorForm{

   private String username;
   private String email;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
   }


   /**
    * Ensure that both fields have been input.
    *
    * @param mapping The mapping used to select this instance
    * @param request The servlet request we are processing
    */
   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public String getUsername() {
      return username;
   }

   public void setUsername(String username) {
      this.username = username;
   }

   public String getEmail() {
      return email;
   }

   public void setEmail(String email) {
      this.email = email;
   }
}
```

6.4.300 LogonForm.java

```
/*
 * LogonForm.java
 * Created on 8 de Novembro de 2005, 15:55
 */
```

```
package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;


/**
 *
 *
 */
public class LogonForm extends ValidatorForm{

   private String username;
   private String password;

   public String getUsername() {
      return username;
   }

   public void setUsername(String username) {
      this.username = username;
   }

   public String getPassword() {
      return password;
   }

   public void setPassword(String password) {
      this.password = password;
   }

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
      setPassword("");
      setUsername("");
   }


   /**
    * Ensure that both fields have been input.
    *
    * @param mapping The mapping used to select this instance
    * @param request The servlet request we are processing
    */
   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }
}
```

6.4.301 NovaEmpresaForm.java

```
/*
 * NovaEmpresaForm.java
 * Created on 19 de Janeiro de 2006, 10:35
 */

package estagio.form;

import estagio.bean.Empresa;
import estagio.dao.EmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
import org.apache.struts.validator.ValidatorActionForm;


/**
 *
 *
 */
public class NovaEmpresaForm extends ValidatorActionForm{

   private String action;
   private String cnpj;
   private String cargoRepresentante;
   private String representante;
   private String email;
   private String fone;
   private String razaoSocial;
   private String nomeFantasia;
   private String ramo;
   private String site;
   private String bairro;
   private String cep;
   private String rua;
   private String complemento;
   private String estado;
   private String municipio;
   private Integer numero;
   private String pais;
   private String fax;
   private String dataInicio;
   private String dataFim;
   private String numProcesso;
   private String agenteIntegrador;

   public void reset(ActionMapping mapping, HttpServletRequest
request) {
      setAction("Create");
   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      //parametros para o caso de falha na validacao
      String[] estados =
{"AC","AL","AP","AM","BA","CE","DF","ES","GO","MA","M
T","MS","MG","PA","PB","PR","PE","PI","RJ","RN","RS","RO
","RR","SC","SP","SE","TO"};
      List<Empresa> empresas = null;
      try{
         EmpresaDao empresaDao = new EmpresaDao();
         empresas = empresaDao.listarEmpresasSemConvenio();

empresas.addAll(empresaDao.listarEmpresasConvenioVencido())
;
      }catch (SQLException e){
         errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
      }catch (Exception e) {
         errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.empresa"));
      }
      if(empresas.size()>0){
         ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
         for(Empresa e: empresas){
            options.add(new LabelValueBean(e.getRazaoSocial(),
e.getCnpj()));
         }
```

```
            request.setAttribute("empresas", options);
        }
        request.setAttribute("estados", estados);
        //fim dos parametros de falha na validacao
        return errors;
    }

    public String getAction() {
        return action;
    }

    public void setAction(String action) {
        this.action = action;
    }

    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

    public String getCargoRepresentante() {
        return cargoRepresentante;
    }

    public void setCargoRepresentante(String cargoRepresentante)
{
        this.cargoRepresentante = cargoRepresentante;
    }

    public String getRepresentante() {
        return representante;
    }

    public void setRepresentante(String representante) {
        this.representante = representante;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getFone() {
        return fone;
    }

    public void setFone(String fone) {
        this.fone = fone;
    }

    public String getRazaoSocial() {
        return razaoSocial;
    }

    public void setRazaoSocial(String razaoSocial) {
        this.razaoSocial = razaoSocial;
    }

    public String getNomeFantasia() {
        return nomeFantasia;
    }

    public void setNomeFantasia(String nomeFantasia) {
        this.nomeFantasia = nomeFantasia;
    }

    public String getRamo() {
        return ramo;
    }

    public void setRamo(String ramo) {
        this.ramo = ramo;
    }

    public String getSite() {
        return site;
    }

    public void setSite(String site) {
        this.site = site;
    }

    public String getBairro() {
        return bairro;
    }

    public void setBairro(String bairro) {
        this.bairro = bairro;
    }

    public String getCep() {
        return cep;
    }

    public void setCep(String cep) {
        this.cep = cep;
    }

    public String getRua() {
        return rua;
    }

    public void setRua(String rua) {
        this.rua = rua;
    }

    public String getComplemento() {
        return complemento;
    }

    public void setComplemento(String complemento) {
        this.complemento = complemento;
    }

    public String getEstado() {
        return estado;
    }

    public void setEstado(String estado) {
        this.estado = estado;
    }

    public String getMunicipio() {
        return municipio;
    }

    public void setMunicipio(String municipio) {
        this.municipio = municipio;
    }

    public Integer getNumero() {
        return numero;
    }

    public void setNumero(Integer numero) {
        this.numero = numero;
    }
```

```
    public String getPais() {
        return pais;
    }

    public void setPais(String pais) {
        this.pais = pais;
    }

    public String getFax() {
        return fax;
    }

    public void setFax(String fax) {
        this.fax = fax;
    }

    public String getDataInicio() {
        return dataInicio;
    }

    public void setDataInicio(String dataInicio) {
        this.dataInicio = dataInicio;
    }

    public String getDataFim() {
        return dataFim;
    }

    public void setDataFim(String dataFim) {
        this.dataFim = dataFim;
    }

    public String getNumProcesso() {
        return numProcesso;
    }

    public void setNumProcesso(String numProcesso) {
        this.numProcesso = numProcesso;
    }

    public String getAgenteIntegrador() {
        return agenteIntegrador;
    }

    public void setAgenteIntegrador(String agenteIntegrador) {
        this.agenteIntegrador = agenteIntegrador;
    }

}
```

6.4.302 NovaHabilidadeAlunoForm.java

```
/*
 * NovaHabilidadeAlunoForm.java
 * Created on 9 de Fevereiro de 2006, 17:01
 */

package estagio.form;

import java.util.List;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovaHabilidadeAlunoForm extends ValidatorForm{

    private Integer codAlunoEstagio;
    private List habilidades;
    private Integer[] codHabilidade;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {

setCodAlunoEstagio(request.getParameter("codAlunoEstagio")!=
null?Integer.valueOf(request.getParameter("codAlunoEstagio")):I
nteger.valueOf(0));
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        return errors;
    }

    public Integer getCodAlunoEstagio() {
        return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
        this.codAlunoEstagio = codAlunoEstagio;
    }

    public Integer[] getCodHabilidade() {
        return codHabilidade;
    }

    public void setCodHabilidade(Integer[] codHabilidade) {
        this.codHabilidade = codHabilidade;
    }

    public List getHabilidades() {
        return habilidades;
    }

    public void setHabilidades(List habilidades) {
        this.habilidades = habilidades;
    }
}
```

6.4.303 NovaHabilidadeForm.java

```
/*
 * NovaHabilidadeForm.java
 * Created on 9 de Fevereiro de 2006, 12:04
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovaHabilidadeForm extends ValidatorForm{

    private Integer codVaga;
    private Integer codAlunoEstagio;
    private Integer codHabilidade;
    private Integer codHabilidadeVaga;
    private String descricaoHabilidade;


    public void reset(ActionMapping mapping, HttpServletRequest
request) {
```

```
setCodVaga(request.getParameter("codvaga")!=null?Integer.value
Of(request.getParameter("codvaga")):Integer.valueOf(0));

setCodAlunoEstagio(request.getParameter("codalunoestagio")!=n
ull?Integer.valueOf(request.getParameter("codalunoestagio")):Int
eger.valueOf(0));
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        return errors;
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public Integer getCodHabilidade() {
        return codHabilidade;
    }

    public void setCodHabilidade(Integer codHabilidade) {
        this.codHabilidade = codHabilidade;
    }

    public Integer getCodHabilidadeVaga() {
        return codHabilidadeVaga;
    }

    public void setCodHabilidadeVaga(Integer
codHabilidadeVaga) {
        this.codHabilidadeVaga = codHabilidadeVaga;
    }

    public String getDescricaoHabilidade() {
        return descricaoHabilidade;
    }

    public void setDescricaoHabilidade(String
descricaoHabilidade) {
        this.descricaoHabilidade = descricaoHabilidade;
    }

    public Integer getCodAlunoEstagio() {
        return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
        this.codAlunoEstagio = codAlunoEstagio;
    }
}
```

6.4.304 NovaHabilidadeVagaForm.java

```
/*
 * NovaHabilidadeVagaForm.java
 * Created on 8 de Fevereiro de 2006, 23:10
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovaHabilidadeVagaForm extends ValidatorForm{

    private Integer codVaga;
    private Integer[] codHabilidade;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {

setCodVaga(request.getParameter("codvaga")!=null?Integer.value
Of(request.getParameter("codvaga")):Integer.valueOf(0));
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        return errors;
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public Integer[] getCodHabilidade() {
        return codHabilidade;
    }

    public void setCodHabilidade(Integer[] codHabilidade) {
        this.codHabilidade = codHabilidade;
    }
}
```

6.4.305 NovaVagaEstagioInternoForm.java

```
/*
 * NovaVagaEstagioInternoForm.java
 * Created on 27 de Abril de 2006, 11:25
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovaVagaEstagioInternoForm extends
ValidatorForm{

    private String cnpj;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {
        setCnpj((String) request.getAttribute("cnpj"));
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        return errors;
    }
```

```
    public String getCnpj() {
        return cnpj;
    }

    public void setCnpj(String cnpj) {
        this.cnpj = cnpj;
    }

}
```

6.4.306 NovaVagaForm.java

```
/*
 * NovaVagaForm.java
 * Created on 3 de Fevereiro de 2006, 10:51
 */

package estagio.form;

import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Curso;
import estagio.bean.Empresa;
import estagio.dao.CursoDao;
import estagio.dao.EmpresaDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
import org.apache.struts.validator.ValidatorForm;


/**
 *
 *
 */
public class NovaVagaForm extends ValidatorForm{

    private String action;
    private String codVaga;
    private String cnpj;
    private String setor;
    private String horas;
    private String salario;
    private String sexo;
    private String locomocao;
    private String deficiente;
    private String atividadesFds;
    private String fumante;
    private String[] codCurso;
    private Integer semestre;
    private String valeTransporte;
    private String outrosBenefícios;
    private String estagio;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {
        setAction("Create");
        setCnpj(request.getParameter("cnpj"));
        codCurso = new String[0];
        setEstagio((String) request.getAttribute("estagio"));
        }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        if(!errors.isEmpty()){
            ArrayList<LabelValueBean> options = new
ArrayList<LabelValueBean>();
            options.add(new LabelValueBean("Masculino", "M"));
            options.add(new LabelValueBean("Feminino", "F"));
            options.add(new LabelValueBean("Qualquer", "A"));
            request.setAttribute("sexos", options);
            ArrayList simNaoAll = new ArrayList();
            simNaoAll.add(new LabelValueBean("Sim", "S"));
            simNaoAll.add(new LabelValueBean("Não", "N"));
            simNaoAll.add(new LabelValueBean("Não Importa",
"A"));
            request.setAttribute("simNaoAll", simNaoAll);
            ArrayList simNao = new ArrayList();
            simNao.add(new LabelValueBean("Sim", "S"));
            simNao.add(new LabelValueBean("Não", "N"));
            request.setAttribute("simNao", simNao);
            ArrayList locomotion = new ArrayList();
            locomotion.add(new LabelValueBean("Carro", "C"));
            locomotion.add(new LabelValueBean("Moto", "M"));
            locomotion.add(new LabelValueBean("Não Importa",
"A"));
            request.setAttribute("locomocao", locomotion);
            AlunoEstagio user = null;
            HttpSession session = request.getSession();
            try{
                user = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
            } catch (Exception e) {
                errors.add(ActionMessages.GLOBAL_MESSAGE,
                        new ActionMessage("error.generico"));
            }
            String empresa = "";
            String cnpj = "";
            Empresa emp = new Empresa();
            if(user.getStatus() != null){
                if("empresa".equals(user.getStatus())){
                    try{
                        UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
                        emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(user.getCodAlunoEst
agio());
                    } catch (Exception e){

                    }
                    if(emp != null){
                        empresa = emp.getRazaoSocial();
                        cnpj = emp.getCnpj();
                    }
                }
            }
            request.setAttribute("empresa", empresa);
            request.setAttribute("cnpj", cnpj);
            //lista os cursos
            List<Curso> cursos = new ArrayList<Curso>();
            try{
                CursoDao cursoDao = new CursoDao();
                cursos = cursoDao.listarCursos();
            } catch (SQLException e){
                errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.listar.cursos"));
            }catch (Exception e) {
                errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.cursos"));
            }
            if(cursos.size()>0){
                request.setAttribute("cursos", cursos);
            }
            //cursos listados
        }
```

```
    return errors;
}

public String getCodVaga() {
    return codVaga;
}

public void setCodVaga(String codVaga) {
    this.codVaga = codVaga;
}

public String getCnpj() {
    return cnpj;
}

public void setCnpj(String cnpj) {
    this.cnpj = cnpj;
}

public String getSetor() {
    return setor;
}

public void setSetor(String setor) {
    this.setor = setor;
}

public String getHoras() {
    return horas;
}

public void setHoras(String horas) {
    this.horas = horas;
}

public String getSalario() {
    return salario;
}

public void setSalario(String salario) {
    this.salario = salario;
}

public String getSexo() {
    return sexo;
}

public void setSexo(String sexo) {
    this.sexo = sexo;
}

public String getLocomocao() {
    return locomocao;
}

public void setLocomocao(String locomocao) {
    this.locomocao = locomocao;
}

public String getDeficiente() {
    return deficiente;
}

public void setDeficiente(String deficiente) {
    this.deficiente = deficiente;
}

public String getAtividadesFds() {
    return atividadesFds;
}

public void setAtividadesFds(String atividadesFds) {
    this.atividadesFds = atividadesFds;
}

public String getFumante() {
    return fumante;
}

public void setFumante(String fumante) {
    this.fumante = fumante;
}

public String getAction() {
    return action;
}

public void setAction(String action) {
    this.action = action;
}

public String[] getCodCurso() {
    return codCurso;
}

public void setCodCurso(String[] codCurso) {
    this.codCurso = codCurso;
}

public Integer getSemestre() {
    return semestre;
}

public void setSemestre(Integer semestre) {
    this.semestre = semestre;
}

public String getValeTransporte() {
    return valeTransporte;
}

public void setValeTransporte(String valeTransporte) {
    this.valeTransporte = valeTransporte;
}

public String getOutrosBenefícios() {
    return outrosBenefícios;
}

public void setOutrosBenefícios(String outrosBenefícios) {
    this.outrosBenefícios = outrosBenefícios;
}

public String getEstagio() {
    return estagio;
}

public void setEstagio(String estagio) {
    this.estagio = estagio;
}
}
```

6.4.307 NovoAgenteIntegracaoForm.java

```
/*
 * NovoAgenteIntegracaoForm.java
 * Created on 27 de Fevereiro de 2006, 12:02
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
```

```
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovoAgenteIntegracaoForm extends ValidatorForm{

   private String agente;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
   }


   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public String getAgente() {
      return agente;
   }

   public void setAgente(String agente) {
      this.agente = agente;
   }

}
```

6.4.308 NovoConvenioForm.java

```
/*
 * NovoConvenioForm.java
 * Created on 18 de Janeiro de 2006, 19:46
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovoConvenioForm extends ValidatorForm{

   private String action;
   private Integer codConvenio;
   private String descricao;
   private String dataInicio;
   private String dataFim;
   private String cnpj;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
      setAction("Create");
      setCodConvenio(null);
      setDataFim("");
      setDataInicio("");
      setDescricao("");
      setCnpj("");
   }


   /**
    * Ensure that both fields have been input.
    *
    * @param mapping The mapping used to select this instance
    * @param request The servlet request we are processing
    */
   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);

     /*if ((password == null) || (password.length() < 1))
        errors.add("password",
           new ActionError("error.password.required"));*/

      return errors;
   }

   public String getAction() {
      return action;
   }

   public void setAction(String action) {
      this.action = action;
   }

   public Integer getCodConvenio() {
      return codConvenio;
   }

   public void setCodConvenio(Integer codConvenio) {
      this.codConvenio = codConvenio;
   }

   public String getDescricao() {
      return descricao;
   }

   public void setDescricao(String descricao) {
      this.descricao = descricao;
   }

   public String getDataInicio() {
      return dataInicio;
   }

   public void setDataInicio(String dataInicio) {
      this.dataInicio = dataInicio;
   }

   public String getDataFim() {
      return dataFim;
   }

   public void setDataFim(String dataFim) {
      this.dataFim = dataFim;
   }

   public String getCnpj() {
      return cnpj;
   }

   public void setCnpj(String cnpj) {
      this.cnpj = cnpj;
   }
}
```

6.4.309 NovoCursoForm.java

```
/*
 * NovoCursoForm.java
 * Created on 2 de Março de 2006, 19:16
 */
```

```
package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovoCursoForm extends ValidatorForm{

   private String nomeCurso;
   private Integer duracao;

  public void reset(ActionMapping mapping, HttpServletRequest
request) {
   }


   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);
      if(duracao < 2){
         errors.add("duracao", new
ActionMessage("error.duracao.invalida"));
      }
      return errors;
   }

   public String getNomeCurso() {
      return nomeCurso;
   }

   public void setNomeCurso(String nomeCurso) {
      this.nomeCurso = nomeCurso;
   }

   public Integer getDuracao() {
      return duracao;
   }

   public void setDuracao(Integer duracao) {
      this.duracao = duracao;
   }
}
```

6.4.310 NovoTermoCompromissoForm.java

```
/*
 * NovoTermoCompromissoForm.java
 * Created on 21 de Fevereiro de 2006, 17:01
 */

package estagio.form;

import estagio.bean.AgenteIntegracao;
import estagio.bean.AlunoEstagio;
import estagio.bean.Constants;
import estagio.bean.Empresa;
import estagio.dao.AgentesIntegracaoDao;
import estagio.dao.UsuarioEmpresaDao;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.action.ActionMessages;
import org.apache.struts.util.LabelValueBean;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovoTermoCompromissoForm extends
ValidatorForm{

   private Integer codAlunoEstagio;
   private Integer codVaga;
   private Integer codTermo;
   private String dataInicio;
   private String dataFim;
   private String cnpj;
   private String canal;
   private String action;
   private String obrigatorio;
   private String orientador;
   private String formacaoOrientador;
   private String areaAtividadeEstagiario;
   private String objetivos;
   private String atividades;
   private String seguroCompanhia;
   private Long seguroNumApolice;
   private String seguroProvidenciado;
   private String estagio;
   private String cargoOrientador;
   private String ramal;
   private String departamento;

   public void reset(ActionMapping mapping, HttpServletRequest
request) {

setCodAlunoEstagio(request.getParameter("codAlunoEstagio") !=
null ? Integer.valueOf(request.getParameter("codAlunoEstagio")):
Integer.valueOf(0));
      setCodVaga(request.getParameter("codVaga")!= null ?
Integer.valueOf(request.getParameter("codVaga")):
Integer.valueOf(0));
      setCnpj(request.getAttribute("cnpj") != null ?
request.getAttribute("cnpj").toString(): "");
      setAction("Create");
      setCodTermo(request.getParameter("codTermo")!= null ?
Integer.valueOf(request.getParameter("codTermo")):
Integer.valueOf(0));
      setEstagio(request.getAttribute("estagio") != null ?(String)
request.getAttribute("estagio"):"");
   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      if(!"true".equals(getObrigatorio())){
         if(getOrientador() == null || "".equals(getOrientador())){
            errors.add("orientador", new
ActionMessage("errors.orientador.required"));
         }
         if(getFormacaoOrientador() == null ||
"".equals(getFormacaoOrientador())){
            errors.add("formacaoOrientador", new
ActionMessage("errors.formacaoOrientador.required"));
         }
         if(getAreaAtividadeEstagiario() == null ||
"".equals(getAreaAtividadeEstagiario())){
```

```
            errors.add("areaAtividadeEstagiario", new
ActionMessage("errors.areaAtividadeEstagiario.required"));
        }
        if(getAtividades() == null || "".equals(getAtividades())){
            errors.add("atividades", new
ActionMessage("errors.atividades.required"));
        }
        if(getObjetivos() == null || "".equals(getObjetivos())){
            errors.add("objetivos", new
ActionMessage("errors.objetivos.required"));
        }
      }
      if("interno".equals(getEstagio())){
        if(getCargoOrientador() == null ||
"".equals(getCargoOrientador())){
            errors.add("cargoOrientador", new
ActionMessage("errors.cargoOrientador.required"));
        }
        if(getRamal() == null || "".equals(getRamal())){
            errors.add("ramal", new
ActionMessage("errors.ramal.required"));
        }
        if(getDepartamento() == null ||
"".equals(getDepartamento())){
            errors.add("departamento", new
ActionMessage("errors.departamento.required"));
        }
      }
      if(!errors.isEmpty()){
        //seta parametros para NovoTermoCompromisso
        HttpSession session = request.getSession();
        AlunoEstagio userCheck = (AlunoEstagio)
session.getAttribute(Constants.USER_KEY);
        String empresa = "";
        String cnpj = "";
        Empresa emp = new Empresa();
        if(userCheck.getStatus() != null){
          if("empresa".equals(userCheck.getStatus())){
            try{
              UsuarioEmpresaDao usuarioEmpresaDao = new
UsuarioEmpresaDao();
              emp =
usuarioEmpresaDao.buscarUsuarioEmpresa(userCheck.getCodAl
unoEstagio());
            } catch (SQLException e){

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("sqlerror.buscar.usuarios"));
            }catch (Exception e) {

errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.buscar.usuarios"));
            }
            if(emp != null){
              empresa = emp.getRazaoSocial();
              cnpj = emp.getCnpj();
            }
          }
        }
        request.setAttribute("empresa", empresa);
        request.setAttribute("cnpj", cnpj);
        List<LabelValueBean> selectAgentes = new
ArrayList<LabelValueBean>();
        List<AgenteIntegracao> agentes = new
ArrayList<AgenteIntegracao>();
        try{
          AgentesIntegracaoDao agentesIntegracaoDao = new
AgentesIntegracaoDao();
          agentes =
agentesIntegracaoDao.listarAgentesIntegracao();
        } catch (SQLException e){
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.banco"));
        }catch (Exception e) {
            errors.add(ActionMessages.GLOBAL_MESSAGE, new
ActionMessage("error.listar.agente"));
        }
        for(AgenteIntegracao agente: agentes){
          selectAgentes.add(new
LabelValueBean(agente.getAgente(), agente.getCnpj()));
        }
        if(selectAgentes.size()>0){
          request.setAttribute("selectAgentes", selectAgentes);
        } else {
          request.setAttribute("selectAgentes", new
LabelValueBean("Nao informado", "0"));
        }
        request.setAttribute("codvaga", getCodVaga());
        //setados parametros para NovaTermoCompromisso
      }
      return errors;
    }

    public Integer getCodAlunoEstagio() {
      return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
      this.codAlunoEstagio = codAlunoEstagio;
    }

    public Integer getCodVaga() {
      return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
      this.codVaga = codVaga;
    }

    public String getDataInicio() {
      return dataInicio;
    }

    public void setDataInicio(String dataInicio) {
      this.dataInicio = dataInicio;
    }

    public String getDataFim() {
      return dataFim;
    }

    public void setDataFim(String dataFim) {
      this.dataFim = dataFim;
    }

    public String getCnpj() {
      return cnpj;
    }

    public void setCnpj(String cnpj) {
      this.cnpj = cnpj;
    }

    public String getCanal() {
      return canal;
    }

    public void setCanal(String canal) {
      this.canal = canal;
    }

    public String getAction() {
      return action;
```

```
    }

    public void setAction(String action) {
        this.action = action;
    }

    public String getObrigatorio() {
        return obrigatorio;
    }

    public void setObrigatorio(String obrigatorio) {
        this.obrigatorio = obrigatorio;
    }

    public String getOrientador() {
        return orientador;
    }

    public void setOrientador(String orientador) {
        this.orientador = orientador;
    }

    public String getFormacaoOrientador() {
        return formacaoOrientador;
    }

    public void setFormacaoOrientador(String
formacaoOrientador) {
        this.formacaoOrientador = formacaoOrientador;
    }

    public String getAreaAtividadeEstagiario() {
        return areaAtividadeEstagiario;
    }

    public void setAreaAtividadeEstagiario(String
areaAtividadeEstagiario) {
        this.areaAtividadeEstagiario = areaAtividadeEstagiario;
    }

    public String getObjetivos() {
        return objetivos;
    }

    public void setObjetivos(String objetivos) {
        this.objetivos = objetivos;
    }

    public String getAtividades() {
        return atividades;
    }

    public void setAtividades(String atividades) {
        this.atividades = atividades;
    }

    public Integer getCodTermo() {
        return codTermo;
    }

    public void setCodTermo(Integer codTermo) {
        this.codTermo = codTermo;
    }

    public String getSeguroCompanhia() {
        return seguroCompanhia;
    }

    public void setSeguroCompanhia(String seguroCompanhia) {
        this.seguroCompanhia = seguroCompanhia;
    }

    public Long getSeguroNumApolice() {
        return seguroNumApolice;
    }

    public void setSeguroNumApolice(Long seguroNumApolice) {
        this.seguroNumApolice = seguroNumApolice;
    }

    public String getSeguroProvidenciado() {
        return seguroProvidenciado;
    }

    public void setSeguroProvidenciado(String
seguroProvidenciado) {
        this.seguroProvidenciado = seguroProvidenciado;
    }

    public String getEstagio() {
        return estagio;
    }

    public void setEstagio(String estagio) {
        this.estagio = estagio;
    }

    public String getCargoOrientador() {
        return cargoOrientador;
    }

    public void setCargoOrientador(String cargoOrientador) {
        this.cargoOrientador = cargoOrientador;
    }

    public String getRamal() {
        return ramal;
    }

    public void setRamal(String ramal) {
        this.ramal = ramal;
    }

    public String getDepartamento() {
        return departamento;
    }

    public void setDepartamento(String departamento) {
        this.departamento = departamento;
    }
}
```

6.4.311 NovoTurnoForm.java

```
/*
 * NovoTurnoForm.java
 * Created on 16 de Fevereiro de 2006, 17:23
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class NovoTurnoForm extends ValidatorForm{

    private Integer codVaga;
    private String horaInicio;
```

```
    private String horaFim;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {

setCodVaga(request.getParameter("codvaga")!=null?Integer.value
Of(request.getParameter("codvaga")):Integer.valueOf(0));
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        if(getHoraInicio() != null && getHoraInicio().length() > 0){
            if(':' != getHoraInicio().charAt(2)){
                errors.add("horaInicio", new
ActionMessage("error.horaInicio.invalida"));
            } else {
                try{
                    Integer horasInicio =
Integer.valueOf(getHoraInicio().substring(0,2));
                    Integer minutosInicio =
Integer.valueOf(getHoraInicio().substring(3));
                    if(horasInicio > 24){
                        errors.add("horaInicio", new
ActionMessage("error.horaInicio.invalida"));
                    }
                    if(minutosInicio > 59){
                        errors.add("horaInicio", new
ActionMessage("error.horaInicio.invalida"));
                    }
                } catch (NumberFormatException numberExcpetion){
                    errors.add("horaInicio", new
ActionMessage("error.horaInicio.integer"));
                }
            }
        }
        if(getHoraFim() != null && getHoraFim().length() > 0){
            if(':' != getHoraFim().charAt(2)){
                errors.add("horaFim", new
ActionMessage("error.horaFim.invalida"));
            } else {
                try{
                    Integer horasFim =
Integer.valueOf(getHoraFim().substring(0,2));
                    Integer minutosFim =
Integer.valueOf(getHoraFim().substring(3));
                    if(horasFim > 24){
                        errors.add("horaFim", new
ActionMessage("error.horaFim.invalida"));
                    }
                    if(minutosFim > 59){
                        errors.add("horaFim", new
ActionMessage("error.horaFim.invalida"));
                    }
                } catch (NumberFormatException numberExcpetion){
                    errors.add("horaFim", new
ActionMessage("error.horaFim.integer"));
                }
            }
        }
        return errors;
    }

    public Integer getCodVaga() {
        return codVaga;
    }

    public void setCodVaga(Integer codVaga) {
        this.codVaga = codVaga;
    }

    public String getHoraInicio() {
        return horaInicio;
    }

    public void setHoraInicio(String horaInicio) {
        this.horaInicio = horaInicio;
    }

    public String getHoraFim() {
        return horaFim;
    }

    public void setHoraFim(String horaFim) {
        this.horaFim = horaFim;
    }
}
```

6.4.312 PesquisarConveniosForm.java

```
/*
 * PesquisarConveniosForm.java
 * Created on 23 de Março de 2006, 21:38
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class PesquisarConveniosForm  extends ValidatorForm{

    private String parametro;
    private String valor;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {
        setParametro("cnpj");
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        if("cnpj".equals(getParametro())){
            if(getValor() != null && !"".equals(getValor())){
                String xCGC = "";
                if (getValor() != null && getValor().length() > 0) {
                    xCGC= getValor();
                }
                try {
                    //Testa se o CGC é válido ou não
                    int d1,d4,xx,nCount,fator,resto,digito1,digito2;
                    String Check, s_aux;
                    String Separadores = "/-.";
                    d1 = 0;
                    d4 = 0;
                    xx = 0;
                    for (nCount = 0; nCount < xCGC.length()-2;
nCount++) {
                        s_aux = xCGC.substring(nCount, nCount+1);
                        if (Separadores.indexOf(s_aux) == -1) {
                            if (xx < 4) {
                                fator = 5 - xx;
                            } else {
                                fator = 13 - xx;
                            }
```

```
                d1 = d1 + Integer.valueOf(s_aux).intValue() *
fator;
                if (xx < 5) {
                  fator = 6 - xx;
                } else {
                  fator = 14 - xx;
                }
                d4 += Integer.valueOf(s_aux).intValue() * fator;
                xx++;
              };
            }
            resto = (d1 % 11);
            if (resto < 2) {
              digito1 = 0;
            } else {
              digito1 = 11 - resto;
            }

            d4 = d4 + 2 * digito1;
            resto = (d4 % 11);
            if (resto < 2) {
              digito2 = 0;
            } else {
              digito2 = 11 - resto;
            }

            Check = String.valueOf(digito1) +
String.valueOf(digito2);
            if
(Check.compareTo(xCGC.substring(xCGC.length()-2,
xCGC.length() )) !=0) {
                errors.add("cnpj", new
ActionMessage("errors.cnpj"));
            }
          } catch (Exception e) {
            errors.add("cnpj", new
ActionMessage("errors.cnpj"));
          }
        }
      }
      return errors;
    }

    public String getParametro() {
      return parametro;
    }

    public void setParametro(String parametro) {
      this.parametro = parametro;
    }

    public String getValor() {
      return valor;
    }

    public void setValor(String valor) {
      this.valor = valor;
    }

}
```

6.4.313 PesquisarEmpresasForm.java

```
/*
 * PesquisarEmpresasForm.java
 * Created on 23 de Março de 2006, 15:23
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class PesquisarEmpresasForm extends ValidatorForm{

    private String parteDoNome;


    public void reset(ActionMapping mapping, HttpServletRequest
request) {

    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      return errors;
    }

    public String getParteDoNome() {
      return parteDoNome;
    }

    public void setParteDoNome(String parteDoNome) {
      this.parteDoNome = parteDoNome;
    }

}
```

6.4.314 PesquisarHabilidadeAlunoForm.java

```
/*
 * PesquisarHabilidadeAlunoForm.java
 * Created on 24 de Março de 2006, 19:47
 */

package estagio.form;

import java.util.List;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;

/**
 *
 *
 */
public class PesquisarHabilidadeAlunoForm extends
ActionForm{

    private Integer codAlunoEstagio;
    private Integer[] codHabilidade;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {
      try {

setCodAlunoEstagio(Integer.valueOf(request.getParameter("codA
lunoEstagio")));
      }catch(NumberFormatException e){

      }
    }
```

```
    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        return errors;
    }

    public Integer getCodAlunoEstagio() {
        return codAlunoEstagio;
    }

    public void setCodAlunoEstagio(Integer codAlunoEstagio) {
        this.codAlunoEstagio = codAlunoEstagio;
    }

    public Integer[] getCodHabilidade() {
        return codHabilidade;
    }

    public void setCodHabilidade(Integer[] codHabilidade) {
        this.codHabilidade = codHabilidade;
    }
}
```

6.4.315 PesquisarTermosCompromissoForm.java

```
/*
 * PesquisarTermosCompromissoForm.java
 * Created on 23 de Março de 2006, 16:32
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class PesquisarTermosCompromissoForm extends
ValidatorForm{

    private String parametro;
    private String valor;

    public void reset(ActionMapping mapping, HttpServletRequest
request) {
        setParametro("cnpj");
    }

    public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
        ActionErrors errors = super.validate(mapping, request);
        if("cnpj".equals(getParametro())){
            String xCGC = "";
            if (getValor() != null && getValor().length() > 0) {
                xCGC= getValor();
            }
            try {
                //Testa se o CGC é válido ou não
                int d1,d4,xx,nCount,fator,resto,digito1,digito2;
                String Check, s_aux;
                String Separadores = "/-.";
                d1 = 0;
                d4 = 0;
                xx = 0;
                for (nCount = 0; nCount < xCGC.length()-2; nCount++)
{
                    s_aux = xCGC.substring(nCount, nCount+1);
                    if (Separadores.indexOf(s_aux) == -1) {
                        if (xx < 4) {
                            fator = 5 - xx;
                        } else {
                            fator = 13 - xx;
                        }

                        d1 = d1 + Integer.valueOf(s_aux).intValue() *
fator;
                        if (xx < 5) {
                            fator = 6 - xx;
                        } else {
                            fator = 14 - xx;
                        }
                        d4 += Integer.valueOf(s_aux).intValue() * fator;
                        xx++;
                    };
                }
                resto = (d1 % 11);
                if (resto < 2) {
                    digito1 = 0;
                } else {
                    digito1 = 11 - resto;
                }

                d4 = d4 + 2 * digito1;
                resto = (d4 % 11);
                if (resto < 2) {
                    digito2 = 0;
                } else {
                    digito2 = 11 - resto;
                }

                Check = String.valueOf(digito1) +
String.valueOf(digito2);
                if (Check.compareTo(xCGC.substring(xCGC.length()-
2, xCGC.length() )) !=0) {
                    errors.add("cnpj", new
ActionMessage("errors.cnpj"));
                }
            } catch (Exception e) {
                errors.add("cnpj", new ActionMessage("errors.cnpj"));
            }
        }
        return errors;
    }

    public String getParametro() {
        return parametro;
    }

    public void setParametro(String parametro) {
        this.parametro = parametro;
    }

    public String getValor() {
        return valor;
    }

    public void setValor(String valor) {
        this.valor = valor;
    }

}
```

6.4.316 SelecionarEstagiarioForm.java

```
/*
 * SelecionarEstagiarioForm.java
 * Created on 21 de Fevereiro de 2006, 16:34
```

```
*/

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;

/**
 *
 *
 */
public class SelecionarEstagiarioForm extends ActionForm{

   private Integer codAlunoEstagio;
   private Integer codVaga;


   public void reset(ActionMapping mapping, HttpServletRequest
request) {
      setCodVaga(request.getParameter("codVaga")!= null ?
Integer.valueOf(request.getParameter("codVaga")):
Integer.valueOf(0));

setCodAlunoEstagio((Integer)request.getAttribute("codAlunoEsta
gio"));
   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public Integer getCodAlunoEstagio() {
      return codAlunoEstagio;
   }

   public void setCodAlunoEstagio(Integer codAlunoEstagio) {
      this.codAlunoEstagio = codAlunoEstagio;
   }

   public Integer getCodVaga() {
      return codVaga;
   }

   public void setCodVaga(Integer codVaga) {
      this.codVaga = codVaga;
   }

}
```

6.4.317 SelecionarRelatorioForm.java

```
/*
 * SelecionarRelatorioForm.java
 * Created on 1 de Maio de 2006, 14:57
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;

/**
 *
 *
 */
public class SelecionarRelatorioForm extends ActionForm{

   private String relatorio;
   private String cnpj;
   private String mes;
   private String ano;
   private String estagio;

   public void reset(ActionMapping mapping, HttpServletRequest
request) {
      setCnpj(request.getParameter("cnpj"));
      setRelatorio("cancelar");
      setEstagio(request.getParameter("estagio"));
   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public String getRelatorio() {
      return relatorio;
   }

   public void setRelatorio(String relatorio) {
      this.relatorio = relatorio;
   }

   public String getCnpj() {
      return cnpj;
   }

   public void setCnpj(String cnpj) {
      this.cnpj = cnpj;
   }

   public String getMes() {
      return mes;
   }

   public void setMes(String mes) {
      this.mes = mes;
   }

   public String getAno() {
      return ano;
   }

   public void setAno(String ano) {
      this.ano = ano;
   }

   public String getEstagio() {
      return estagio;
   }

   public void setEstagio(String estagio) {
      this.estagio = estagio;
   }
}
```

6.4.318 ServidorSmtpForm.java

```
/*
 * ServidorSmtpForm.java
 * Created on 15 de Maio de 2006, 17:24
 */

package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
```

```
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class ServidorSmtpForm extends ValidatorForm{

   private Integer cod_servidor_smtp;
   private String url;
   private String username;
   private String senha;
   private Integer porta_smtp;
   private String endereco_email;
   private Boolean requer_autenticacao;

   public void reset(ActionMapping mapping, HttpServletRequest
request) {
   }


   /**
    * Ensure that both fields have been input.
    *
    * @param mapping The mapping used to select this instance
    * @param request The servlet request we are processing
    */
   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {

      // Perform validator framework validations
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public Integer getCod_servidor_smtp() {
      return cod_servidor_smtp;
   }

   public void setCod_servidor_smtp(Integer cod_servidor_smtp)
{
      this.cod_servidor_smtp = cod_servidor_smtp;
   }

   public String getUrl() {
      return url;
   }

   public void setUrl(String url) {
      this.url = url;
   }

   public String getUsername() {
      return username;
   }

   public void setUsername(String username) {
      this.username = username;
   }

   public String getSenha() {
      return senha;
   }

   public void setSenha(String senha) {
      this.senha = senha;
   }

   public Integer getPorta_smtp() {
      return porta_smtp;
   }

   public void setPorta_smtp(Integer porta_smtp) {
      this.porta_smtp = porta_smtp;
   }

   public String getEndereco_email() {
      return endereco_email;
   }

   public void setEndereco_email(String endereco_email) {
      this.endereco_email = endereco_email;
   }

   public Boolean getRequer_autenticacao() {
      return requer_autenticacao;
   }

   public void setRequer_autenticacao(Boolean
requer_autenticacao) {
      this.requer_autenticacao = requer_autenticacao;
   }
}
```

6.4.319 TipoUsuarioForm.java

```
/*
 * TipoUsuarioForm.java
 * Created on 31 de Janeiro de 2006, 21:10
 */

package estagio.form;

import estagio.bean.Usuario;
import java.util.ArrayList;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;

/**
 *
 *
 */
public class TipoUsuarioForm  extends ActionForm{

   private String status;


   public void reset(ActionMapping mapping, HttpServletRequest
request) {
      setStatus("aluno");
   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      return errors;
   }

   public String getStatus() {
      return status;
   }

   public void setStatus(String status) {
      this.status = status;
   }
}
```

6.4.320 TurnoDisponivelForm.java

```
/*
 * TurnoDisponivelForm.java
 * Created on 14 de Fevereiro de 2006, 15:41
 */
```

```
package estagio.form;

import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;
import org.apache.struts.action.ActionMessage;
import org.apache.struts.validator.ValidatorForm;

/**
 *
 *
 */
public class TurnoDisponivelForm extends ValidatorForm{

   private Integer codAlunoEstagio;
   private String horaInicio;
   private String horaFim;

   public void reset(ActionMapping mapping, HttpServletRequest
request) {

setCodAlunoEstagio(request.getParameter("codalunoestagio")!=n
ull?Integer.valueOf(request.getParameter("codalunoestagio")):Int
eger.valueOf(0));
   }

   public ActionErrors validate(ActionMapping
mapping,HttpServletRequest request) {
      ActionErrors errors = super.validate(mapping, request);
      if(getHoraInicio() != null && getHoraInicio().length() > 0){
        if(':' != getHoraInicio().charAt(2)){
          errors.add("horaInicio", new
ActionMessage("error.horaInicio.invalida"));
        } else {
          try{
            Integer horasInicio =
Integer.valueOf(getHoraInicio().substring(0,2));
            Integer minutosInicio =
Integer.valueOf(getHoraInicio().substring(3));
            if(horasInicio > 24){
              errors.add("horaInicio", new
ActionMessage("error.horaInicio.invalida"));
            }
            if(minutosInicio > 59){
              errors.add("horaInicio", new
ActionMessage("error.horaInicio.invalida"));
            }
          } catch (NumberFormatException numberExcpetion){
            errors.add("horaInicio", new
ActionMessage("error.horaInicio.integer"));
          }
        }
      }
      if(getHoraFim() != null && getHoraFim().length() > 0){
        if(':' != getHoraFim().charAt(2)){
          errors.add("horaFim", new
ActionMessage("error.horaFim.invalida"));
        } else {
          try{
            Integer horasFim =
Integer.valueOf(getHoraFim().substring(0,2));
            Integer minutosFim =
Integer.valueOf(getHoraFim().substring(3));
            if(horasFim > 24){
              errors.add("horaFim", new
ActionMessage("error.horaFim.invalida"));
            }
            if(minutosFim > 59){
              errors.add("horaFim", new
ActionMessage("error.horaFim.invalida"));
            }
          } catch (NumberFormatException numberExcpetion){
            errors.add("horaFim", new
ActionMessage("error.horaFim.integer"));
          }
        }
      }
      return errors;
   }

   public Integer getCodAlunoEstagio() {
      return codAlunoEstagio;
   }

   public void setCodAlunoEstagio(Integer codAlunoEstagio) {
      this.codAlunoEstagio = codAlunoEstagio;
   }

   public String getHoraInicio() {
      return horaInicio;
   }

   public void setHoraInicio(String horaInicio) {
      this.horaInicio = horaInicio;
   }

   public String getHoraFim() {
      return horaFim;
   }

   public void setHoraFim(String horaFim) {
      this.horaFim = horaFim;
   }

}
```

6.4.321 package estagio.mail

6.4.322 MailManager.java

```
/*
 * MailManager.java
 * Created on 15 de Maio de 2006, 12:22
 */

package estagio.mail;

import estagio.bean.ServidorSmtp;
import java.util.Properties;
import javax.mail.Authenticator;
import javax.mail.Message;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

/**
 *
 *
 */
public class MailManager {

   /** Creates a new instance of MailManager */
   public MailManager() {
   }

   public void send(MailServer server, MailMessage msg,
ServidorSmtp servidorSmtp) throws Exception {
      try {
        Properties mailProps = new Properties();
        mailProps.put("mail.smtp.host", server.getMailServer());
        SimpleAuth auth = null;
```

```
if(!servidorSmtp.getPorta_smtp().equals(Integer.valueOf(110))){
            mailProps.put("mail.smtp.port",
servidorSmtp.getPorta_smtp().intValue() );
          }
          if (!server.isNeedsAuth()) {
            // [#] No authorization needed for this mail server
            mailProps.put("mail.smtp.auth", "false");
          } else {
            // [#] Authorization needed for this mail server
            auth = new SimpleAuth(server.getUsername(),
server.getPassword());
            mailProps.put("mail.smtp.auth", "true");
            mailProps.put("mail.user", auth.username);
            mailProps.put("mail.from", msg.getFrom());
            mailProps.put("mail.to", msg.getTo());
          }
          Session mailSession = Session.getInstance(mailProps,
auth);
          mailSession.setDebug(false);
          Message email = new MimeMessage(mailSession);
          email.setRecipients( Message.RecipientType.TO,
InternetAddress.parse(msg.getTo()) );
          if (!msg.getCc().trim().equals("")) {
            email.setRecipients( Message.RecipientType.CC,
InternetAddress.parse(msg.getCc()) );
          }
          if (!msg.getBcc().trim().equals("")) {
            email.setRecipients( Message.RecipientType.BCC,
InternetAddress.parse(msg.getBcc()) );
          }
          email.setFrom( new InternetAddress(msg.getFrom()));
          email.setSubject(msg.getSubject());
          email.setContent( msg.getText(), "text/plain" );

          Transport.send( email );
        } catch (Exception e) {
          System.out.println(" [E] ERROR: "+e);
          e.printStackTrace(System.out);
          throw e;
        }
        System.out.println(" [#] Email successfully sent");
        return;
    }

    public void send(MailServer server, String to, String cc, String
bcc, String from, String subject, String text, ServidorSmtp
servidorSmtp) throws Exception {
        MailMessage msg = new
MailMessage(to,cc,bcc,from,subject,text);
        this.send(server, msg, servidorSmtp);
        return;
    }

    /*public static void main(String args[]) {
        MailServer server = new
MailServer("ufpr.br","r_hill","563247",false);
        MailManager m = new MailManager();
        try {

m.send(server,"r_hill@ufpr.br","","","r_hill@ufpr.br","Esse e um
email de teste","Corpo do email");
        } catch (Exception e) {
          e.printStackTrace(System.out);
        }
    }*/

    class SimpleAuth extends Authenticator {
        public String username = null;
        public String password = null;
        public SimpleAuth(String user, String pwd) {
          username = user;
          password = pwd;
        }
        protected PasswordAuthentication
getPasswordAuthentication() {
          return new PasswordAuthentication(username,password);
        }
    }
}
```

6.4.323 MailMessage.java

```
/*
 * MailMessage.java
 * Created on 15 de Maio de 2006, 12:21
 */

package estagio.mail;

/**
 *
 *
 */
public class MailMessage {

    private String to;
    private String cc;
    private String bcc;
    private String from;
    private String subject;
    private String text;

    /** Creates a new instance of MailMessage */
    public MailMessage() {
    }

    /** Creates a new instance of MailMessage */
    public MailMessage(String to,String cc,String bcc,String
from,String subject,String text) {
        this.to=to;
        this.cc=cc;
        this.bcc=bcc;
        this.from=from;
        this.subject=subject;
        this.text=text;
    }

    /** Getter for property bcc.
     * @return Value of property bcc.
     *
     */
    public java.lang.String getBcc() {
        return bcc;
    }

    /** Setter for property bcc.
     * @param bcc New value of property bcc.
     *
     */
    public void setBcc(java.lang.String bcc) {
        this.bcc = bcc;
    }

    /** Getter for property cc.
     * @return Value of property cc.
     *
     */
    public java.lang.String getCc() {
        return cc;
    }

    /** Setter for property cc.
     * @param cc New value of property cc.
     *
```

```
 */
public void setCc(java.lang.String cc) {
    this.cc = cc;
}

/** Getter for property from.
 * @return Value of property from.
 *
 */
public java.lang.String getFrom() {
    return from;
}

/** Setter for property from.
 * @param from New value of property from.
 *
 */
public void setFrom(java.lang.String from) {
    this.from = from;
}

/** Getter for property subject.
 * @return Value of property subject.
 *
 */
public java.lang.String getSubject() {
    return subject;
}

/** Setter for property subject.
 * @param subject New value of property subject.
 *
 */
public void setSubject(java.lang.String subject) {
    this.subject = subject;
}

/** Getter for property text.
 * @return Value of property text.
 *
 */
public java.lang.String getText() {
    return text;
}

/** Setter for property text.
 * @param text New value of property text.
 *
 */
public void setText(java.lang.String text) {
    this.text = text;
}

/** Getter for property to.
 * @return Value of property to.
 *
 */
public java.lang.String getTo() {
    return to;
}

/** Setter for property to.
 * @param to New value of property to.
 *
 */
public void setTo(java.lang.String to) {
    this.to = to;
}

}
```

6.4.324 MailServer.java

```
/*
 * MailServer.java
 * Created on 15 de Maio de 2006, 12:21
 */

package estagio.mail;

/**
 *
 *
 */
public class MailServer {

    private String mailServer = null;
    private String username = null;
    private String password = null;
    private boolean needsAuth = true;

    /** Creates a new instance of MailServer */
    public MailServer() {
    }

    /** Creates a new instance of MailServer */
    public MailServer(String mailServer, String username, String
password, boolean needsAuth) {
        this.mailServer=mailServer;
        this.username=username;
        this.password=password;
        this.needsAuth=needsAuth;
    }

    /** Getter for property mailServer.
     * @return Value of property mailServer.
     *
     */
    public java.lang.String getMailServer() {
        return mailServer;
    }

    /** Setter for property mailServer.
     * @param mailServer New value of property mailServer.
     *
     */
    public void setMailServer(java.lang.String mailServer) {
        this.mailServer = mailServer;
    }

    /** Getter for property needsAuth.
     * @return Value of property needsAuth.
     *
     */
    public boolean isNeedsAuth() {
        return needsAuth;
    }

    /** Setter for property needsAuth.
     * @param needsAuth New value of property needsAuth.
     *
     */
    public void setNeedsAuth(boolean needsAuth) {
        this.needsAuth = needsAuth;
    }

    /** Getter for property password.
     * @return Value of property password.
     *
     */
    public java.lang.String getPassword() {
        return password;
    }
```

```
  /** Setter for property password.
   * @param password New value of property password.
   *
   */
  public void setPassword(java.lang.String password) {
    this.password = password;
  }

  /** Getter for property username.
   * @return Value of property username.
   *
   */
  public java.lang.String getUsername() {
    return username;
  }

  /** Setter for property username.
   * @param username New value of property username.
   *
   */
  public void setUsername(java.lang.String username) {
    this.username = username;
  }

}
```

6.4.325 package estagio.validation

6.4.326 Valida.java

```
/*
 * Valida.java
 * Created on 14 de Janeiro de 2006, 20:02
 */

package estagio.validation;

import javax.servlet.http.HttpServletRequest;
import org.apache.commons.validator.Field;
import org.apache.commons.validator.ValidatorAction;
import org.apache.commons.validator.ValidatorUtil;
import org.apache.struts.action.ActionMessages;
import org.apache.commons.validator.Validator;
import org.apache.struts.validator.Resources;

/**
 *
 *
 */
public class Valida {

   public static boolean validateCpf(Object bean,
        ValidatorAction va, Field field, ActionMessages errors,
Validator validator,HttpServletRequest request){
      String xCPF = "";
      if (field.getProperty() != null && field.getProperty().length()
> 0) {
        xCPF=
ValidatorUtil.getValueAsString(bean,field.getProperty());
      }
      try {
        //Testa se o CPF é válido ou não
        int d1,d4,xx,nCount,resto,digito1,digito2;
        String Check;
        String Separadores = "/-.";
        d1 = 0; d4 = 0; xx = 1;
        for (nCount = 0; nCount < xCPF.length() -2; nCount++) {
          String s_aux = xCPF.substring(nCount, nCount+1);
          if (Separadores.indexOf(s_aux) == -1) {
            d1 = d1 + ( 11 - xx ) *
Integer.valueOf(s_aux).intValue();
            d4 = d4 + ( 12 - xx ) *
Integer.valueOf(s_aux).intValue();
            xx++;
          };
        };
        resto = (d1 % 11);
        if (resto < 2) {
          digito1 = 0;
        } else {
          digito1 = 11 - resto;
        }

        d4 = d4 + 2 * digito1;
        resto = (d4 % 11);
        if (resto < 2) {
          digito2 = 0;
        } else {
          digito2 = 11 - resto;
        }
        Check = String.valueOf(digito1) +
String.valueOf(digito2);
        String s_aux2 = xCPF.substring(xCPF.length()-2,
xCPF.length());
        if (s_aux2.compareTo(Check) != 0) {
          errors.add(field.getKey(),
Resources.getActionError(request,va,field));
          return false;
        }
        return true;
      } catch (Exception e) {
        errors.add(field.getKey(),
Resources.getActionError(request,va,field));
        return false;
      }
    }

    public static boolean validateCnpj( Object bean,
        ValidatorAction va, Field field, ActionMessages errors,
Validator validator,HttpServletRequest request){
      String xCGC = "";
      if (field.getProperty() != null && field.getProperty().length()
> 0) {
        xCGC=
ValidatorUtil.getValueAsString(bean,field.getProperty());
      }
      try {

        //Testa se o CGC é válido ou não
        int d1,d4,xx,nCount,fator,resto,digito1,digito2;
        String Check, s_aux;
        String Separadores = "/-.";
        d1 = 0;
        d4 = 0;
        xx = 0;
        for (nCount = 0; nCount < xCGC.length()-2; nCount++) {
          s_aux = xCGC.substring(nCount, nCount+1);
          if (Separadores.indexOf(s_aux) == -1) {
            if (xx < 4) {
              fator = 5 - xx;
            } else {
              fator = 13 - xx;
            }

            d1 = d1 + Integer.valueOf(s_aux).intValue() * fator;
            if (xx < 5) {
              fator = 6 - xx;
            } else {
              fator = 14 - xx;
            }
            d4 += Integer.valueOf(s_aux).intValue() * fator;
            xx++;
          };
```

```
        }
        resto = (d1 % 11);
        if (resto < 2) {
          digito1 = 0;
        } else {
          digito1 = 11 - resto;
        }

        d4 = d4 + 2 * digito1;
        resto = (d4 % 11);
        if (resto < 2) {
          digito2 = 0;
        } else {
          digito2 = 11 - resto;
        }

        Check = String.valueOf(digito1) +
String.valueOf(digito2);
        if (Check.compareTo(xCGC.substring(xCGC.length()-2,
xCGC.length() )) !=0) {
          errors.add(field.getKey(),
Resources.getActionError(request,va,field));
          return false;
        }
        return true;
      } catch (Exception e) {
        errors.add(field.getKey(),
Resources.getActionError(request,va,field));
        return false;
      }
    }
}
```

6.4.327 package jasper

6.4.328 teste_seguro.jrxml

```
<?xml version="1.0" encoding="UTF-8"  ?>
<!-- Created with iReport - A designer for JasperReports -->
<!DOCTYPE jasperReport PUBLIC "//JasperReports//DTD
Report Design//EN"
"http://jasperreports.sourceforge.net/dtds/jasperreport.dtd">
<jasperReport
                name="teste_seguro"
                columnCount="1"
                printOrder="Vertical"
                orientation="Portrait"
                pageWidth="595"
                pageHeight="842"
                columnWidth="539"
                columnSpacing="0"
                leftMargin="28"
                rightMargin="28"
                topMargin="20"
                bottomMargin="20"
                whenNoDataType="NoPages"
                isTitleNewPage="false"
                isSummaryNewPage="false">
        <property name="ireport.scriptlethandling" value="0"
/>
        <property name="ireport.encoding" value="UTF-8" />
        <import value="java.util.*" />
        <import value="net.sf.jasperreports.engine.*" />
        <import value="net.sf.jasperreports.engine.data.*" />


        <field name="nome_pessoa" class="java.lang.String"/>
        <field name="cpf" class="java.math.BigDecimal"/>
        <field name="nasc_pessoa" class="java.sql.Date"/>
        <field name="datainicio" class="java.sql.Date"/>
        <field name="datafim" class="java.sql.Date"/>
        <field name="empresa_cnpj" class="java.lang.String"/>
        <field name="processo" class="java.lang.String"/>
        <field name="data" class="java.util.Date"/>

                <background>
                        <band height="0"
isSplitAllowed="true" >
                        </band>
                </background>
                <title>
                        <band height="53"
isSplitAllowed="true" >
                                <staticText>
                                        <reportElement
                                                x="54"
                                                y="30"

        width="412"

        height="19"

        key="staticText-6"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement
textAlignment="Center">
                                                <font
size="12"/>
                                        </textElement>

        <text><![CDATA[LISTAGEM DE ESTAGIÃ□RIOS
PARA FINS DE BENEFÃ□CIO DE SEGURO]]></text>
                                </staticText>
                                <staticText>
                                        <reportElement

        x="128"
                                                y="2"

        width="299"

        height="22"

        key="staticText-10"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement
textAlignment="Center">
                                                <font
pdfFontName="Helvetica-Bold" size="12" isBold="true"/>
                                        </textElement>

        <text><![CDATA[SISTEMA DE CONTROLE DE
ESTÃ□GIOS DA UFPR]]></text>
                                </staticText>
                        </band>
                </title>
                <pageHeader>
                        <band height="20"
isSplitAllowed="true" >
                                <staticText>
                                        <reportElement

        x="333"
                                                y="2"

        width="36"
```

```
        height="13"

        key="staticText-7"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font/>
                                        </textElement>

        <text><![CDATA[MÃ*s:]]></text>
                                </staticText>
                                <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Now" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement

        x="371"
                                                y="2"

        width="82"

        height="13"

        key="textField-1"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font/>
                                        </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[new
SimpleDateFormat("MMMMMMMMMM/yyyy").format($F{dat
a})]]></textFieldExpression>
                                </textField>
                        </band>
                </pageHeader>
                <columnHeader>
                        <band height="27"
isSplitAllowed="true" >
                                <rectangle radius="5" >
                                        <reportElement
                                                x="9"
                                                y="3"

        width="520"

        height="21"

        key="rectangle-1"/>
                                        <graphicElement
stretchType="NoStretch"/>
                                </rectangle>
                                <staticText>
                                        <reportElement
                                                x="14"
                                                y="6"

        width="40"

        height="13"

        key="staticText-1"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font
pdfFontName="Helvetica-Bold" size="10" isBold="true"/>
                                        </textElement>

        <text><![CDATA[Ordem]]></text>
                                </staticText>
                                <staticText>
                                        <reportElement
                                                x="66"
                                                y="6"

        width="78"

        height="13"

        key="staticText-2"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font
pdfFontName="Helvetica-Bold" size="10" isBold="true"/>
                                        </textElement>
                                <text><![CDATA[Nome
do Aluno]]></text>
                                </staticText>
                                <staticText>
                                        <reportElement

        x="371"
                                                y="6"

        width="57"

        height="13"

        key="staticText-4"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font
pdfFontName="Helvetica-Bold" size="10" isBold="true"/>
                                        </textElement>

        <text><![CDATA[Nascimento]]></text>
                                </staticText>
                                <staticText>
                                        <reportElement

        x="461"
                                                y="6"

        width="53"

        height="13"

        key="staticText-5"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
```

```
                                        <font
pdfFontName="Helvetica-Bold" size="10" isBold="true"/>
                                        </textElement>

        <text><![CDATA[Processo]]></text>
                                </staticText>
                                <staticText>
                                        <reportElement

        x="258"
                                                y="6"

        width="33"

        height="13"

        key="staticText-8"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font
pdfFontName="Helvetica-Bold" isBold="true"/>
                                        </textElement>

        <text><![CDATA[CPF]]></text>
                                </staticText>
                        </band>
                </columnHeader>
                <detail>
                        <band height="16"
isSplitAllowed="true" >
                                <textField
isStretchWithOverflow="false" pattern="###0"
isBlankWhenNull="false" evaluationTime="Now"
hyperlinkType="None"  hyperlinkTarget="Self" >
                                        <reportElement
                                                x="14"
                                                y="1"

        width="28"

        height="13"

        key="textField-2"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font/>
                                        </textElement>
                                <textFieldExpression
class="java.lang.Integer"><![CDATA[$V{REPORT_COUNT}]]
></textFieldExpression>
                                </textField>
                                <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Now" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement
                                                x="69"
                                                y="1"

        width="89"

        height="13"

        key="textField-3"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font/>
                                        </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{nome_pessoa}]]></text
FieldExpression>
                                </textField>
                                <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Now" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement

        x="255"
                                                y="1"

        width="81"

        height="13"

        key="textField-4"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font/>
                                        </textElement>
                                <textFieldExpression
class="java.math.BigDecimal"><![CDATA[$F{cpf}]]></textFiel
dExpression>
                                </textField>
                                <textField
isStretchWithOverflow="false" pattern=""
isBlankWhenNull="false" evaluationTime="Now"
hyperlinkType="None"  hyperlinkTarget="Self" >
                                        <reportElement

        x="370"
                                                y="1"

        width="81"

        height="13"

        key="textField-5"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                                <font/>
                                        </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[new
SimpleDateFormat("dd/MM/yyyy").format($F{nasc_pessoa})]]>
</textFieldExpression>
                                </textField>
                                <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Now" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement

        x="462"
```

```
                                        y="1"

        width="58"

        height="13"

        key="textField-12"/>
                                    <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                    <textElement>
                                        <font/>
                                    </textElement>
                                <textFieldExpression
class="java.lang.String"><![CDATA[$F{processo}]]></textField
Expression>
                                </textField>
                                <line
direction="TopDown">
                                    <reportElement
                                        x="1"
                                        y="14"

        width="534"

        height="0"

        key="line-1"/>
                                    <graphicElement
stretchType="NoStretch" pen="Thin"/>
                                </line>
                            </band>
                        </detail>
                        <columnFooter>
                            <band height="30"
isSplitAllowed="true" >
                                <rectangle radius="5" >
                                    <reportElement
                                        x="8"
                                        y="3"

        width="150"

        height="23"

        key="rectangle-2"/>
                                    <graphicElement
stretchType="NoStretch"/>
                                </rectangle>
                                <staticText>
                                    <reportElement
                                        x="21"
                                        y="8"

        width="39"

        height="13"

        key="staticText-9"/>
                                    <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                    <textElement>
                                        <font/>
                                    </textElement>

        <text><![CDATA[TOTAL:]]></text>
                                </staticText>
                                    <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Report" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement
                                            x="62"
                                            y="8"

        width="71"

        height="13"

        key="textField-13"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                            <font/>
                                        </textElement>
                                    <textFieldExpression
class="java.lang.Integer"><![CDATA[$V{REPORT_COUNT}]]
></textFieldExpression>
                                    </textField>
                                </band>
                            </columnFooter>
                            <pageFooter>
                                <band height="21"
isSplitAllowed="false" >
                                    <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Report" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement

        x="464"
                                            y="1"

        width="21"

        height="16"

        key="textField-7"/>
                                        <box
topBorder="None" topBorderColor="#000000"
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                        <textElement>
                                            <font/>
                                        </textElement>
                                    <textFieldExpression
class="java.lang.Integer"><![CDATA[$V{PAGE_NUMBER}]]>
</textFieldExpression>
                                    </textField>
                                    <textField
isStretchWithOverflow="false" isBlankWhenNull="false"
evaluationTime="Now" hyperlinkType="None"
hyperlinkTarget="Self" >
                                        <reportElement

        x="382"
                                            y="2"

        width="82"

        height="15"

        key="textField-11"/>
                                        <box
topBorder="None" topBorderColor="#000000"
```

```
leftBorder="None" leftBorderColor="#000000"
rightBorder="None" rightBorderColor="#000000"
bottomBorder="None" bottomBorderColor="#000000"/>
                                                <textElement>
                                                        <font/>
                                                </textElement>
                                        <textFieldExpression
class="java.lang.String"><![CDATA["PÃ¡gina " +
$V{PAGE_NUMBER} + " de "]]></textFieldExpression>
                                        </textField>
                                </band>
                        </pageFooter>
                        <lastPageFooter>
                                <band height="50"
isSplitAllowed="true" >
                                </band>
                        </lastPageFooter>
                        <summary>
                                <band height="50"
isSplitAllowed="true" >
                                </band>
                        </summary>
</jasperReport>
```