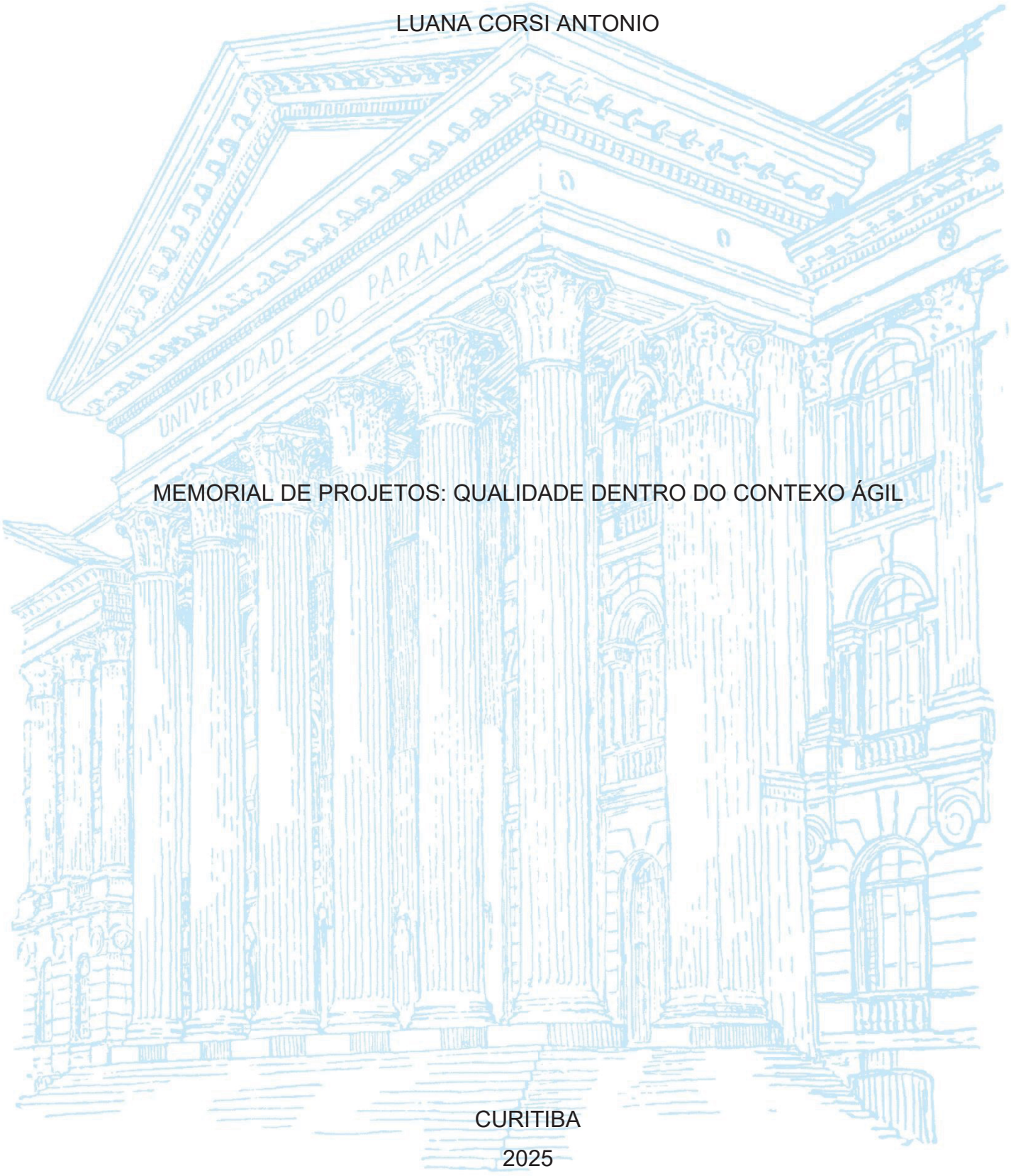
UNIVERSIDADE FEDERAL DO PARANÁ

LUANA CORSI ANTONIO



MEMORIAL DE PROJETOS: QUALIDADE DENTRO DO CONTEXO ÁGIL

CURITIBA

2025

LUANA CORSI ANTONIO

MEMORIAL DE PROJETOS: QUALIDADE DENTRO DO CONTEXO ÁGIL

Trabalho de Conclusão de Curso apresentado ao curso de Pós-Graduação em Desenvolvimento Ágil de Software, Setor de Educação Profissional e Tecnológica, Universidade Federal do Paraná, como requisito parcial à obtenção do título de Especialista em Desenvolvimento Ágil de Software.

Orientador: Profa. Dra. Rafaela Mantovani Fontana

CURITIBA

2025

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação Desenvolvimento Ágil de Software da Universidade Federal do Paraná foram convocados para realizar a arguição da Monografia de Especialização de **LUANA CORSI ANTONIO**, intitulada: **MEMORIAL DE PROJETOS: QUALIDADE DENTRO DO CONTEXO ÁGIL**, que após terem inquirido a aluna e realizada a avaliação do trabalho, são de parecer pela sua **aprovação** no rito de defesa.

A outorga do título de especialista está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 13 de Agosto de 2025.

RAFAELA MANTOVANI FONTANA
Presidente da Banca Examinadora

JAIME WOJCIECHOWSKI
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

RESUMO

Este memorial de projetos reúne os trabalhos finais do curso de Especialização em Desenvolvimento Ágil de Software da UFPR e realça como a qualidade do software é um elemento central no desenvolvimento ágil, mostrando-a resultado da integração contínua de práticas ágeis em todas as fases do projeto. A jornada começa com a base teórica de Métodos Ágeis, fundamentada no Manifesto Ágil. Em seguida, a Modelagem Ágil garante a qualidade na definição e comunicação de requisitos. O Gerenciamento Ágil de Projetos complementa com a utilização de métricas para monitorar e aprimorar continuamente os processos. A qualidade técnica do software é explorada em disciplinas de programação para diferentes plataformas e modelagem de dados, essenciais para a manutenibilidade e escalabilidade do software. A disciplina de *User Experience* atua diretamente na qualidade percebida pelo cliente, por meio de testes de usabilidade. Por fim, Infraestrutura para Desenvolvimento e Implantação (DevOps) e Testes Automatizados reforçam a qualidade, automatizando *pipelines* de entrega e garantindo a estabilidade do software. O memorial ilustra as etapas do desenvolvimento, e os resultados dos trabalhos desenvolvidos pela autora ao longo do curso, por meio de artefatos como diagramas, protótipos e trechos de código.

Palavras-chave: Desenvolvimento Ágil, Qualidade, Software, Parecer Técnico.

ABSTRACT

This project memorial brings together the final projects of the UFPR Specialization in Agile Software Development and highlights how software quality is a central element in agile development, demonstrating it as a result of the continuous integration of agile practices across all project phases. The journey begins with the theoretical foundation of Agile Methods, rooted in the Agile Manifesto. Next, Agile Modeling ensures quality in requirement definition and communication. Agile Project Management complements this by utilizing metrics to continuously monitor and enhance processes. The technical quality of the software is explored through programming disciplines for different platforms and data modeling, which are essential for software maintainability and scalability. The User Experience discipline directly impacts customer-perceived quality through usability testing. Finally, Infrastructure for Development and Deployment (DevOps) and Automated Tests reinforce quality by automating delivery pipelines and ensuring software stability. The memorial illustrates the development stages, and the results of the work carried out by the author throughout the course, using artifacts like diagrams, prototypes, and code snippets.

Keywords: Agile Development, Quality, Software, Technical Report.

SUMÁRIO

1 PARECER TÉCNICO

A adoção de metodologias ágeis no desenvolvimento de software tem se mostrado fundamental para garantir qualidade, flexibilidade e entrega contínua de valor a clientes e usuários. No programa de pós-graduação em Desenvolvimento Ágil de Software, compreende-se que a qualidade do software não se limita a atributos técnicos, mas resulta da integração de práticas ágeis em todas as etapas do ciclo de vida do projeto. Essa integração é abordada por meio do alinhamento entre os princípios ágeis e as disciplinas específicas abordadas na formação.

As disciplinas da especialização fornecem as bases teóricas e práticas necessárias para alinhar agilidade e qualidade de forma consistente. A disciplina de Métodos Ágeis para Desenvolvimento de Software (MADS) fundamenta os princípios do Manifesto Ágil. O manifesto estabeleceu quatro valores essenciais que privilegiam indivíduos e interações em detrimento de processos e ferramentas, software funcionando em vez de documentação extensiva, colaboração com o cliente sobre negociação contratual, e capacidade de responder a mudanças em vez de seguir rigidamente um plano. Complementando esses valores, doze princípios orientadores foram estabelecidos para operacionalizar essa visão, enfatizando a entrega contínua de software funcional, a aceitação de mudanças mesmo em fases tardias do desenvolvimento, e a colaboração diária entre as partes interessadas (Beck et al., 2001). Também, a disciplina MADS aborda as principais metodologias ágeis, como Scrum, XP e Lean. Esses conhecimentos são essenciais para estabelecer uma cultura de trabalho colaborativo e iterativo.

As disciplinas Modelagem Ágil de Software 1 (MAG1) e 2 (MAG2) abordam técnicas como *User Stories* e *Domain-Driven Design*, fundamentais para a definição clara e visualização dos processos e requisitos, utilizando diagramas da UML (do inglês *Unified Modeling Language*, ou Linguagem de Modelagem Unificada) que facilitam a comunicação entre as equipes e a adaptação do sistema às mudanças (Booch et al., 2006). Inayat et al. (2015) destacam que requisitos bem modelados e priorizados impactam diretamente na qualidade do produto, reduzindo retrabalhos e ambiguidades durante o desenvolvimento.

As disciplinas Gerenciamento Ágil de Projetos de Software 1 (GAP1) e 2 (GAP2) abordam a gestão de projetos, enfatizando a importância de práticas como a criação de planos de *release* e a utilização de métodos como Kanban para a

transparência e agilidade na entrega de valor. Além disso, elas complementam essa visão ao apresentar métricas de qualidade como lead time, velocidade e WIP (do inglês, *Work In Progress*), que permitem monitorar e melhorar continuamente os processos (Poppendieck; Poppendieck, 2003). Staron et al. (2017) mostram que o uso dessas métricas ajuda as equipes a identificar gargalos e manter um ritmo sustentável de entregas sem comprometer a qualidade.

Disciplinas técnicas como Introdução à Programação (INTRO) e Desenvolvimento Web e Mobile (WEB1, WEB2, MOB1 e MOB2), evidenciaram a importância de uma base sólida de programação. Essas etapas integraram práticas modernas como TDD (*Test-Driven Development*) e integração contínua (CI), além do desenvolvimento responsivo para diferentes plataformas. A disciplina Aspectos Ágeis de Programação (AAP) reforçou a importância de boas práticas de codificação, incluindo *Clean Code*, SOLID e *Pair Programming*, vitais para a manutenibilidade e escalabilidade do software. Isso é crucial, pois Leite et al. (2025) argumentam que, em ambientes ágeis, a velocidade de entrega não deve comprometer a qualidade do código, evitando dívidas técnicas futuras.

Projetos ágeis exigem esquemas de banco de dados eficientes e arquiteturas responsivas que suportem mudanças frequentes. Dessa forma, a disciplina Banco de Dados (DB) desempenha um papel crucial ao abordar a fundamentação, modelagem e manipulação de dados. Isso inclui modelagem de dados e linguagem SQL. Conforme evidenciado por Mognon e Stadzisz (2017) em sua revisão sistemática, a modelagem de dados em ambientes de desenvolvimento ágil é crucial para a qualidade do software, pois fornece a clareza e a estrutura necessárias para mitigar ambiguidades, reduzir defeitos e otimizar o design e a manutenibilidade do sistema.

A disciplina de *User Experience* (UX) no Desenvolvimento Ágil de Software enfatiza a importância de testes de usabilidade iterativos e do *feedback* contínuo do cliente, assegurando que o produto atenda às expectativas dos usuários. A disciplina abordou a criação de protótipos no Figma como ferramenta de design e a criação de questionários para coleta de *feedback* dos usuários.

Os projetos de Infraestrutura para Desenvolvimento e Implantação (DevOps) (INFRA) e Testes Automatizados (TEST) representam áreas importantes para a qualidade em ambientes ágeis. Eles enfatizam a cultura DevOps e a automação, que são essenciais para a melhoria contínua dos processos e a redução de erros em ambientes de produção. O DevOps automatiza *pipelines* de entrega, reduzindo falhas

humanas e acelerando a detecção de problemas (Martin, 2008). A automação de testes, por sua vez, é essencial para garantir a estabilidade do software em entregas frequentes. Juntos os tópicos de DevOps e Testes Automatizados garantem que os desenvolvedores e equipes sejam capazes de construir e manter software de alta qualidade, minimizando riscos e otimizando o processo de entrega. Conforme demonstrado por Forsgren et al. (2018), equipes de alta performance que adotam princípios DevOps, alcançam notavelmente maior qualidade de software, reduzindo a taxa de falhas em produção e melhorando o tempo de recuperação.

A cultura de qualidade emerge como fator determinante para o sucesso de projetos ágeis, requerendo o envolvimento de todos os *stakeholders*, incluindo *Product Owners*, desenvolvedores e QA (Beck, 2001). Essa cultura é fortemente influenciada pelas práticas de gerenciamento ágil abordadas durante essa pós-graduação. Sem uma mentalidade colaborativa e focada em melhoria contínua, mesmo as melhores ferramentas e técnicas podem falhar em entregar software de alta qualidade.

Muito além dos aspectos técnicos, a qualidade de um software pode ser mensurada pela satisfação do cliente, um indicador crítico de sucesso em projetos de desenvolvimento. As metodologias ágeis, com sua ênfase em entrega contínua de valor, *feedback* iterativo e adaptação a mudanças, demonstram impacto positivo nessa dimensão. Estudos recentes comprovam que práticas como revisões frequentes com *stakeholders* (Sprint Reviews) e incorporação contínua de *feedback* elevam significativamente a satisfação do cliente (Crispin; Gregory, 2009).

Uma base fundamental da qualidade de software é formação dos profissionais, que devem ser capacitados tanto em aspectos técnicos quanto em práticas modernas de desenvolvimento, gestão e colaboração. O curso de especialização em Desenvolvimento Ágil de Software, ao integrar disciplinas como Métodos Ágeis, Modelagem de Software, Gerenciamento de Projetos, Programação, Banco de Dados, DevOps, Testes Automatizados e UX proporciona a formação de profissionais completos capazes de aplicar esses conhecimentos de forma integrada. Essa visão multidisciplinar permite não apenas o desenvolvimento de sistemas robustos e eficientes, mas também uma contribuição relevante para a evolução da área, alinhando qualidade técnica, satisfação do cliente e boas práticas de engenharia. Dessa forma, a especialização prepara indivíduos para os desafios do mercado enquanto fortalece todo o ecossistema de desenvolvimento de software,

promovendo soluções inovadoras e sustentáveis que atendam às demandas atuais da indústria.

2 DISCIPLINA: MADS – MÉTODOS ÁGEIS PARA DESENVOLVIMENTO DE SOFTWARE

O projeto da disciplina de Métodos Ágeis de Desenvolvimento de Software (MADS) tem como objetivo central a elaboração de um mapa mental que sintetize os conceitos principais de processos de software, princípios ágeis e os métodos ágeis de desenvolvimento de software. A relevância deste projeto reside em sua capacidade de consolidar o entendimento dos fundamentos que regem a agilidade no contexto do desenvolvimento de software, servindo como uma base conceitual para as demais disciplinas do curso.

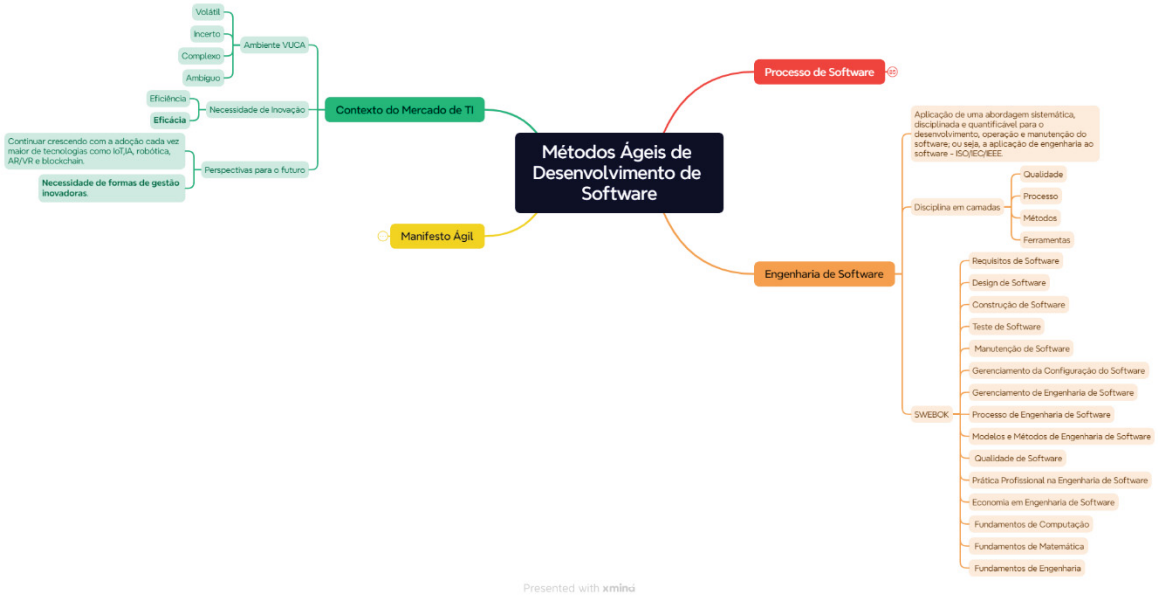
A importância deste projeto no desenvolvimento ágil de software é multifacetada. Primeiramente, a criação do mapa mental exige uma profunda imersão e organização do conhecimento sobre temas como o Processo de Software, contrastando-o com os Modelos Tradicionais de Processo de Software. Essa análise comparativa é fundamental para compreender a transição e as vantagens da abordagem ágil. Além disso, a disciplina MADS e seu projeto focam no Manifesto Ágil e seus Princípios Ágeis, que são a espinha dorsal de qualquer metodologia ágil. O entendimento desses valores e princípios é essencial para o desenvolvimento de uma mentalidade ágil, que privilegia a colaboração, a adaptabilidade e a entrega contínua de valor.

O mapa mental também exige o detalhamento de metodologias ágeis específicas, como *Lean Software Development*, Scrum, *Extreme Programming* (XP) e Kanban. Aprofundar-se em cada uma dessas abordagens permite ao estudante não apenas conhecer suas particularidades, mas também identificar suas sinergias e quando aplicá-las de forma mais eficaz. A inclusão da Entrega Contínua de Software reforça a importância da automação e da agilidade na disponibilização de valor ao cliente.

O projeto da disciplina MADS é um pilar fundamental que conecta e contextualiza o conhecimento adquirido em todo o curso. Ele capacita o estudante a visualizar a interdependência entre os conceitos teóricos e as práticas aplicadas, promovendo uma compreensão abrangente do desenvolvimento ágil de software e sua importância para a entrega de soluções de alta qualidade e valor contínuo.

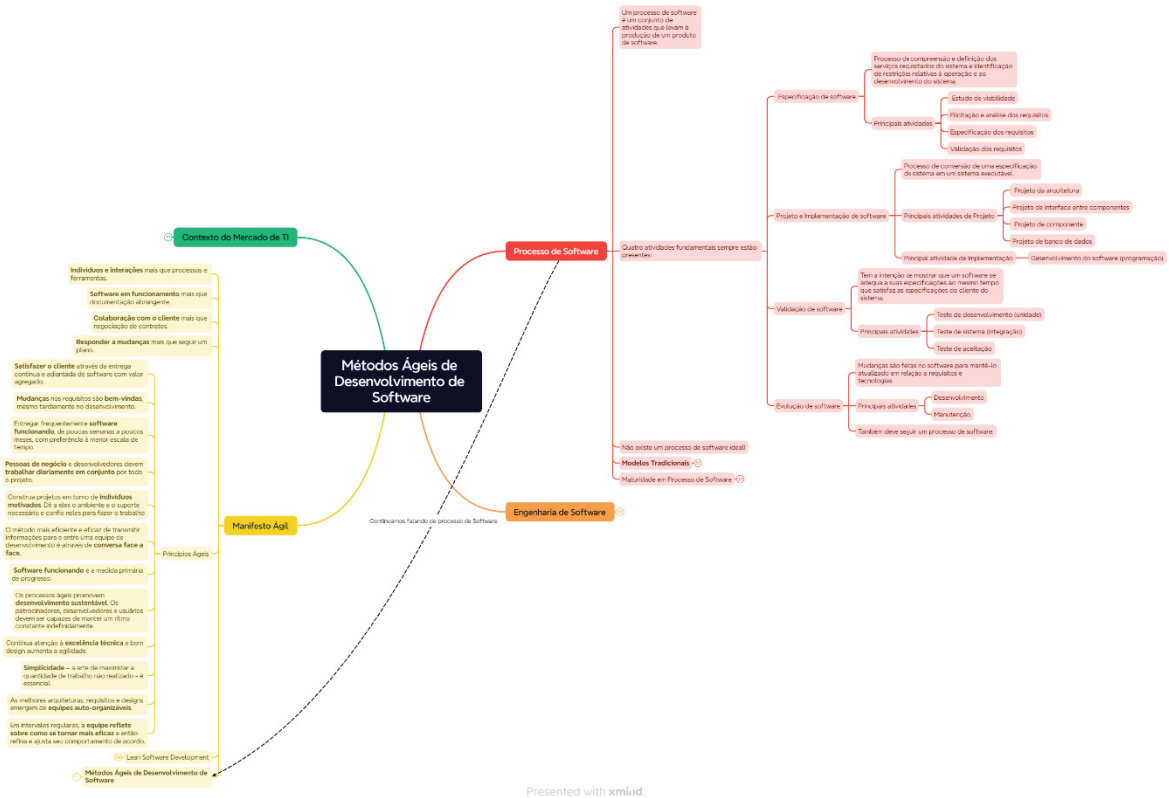
2.1 ARTEFATOS DO PROJETO

FIGURA 1 – MAPA MENTAL MADS PARTE 1



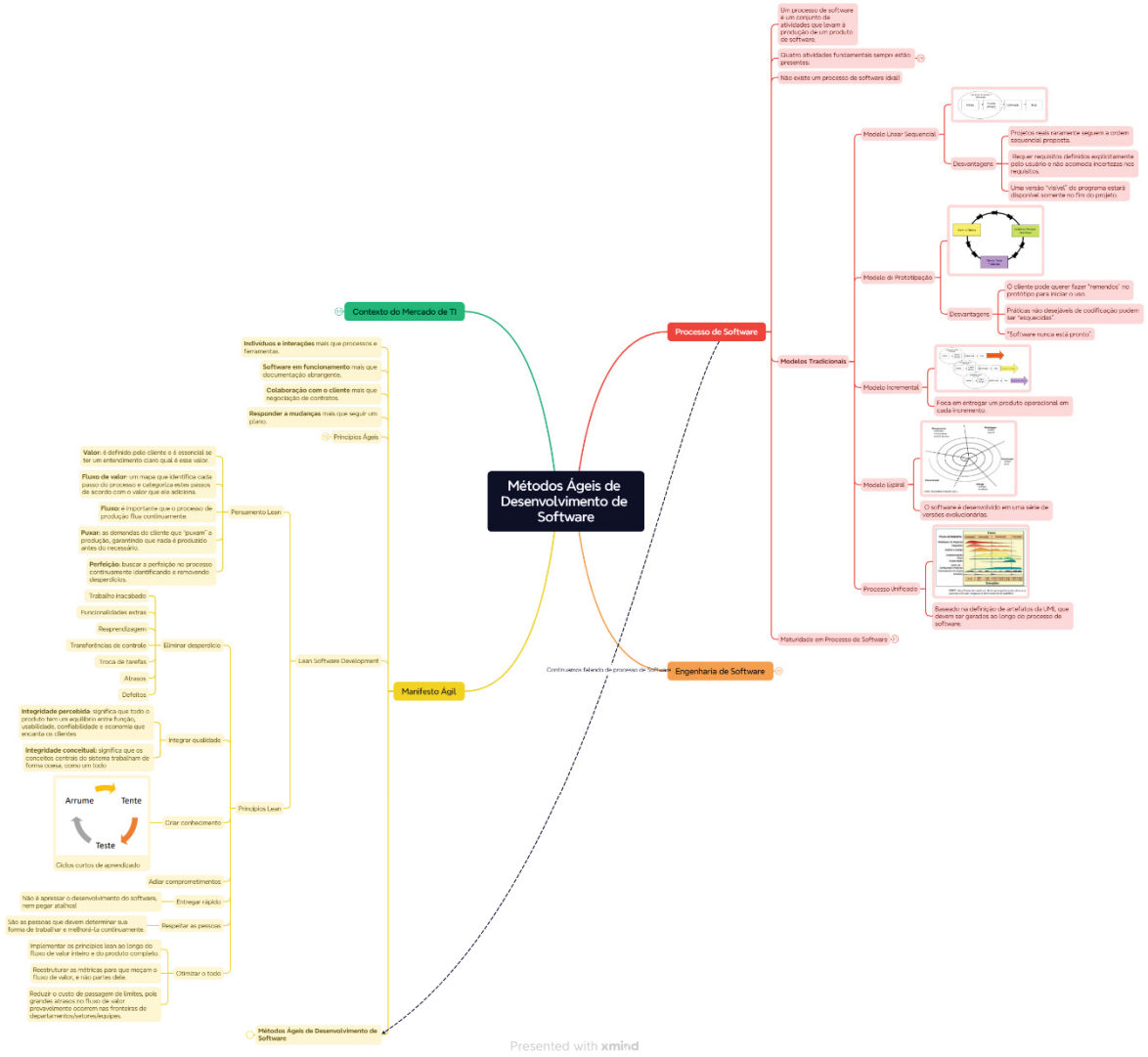
FONTE: A autora (2025).

FIGURA 2 – MAPA MENTAL MADS PARTE 2



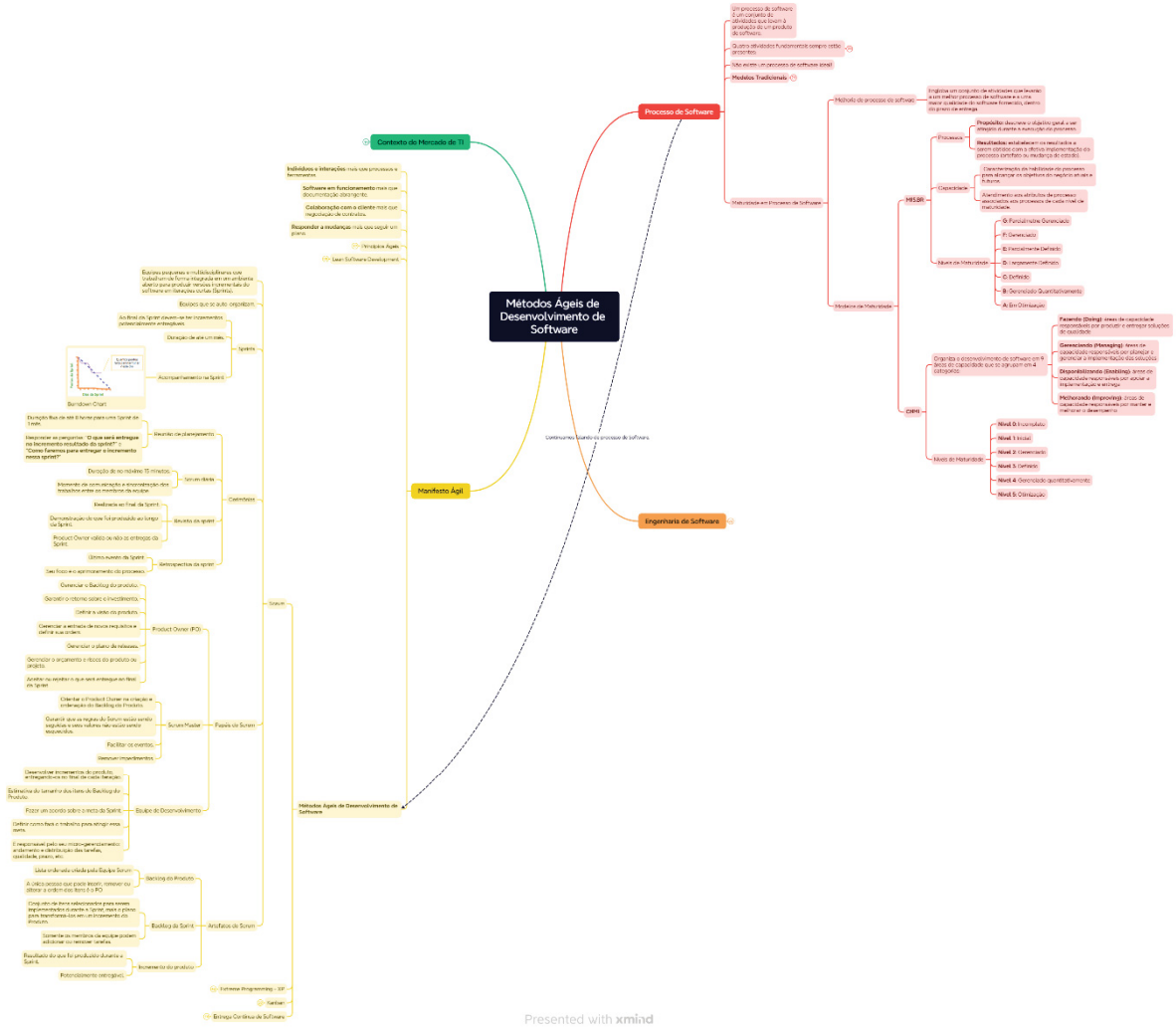
FONTE: A autora (2025).

FIGURA 3 – MAPA MENTAL MADS PARTE 3



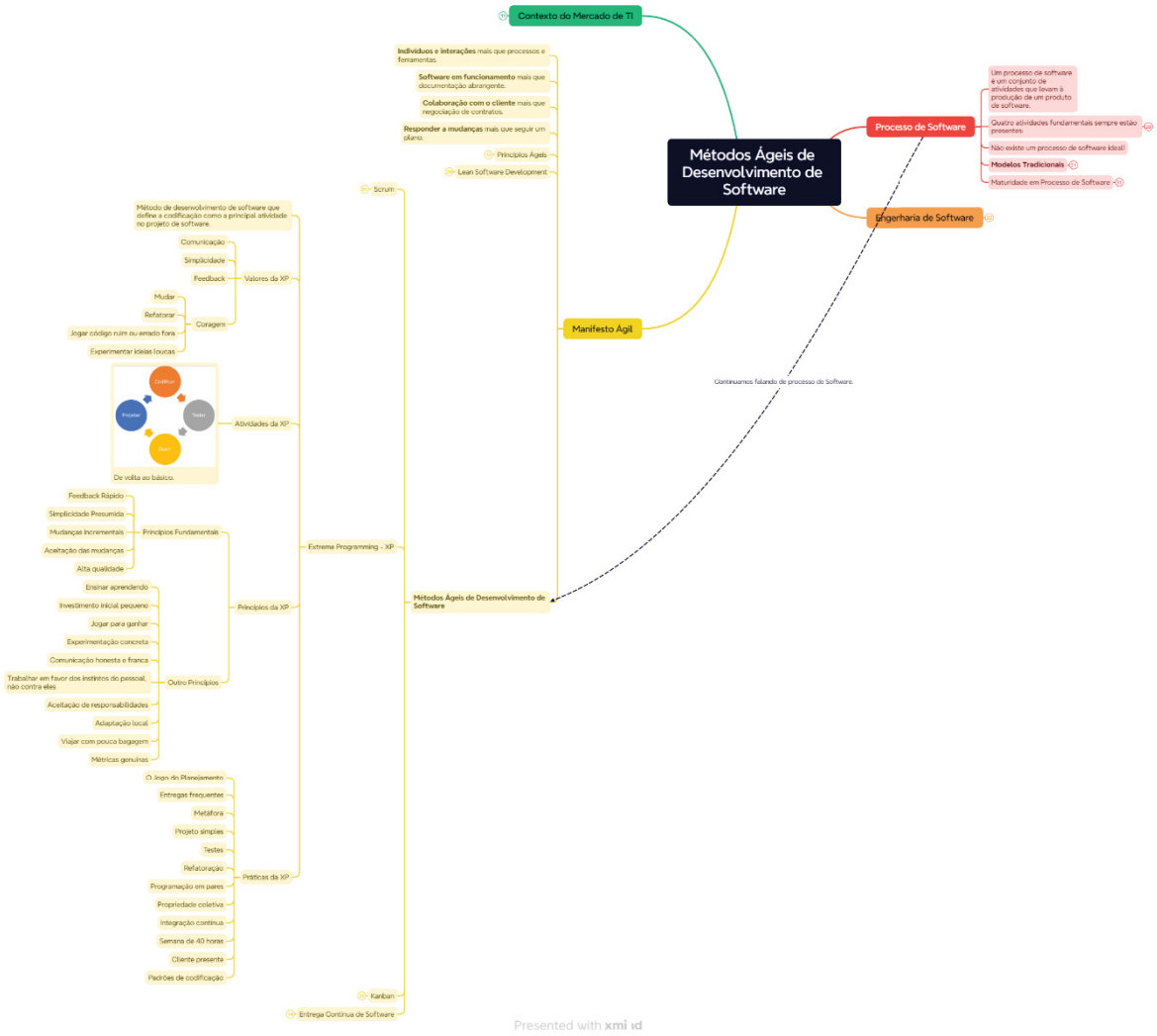
FONTE: A autora (2025).

FIGURA 4 – MAPA MENTAL MADS PARTE 4



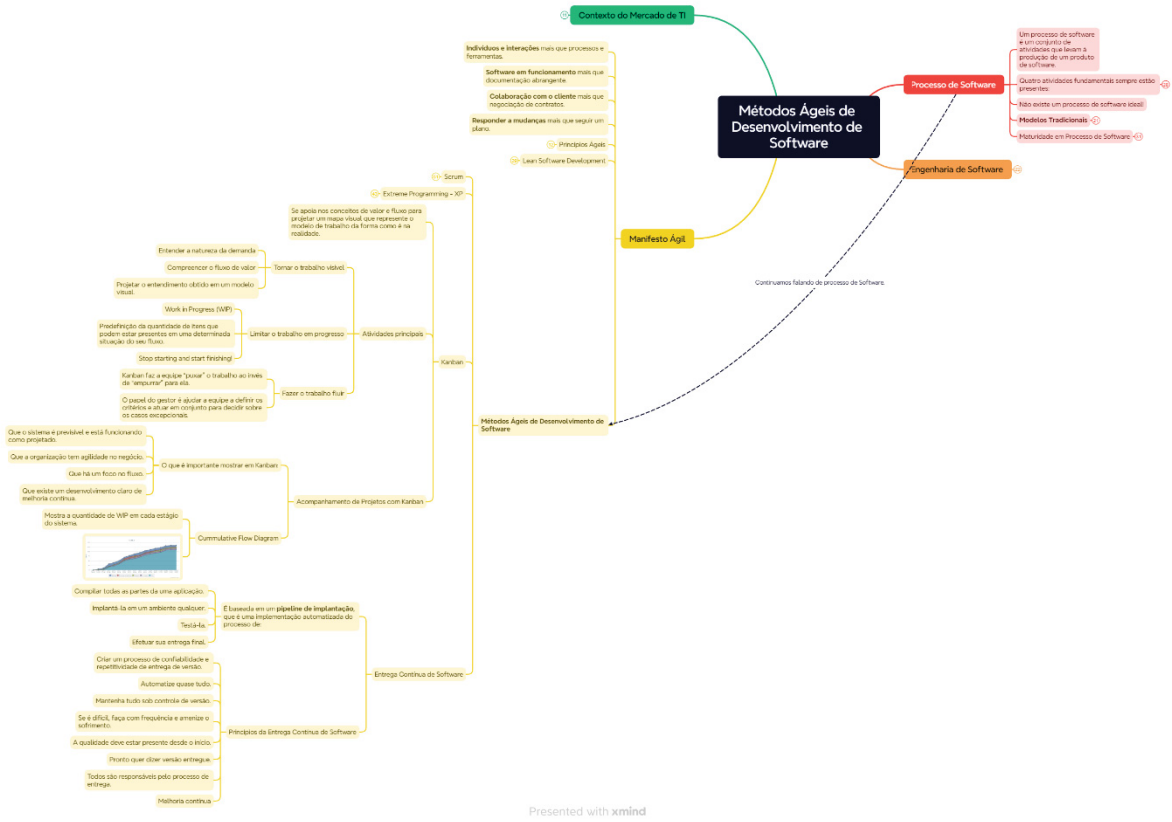
FONTE: A autora (2025).

FIGURA 5 – MAPA MENTAL MADS PARTE 5



FONTE: A autora (2025).

FIGURA 6 – MAPA MENTAL MADS PARTE 6



FONTE: A autora (2025).

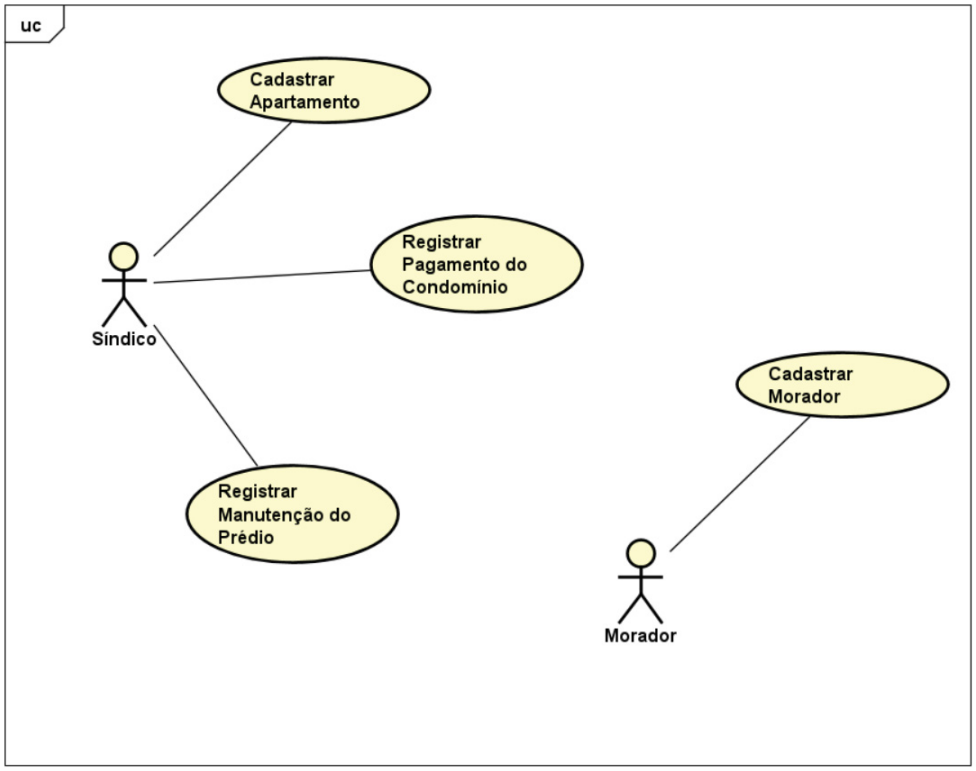
3 DISCIPLINA: MAG1 E MAG2 – MODELAGEM ÁGIL DE SOFTWARE 1 E 2

O projeto das disciplinas de Modelagem Ágil de Software I (MAG1) - Visão Funcional e Modelagem Ágil de Software II (MAG2) - Visão Estrutural, consiste na modelagem completa de um Sistema de Gestão de Condomínio. Este trabalho permite a aplicação prática de técnicas de análise, design e documentação de software de forma iterativa e colaborativa. O projeto é relevante no contexto do desenvolvimento ágil pois, com um foco primordial nos requisitos e Histórias de Usuário por meio da criação de Diagramas de Caso de Uso (Níveis 1 e 2) e Histórias de Usuário completas, reflete a prioridade em entender as necessidades do cliente de forma clara e concisa. As Histórias de Usuário, em particular, são a base para o planejamento de sprints e a comunicação eficaz entre a equipe de desenvolvimento e os *stakeholders*.

Além disso, a modelagem é evolutiva e adaptável, com MAG2 demonstrando a capacidade de evoluir o design do sistema de forma incremental através da elaboração de Diagramas de Classes com atributos associados, Tabelas do Banco de Dados e Diagramas de Sequência para todas as Histórias de Usuário. Dentro do contexto ágil, a modelagem não é um processo estático e completo no início, mas sim um esforço contínuo que se adapta às mudanças e aprendizados ao longo do projeto. A comunicação e colaboração são facilitadas pela utilização de diagramas UML (*Unified Modeling Language*), que servem como ferramentas eficazes para alinhar entendimentos e validar soluções no ambiente ágil, onde a colaboração é fundamental. Por fim, este projeto estabelece uma base sólida para implementação e testes, já que a modelagem detalhada do sistema fornece um guia claro para as fases seguintes, e uma estrutura bem definida desde a modelagem reduz ambiguidades e potenciais retrabalhos em etapas posteriores do desenvolvimento.

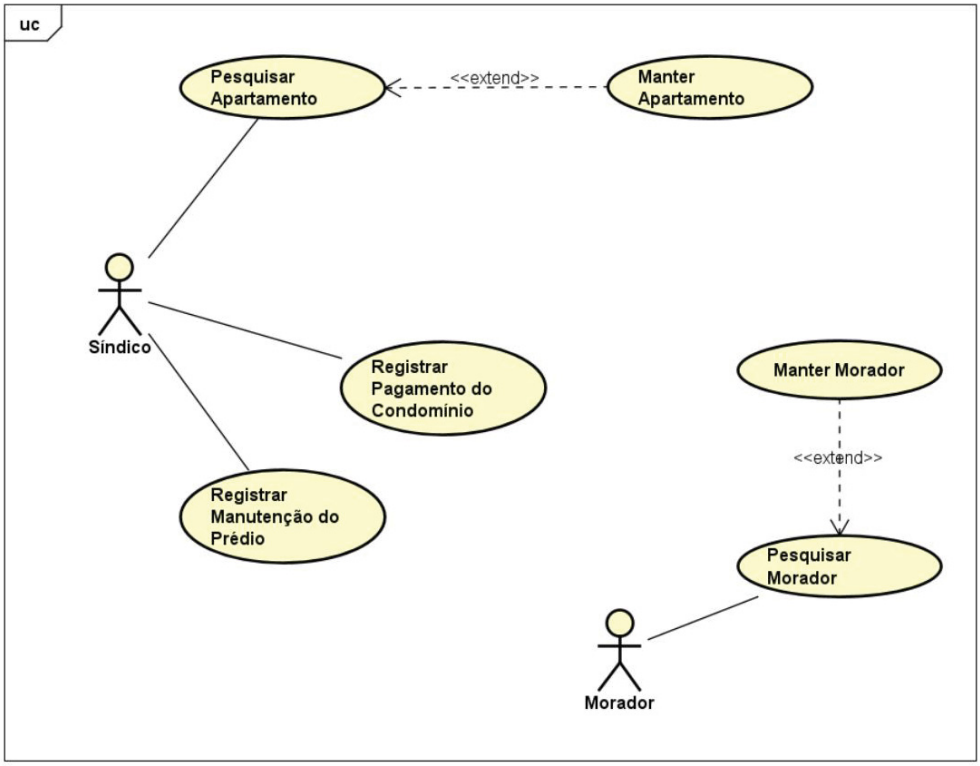
3.1 ARTEFATOS DO PROJETO

FIGURA 7 - DIAGRAMA DE CASO DE USO NÍVEL 1



FONTE: A autora (2025).

FIGURA 8 - DIAGRAMA DE CASO DE USO NÍVEL 2



FONTE: A autora (2025).

Histórias de Usuário

HU001 – Pesquisar Apartamento

SENDO o Síndico
QUERO pesquisar os apartamentos
PARA fazer manutenções nos seus dados.

DESENHO DA TELA

LISTA DE CRITÉRIOS DE ACEITAÇÃO

1. Deve preencher a tabela com todos os apartamentos cadastrados.
2. Deve permitir incluir um novo apartamento.
3. Deve permitir alterar os dados de um apartamento cadastrado.
4. Deve permitir excluir um apartamento cadastrado.
5. Deve permitir consultar os dados de um apartamento cadastrado.
6. Deve pesquisar os apartamentos na tabela da tela.
7. Deve voltar à tela anterior.

CRITÉRIOS DE ACEITAÇÃO - DETALHAMENTO

1. Deve preencher a tabela com todos os apartamentos cadastrados.

Dado que	
Quando	A tela é apresentada
Então	A tabela da tela deve ser preenchida com todos os apartamentos do banco de dados.

2. Deve permitir incluir um novo apartamento.

Dado que	Os apartamentos estão na tabela
Quando	O botão “Novo” é pressionado
Então	O sistema chama a HU002 – Manter Apartamento passando o parâmetro “Novo”.

3. Deve permitir alterar os dados de um apartamento cadastrado.

Dado que	Os apartamentos estão na tabela
Quando	O botão “Alterar” é pressionado em uma determinada linha
Então	O sistema chama a HU002 – Manter Apartamento passando o parâmetro “Alterar” e os dados do evento da linha.

4. Deve permitir excluir um apartamento.

Dado que	Os apartamentos estão na tabela
Quando	O botão “Excluir” é pressionado em uma determinada linha
Então	O sistema pede uma confirmação, exclui o apartamento da base de dados e refaz a tabela da tela lendo novamente o banco de dados.

5. Deve permitir consultar os dados de um apartamento cadastrado.

Dado que	Os apartamentos estão na tabela
Quando	O usuário clica em qualquer dado de um apartamento em uma determinada linha
Então	O sistema chama a HU002 – Manter Apartamento passando o parâmetro “Consultar” e os dados do evento da linha.

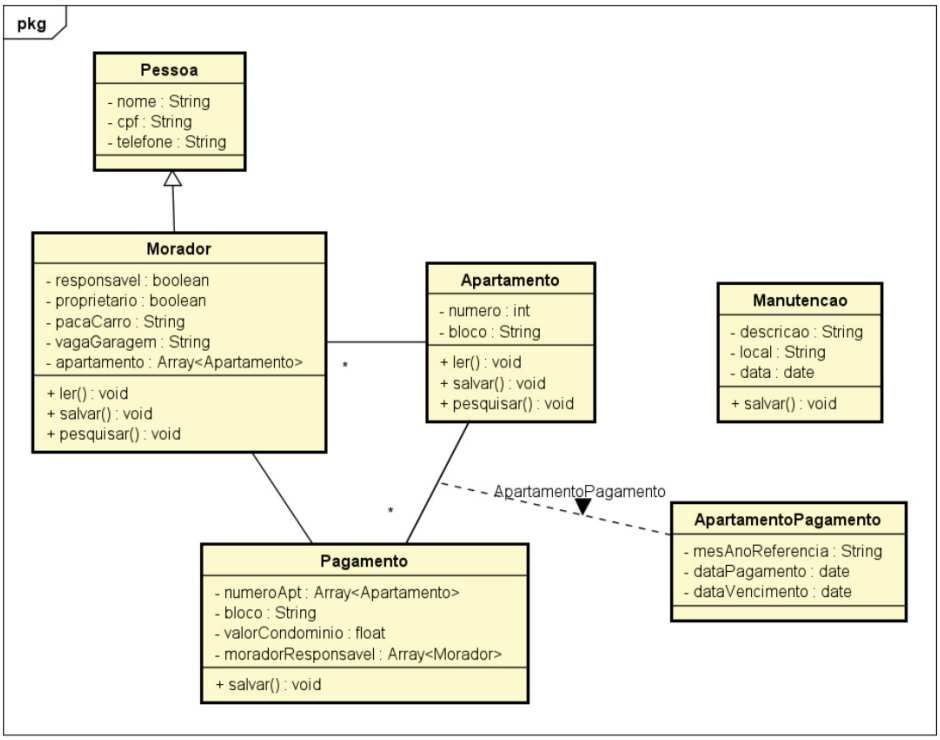
6. Deve pesquisar os apartamentos na tabela da tela.

Dado que	Os apartamentos estão na tabela
Quando	For informado um argumento de pesquisa
Então	O sistema filtra e apresenta na tela somente os apartamentos que obedeçam ao critério.

7. Deve voltar à tela anterior

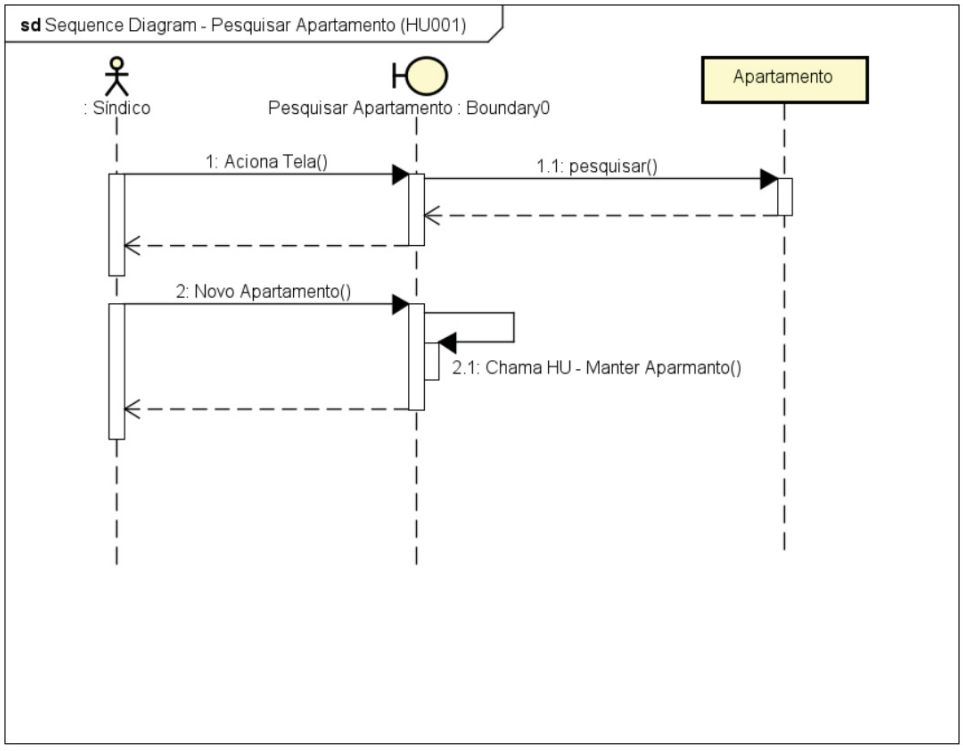
Dado que	Os apartamentos estão na tabela
Quando	O botão “Voltar” é pressionado
Então	O sistema volta para a tela anterior

FIGURA 9 - DIAGRAMA DE CLASSES



FONTE: A autora (2025).

FIGURA 10 - DIAGRAMA DE SEQUÊNCIA (HU001 – PESQUISAR APARTAMENTO)



FONTE: A autora (2025).

4 DISCIPLINA: GAP1 E GAP2 – GERENCIAMENTO ÁGIL DE PROJETOS DE SOFTWARE 1 E 2

O projeto das disciplinas de Gerenciamento Ágil de Projetos I (GAP1) e Gerenciamento Ágil de Projetos II (GAP2) oferece uma experiência prática e aprofundada na gestão de projetos sob a ótica ágil. Em GAP1, o objetivo é elaborar um Plano de *Release* para um software concebido pela aluna, detalhando o cálculo da velocidade de desenvolvimento e a estruturação de sprints com datas de início e fim, além de Histórias de Usuário estimadas em pontos. Já em GAP2, o foco se desloca para a execução prática, com a simulação de um ciclo de 35 dias utilizando a ferramenta kanbanboardgame.com, onde o monitora-se o progresso, registra-se resultados e analisa-se o Diagrama de Fluxo Cumulativo (CFD).

Estes projetos traduzem os conceitos ágeis em práticas de gestão. O Plano de *Release* de GAP1 é uma ferramenta essencial para a previsibilidade e transparência em projetos ágeis, permitindo que *stakeholders* compreendam o cronograma de entregas e o valor esperado em cada iteração. A definição de Histórias de Usuário no formato "SENDO/QUERO/PARA" reforça a orientação ao valor e ao cliente, um pilar do Manifesto Ágil. A prática de calcular a velocidade e estimar em pontos é importante para o planejamento adaptativo e a melhoria contínua do processo de desenvolvimento. Em GAP2, a simulação do Kanban proporciona uma vivência real do fluxo de trabalho contínuo, da gestão visual e da identificação de gargalos. A análise do CFD e das métricas de receita ensina a importância de monitorar o desempenho do processo e do produto para tomar decisões baseadas em dados, garantindo que o projeto mantenha um ritmo sustentável e entregue valor de forma consistente.

4.1 ARTEFATOS DO PROJETO

Sistema de Gestão de Condomínio

Cálculo da Velocidade:

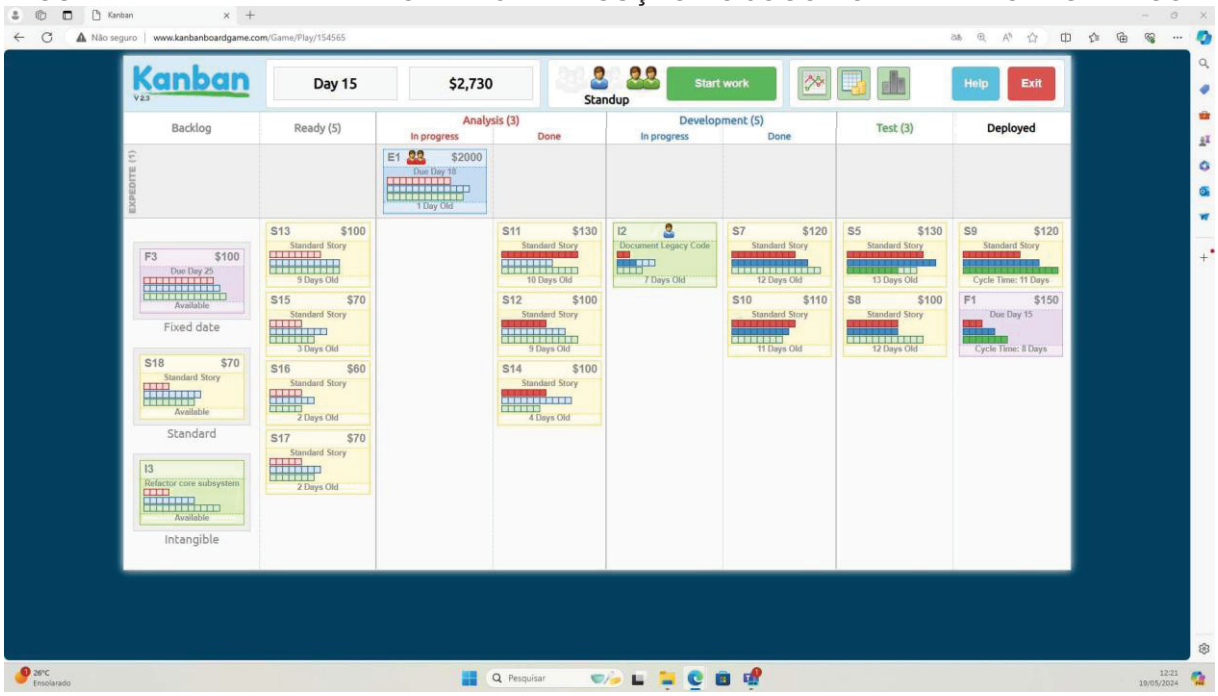
Horas disponíveis por dia:	6 h	Tamanho da Sprint:	2 semanas
Horas disponíveis por Sprint:	60 h	Velocidade:	7 pontos

Plano de *Release*:

Iteração/Sprint 1	Iteração/Sprint 2	Iteração/Sprint 3	Iteração/Sprint 4
Data Início: 06/05/2024	Data Início: 20/05/2024	Data Início: 03/06/2024	Data Início: 17/06/2024
Data Fim: 17/05/2024	Data Fim: 31/05/2024	Data Fim: 14/06/2024	Data Fim: 28/06/2024
< HU001 - Pesquisar Apartamento > SENDO o Síndico QUERO pesquisar os apartamentos PARA fazer manutenções nos seus dados. ESTIMATIVA (3 pontos)	**< HU005 – Pesquisar Morador >** SENDO o Síndico QUERO pesquisar os moradores PARA fazer manutenções nos seus dados. ESTIMATIVA (3 pontos)	**< HU003 – Registrar Pagamento do Condomínio >** SENDO o Síndico QUERO registrar o pagamento do condomínio de um apartamento PARA controle das finanças do condomínio ESTIMATIVA (2 pontos)	**< HU0018 – Convocar assembleia do condomínio >** SENDO o Síndico QUERO convocar uma assembleia PARA discutir assuntos importantes para moradores. ESTIMATIVA (2 pontos)
< HU002 - Manter Apartamento > SENDO o Síndico QUERO manter os dados dos apartamentos PARA que seus dados fiquem atualizados. ESTIMATIVA (4 pontos)	**< HU006 – Manter Morador >** SENDO o Síndico QUERO manter os dados dos moradores PARA que seus dados fiquem atualizados ESTIMATIVA (4 pontos)	**< HU004 – Registrar Manutenção do Prédio >** SENDO o Síndico QUERO registrar as manutenções do prédio PARA organização do condomínio. ESTIMATIVA (2 pontos)	**< HU0019 – Salvar ata da assembleia do condomínio >** SENDO o Síndico QUERO salvar a ata da assembleia PARA os moradores terem acesso ESTIMATIVA (2 pontos)
		< HU0014 – Consultar manutenções agendadas > SENDO o Morador QUERO ver se existem manutenções agendadas PARA tomar conhecimento. ESTIMATIVA (2 pontos)	**< HU0020 – Consultar assembleia >** SENDO o Morador QUERO ver assembleias agendadas PARA tomar conhecimento ESTIMATIVA (1 pontos)
		< HU0015 – Consultar histórico de pagamento de condomínio > SENDO o Morador QUERO consultar histórico de pagamento de condomínio do meu apartamento PARA verificar se existem pagamentos anteriores vencidos. ESTIMATIVA (1 ponto)	**< HU0021 – Baixar atas de assembleias >** SENDO o Morador QUERO baixar as atas de assembleias passadas PARA tomar conhecimento das decisões tomadas. ESTIMATIVA (2 pontos)

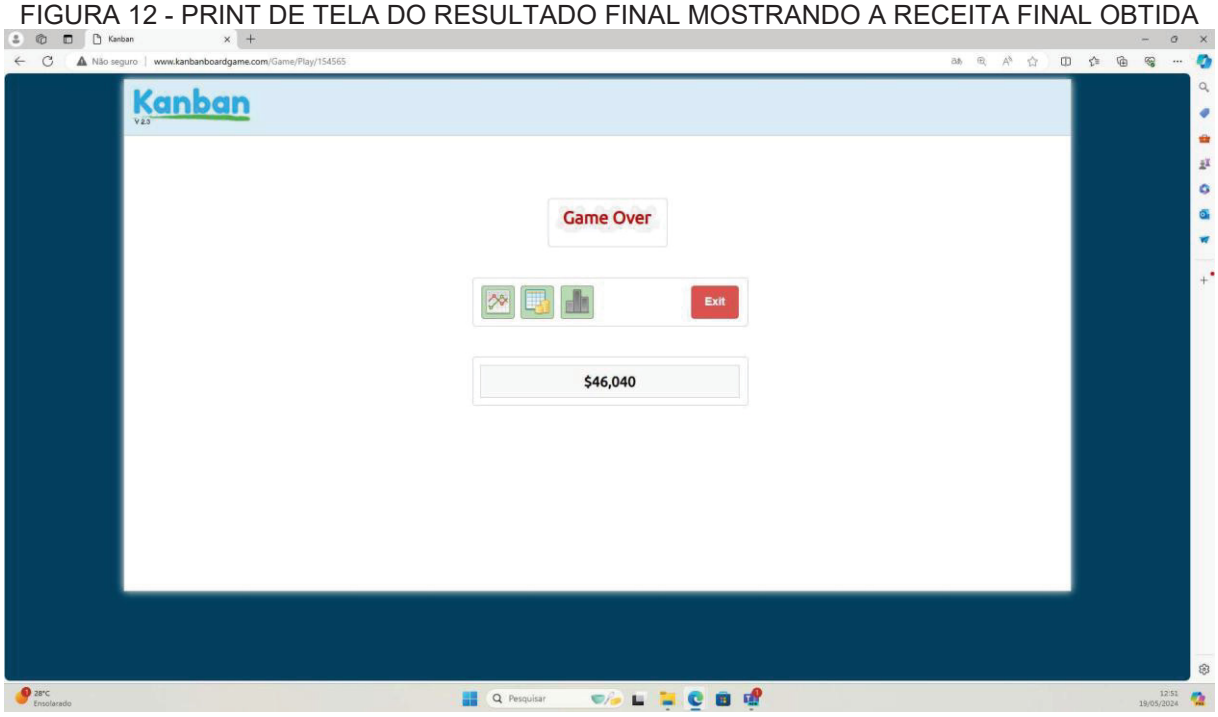
Iteração/Sprint 5	Iteração/Sprint 6	Iteração/Sprint 7	Iteração/Sprint 8
Data Início: 01/07/2024	Data Início: 15/07/2024	Data Início: 29/07/2024	Data Início: 12/08/2024
Data Fim: 12/07/2024	Data Fim: 26/07/2024	Data Fim: 09/08/2024	Data Fim: 23/08/2024
< HU009 – Registrar pagamento de multa > SENDO o Síndico QUERO registrar o pagamento de multa de um apartamento PARA controle das finanças do condomínio. ESTIMATIVA (2 pontos)	**< HU0022 – Salvar relatório de despesas do condomínio >** SENDO o Síndico QUERO salvar o relatório mensal de despesas do condomínio PARA os moradores terem acesso. ESTIMATIVA (3 pontos)	**< HU0023 – Baixar o relatório de despesas do condomínio >** SENDO o Morador QUERO baixar o relatório mensal de despesas do condomínio PARA tomar conhecimento dos gastos do condomínio. ESTIMATIVA (3 pontos)	**< HU0013 – Responder mensagens de moradores >** SENDO o Síndico QUERO responder as mensagens dos moradores PARA prestar assistência aos moradores. ESTIMATIVA (3 pontos)
< HU0010 – Verificar se existem multas/notificações para apartamento > SENDO o Morador QUERO verificar se existem notificações ou multas não pagas aplicadas para meu apartamento PARA tomar conhecimento do ocorrido e efetuar o pagamento. ESTIMATIVA (3 pontos)	**< HU0017 – Agendar o salão de festas do condomínio >** SENDO o Morador QUERO agendar o salão de festas do condomínio PARA realizar uma festa. ESTIMATIVA (3 pontos)	**< HU0011 – Enviar mensagem ao síndico >** SENDO o Morador QUERO enviar uma mensagem ao síndico PARA reportar um problema ou fazer uma solicitação ao síndico. ESTIMATIVA (3 pontos)	**< HU0012 – Verificar se existem mensagens de moradores ainda não respondidas>** SENDO o Síndico QUERO verificar se existem mensagens de moradores ainda não respondidas PARA tomar conhecimento e respondê-las. ESTIMATIVA (3 pontos)
< HU008 – Aplicar multa > SENDO o Síndico QUERO aplicar multa a um apartamento devido a descumprimento das regras do condomínio PARA manter a ordem do condomínio. ESTIMATIVA (1 pontos)	**< HU0016 – Verificar quando o salão de festas está disponível >** SENDO o Morador QUERO verificar quais datas o salão de festas do condomínio está disponível PARA agendar o uso do espaço. ESTIMATIVA (1 pontos)	**< HU0025 – Salvar nova cópia do regimento do condomínio >** SENDO o Síndico QUERO salvar nova cópia do regimento do condomínio PARA os moradores terem acesso. ESTIMATIVA (1 ponto)	**< HU0024 – Baixar o regimento do condomínio >** SENDO o Morador QUERO baixar o regimento do condomínio PARA consultá-lo. ESTIMATIVA (1 pontos)
< HU007 – Aplicar notificação > SENDO o Síndico QUERO aplicar notificação a um apartamento devido a descumprimento das regras do condomínio PARA manter a ordem do condomínio. ESTIMATIVA (1 pontos)			

FIGURA 11 - PRINT DE TELA DO DIA 15 A EXECUÇÃO DO JOGO NO KANBANBOADGAME.COM



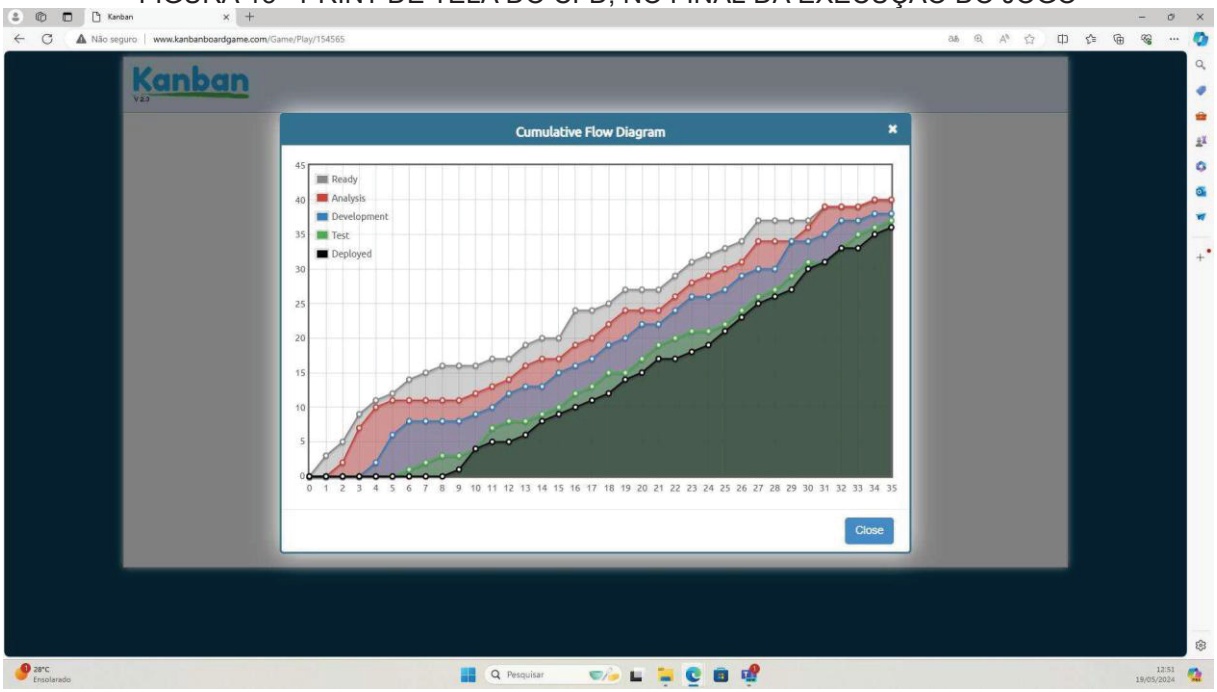
FONTE: A autora (2025).

FIGURA 12 - PRINT DE TELA DO RESULTADO FINAL MOSTRANDO A RECEITA FINAL OBTIDA



FONTE: A autora (2025).

FIGURA 13 - PRINT DE TELA DO CFD, NO FINAL DA EXECUÇÃO DO JOGO



FONTE: A autora (2025).

5 DISCIPLINA: INTRO – INTRODUÇÃO À PROGRAMAÇÃO

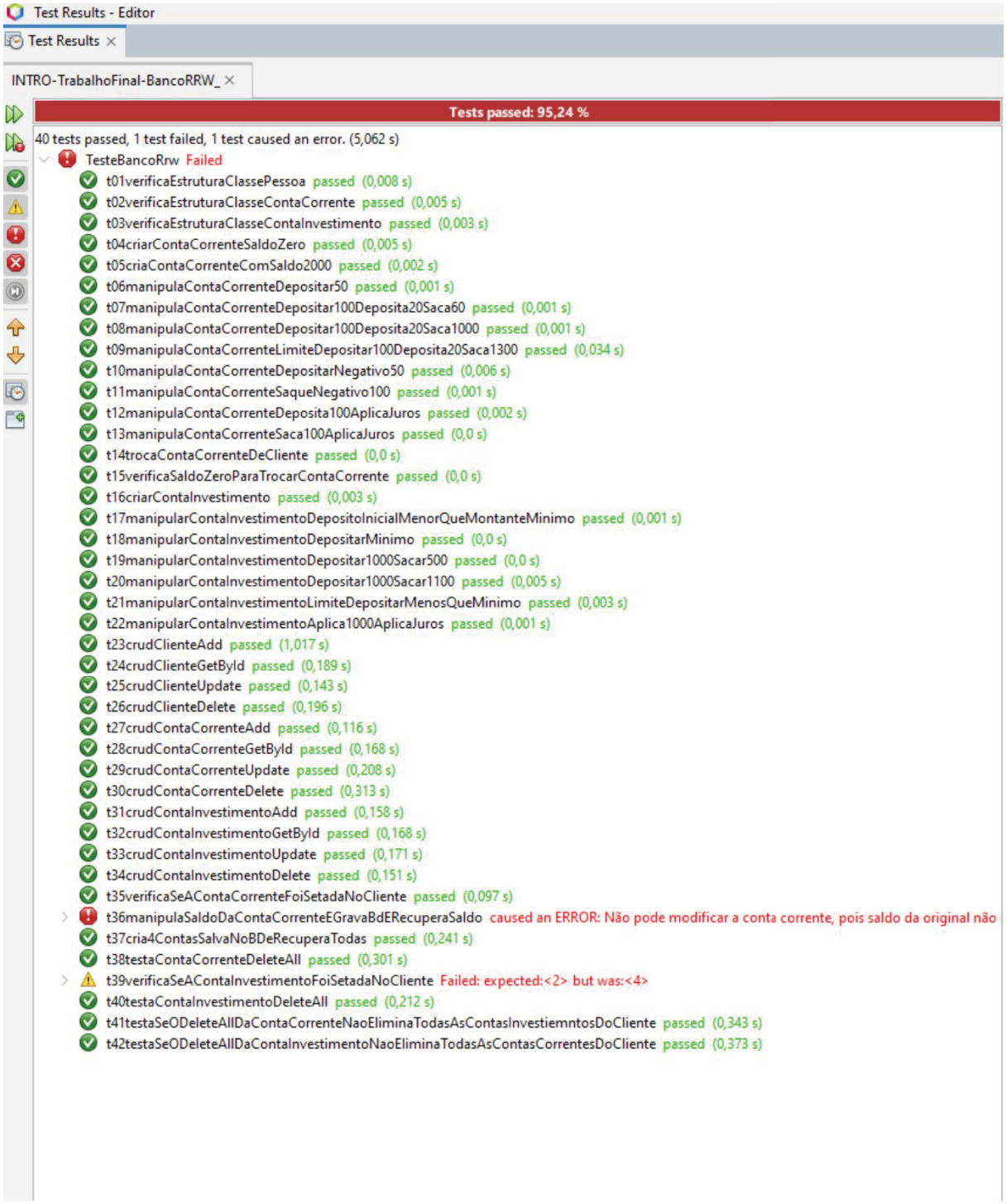
O projeto da disciplina de Introdução à Programação (INTRO) tem como objetivo principal a implementação do *backend* de um sistema bancário simplificado em Java. Este sistema visa o controle de cadastro de clientes, contas correntes e contas investimento. A essência do trabalho reside em construir as classes do sistema de forma que todos os testes unitários fornecidos, escritos em JUnit (total de 42 casos de teste), funcionem corretamente, com a meta de deixar 40 casos de teste "verdes" simultaneamente para atingir a nota máxima. Além disso, é fundamental a utilização de um Banco de Dados MySQL para persistência dos dados, cujo script DDL é fornecido.

O projeto enfatiza a qualidade do código desde as fases iniciais do aprendizado. A exigência de que os testes unitários funcionem promove a prática de *Test-Driven Development* (TDD) de forma implícita, incentivando os alunos a escreverem código que seja testável, robusto e livre de defeitos. Esta é uma prática fundamental em ambientes ágeis, onde a entrega contínua de software funcional e de alta qualidade é primordial. O projeto capacita a aluna a desenvolver uma base sólida em programação orientada a objetos e a interagir com bancos de dados, habilidades essenciais para qualquer desenvolvedor ágil.

A disciplina de Introdução à Programação é um pilar fundamental da especialização, pois fornece a base técnica e a mentalidade de qualidade necessárias para o desenvolvimento ágil de software. Ela não apenas ensina a programar em Java, mas também a programar com qualidade, testabilidade e integração, preparando os alunos para os desafios de construir sistemas robustos e eficientes em um ambiente ágil.

5.1 ARTEFATOS DO PROJETO

FIGURA 14 - EVIDÊNCIAS DOS TESTES QUE FUNCIONARAM COLETADAS DO NETBEANS



FONTE: A autora (2025).

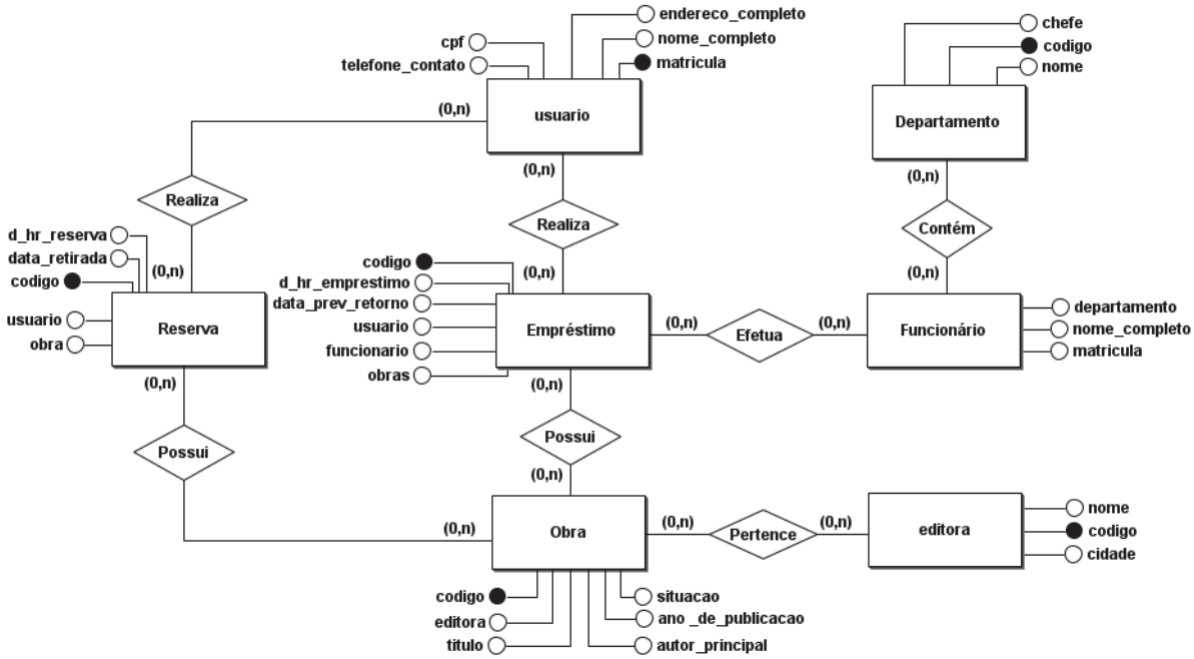
6 DISCIPLINA: BD – BANCO DE DADOS

O projeto da disciplina de Banco de Dados (BD) é dividido em duas questões principais que abordam a modelagem e a implementação de bancos de dados. A primeira questão foca na modelagem de dados para um sistema de controle de biblioteca, exigindo a criação de um Modelo Entidade-Relacionamento Conceitual e um Modelo Lógico (Diagrama de Entidade Relacionamento com tabelas, campos, chaves primárias e estrangeiras). A segunda questão teve como tema um sistema de controle de vacinação (perspectiva do posto de saúde) e foi desenvolvido um modelo lógico que inclui pelo menos um relacionamento de cada tipo (1x1, 1xN e NxN) entre as tabelas. Também foi gerado o script SQL para criação das tabelas e fornecido dados como exemplo.

A importância deste projeto no desenvolvimento ágil de software é estabelecer a base para a persistência e organização dos dados, um componente crítico em qualquer sistema de software. Em um ambiente ágil, onde os requisitos podem evoluir e as entregas são contínuas, a capacidade de modelar bancos de dados de forma eficiente e flexível é fundamental. A modelagem conceitual e lógica, como praticada neste projeto, permite a visualização da estrutura dos dados de maneira clara, facilitando a comunicação e a compreensão entre a equipe de desenvolvimento e os *stakeholders.* A exigência de criar scripts SQL e exemplos de dados garante que não apenas a teoria da modelagem seja compreendida, mas também que seja aplicada, desenvolvendo habilidades essenciais para a implementação e manutenção de bancos de dados em cenários reais. A ênfase em diferentes tipos de relacionamento entre tabelas prepara a aluna para projetar esquemas de banco de dados robustos e escaláveis, capazes de suportar as mudanças frequentes inerentes aos projetos ágeis.

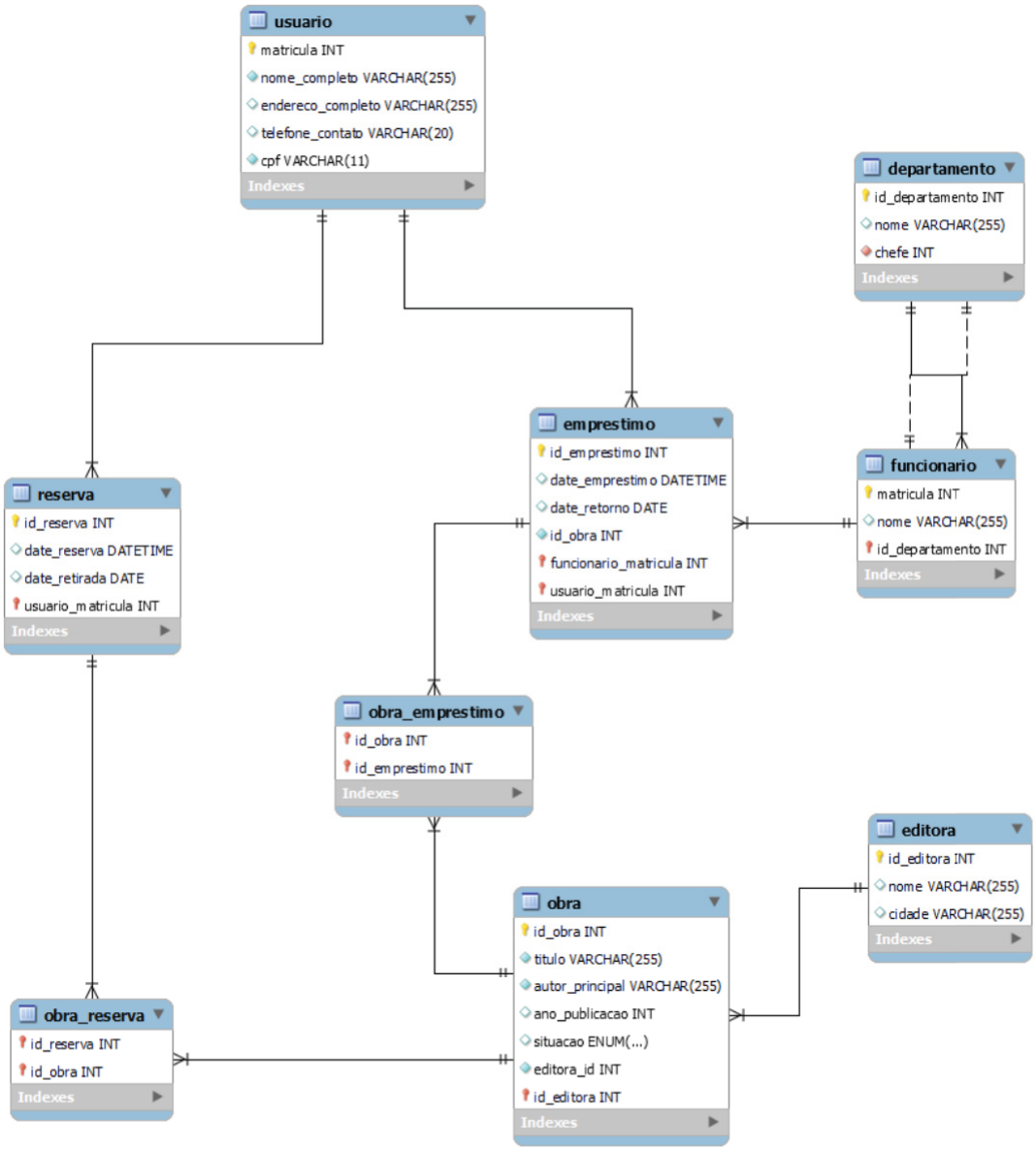
6.1 ARTEFATOS DO PROJETO

FIGURA 15 – MODELO CONCEITUAL PARA CONTROLE DE BIBLIOTECA



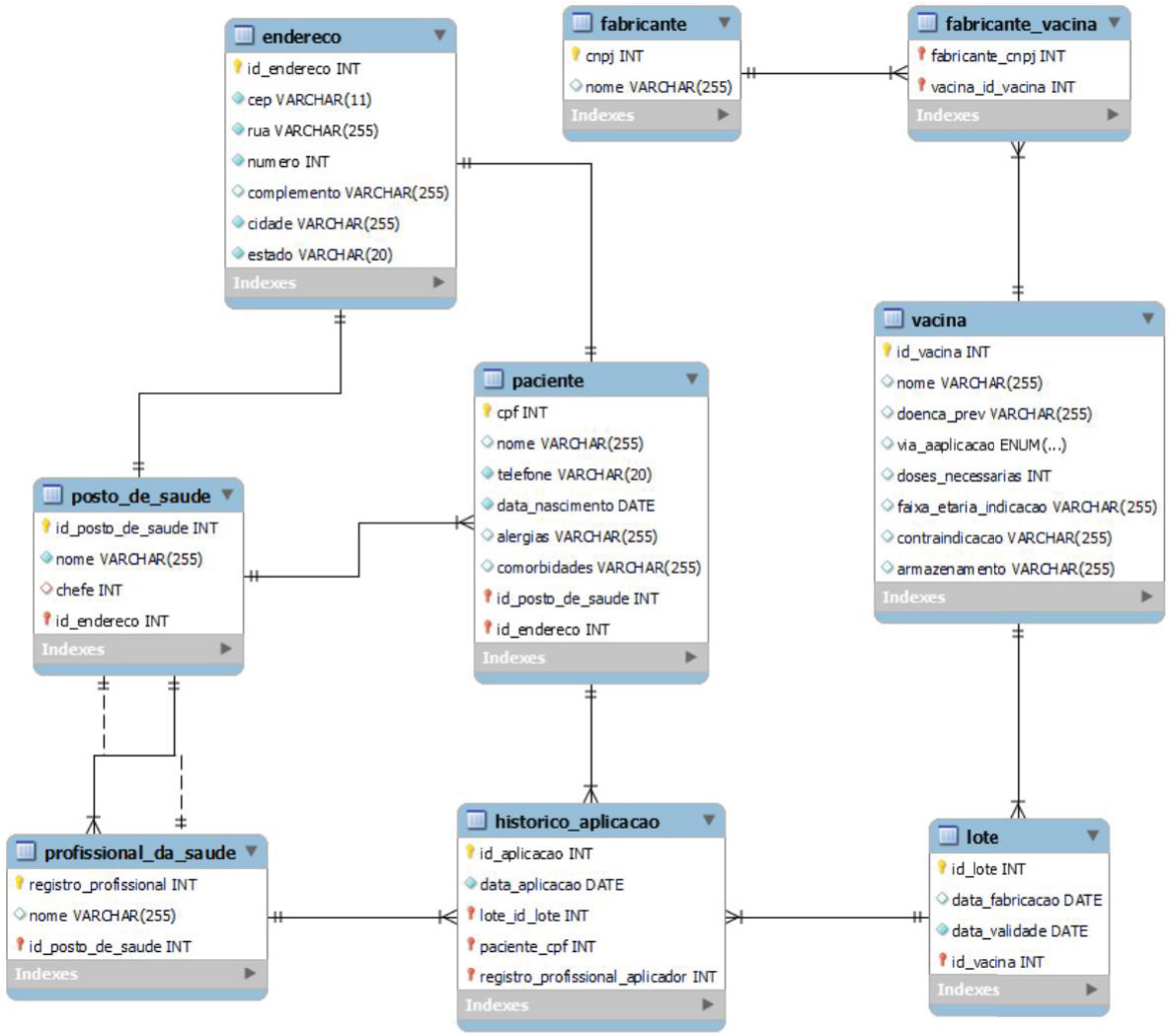
FONTE: A autora (2025).

FIGURA 16 – MODELO LÓGICO PARA CONTROLE DE BIBLIOTECA



FONTE: A autora (2025).

FIGURA 17 – MODELO LÓGICO PARA CONTROLE DE VACINAÇÃO



FONTE: A autora (2025).

Script SQL para criação das tabelas

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,E
RROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`profissional_da_saude`
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`profissional_da_saude` (
  `registro_profissional` INT NOT NULL,
  `nome` VARCHAR(255) NULL,
  `id_posto_de_saude` INT NOT NULL,
  PRIMARY KEY (`registro_profissional`, `id_posto_de_saude`),
  INDEX `fk_profissional_da_saude_posto_de_saude1_idx` (`id_posto_de_saude` ASC) VISIBLE,
  CONSTRAINT `fk_profissional_da_saude_posto_de_saude1`
    FOREIGN KEY (`id_posto_de_saude`)
    REFERENCES `mydb`.`posto_de_saude` (`id_posto_de_saude`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`endereco`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`endereco` (
  `id_endereco` INT NOT NULL,
  `cep` VARCHAR(11) NOT NULL,
  `rua` VARCHAR(255) NOT NULL,
  `numero` INT NOT NULL,
  `complemento` VARCHAR(255) NULL,
  `cidade` VARCHAR(255) NOT NULL,
  `estado` VARCHAR(20) NOT NULL,
  PRIMARY KEY (`id_endereco`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`posto_de_saude`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`posto_de_saude` (
  `id_posto_de_saude` INT NOT NULL,
  `nome` VARCHAR(255) NOT NULL,
  `chefe` INT NULL,
  `id_endereco` INT NOT NULL,
  PRIMARY KEY (`id_posto_de_saude`, `id_endereco`),
  INDEX `fk_posto_de_saude_chefe_idx` (`chefe` ASC) VISIBLE,
  INDEX `fk_posto_de_saude_endereco1_idx` (`id_endereco` ASC) VISIBLE,
  CONSTRAINT `fk_posto_de_saude_chefe`
    FOREIGN KEY (`chefe`)
    REFERENCES `mydb`.`profissional_da_saude` (`registro_profissional`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_posto_de_saude_endereco1`
    FOREIGN KEY (`id_endereco`)
    REFERENCES `mydb`.`endereco` (`id_endereco`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`vacina`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`vacina` (
  `id_vacina` INT NOT NULL,
  `nome` VARCHAR(255) NULL,
  `doenca_prev` VARCHAR(255) NULL,
  `via_aaplicacao` ENUM('oral', 'intradermica', 'subcutanea', 'intramuscular') NULL,
  `doses_necessarias` INT NULL,
  `faixa_etaria_indicacao` VARCHAR(255) NULL,
  `contraindicacao` VARCHAR(255) NULL,
  `armazenamento` VARCHAR(255) NULL,
 PRIMARY KEY (`id_vacina`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`fabricante`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`fabricante` (
  `cnpj` INT NOT NULL,
  `nome` VARCHAR(255) NULL,
 PRIMARY KEY (`cnpj`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`lote`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`lote` (
  `id_lote` INT NOT NULL,
  `data_fabricacao` DATE NULL,
  `data_validade` DATE NOT NULL,
  `id_vacina` INT NOT NULL,
 PRIMARY KEY (`id_lote`, `id_vacina`),
 INDEX `fk_lote_vacina1_idx` (`id_vacina` ASC) VISIBLE,
 CONSTRAINT `fk_lote_vacina1`
   FOREIGN KEY (`id_vacina`)
   REFERENCES `mydb`.`vacina` (`id_vacina`)
   ON DELETE NO ACTION
   ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`paciente`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`paciente` (
  `cpf` INT NOT NULL,
  `nome` VARCHAR(255) NULL,
  `telefone` VARCHAR(20) NOT NULL,
  `data_nascimento` DATE NOT NULL,
  `alergias` VARCHAR(255) NULL,
  `comorbidades` VARCHAR(255) NULL,
  `id_posto_de_saude` INT NOT NULL,
  `id_endereco` INT NOT NULL,
 PRIMARY KEY (`cpf`, `id_posto_de_saude`, `id_endereco`),
 INDEX `fk_paciente_posto_de_saude1_idx` (`id_posto_de_saude` ASC) VISIBLE,
 INDEX `fk_paciente_endereco1_idx` (`id_endereco` ASC) VISIBLE,
 CONSTRAINT `fk_paciente_posto_de_saude1`
```

```
  FOREIGN KEY (`id_posto_de_saude`)
  REFERENCES `mydb`.`posto_de_saude` (`id_posto_de_saude`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
 CONSTRAINT `fk_paciente_endereco1`
  FOREIGN KEY (`id_endereco`)
  REFERENCES `mydb`.`endereco` (`id_endereco`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`historico_aplicacao`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`historico_aplicacao` (
 `id_aplicacao` INT NOT NULL,
 `data_aplicacao` DATE NOT NULL,
 `lote_id_lote` INT NOT NULL,
 `paciente_cpf` INT NOT NULL,
 `registro_profissional_aplicador` INT NOT NULL,
 PRIMARY KEY (`id_aplicacao`, `lote_id_lote`, `paciente_cpf`, `registro_profissional_aplicador`),
 INDEX `fk_aplicacao_lote1_idx` (`lote_id_lote` ASC) VISIBLE,
 INDEX `fk_aplicacao_paciente1_idx` (`paciente_cpf` ASC) VISIBLE,
 INDEX `fk_aplicacao_profissional_da_saude1_idx` (`registro_profissional_aplicador` ASC) VISIBLE,
 CONSTRAINT `fk_aplicacao_lote1`
  FOREIGN KEY (`lote_id_lote`)
  REFERENCES `mydb`.`lote` (`id_lote`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
 CONSTRAINT `fk_aplicacao_paciente1`
  FOREIGN KEY (`paciente_cpf`)
  REFERENCES `mydb`.`paciente` (`cpf`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
 CONSTRAINT `fk_aplicacao_profissional_da_saude1`
  FOREIGN KEY (`registro_profissional_aplicador`)
  REFERENCES `mydb`.`profissional_da_saude` (`registro_profissional`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`fabricante_vacina`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`fabricante_vacina` (
 `fabricante_cnpj` INT NOT NULL,
 `vacina_id_vacina` INT NOT NULL,
 PRIMARY KEY (`fabricante_cnpj`, `vacina_id_vacina`),
 INDEX `fk_fabricante_vacina_vacina1_idx` (`vacina_id_vacina` ASC) VISIBLE,
 CONSTRAINT `fk_fabricante_vacina_fabricante1`
  FOREIGN KEY (`fabricante_cnpj`)
  REFERENCES `mydb`.`fabricante` (`cnpj`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
 CONSTRAINT `fk_fabricante_vacina_vacina1`
```

```
  FOREIGN KEY (`vacina_id_vacina`)
  REFERENCES `mydb`.`vacina` (`id_vacina`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

7 DISCIPLINA: AAP – ASPECTOS ÁGEIS DE PROGRAMAÇÃO

O projeto da disciplina de Aspectos Ágeis de Programação (AAP) tem como objetivo aprimorar a qualidade do código através da aplicação de princípios de *Clean Code*. A tarefa consiste em refatorar um código existente (neste caso, uma implementação do algoritmo Bubble Sort) para torná-lo mais "limpo", legível e manutenível, evidenciando as alterações realizadas e explicando sua justificativa.

A qualidade do código é um fundamental para a agilidade e sustentabilidade de qualquer projeto. Em ambientes ágeis, onde a velocidade de entrega é valorizada, é fácil acumular "dívidas técnicas" caso as boas práticas de codificação não sejam seguidas. Um código limpo e bem estruturado, conforme preconizado por *Clean Code*, facilita a compreensão, a manutenção, a evolução e a colaboração entre os desenvolvedores. Isso se traduz em menos problemas, menor tempo gasto em refatorações futuras e maior capacidade de resposta a mudanças, características essenciais para equipes ágeis de alta performance.

O projeto da disciplina de Aspectos Ágeis de Programação desenvolve a capacidade de escrita de código de alta qualidade, legível e manutenível. Ele não apenas solidifica as habilidades de programação, mas também atua como um integrador, garantindo que as práticas de codificação estejam alinhadas com os princípios ágeis e contribuam diretamente para a entrega contínua de software de valor e com estabilidade.

7.1 ARTEFATOS DO PROJETO

```java
package BubbleSort;

// Luana Corsi Antonio
// Implementação otimizada em Java do Bubble sort
// Código extraído de https://www.geeksforgeeks.org/bubble-sort/

import java.io.*;

class BubbleSort_LuanaCorsi {

    // Driver program
    public static void main(String args[])
    {
        int arr[] = { 64, 34, 25, 12, 22, 11, 90 };
```

```
        bubbleSort(arr);
    }

    static void bubbleSort(int arr[])
    {
        sortArr(arr);
        printArray(arr);

    }

    private static void sortArr(int[] arr) {
        boolean isSwapped;

        for (int i = 0; i < arr.length; i++) {

            for (int j = i; j < arr.length; j++) {
                if (arr[i] > arr[j]) {
                    swap(arr, i, j);
                }
            }
        }
    }

    private static void swap(int[] arr, int i, int j) {
        int temp;
        // Swap arr[j] and arr[j+1]
        temp = arr[j];
        arr[j] = arr[i];
        arr[i] = temp;
    }

    // Function to print an array
    static void printArray(int arr[])
    {
        System.out.println("Array ordenado: ");
        for (int k = 0; k < arr.length; k++)
            System.out.print(arr[k] + " ");
        System.out.println();
    }

}

// This code is contributed
// by Nikita Tiwari.

/*Alterações feitas:
1. Extraí método swap de bubbluSort.
2. Retirei o orgumento n do método bubbluSort e obtive o tamanho do array
com a função arr.lenght
```

```
adicionada diretamente onde o tamanho do array era necessário (linha 14 e
16).
3. Retirei o orgumento n do método printArray e obtive o tamanho do array
com a função arr.lenght
adicionada diretamente onde o tamanho do array era necessário (linha 41).
4. Declarei as variáveis i e j ano FOR em que seriam usadas (linhas 14 e
16).
5. Re-ordenei os métodos para que que os métodos chamados fiquem abaixo dos
chamadores.
6. Extraí o método sortArr.
7. Reduzi a complexidade ciclomática do método sortArr.

*/
```

8 DISCIPLINA: WEB1 E WEB2 – DESENVOLVIMENTO WEB 1 E 2

O projeto das disciplinas de Desenvolvimento Web 1 (WEB1) e 2 (WEB2) tem como objetivo a implementação de sistemas web completos, focando na criação de operações CRUD (do inglês, Create, Read, Update, Delete) para diferentes entidades. Em WEB1, o trabalho consiste em desenvolver dois CRUDs (Aluno e Curso) utilizando Local Storage para persistência de dados e frameworks front-end usando Bootstrap. Já em WEB2, o projeto avança para a implementação de três CRUDs (Aluno, Curso e Matrícula), exigindo a construção de um *backend* em Spring Boot com acesso a um banco de dados PostgreSQL, além da criação de um menu de navegação.

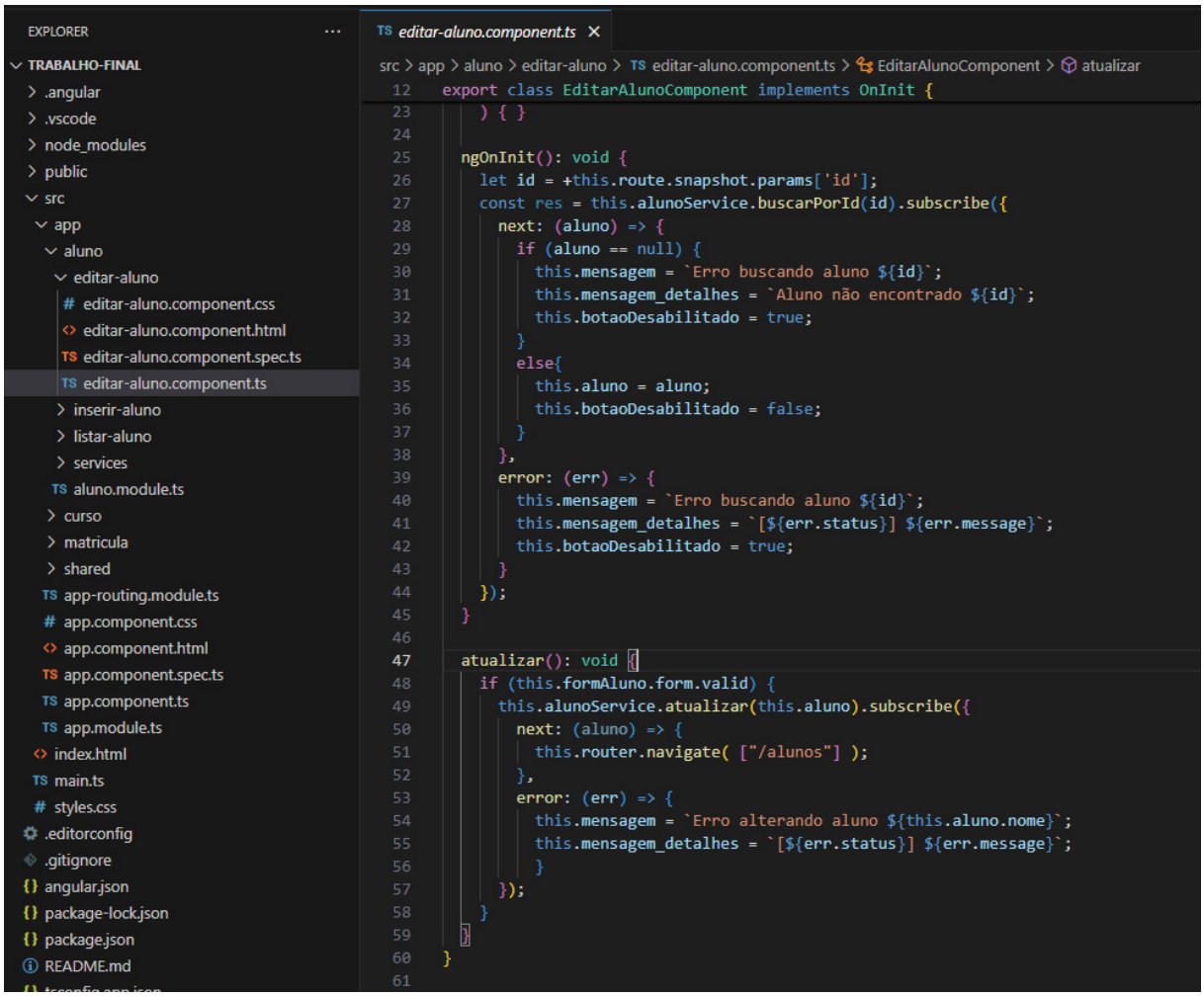
Esses projetos proporcionam uma experiência prática e abrangente no ciclo de vida do desenvolvimento web, alinhada aos princípios ágeis. A implementação de operações CRUD é uma habilidade básica e essencial para qualquer desenvolvedor, e a complexidade crescente entre WEB1 e WEB2 (passando de Local Storage para *backend* com banco de dados real) simula a evolução de um projeto ágil, onde as funcionalidades são construídas e aprimoradas incrementalmente. A exigência de frameworks de design (Bootstrap/Material Design) promove a atenção à *User Experience* (UX) e à responsividade, aspectos cruciais para a entrega de valor ao cliente em um ambiente ágil. A transição para um *backend* em Spring Boot e PostgreSQL em WEB2 introduz conceitos de arquitetura de sistemas distribuídos e persistência de dados, habilidades essenciais para a construção de aplicações robustas e escaláveis, características desejáveis em software desenvolvido de forma ágil.

8.1 ARTEFATOS DO PROJETO

FIGURA 18 – CRUD ALUNO (EDITAR-ALUNO.COMPONENT.HTML)

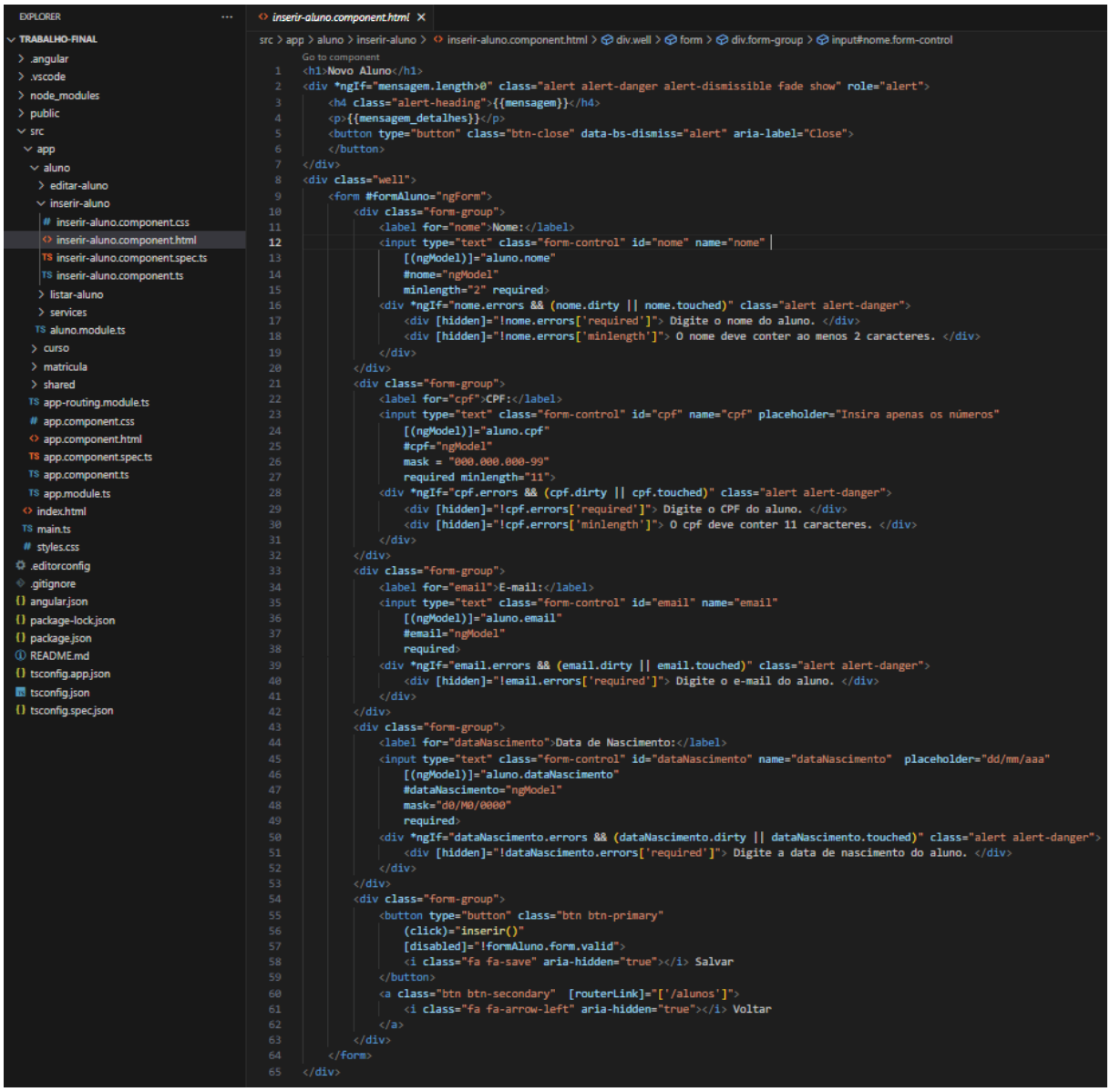
FONTE: A autora (2025).

FIGURA 19 – CRUD ALUNO (EDITAR-ALUNO.COMPONENT.TS)



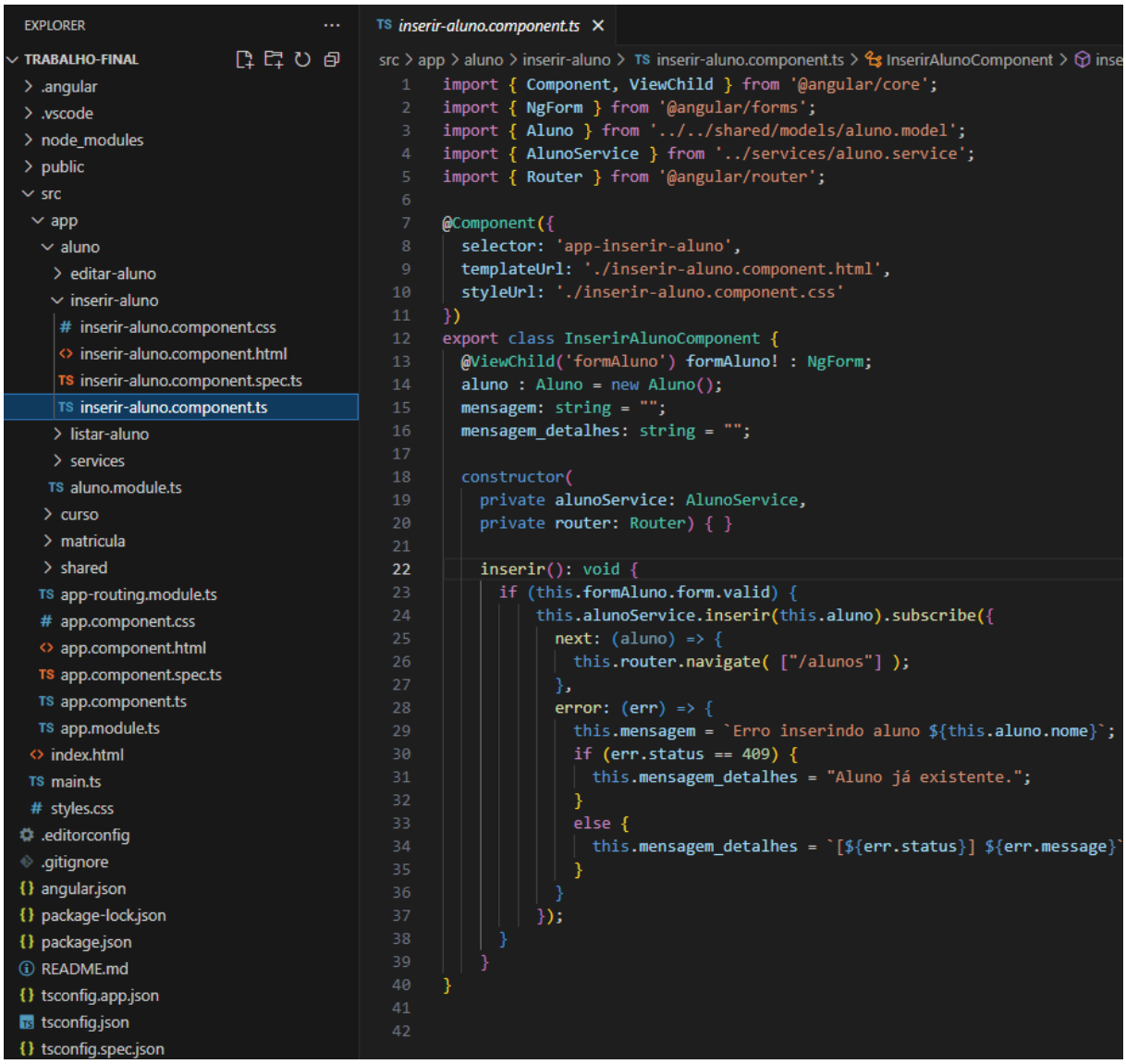
FONTE: A autora (2025).

FIGURA 20 – CRUD ALUNO (INSERIR-ALUNO.COMPONENT.HTML)



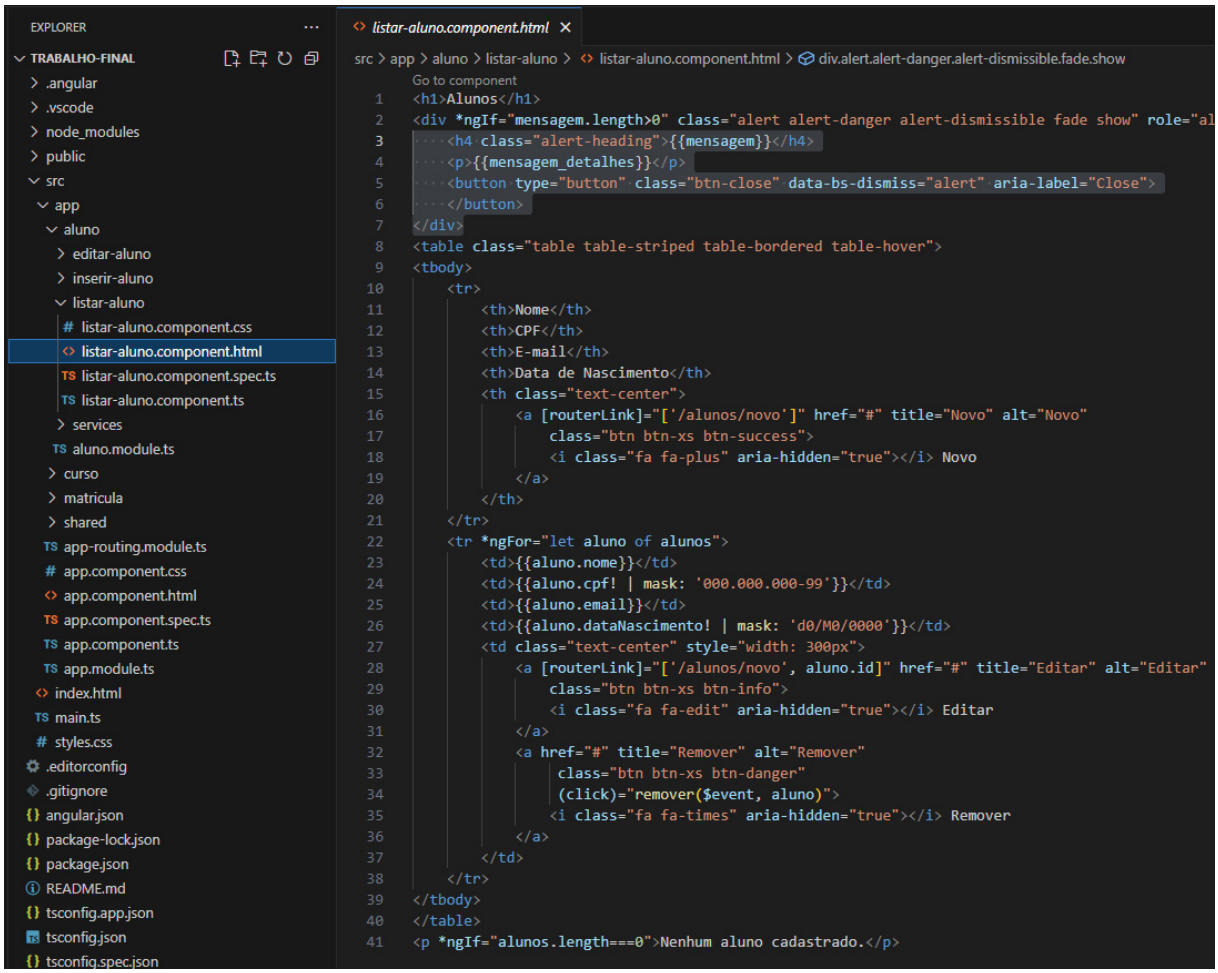
FONTE: A autora (2025).

FIGURA 21 – CRUD ALUNO (INSERIR-ALUNO.COMPONENT.TS)



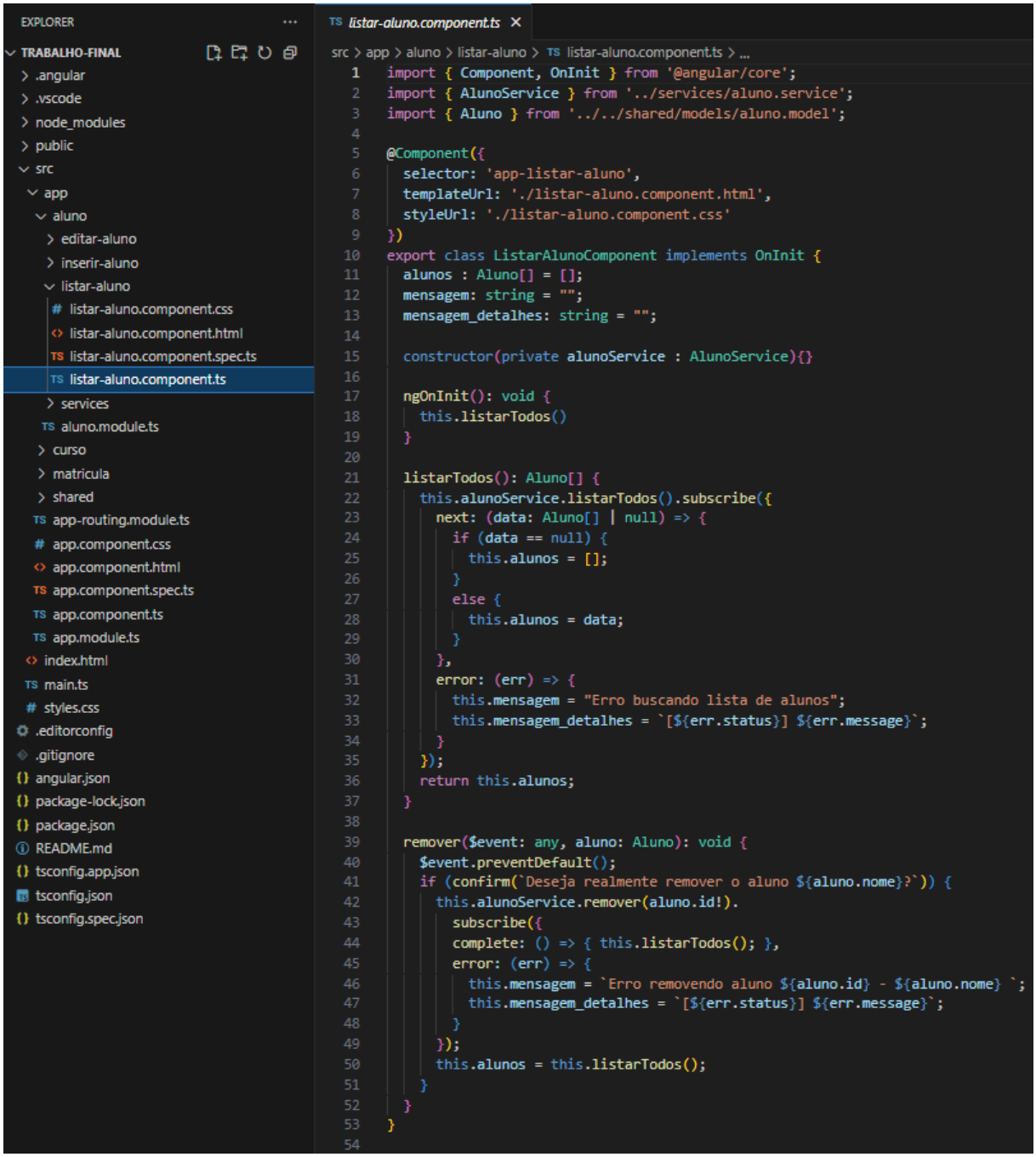
FONTE: A autora (2025).

FIGURA 22 – CRUD ALUNO (LISTAR-ALUNO.COMPONENT.HTML)



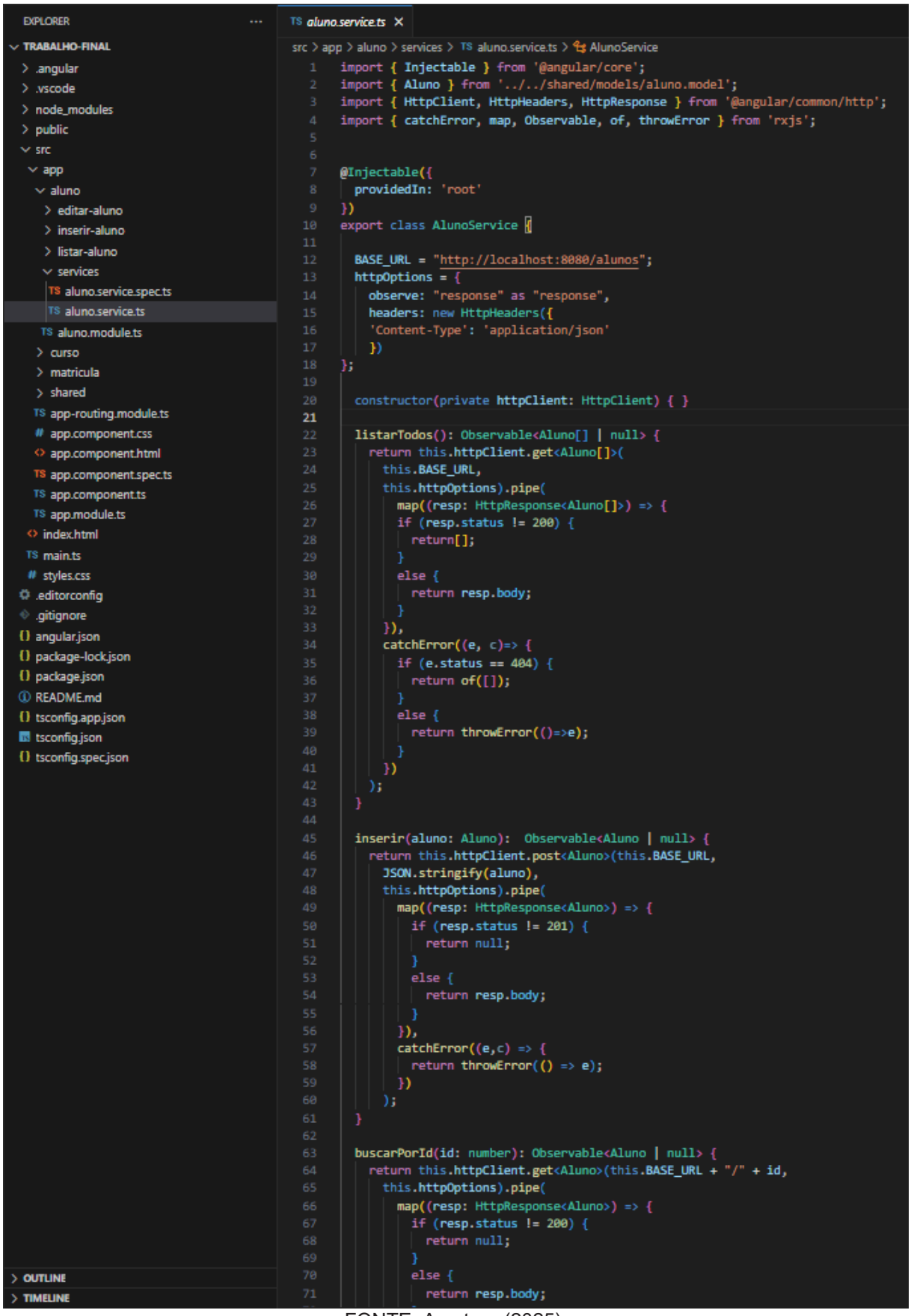
FONTE: A autora (2025).

FIGURA 23 – CRUD ALUNO (LISTAR-ALUNO.COMPONENT.TS)



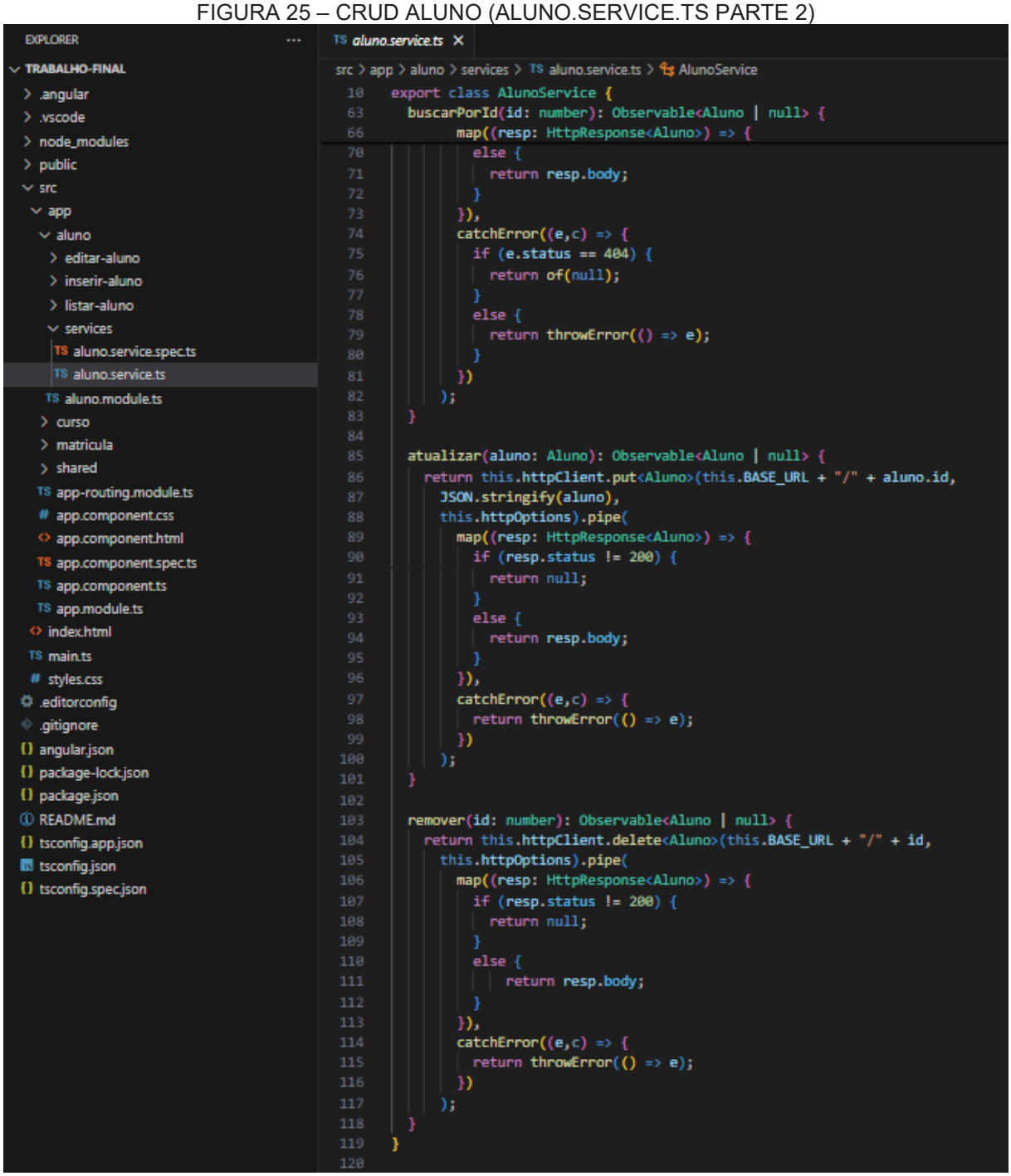
FONTE: A autora (2025).

FIGURA 24 – CRUD ALUNO (ALUNO.SERVICE.TS PARTE 1)



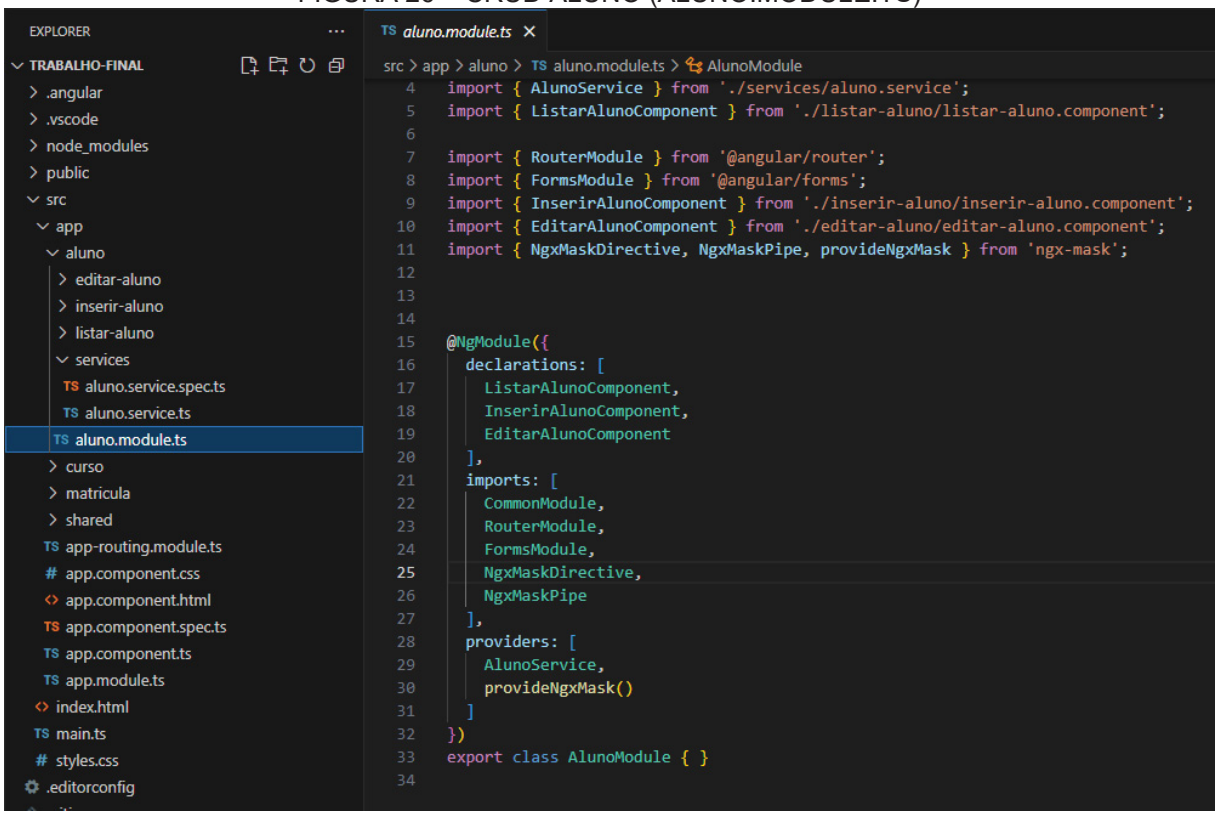
FONTE: A autora (2025).

FIGURA 25 – CRUD ALUNO (ALUNO.SERVICE.TS PARTE 2)



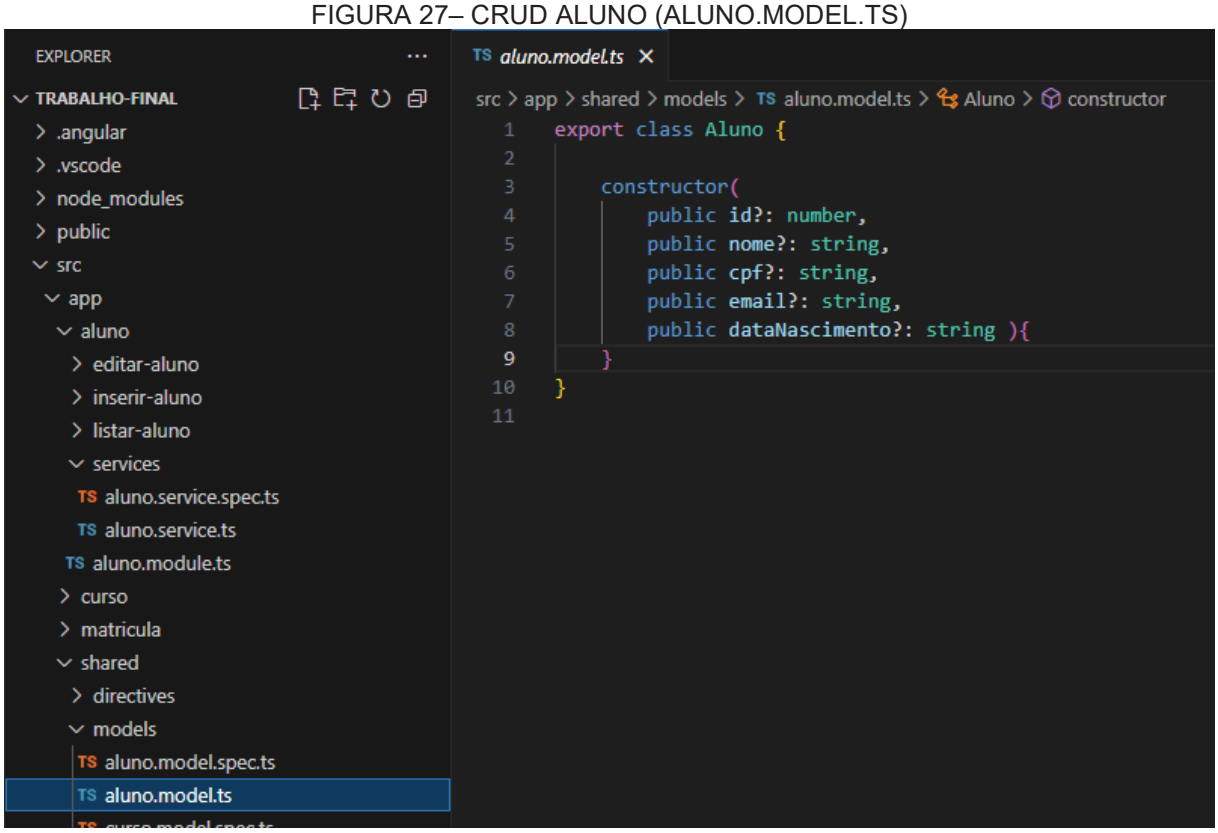
FONTE: A autora (2025).

FIGURA 26 – CRUD ALUNO (ALUNO.MODULE.TS)



FONTE: A autora (2025).

FIGURA 27– CRUD ALUNO (ALUNO.MODEL.TS)



FONTE: A autora (2025).

9 DISCIPLINA: UX – UX NO DESENVOLVIMENTO ÁGIL DE SOFTWARE

O projeto da disciplina de *User Experience* (UX) tem como objetivo a aplicação prática dos conceitos de design centrado no usuário, desde a concepção de um produto digital até a coleta de *feedback* de usuários reais. O trabalho exige a explicação detalhada de um produto (aplicativo ou site), incluindo sua função e justificativa de desenvolvimento. Além disso, é necessário desenvolver no mínimo cinco telas prontas e explicar as escolhas de design (cor, layout, fonte, etc.). Um componente importante é a apresentação dessas telas a pelo menos um possível usuário, explicando a função principal, opções de interação e navegação, e coletando *feedback* para validação e aprimoramento.

O projeto coloca a satisfação do cliente e a usabilidade no centro do processo, alinhando-se diretamente aos princípios do Manifesto Ágil que valorizam a colaboração com o cliente e a entrega contínua de software funcional. A fase de explicação do produto e sua justificativa reforça a necessidade de entender profundamente o problema a ser resolvido e o valor a ser entregue. A criação de telas prontas e a explicação das escolhas de design promovem a prototipagem rápida e a iteração, permitindo que as ideias sejam visualizadas e validadas precocemente. A coleta de *feedback* de usuários reais é um ciclo essencial na agilidade, garantindo que o produto evolua com base nas necessidades e expectativas de quem realmente o utilizará, reduzindo riscos e retrabalhos em fases posteriores do desenvolvimento. Essa abordagem iterativa e focada no usuário é essencial para construir produtos que realmente resolvam problemas e gerem valor.

9.1 ARTEFATOS DO PROJETO

Aplicativo de Gestão de Condomínios

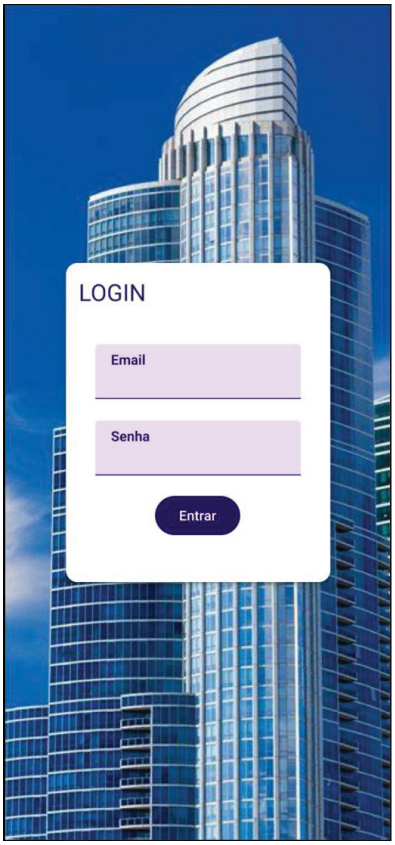
Todas as telas apresentadas neste projeto foram criadas pelo autor utilizando Figma Basic Online.

H001 – Fazer Login
História de Usuário:

Sendo um administrador de condomínio,
Quero acessar o aplicativo de gestão de condomínio inserindo meu e-mail e senha,
Para que eu possa gerenciar as funções do condomínio de forma rápida e segura.

Cenário:

O administrador do condomínio, deseja acessar o aplicativo de gestão para realizar tarefas relacionadas à administração. Ao abrir o app, a primeira tela apresentada é a de login, que solicita a autenticação do usuário para garantir a segurança dos dados e das operações.



H002 – Navegar entre telas

História de Usuário:

Sendo um administrador de condomínio,
Quero acessar uma tela inicial com botões de navegação,
Para que eu possa facilmente organizar as atividades do condomínio de forma eficiente.

Cenário:

Ao entrar no aplicativo, a tela inicial apresenta um layout intuitivo com ícones e botões claramente rotulados.

H003 – Pesquisar Apartamento

História de Usuário:

Sendo um administrador de condomínio,

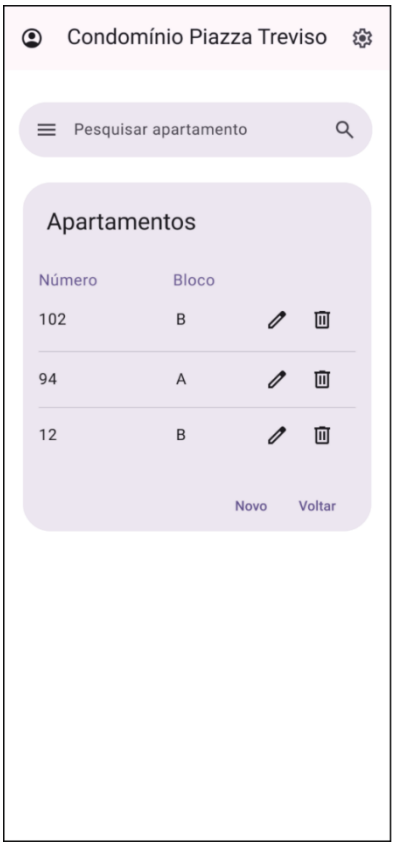
Quero pesquisar apartamentos,

Para fazer manutenção em seus dados.

Cenário:

O usuário deseja gerenciar o cadastro de apartamentos no sistema. Na tela de listagem de apartamentos, o administrador pode visualizar, pesquisar, editar, adicionar ou excluir registros de apartamentos. O objetivo principal é garantir que o cadastro de apartamentos esteja sempre atualizado e que o usuário tenha fácil acesso às informações para controle e administração.

Essa tela é acessada quando o usuário clica no botão “Apartametos” na tela H002.



H004 – Manter Apartamento

História de Usuário:

Sendo um administrador de condomínio,

Quero manter os dados dos apartamentos,

Para que seus dados fiquem atualizados.

Cenário:

O administrador do condomínio está gerenciando o cadastro de apartamentos no sistema. Ele precisa adicionar um novo apartamento ao banco de dados ou atualizar as informações de um apartamento já existente. A tela de editar/adicionar novo apartamento é onde ele pode inserir ou modificar esses dados.

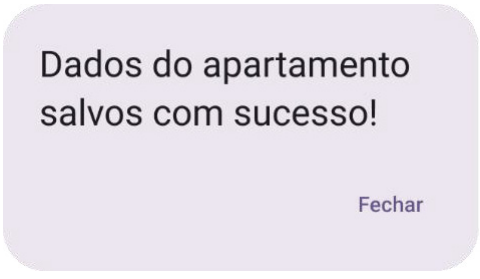
Essa tela é acessada de duas maneiras:

1. **Adicionar Novo Apartamento**: Quando o administrador clica no botão “Novo” a partir da tela de listagem (H003 – Pesquisar Apartamento), ele é

redirecionado para essa tela, que contém um formulário em branco para preenchimento.

2. **Editar Apartamento Existente**: Ao selecionar um apartamento na lista e clicar no ícone de edição, o administrador é direcionado para essa tela, onde os dados do apartamento já estão preenchidos, prontos para serem modificados.

Após o administrador finalizar a adição ou edição de um apartamento, será exibido um card de confirmação visual, confirmando que a ação foi realizada com sucesso. Este card aparece como um pop-up imediatamente após a submissão dos dados do apartamento.



10 DISCIPLINA: MOB1 E MOB2 – DESENVOLVIMENTO MOBILE 1 E 2

Os projetos das disciplinas de Desenvolvimento Mobile I e II têm como objetivo capacitar o profissional na construção de aplicações Android. O projeto de Mobile I foca na criação de um aplicativo financeiro simples (FinApp), com funcionalidades de cadastro de despesas e receitas, e visualização de extrato. O trabalho utiliza estruturas de dados em memória, sem persistência, e enfatiza a organização da interface do usuário. Já em Mobile II, o projeto evolui, exigindo a consulta a uma API pública (Harry Potter API) para exibir dados em diferentes telas, como a listagem de professores ou estudantes de uma casa específica, e a busca por um personagem por ID, introduzindo conceitos de “coroutines” e “web services”.

Embora a execução do trabalho prático seja opcional, a teoria da disciplina, a análise das especificações do projeto e a compreensão dos requisitos proporcionam um entendimento valioso de como as aplicações mobile são construídas em um contexto ágil, reforçando a importância do planejamento e da integração das diferentes áreas do conhecimento para o sucesso de um projeto de software. A familiarização com as ferramentas e a lógica de desenvolvimento mobile já são por si só um aprendizado significativo.

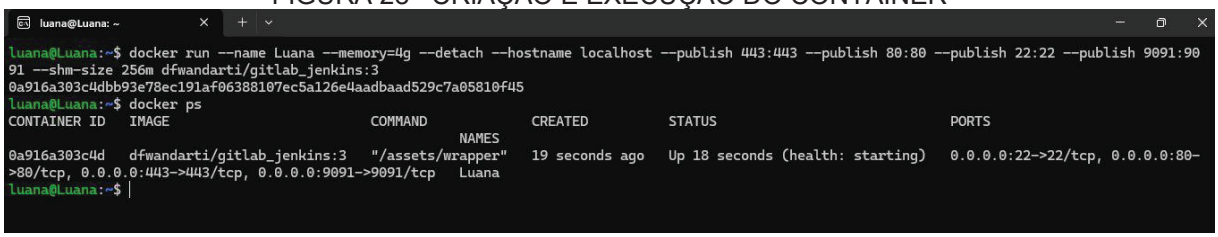
11 DISCIPLINA: INFRA - INFRAESTRUTURA PARA DESENVOLVIMENTO E IMPLANTAÇÃO DE SOFTWARE (DEVOPS)

O projeto da disciplina de Infraestrutura para Desenvolvimento e Implantação (DevOps) (INFRA) tem como objetivo a aplicação prática de ferramentas e conceitos essenciais para a automação do ciclo de vida do software. A tarefa consiste em uma série de etapas que simulam um ambiente de desenvolvimento e integração contínua (CI/CD): baixar e executar uma imagem Docker que contém GitLab e Jenkins, expor as portas necessárias, acessar o GitLab, obter a senha do usuário root do container, logar na interface web e, por fim, realizar um commit e um push em um projeto, registrando a atividade no log do Git.

O projeto aborda diretamente a automação e a colaboração, pilares do movimento DevOps, que é uma extensão natural e poderosa da agilidade. Em um contexto ágil, onde a entrega contínua de software funcional é a prioridade, a capacidade de automatizar o build, o teste e a implantação é essencial para garantir a velocidade e a qualidade. Este projeto capacita a aluna a entender e a utilizar ferramentas como Docker para padronizar ambientes, GitLab para versionamento de código e colaboração, e Jenkins para automação de tarefas, que são amplamente utilizadas para a implementação de *pipelines* de CI/CD. A prática de configurar e interagir com essas ferramentas em um ambiente controlado oferece uma compreensão valiosa sobre como a infraestrutura de um projeto ágil é montada e gerenciada.

11.1 ARTEFATOS DO PROJETO

FIGURA 28– CRIAÇÃO E EXECUÇÃO DO CONTAINER



FONTE: A autora (2025).

FIGURA 29– LOGIN NO GIT PARTE 1

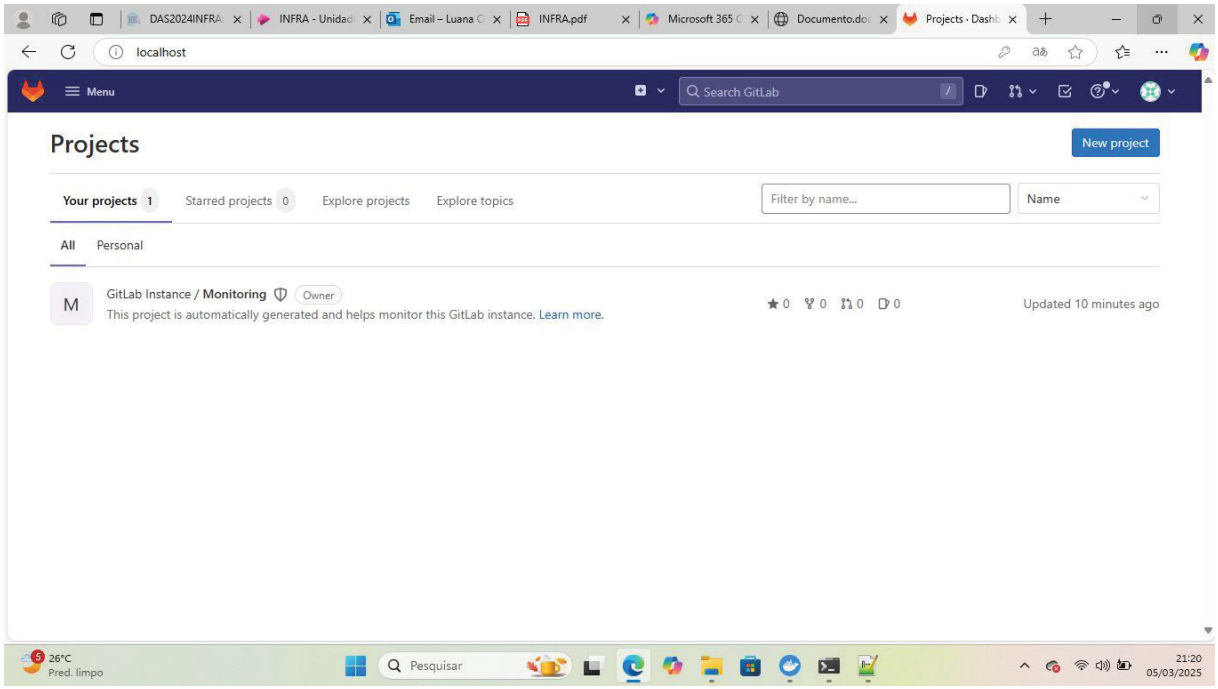
```
luana@Luana:~$ docker exec -it 0a916a303c4d cat /etc/gitlab/initial_root_password
# WARNING: This value is valid only in the following conditions
#          1. If provided manually (either via `GITLAB_ROOT_PASSWORD` environment variable or via `gitlab_rails['initial_root_password']` setting in
 `gitlab.rb`, it was provided before database was seeded for the first time (usually, the first reconfigure run).
#          2. Password hasn't been changed manually, either via UI or via command line.
#
#          If the password shown here doesn't work, you must reset the admin password following https://docs.gitlab.com/ee/security/reset_user_passw
ord.html#reset-your-root-password.

Password: fzbMMpKpB5WD8V5x0As6JrDOx+yKGkYw8jIlgyYoLPQ=

# NOTE: This file will be automatically deleted in the first reconfigure run after 24 hours.
luana@Luana:~$
```

FONTE: A autora (2025).

FIGURA 30 – LOGIN NO GIT PARTE 2



FONTE: A autora (2025).

FIGURA 31 – GIT COMMIT

```
luana@Luana:~/Monitoring$ git log
commit 30f205c473b45b26f160cbe96abb84adb64e14a6 (HEAD -> main, origin/main)
Author: Luana <luana.corsi@ufpr.br>
Date:   Wed Mar 5 21:57:29 2025 -0300

    add README
luana@Luana:~/Monitoring$
```

FONTE: A autora (2025).

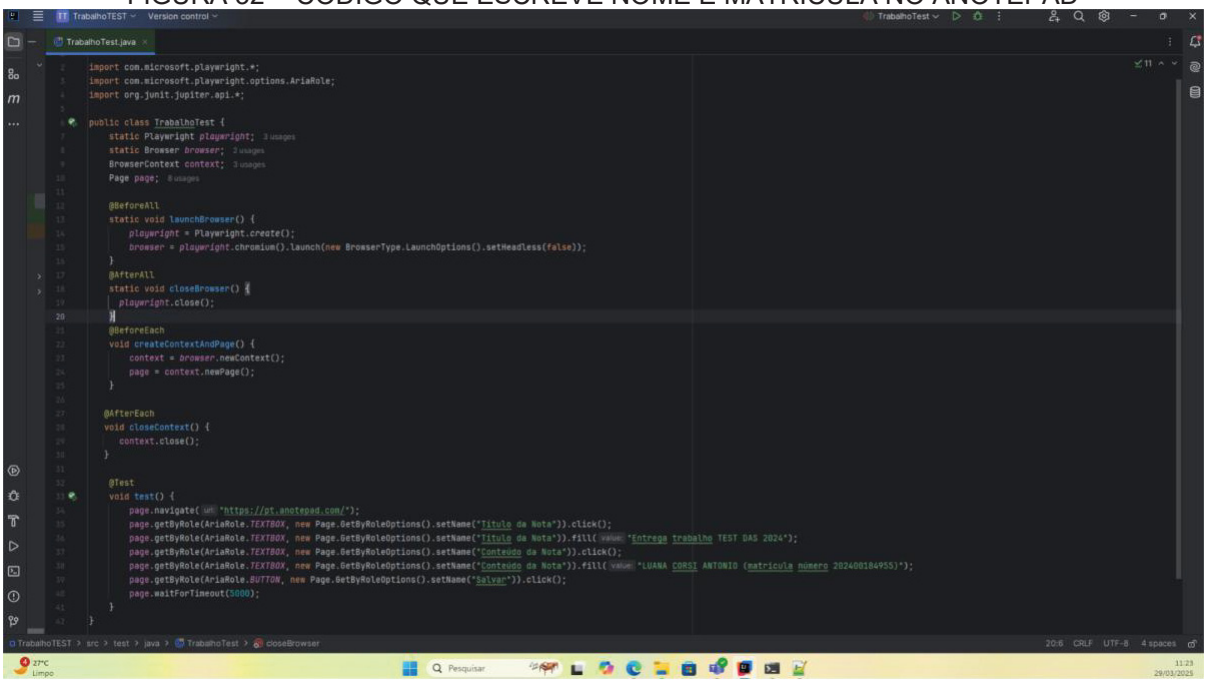
12 DISCIPLINA: TEST – TESTES AUTOMATIZADOS

O projeto da disciplina de Testes Automatizados (TEST) tem como objetivo a aplicação prática da automação de testes para validar funcionalidades de um software. A especificação do trabalho, que exige a criação de um código para interagir com o site anotepad.com e preencher uma nota com informações específicas, é um exercício prático que simula a automação de uma tarefa de validação de interface de usuário. Ele capacita o profissional a programar um script que navega, interage com elementos da página (campos de texto, botões) e verifica o resultado esperado, demonstrando a capacidade de construir testes automatizados de ponta a ponta.

A automação de testes é fundamental para manter a velocidade e a qualidade em ciclos de desenvolvimento curtos e iterativos. Em um ambiente ágil, onde o software é entregue com frequência, a execução manual de testes se torna um gargalo insustentável. A automação, como praticada neste projeto, permite que os testes sejam executados de forma rápida, consistente e repetitiva, fornecendo *feedback* imediato sobre a saúde do sistema a cada nova alteração de código. Isso é essencial para identificar regressões precocemente, reduzindo o tempo de depuração e garantindo que novas funcionalidades não quebrem as existentes. A disciplina, portanto, fortalece a confiança na base de código e na capacidade da equipe de entregar valor de forma contínua e segura.

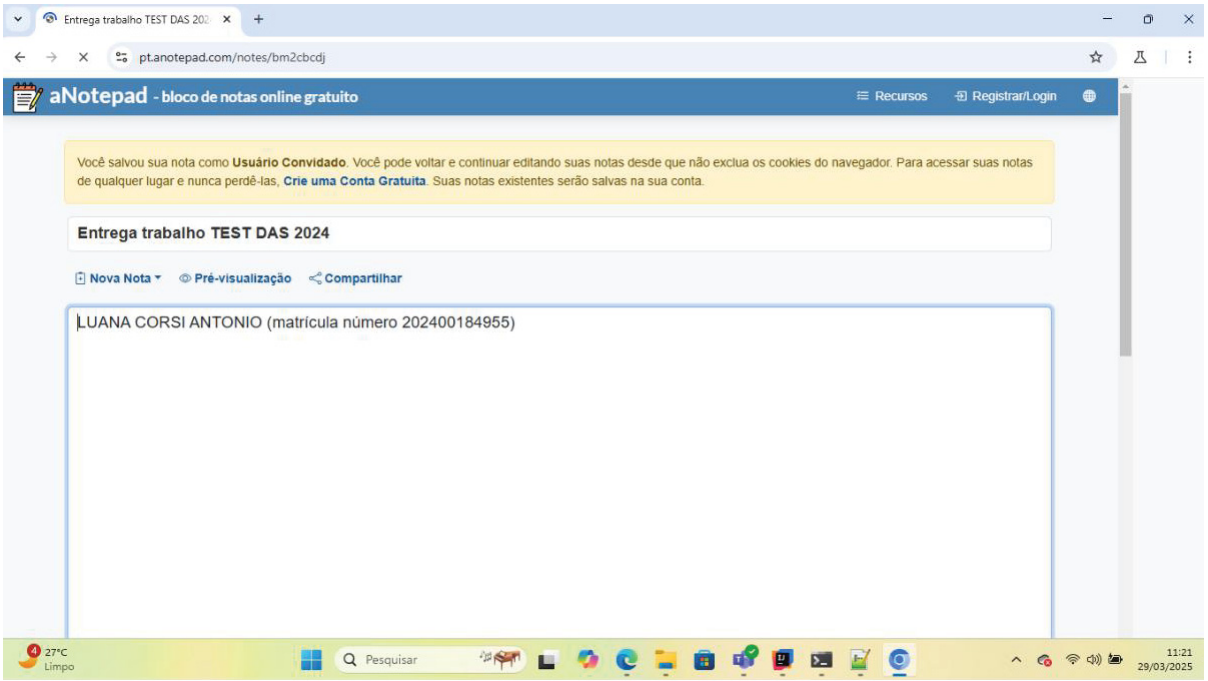
12.1 ARTEFATOS DO PROJETO

FIGURA 32 – CÓDIGO QUE ESCREVE NOME E MATRÍCULA NO ANOTEPAD



FONTE: A autora (2025).

FIGURA 33 – NOME E MATRÍCULA ESCRITAS NO ANOTEPAD



FONTE: A autora (2025).

13 CONCLUSÃO

O memorial de projetos apresentado oferece uma visão abrangente e integrada da jornada de aprendizado da aluna. O foco principal deste documento é demonstrar como a qualidade do software é o resultado da aplicação consistente e integrada de práticas ágeis em todas as etapas do projeto.

A disciplina de Métodos Ágeis de Desenvolvimento de Software (MADS) estabelece o fundamento teórico para este ecossistema de conhecimento, onde cada componente contribui para a entrega de um produto de alta qualidade. A qualidade do software, que emerge de uma cultura de trabalho colaborativo e iterativo, é construída por meio da sinergia entre as disciplinas técnicas e de gestão. As disciplinas de gestão, por exemplo, fornecem as ferramentas para a definição clara e a priorização de requisitos, além de métricas essenciais para o monitoramento contínuo do fluxo de valor. Essa abordagem garante que a velocidade das entregas não comprometa a estabilidade do produto, permitindo que as equipes identifiquem e solucionem gargalos de forma proativa.

Paralelamente, as disciplinas técnicas reforçam a importância da qualidade diretamente no código, desde a programação e modelagem até a estrutura de dados. Elas promovem boas práticas de codificação como *Clean Code,* essenciais para a manutenibilidade e escalabilidade, e garantem a excelência operacional através de testes automatizados e práticas de automação de infraestrutura. A integração com *User Experience* (UX) assegura que o produto atenda não apenas aos requisitos técnicos, mas também às expectativas do cliente, uma dimensão crítica da qualidade.

O currículo evidencia a interdisciplinaridade essencial para a adoção bem-sucedida do desenvolvimento ágil e, consequentemente, a entrega de software de alta qualidade. No entanto, a transição da teoria para a prática não é isenta de desafios. Com base nos temas abordados nos projetos, alguns dos principais obstáculos para a adoção prática do desenvolvimento ágil, com foco na qualidade, incluem a necessidade de aderência à cultura e aos princípios ágeis, já que o maior desafio é o alinhamento de mentalidade em toda a organização para priorizar a colaboração e a adaptabilidade em detrimento de processos rígidos.

Além disso, a gestão da dívida técnica é um desafio constante, exigindo que as equipes equilibrem a velocidade de entrega com a manutenção de um código de alta qualidade para não comprometer a sustentabilidade do projeto. A integração de

disciplinas e papéis também se apresenta como um obstáculo, pois a separação de equipes pode criar silos que contradizem a natureza colaborativa da agilidade. O grande desafio é integrar efetivamente todos os papéis para que a qualidade seja uma responsabilidade compartilhada, do design à implantação.

A especialização, por meio de seus projetos, não apenas fornece as ferramentas e técnicas essenciais, mas também expõe os desafios inerentes à adoção do desenvolvimento ágil. Ela capacita o profissional a construir software e a fazê-lo de forma inteligente, colaborativa e adaptável para a criação de produtos de alta qualidade em um ambiente de constante mudança.

14 REFERÊNCIAS

BECK, K., et al. The Agile Manifesto. **Agile Alliance**. 2001. Disponível em: http://agilemanifesto.org/. Acesso em: 27 jun. 2025.

BOOCH, G.; RUMBAUGH, J. **UML: guia do usuário**. Brasil, Elsevier, 2006.

CRISPIN, L.; GREGORY, J. **Agile Testing: A Practical Guide for Testers and Agile Teams**. Alemanha, Addison-Wesley, 2009.

FORSGREN, P. et al. The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations. Estados Unidos, **IT Revolution Press**, 2018.

INAYAT, I.; SALIM, S.; MARCZAK, S.; DANEVA, M.; SHAMSHIRBAND, S. A systematic literature review on agile requirements engineering practices and challenges. **Computers in human behavior**, 51(Part B), 915-929. 2015. Disponível em: https://doi.org/10.1016/j.chb.2014.10.046. Acesso em: 06 jul. 2025.

LEITE, G.; VIEIRA, R.; CERQUEIRA, L; MACIEL, R.; FREIRE, S.; MENDONÇA, M. Technical Debt Management in Agile Software Development: A Systematic Mapping Study. *In*: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE (SBQS), 23. , 2024, Bahia/BA. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2024 . p. 309–320.

MOGNON, F. C.; STADZISZ, P. Modeling in Agile Software Development: A Systematic Literature Review. **Communications in Computer and Information Science**, vol 680. Springer, Cham. 2017. Disponível em: https://doi.org/10.1007/978-3-319-55907-0_5. Acesso em: 13 jul. 2025.

POPPENDIECK, M.; POPPENDIECK, T**. Lean Software Development: An Agile Toolkit: An Agile Toolkit.** Reino Unido, Pearson Education, 2003.

STARON, M.; STARON, M.; MEDING, W. Metrics for Software Design and Architectures. **Automotive Software Architectures**. Springer, Cham. 2017. Disponível em: https://doi.org/10.1007/978-3-319-58610-6_7. Acesso em: 06 jul. 2025.