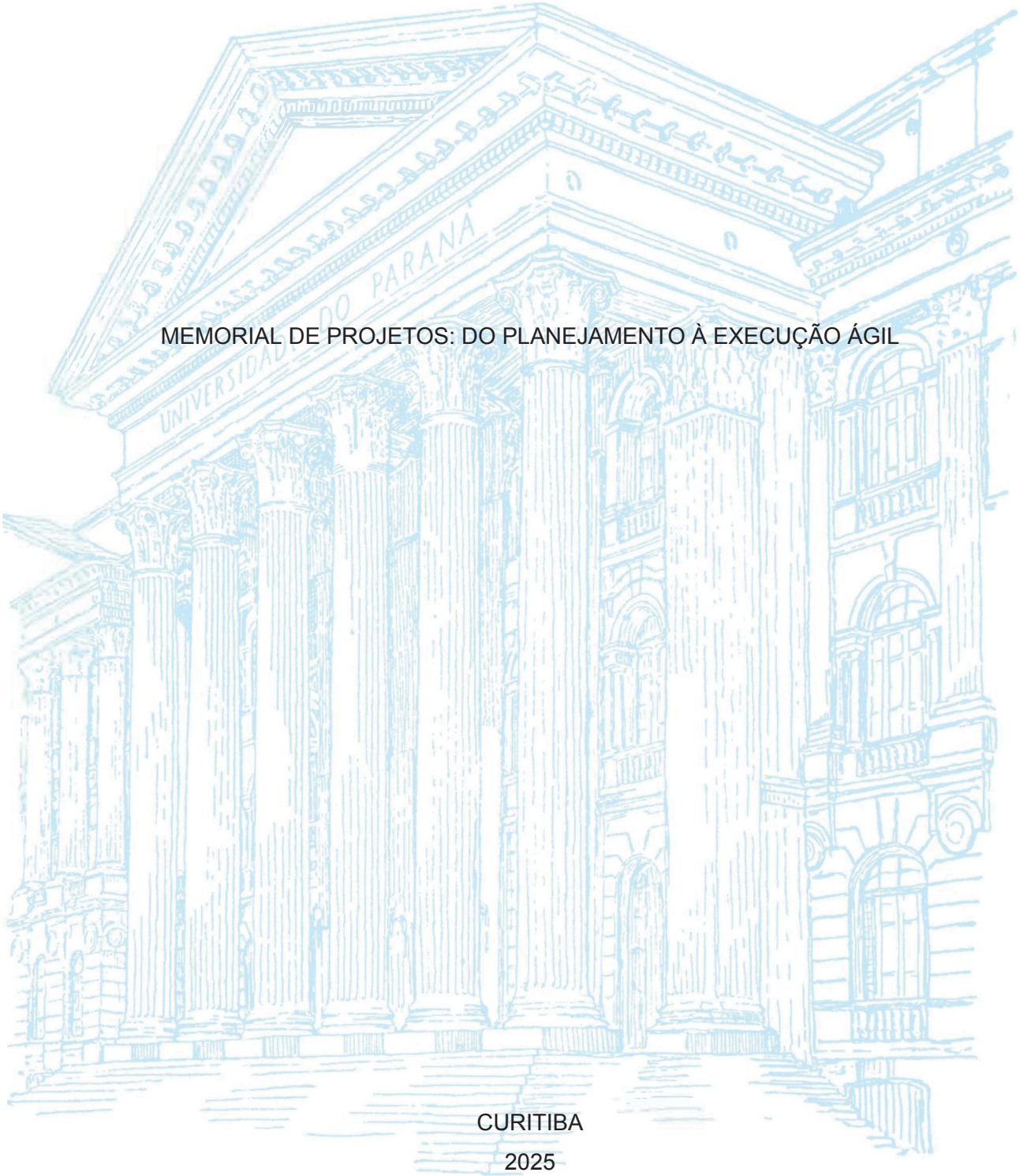
UNIVERSIDADE FEDERAL DO PARANÁ

DANILO DANTAS SONCINI



MEMORIAL DE PROJETOS: DO PLANEJAMENTO À EXECUÇÃO ÁGIL

CURITIBA

2025

DANILO DANTAS SONCINI

MEMORIAL DE PROJETOS: DO PLANEJAMENTO À EXECUÇÃO 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: Prof. Dr. Jaime Wojciechowski

CURITIBA

2025

MINISTÉRIO DA EDUCAÇÃO
SETOR DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PÓS-GRADUAÇÃO
CURSO DE PÓS-GRADUAÇÃO DESENVOLVIMENTO ÁGIL
DE SOFTWARE - 40001016398E1

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 **DANILO DANTAS SONCINI**, intitulada: **MEMORIAL DE PROJETOS: DO PLANEJAMENTO À EXECUÇÃO ÁGIL**, que após terem inquirido o aluno 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, 20 de Outubro de 2025.

JAIME WOJCIECHOWSKI
Presidente da Banca Examinadora

RAFAELA MANTOVANI FONTANA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

RESUMO

A pós-graduação em Desenvolvimento Ágil de Software proporcionou uma formação sólida sobre metodologias ágeis, boas práticas de programação e processos modernos de desenvolvimento. As disciplinas de Métodos Ágeis e Gerenciamento Ágil de Projetos apresentaram *frameworks* como *Scrum* e *Kanban*, destacando a importância da flexibilidade e colaboração. As disciplinas de Modelagem Ágil e Aspectos Ágeis da Programação reforçaram conceitos de documentação, organização e qualidade de código, incluindo práticas como *Test-Driven Development*. Os fundamentos de Introdução à Programação e Banco de Dados foram essenciais para a construção de aplicações robustas. As disciplinas Web e Mobile aprofundaram conhecimentos em desenvolvimento, alinhados às diretrizes da aplicação de UX no Desenvolvimento Ágil. A disciplina *DevOps* abordou automação e entrega contínua, enquanto Testes Automatizados enfatizou a importância da validação constante do software. O curso permitiu uma visão abrangente e integrada do desenvolvimento ágil, preparando para a criação de soluções eficientes e alinhadas às demandas do mercado.

Palavras-chave: Scrum; modelagem; desenvolvimento; devops; testes.

ABSTRACT

The postgraduate program in Agile Software Development provided a solid foundation in agile methodologies, programming best practices, and modern development processes. The courses in Agile Methods and Agile Project Management introduced frameworks such as Scrum and Kanban, emphasizing the importance of flexibility and collaboration. The subjects Agile Modeling and Agile Aspects of Programming reinforced concepts related to documentation, code organization, and quality, including practices such as Test-Driven Development The fundamentals of Introduction to Programming and Databases were essential for building robust applications. Meanwhile, the Web and Mobile courses deepened knowledge in development, aligned with UX application guidelines in Agile Development. The DevOps course covered automation and continuous delivery, while Automated Testing emphasized the importance of constant software validation . Overall, the program offered a comprehensive and integrated view of agile development, preparing students to create efficient solutions aligned with market demands.

Keywords: Scrum; modeling; development; devops; testing.

SUMÁRIO

1 PARECER TÉCNICO

O desenvolvimento de software moderno exige abordagens que garantam agilidade, flexibilidade e qualidade nas entregas. Nesse contexto, as metodologias ágeis se consolidaram como a base para a gestão de projetos, promovendo ciclos iterativos, colaboração entre equipes e adaptação às mudanças de escopo (Rigby, 2020), (Beck, 2004). O presente memorial documenta a trajetória acadêmica dentro da pós-graduação em Desenvolvimento Ágil de Software, evidenciando a conexão entre as disciplinas estudadas e os trabalhos desenvolvidos, desde a fase de planejamento até a execução prática de projetos.

Ao longo do curso, foram explorados diferentes aspectos do desenvolvimento ágil. Disciplinas como Métodos Ágeis para Desenvolvimento de Software e Gerenciamento Ágil de Projetos forneceram os alicerces teóricos para a adoção de *frameworks* como *Scrum* e *Kanban*, destacando a importância da organização e do monitoramento contínuo dos processos (Schwaber; Sutherland, 2020), (Anderson, 2010). Foram abordadas práticas fundamentais como definição de papéis e responsabilidades dentro das equipes, acompanhamento de backlog, refinamento de requisitos e métricas ágeis para mensuração de progresso. A partir desses conceitos, foi possível estruturar projetos práticos, aplicando ferramentas e artefatos ágeis para melhorar a gestão e a execução das tarefas.

A modelagem desempenhou um papel essencial na estruturação dos sistemas, abordada nas disciplinas de Modelagem Ágil de Software e Banco de Dados. O aprendizado sobre *UML* incluiu a construção de diagramas de casos de uso, diagramas de classes e diagramas de sequência, proporcionando uma representação visual detalhada do funcionamento das aplicações. Essa base se conecta diretamente ao desenvolvimento de sistemas, possibilitando a implementação de soluções robustas e alinhadas às necessidades dos usuários.

O aspecto técnico do curso foi fortalecido com disciplinas como Aspectos Ágeis de Programação, Desenvolvimento Web e Desenvolvimento Mobile, onde foram aplicados princípios de código limpo, reutilização de componentes e padrões arquiteturais. Em Aspectos Ágeis de Programação, foram trabalhados conceitos como *Clean Code*, Refatoração, TDD (*Test-Driven Development*) e BDD (*Behavior-Driven Development*), garantindo a criação de códigos mais organizados e

fáceis de manter (Beck, 2002), (Martin, 2009). No Desenvolvimento Web, foi aprofundado o uso do framework Angular e da linguagem TypeScript, com foco na construção de Single Page Applications (SPA), formulários dinâmicos e integração com APIs. Já em Desenvolvimento Mobile, o curso abordou o uso do Android Studio, a estruturação de interfaces com Views, manipulação de listas com RecyclerView e desafios da programação assíncrona no ambiente mobile.

A disciplina de UX (*User Experience)* no Desenvolvimento Ágil proporcionou uma abordagem centrada no usuário, garantindo que os produtos desenvolvidos fossem intuitivos, acessíveis e eficientes. Foram explorados conceitos como criação de personas, mapeamento da jornada do usuário, acessibilidade, responsividade e consistência na interface, reforçando a importância de um design bem estruturado para melhorar a experiência do usuário final (Gothelf; Seiden, 2013).

Por fim, as disciplinas de *DevOps* e Testes Automatizados agregaram a dimensão da automação e da qualidade ao ciclo de desenvolvimento. Em *DevOps*, foram estudadas práticas como versionamento de código com Git e GitHub, conteinerização com Docker, pipelines de integração e entrega contínua (CI/CD) (Humble; Farley, 2011). Já em Testes Automatizados, o foco esteve na implementação de testes unitários e na automação de testes End-to-End, garantindo maior confiabilidade e eficiência nos processos de desenvolvimento.

O presente memorial busca conectar cada uma dessas disciplinas e seus respectivos trabalhos, demonstrando como a aplicação integrada dos conceitos estudados contribuiu para a construção de soluções ágeis e eficazes. Dessa forma, a experiência acadêmica se consolidou como um processo iterativo de aprendizado e aprimoramento, refletindo a essência do desenvolvimento ágil de software.

2 DISCIPLINA: MADS – MÉTODOS ÁGEIS PARA DESENVOLVIMENTO DE SOFTWARE

A disciplina de Métodos Ágeis para Desenvolvimento de Software funciona como o alicerce do curso, servindo como base para a aplicação dos princípios ágeis em todas as demais áreas do desenvolvimento de software. Ao ensinar como adotar metodologias como *Scrum*, *Kanban* e Extreme Programming (Beck, 2004), a disciplina proporciona um entendimento fundamental sobre práticas iterativas, colaboração e entrega contínua de valor ao usuário, comparando com os métodos clássicos de desenvolvimento de software, como o modelo em cascata que priorizam um planejamento detalhado e fases sequenciais de execução, os métodos ágeis propõem uma abordagem mais adaptativa e colaborativa.

O modelo tradicional busca a previsibilidade e o controle por meio de documentação e etapas rigidamente definidas, os métodos ágeis enfatizam a flexibilidade, a comunicação contínua e a capacidade de responder rapidamente às mudanças. De acordo com Pressman e Maxim (2021), compreender essas diferenças permite ao desenvolvedor escolher a metodologia mais adequada ao contexto do projeto, equilibrando planejamento e agilidade conforme as necessidades do produto e da equipe.

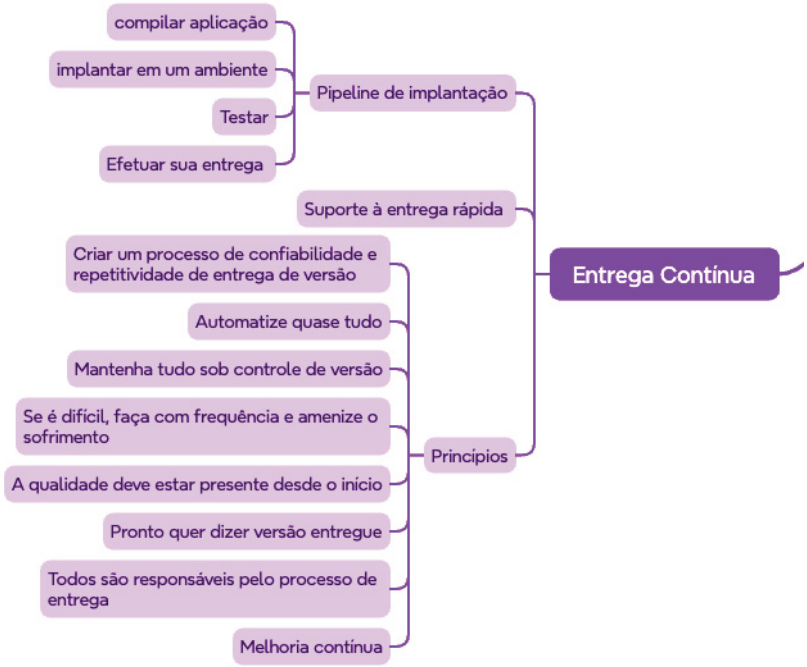
A construção de um mapa mental diferenciando cada um dos métodos ágeis e suas aplicações no desenvolvimento de software se torna um recurso valioso para consolidar o aprendizado e guiar a aplicação prática dessas metodologias. Esse material auxilia na visualização clara das conexões entre os frameworks ágeis e suas influências no Desenvolvimento Ágil de Software.

2.1 ARTEFATOS DO PROJETO

FIGURA 1 - MAPA MENTAL PROCESSO DE SOFTWARE

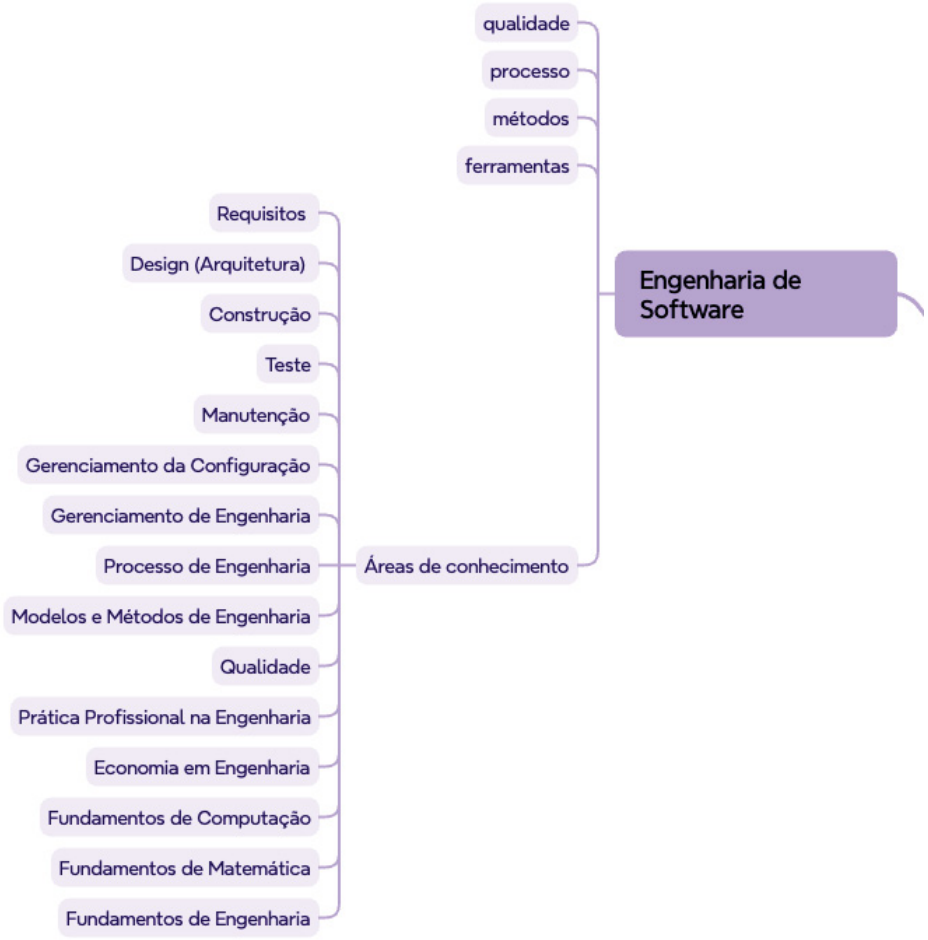
FONTE: O autor (2025)

FIGURA 2 - MAPA MENTAL ENTREGA CONTÍNUA



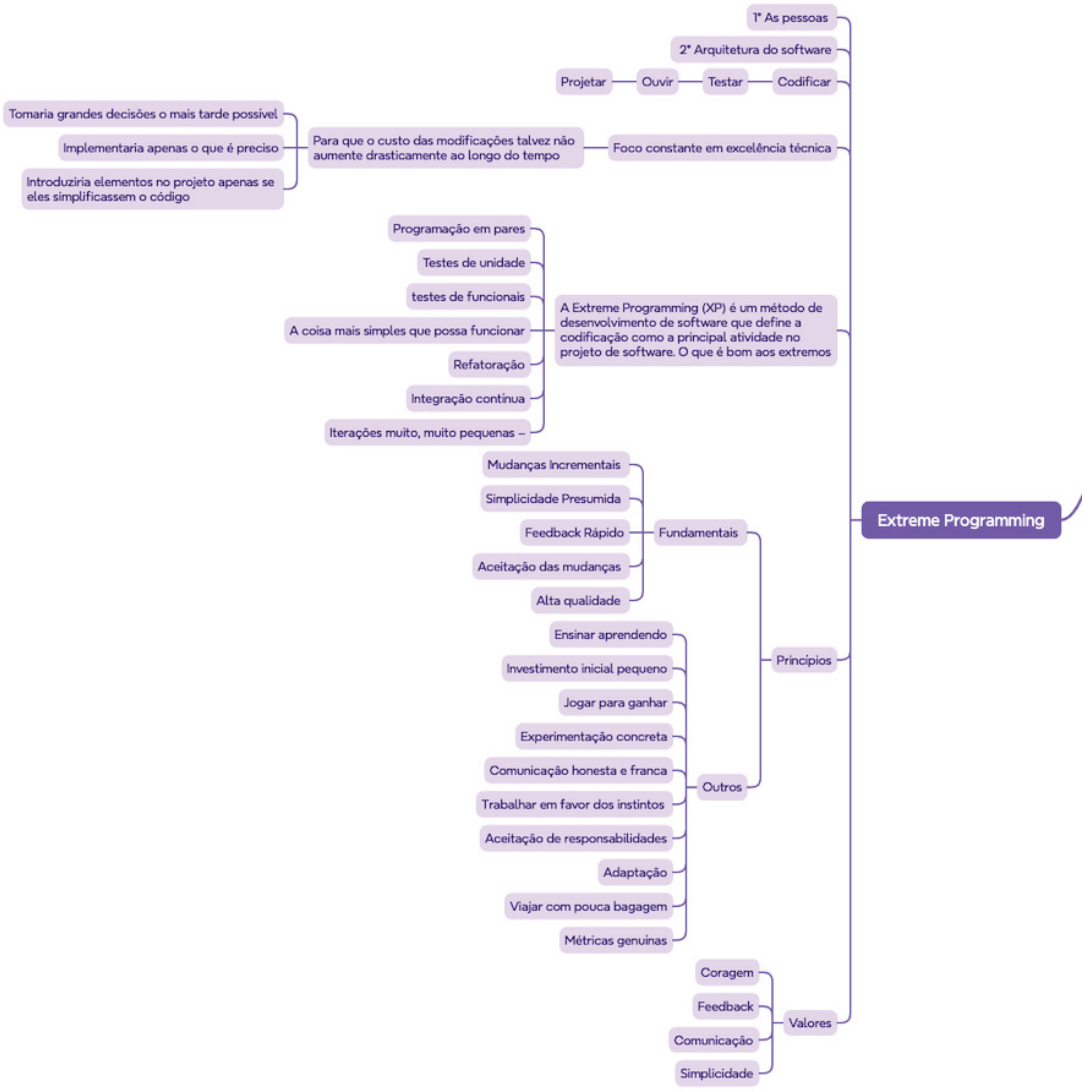
FONTE: O autor (2025)

FIGURA 3 - MAPA MENTAL ENGENHARIA DE SOFTWARE



FONTE: O autor (2025)

FIGURA 4 - MAPA MENTAL EXTREME PROGRAMING

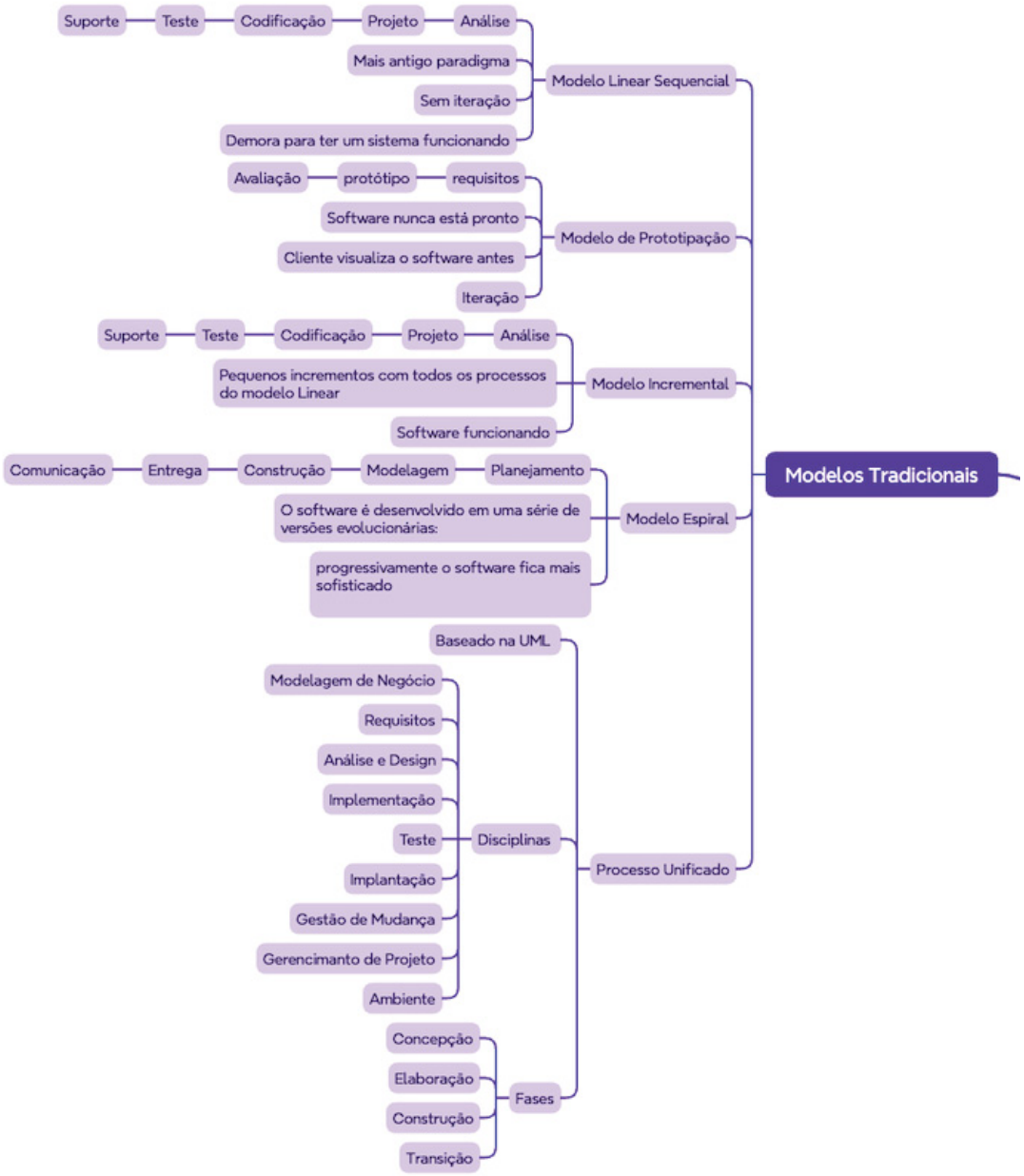


FONTE: O autor (2025)

FIGURA 5 - MAPA MENTAL LEAN

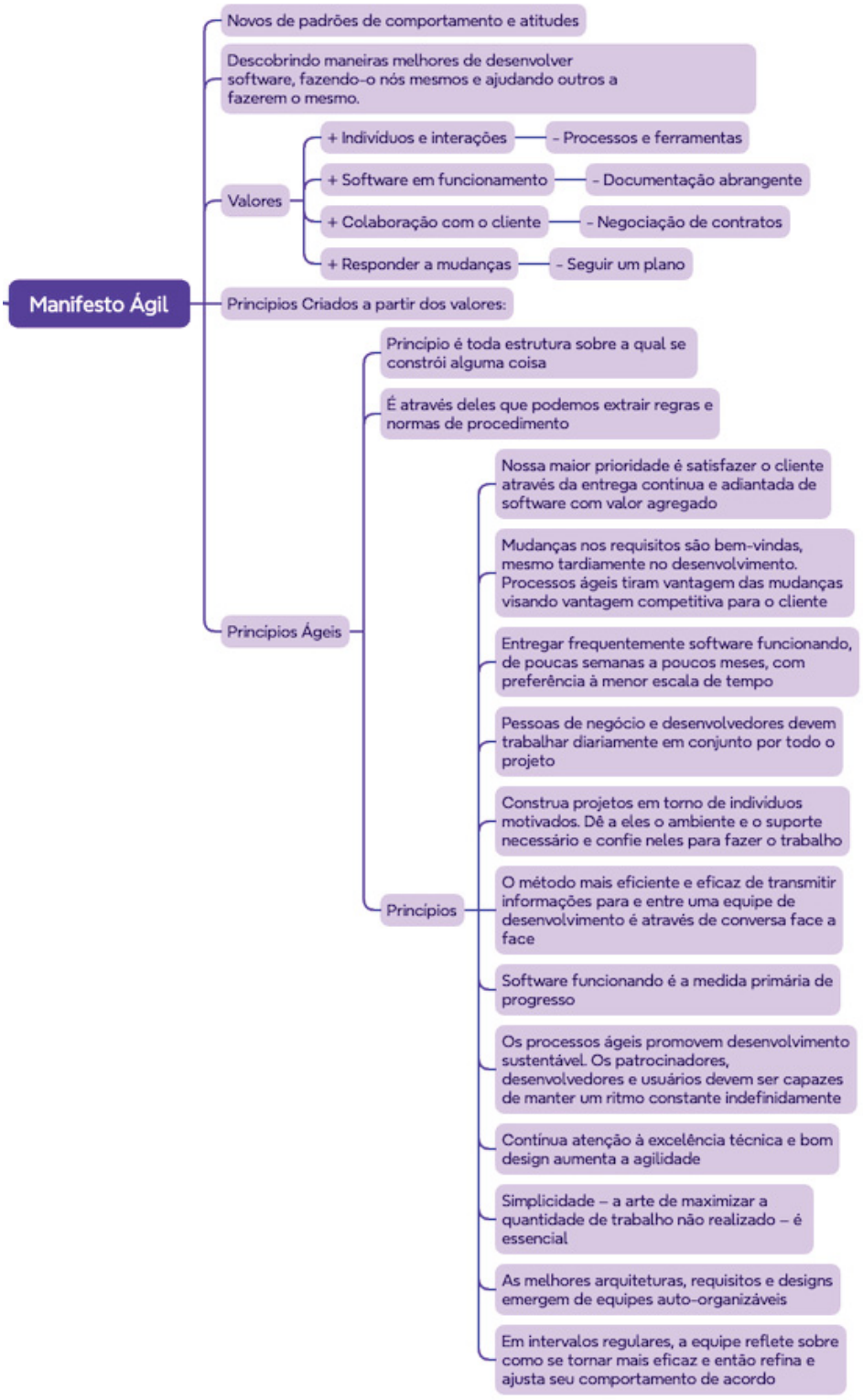
FONTE: O autor (2025)

FIGURA 6 - MAPA MENTAL MODELOS TRADICIONAIS



FONTE: O autor (2025)

FIGURA 7 - MAPA MENTAL MANIFESTO ÁGIL



FONTE: O autor (2025)

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

A disciplina de Modelagem Ágil de Software tem como foco a utilização de técnicas de modelagem para estruturar sistemas de forma clara e eficiente dentro do contexto ágil (Ambler, 2002). Através da UML (*Unified Modeling Language*) , são abordadas as principais ferramentas para representação dos requisitos e do comportamento do sistema, garantindo uma visão compartilhada entre os membros da equipe de desenvolvimento. A partir do diagrama de casos de uso, são definidas as histórias de usuário, que descrevem de forma detalhada as interações dos usuários com o sistema, auxiliando na construção de funcionalidades alinhadas às necessidades do projeto.

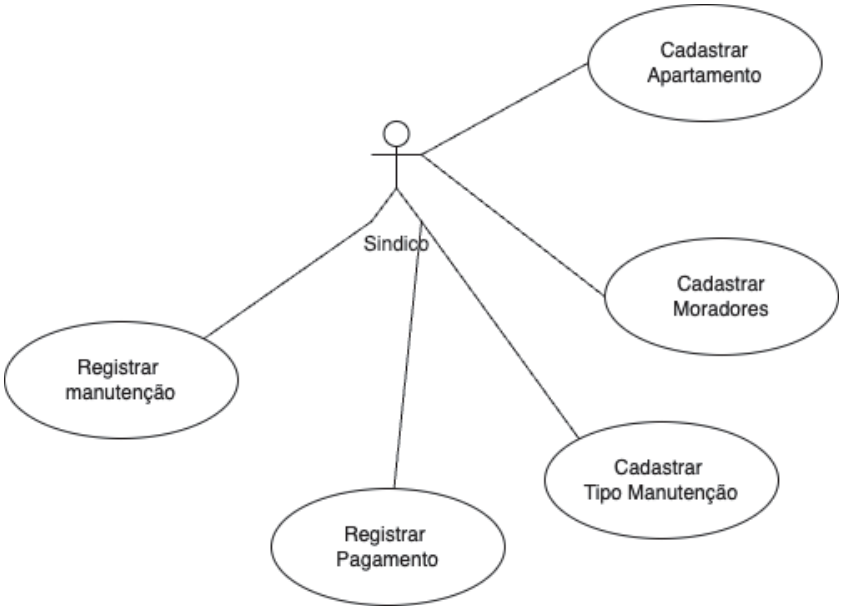
O primeiro grande desafio da disciplina foi a modelagem de um sistema de gestão de condomínios, onde aplicamos os conceitos estudados para estruturar os principais requisitos e interações do software.

Com essa base estabelecida, a disciplina avançou para o aprofundamento na modelagem estrutural e comportamental do sistema. O estudo de objetos e classes possibilitou a criação do diagrama de classes, representando a estrutura do sistema e os relacionamentos entre seus elementos. O diagrama de sequência trouxe uma visão detalhada do fluxo de mensagens entre os componentes, demonstrando a interação entre os objetos ao longo da execução dos casos de uso. Além disso, foram explorados outros diagramas complementares da UML, como o diagrama de atividades, que representa fluxos de processos dentro do sistema, e diagramas suplementares que ajudam a refinar a modelagem e a documentação do projeto.

Na etapa final, os diagramas elaborados no primeiro trabalho foram expandidos e refinados, incorporando novas representações para aprimorar a modelagem do sistema. Esse processo consolidou a importância da modelagem como um suporte essencial ao desenvolvimento ágil, permitindo uma abordagem iterativa e adaptável ao longo do ciclo de vida do projeto.

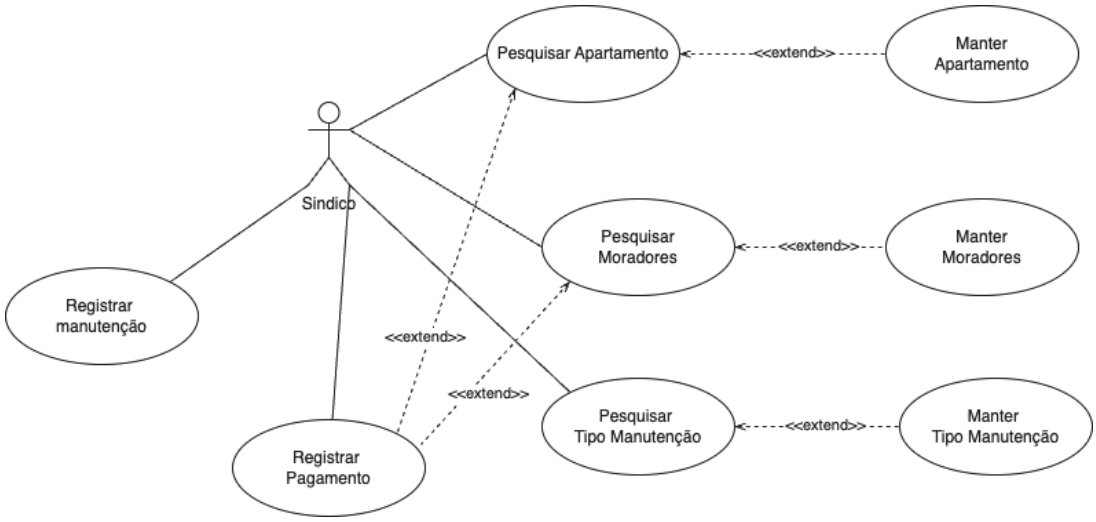
3.1 ARTEFATOS DO PROJETO

FIGURA 11 - DIAGRAMA DE CASO DE USO NÍVEL 1



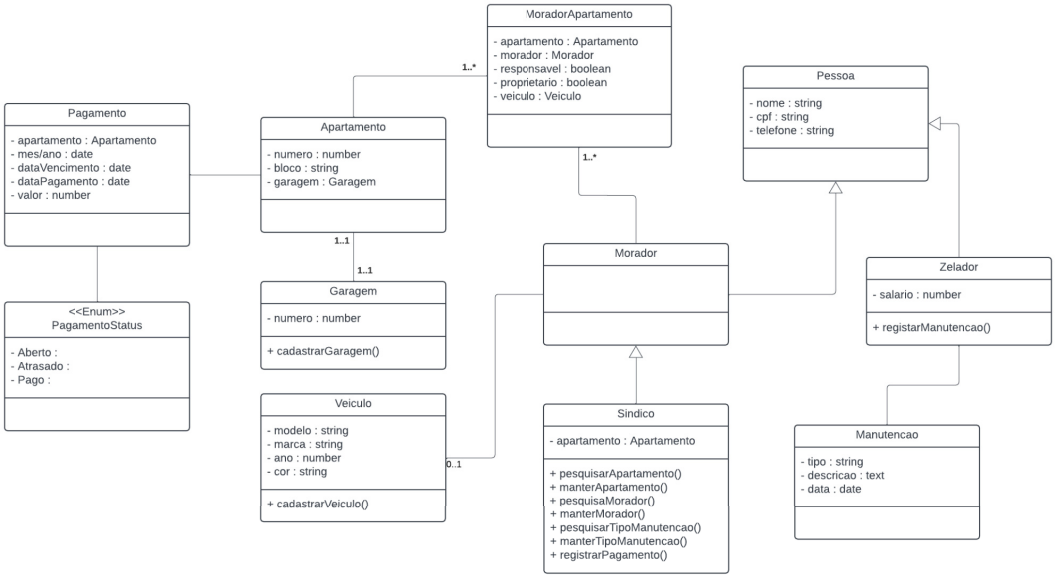
FONTE: O autor (2025)

FIGURA 12 - DIAGRAMA DE CASO DE USO NÍVEL 2



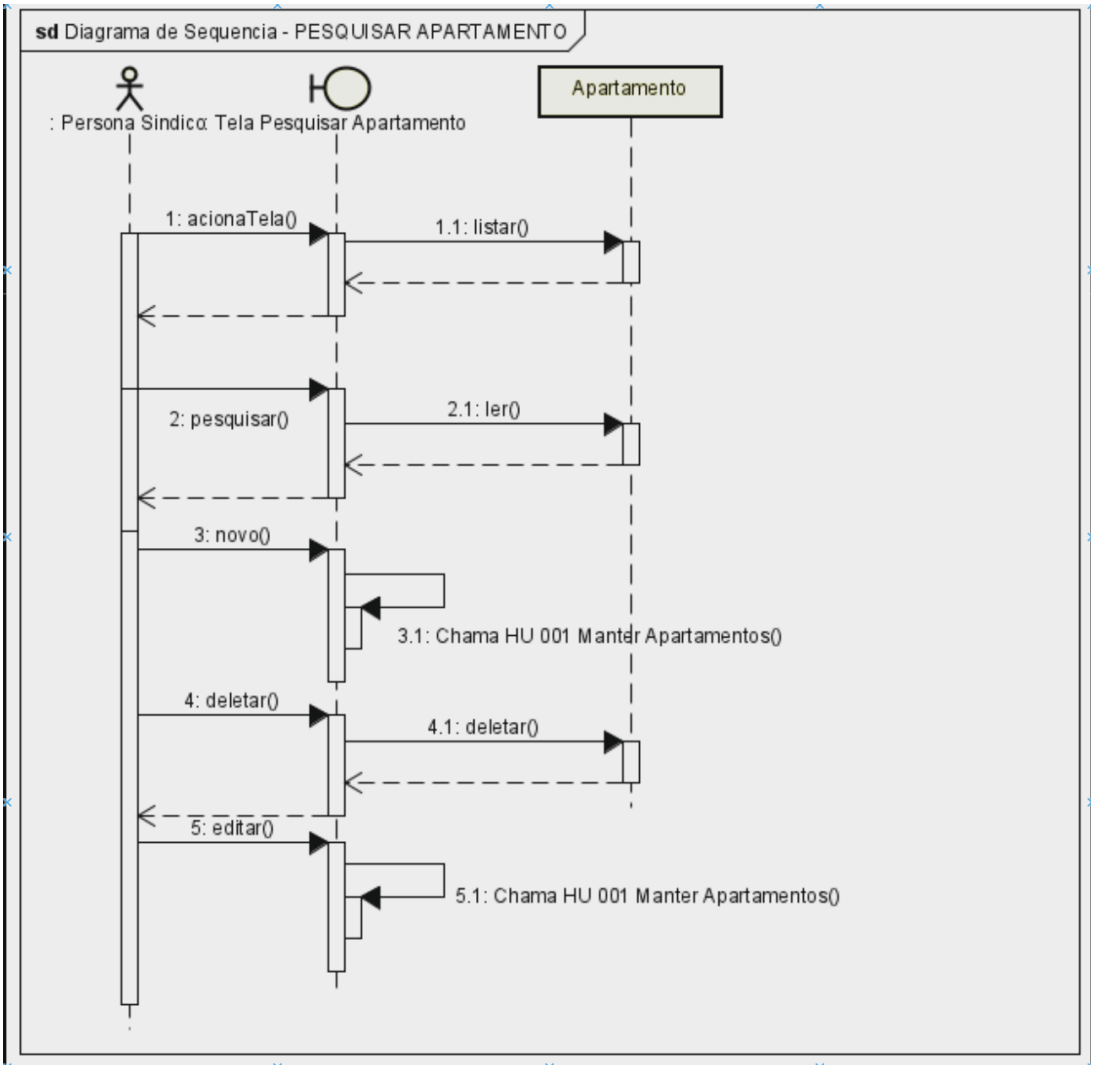
FONTE: O autor (2025)

FIGURA 13 - DIAGRAMA DE CLASSES



FONTE: O autor (2025)

FIGURA 14 - DIAGRAMA DE SEQUÊNCIA



FONTE: O autor (2025)

FIGURA 15 - EXEMPLO DE CENÁRIOS DE TESTE

Detalhamento dos Critérios de Aceitação

Deve receber o parâmetro da tela de pesquisa e configurar a tela

Dado que	o parâmetro da tela de pesquisa foi passado para a tela de manutenção de apartamento
Quando	a tela é apresentada
Então	O sistema deve configurar a tela conforme o parâmetro recebido (R1)

Deve voltar à tela anterior

Dado que	o usuário está na tela de manutenção de apartamento
Quando	o botão “Voltar” é pressionado
Então	o sistema deve retornar à tela anterior de pesquisa

Não deve prosseguir com número de apartamento em branco e único

Dado que	o campo de número do apartamento está em branco ou o número informado já existe no banco de dados
Quando	o usuário tenta salvar as informações
Então	o sistema não deve prosseguir e deve notificar o usuário para preencher o campo com um número válido e único

FONTE: O autor (2025)

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

A disciplina de Gerenciamento Ágil de Projetos de Software aprofunda a aplicação dos princípios ágeis na condução de projetos de desenvolvimento, explorando desde os conceitos fundamentais até práticas avançadas de planejamento, execução e monitoramento. Ao introduzir abordagens como *Scrum* e Kanban, a disciplina proporciona um entendimento essencial sobre organização do fluxo de trabalho, métricas de desempenho e estratégias para garantir entregas contínuas e alinhadas às necessidades do usuário.

Na primeira etapa, o foco está na concepção e planejamento ágil de projetos, trazendo reflexões sobre a diferença entre métodos tradicionais e ágeis e demonstrando como a mentalidade iterativa impacta positivamente o desenvolvimento de software. O aprofundamento no *Scrum* permite compreender sua estrutura, incluindo papéis, eventos e artefatos, possibilitando a construção de um plano de release realista e eficiente Esse exercício prático se torna um elemento fundamental para consolidar o aprendizado, conectando a teoria à aplicação real.

Já na segunda etapa, a disciplina se volta para a gestão do fluxo de trabalho e acompanhamento do progresso, abordando conceitos do PMBOK e a importância da gestão visual na organização das demandas. O estudo do Kanban possibilita a compreensão das dinâmicas de fluxo contínuo e da limitação do trabalho em progresso (WIP), enquanto o aprofundamento em métricas ágeis permite uma análise quantitativa do desempenho da equipe. A execução de um ciclo de 35 dias no Kanban Board Game representa um experimento para testar na prática a eficiência do fluxo de trabalho e avaliar oportunidades de melhoria no processo de desenvolvimento.

A estrutura da disciplina permite uma visão clara sobre como as metodologias ágeis influenciam a gestão de projetos de software, reforçando a importância do planejamento estratégico, do acompanhamento baseado em dados e da adaptação contínua. Ao integrar teoria e prática, a disciplina capacita os alunos a desenvolverem projetos de forma mais ágil, eficiente e alinhada às exigências do mercado.

4.1 ARTEFATOS DO PROJETO

FIGURA 8 - PLANO DE RELEASE: SISTEMA DE SUPORTE A TIMES ÁGEIS

Gerenciamento Ágil de Projetos I
Profa. Dra. Rafaela Mantovani Fontana
Template para o Plano de Release

Sistema de Suporte a Times Ágeis - Planning Poker
Cálculo da Velocidade:

Horas disponíveis por dia:	6	Tamanho da Sprint:	2 Semanas
Horas disponíveis por Sprint:	60	Velocidade:	7

Plano de Release:

Sprint 1	Sprint 2	Sprint 3	Sprint 4
Data Início: 30/04/2024	Data Início: 14/05/2024	Data Início: 28/05/2024	Data Início: 11/06/2024
Data Fim: 13/05/2024	Data Fim: 27/05/2024	Data Fim: 10/06/2024	Data Fim: 24/06/2024
Criar Sala **SENDO** Scrum Master **QUERO** Criar uma sala de Poker Planning **PARA** Conduzir uma dinâmica de Poker Planning ESTIMATIVA 5	**Criar Atividade** **SENDO** Scrum Master **QUERO** Criar uma atividade na sala **PARA** Conduzir uma dinâmica de Poker Planning ESTIMATIVA 2	**Convidar para sala** **SENDO** Scrum Master **QUERO** Convidar outros usuários **PARA** participarem da dinâmica de Poker Planning ESTIMATIVA 3	**Participar da sala** **SENDO** o Time de Desenvolvimento **QUERO** poder participar das sala de Poker Planning **PARA** participarem da dinâmica de Poker Planning ESTIMATIVA 3
Listar Salas **SENDO** Scrum Master **QUERO** poder ver todas as Salas criadas **PARA** para visualizar as dinâmicas que já ocorreram e as agendadas ESTIMATIVA 2	**Listar Atividades** **SENDO** Scrum Master **QUERO** listar as atividades de uma sala **PARA** Conduzir a dinâmica de Poker Planning ESTIMATIVA 5	**Iniciar votação de Atividade** **SENDO** Scrum Master **QUERO** iniciar a votação de uma atividade **PARA** que o time de desenvolvimento possa votar nela ESTIMATIVA 2	**Manter Sala** **SENDO** Scrum Master **QUERO** Editar e deletar salas **PARA** poder manter a gestão dos eventos de Poker Planning ESTIMATIVA 3
		Manter atividades **SENDO** Scrum Master **QUERO** poder editar e deletar uma atividades **PARA** Conduzir a dinâmica de Poker Planning ESTIMATIVA 2	

FONTE: O autor (2025)

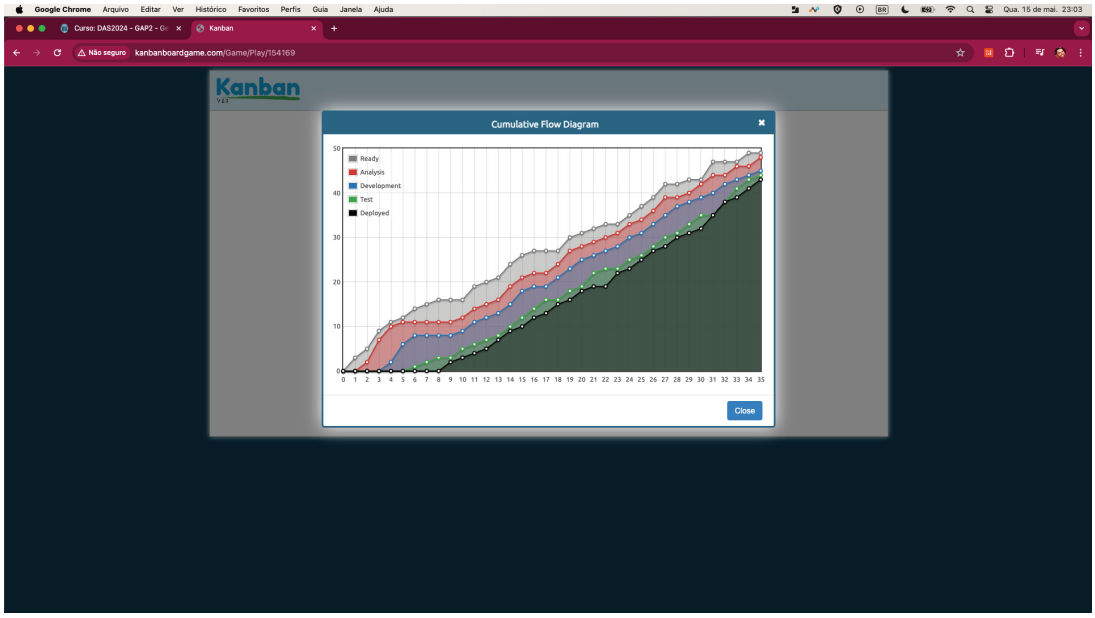
FIGURA 9 - PLANO DE RELEASE: SISTEMA DE SUPORTE A TIMES ÁGEIS

Gerenciamento Ágil de Projetos I
Profa. Dra. Rafaela Mantovani Fontana
Template para o Plano de Release

Sprint 5	Sprint 6	Sprint 7	Sprint 8
Data Início: 25/06/2024	Data Início: 09/07/2024	Data Início: 23/07/2024	Data Início: 06/08/2024
Data Fim: 08/07/2024	Data Fim: 22/07/2024	Data Fim: 05/08/2024	Data Fim: 19/08/2024
Votar em uma Atividade **SENDO** o Time de Desenvolvimento **QUERO** poder votar em uma atividade **PARA** que a média / pontuação da atividade seja calculada ESTIMATIVA 5	**Revelar votos de uma Atividade** **SENDO** Scrum Master **QUERO** poder revelar os votos de uma atividade **PARA** conduzir a dinâmica ágil e iniciar discussões / definição do tamanho de uma atividade. ESTIMATIVA 3	**Finalizar Poker Planning** **SENDO** Scrum Master **QUERO** finalizar uma sala **PARA** ter o retorno das atividades atividades planejadas ESTIMATIVA 8	**Remover Convidado da Sala** **SENDO** Scrum Master **QUERO** remover um convidado da sala **PARA** garantir que apenas participantes do time estejam na sala ESTIMATIVA 5
	Finalizar a votação de uma Atividade **SENDO** Scrum Master **QUERO** poder finalizar a votação de uma atividades **PARA** revelar os as médias das atividades ESTIMATIVA 3		**Permitir Convidado Observador** **SENDO** Scrum Master **QUERO** permitir que um usuário seja observador **PARA** que usuários possam participar sem votar em atividades ESTIMATIVA 3

FONTE: O autor (2025)

FIGURA 10 - DIAGRAMA DE CUMULATIVE FLOW



FONTE: O autor (2025)

5 DISCIPLINA: INTRO – INTRODUÇÃO À PROGRAMAÇÃO

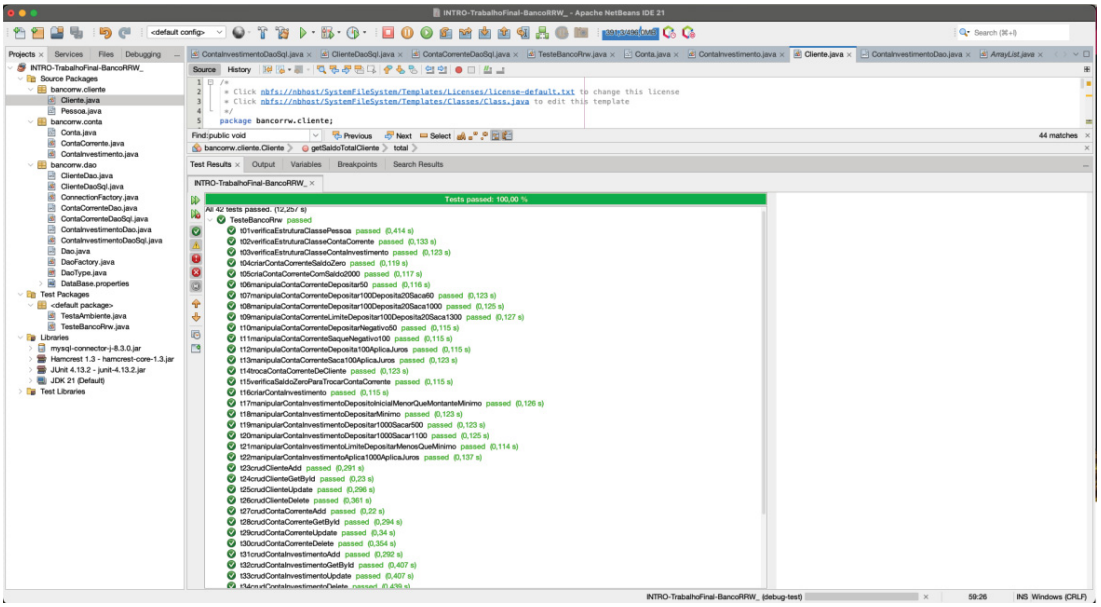
A disciplina de Introdução à Programação teve como objetivo apresentar os fundamentos da programação utilizando a linguagem Java, proporcionando uma base para o desenvolvimento de software. Ao longo da disciplina, foram explorados os princípios essenciais da lógica de programação e os primeiros conceitos de estruturação de código, permitindo a construção de programas.

Um dos momentos mais importantes da disciplina foi a introdução à Programação Orientada a Objetos (POO), um paradigma essencial no desenvolvimento moderno de software (Deitel; Deitel, 2016). Foram explorados conceitos como classes, objetos, encapsulamento, herança e polimorfismo, permitindo estruturar programas de maneira modular e reutilizável. Essa abordagem possibilitou a criação de sistemas mais organizados e escaláveis, preparando o caminho para o aprofundamento em técnicas mais avançadas de desenvolvimento.

Com essa base consolidada, tivemos o desafio de aplicar os conhecimentos adquiridos no sistema onde os testes estavam sendo criados e a aplicação precisava ser escrita, conversando muito bem com os aprendizados de Aspectos Ágeis

5.1 ARTEFATOS DO PROJETO

FIGURA 21 - TESTES DE UNIDADE APROVADOS



FONTE: O autor (2025)

6 DISCIPLINA: BD – BANCO DE DADOS

A disciplina de Banco de Dados teve como objetivo apresentar os fundamentos essenciais para o armazenamento e a manipulação eficiente de dados, proporcionando uma base sólida para o desenvolvimento de sistemas robustos e confiáveis. Foram abordados os princípios de banco de dados, explorando conceitos como estruturação, organização e recuperação de informações

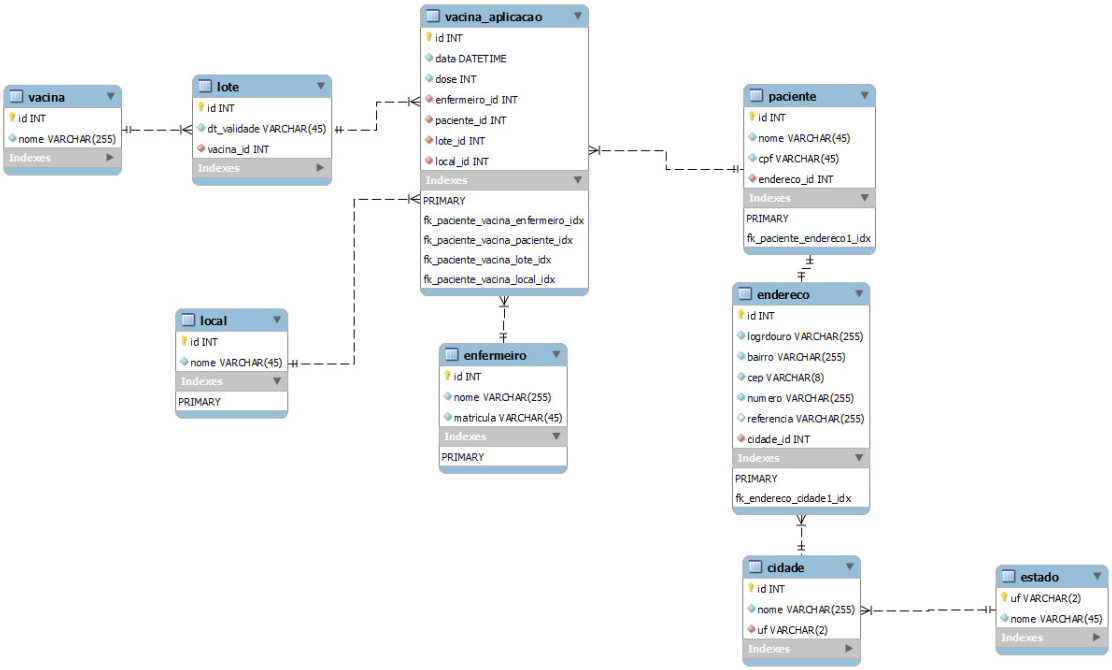
O foco da foi dado em Banco de dados relacionais trabalhando com modelagem de dados, onde foram estudados os conceitos de modelos conceituais, lógicos e físicos, além da normalização para garantir a integridade e a eficiência das tabelas. A partir desse conhecimento, pudemos projetar bancos de dados estruturados de forma otimizada, reduzindo redundâncias e garantindo um melhor desempenho no armazenamento e na consulta das informações.

Além da modelagem, a disciplina aprofundou o estudo da linguagem SQL, explorando os comandos essenciais para a criação (DDL) e manipulação de bancos de dados (DML). Foram abordadas operações como inserção, atualização, exclusão e consulta de dados, além de conceitos mais avançados, como joins, funções agregadas e subconsultas, permitindo a extração eficiente de informações e a interação dinâmica com os dados armazenados.

Como atividade prática, os alunos desenvolveram a modelagem de dois projetos de banco de dados. O primeiro foi um sistema de controle de biblioteca, onde foram definidos os relacionamentos entre livros, usuários e empréstimos, garantindo uma estrutura eficiente para o gerenciamento das operações da biblioteca. O segundo projeto foi de escolha livre, e a opção selecionada foi um sistema de controle de vacinação, onde foram modeladas tabelas para armazenar informações sobre pacientes, vacinas aplicadas, datas de imunização e unidades de saúde. Esses projetos permitiram a aplicação direta dos conceitos estudados, reforçando a importância da modelagem e da linguagem SQL na construção de bancos de dados bem estruturados e funcionais.

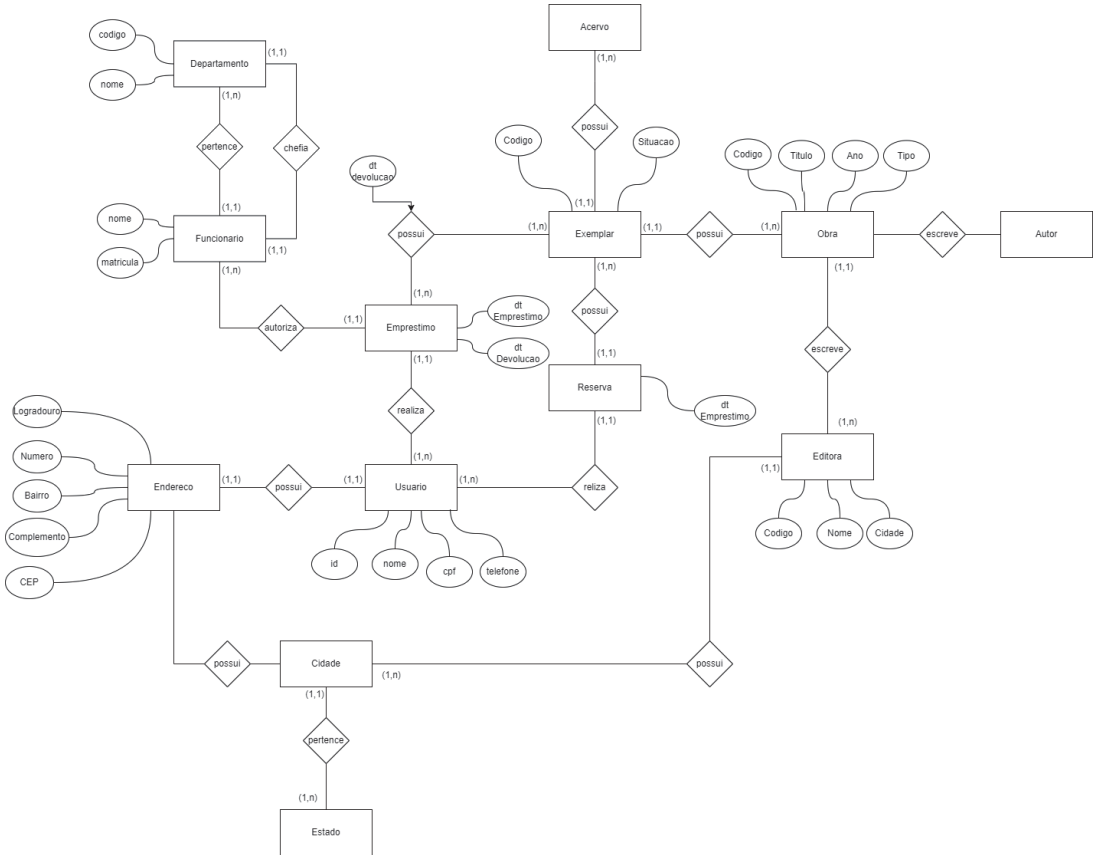
6.1 ARTEFATOS DO PROJETO

FIGURA 22 - MODELO RELACIONAL - VACINA



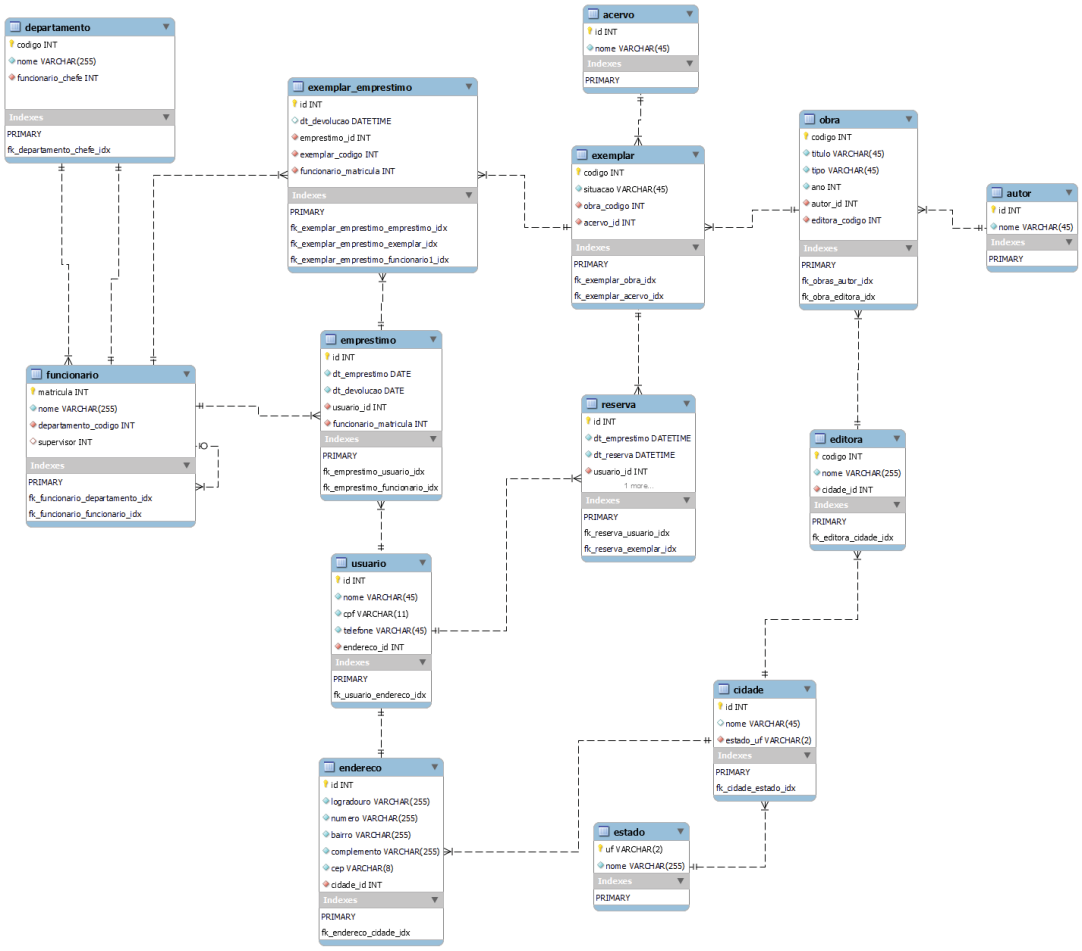
FONTE: O autor (2025)

FIGURA 23 - MODELO ENTIDADE RELACIONAMENTO - BIBLIOTECA



FONTE: O autor (2025)

FIGURA 24 - MODELO RELACIONAL - BIBLIOTECA



FONTE: O autor (2025)

7 DISCIPLINA: AAP – ASPECTOS ÁGEIS DE PROGRAMAÇÃO

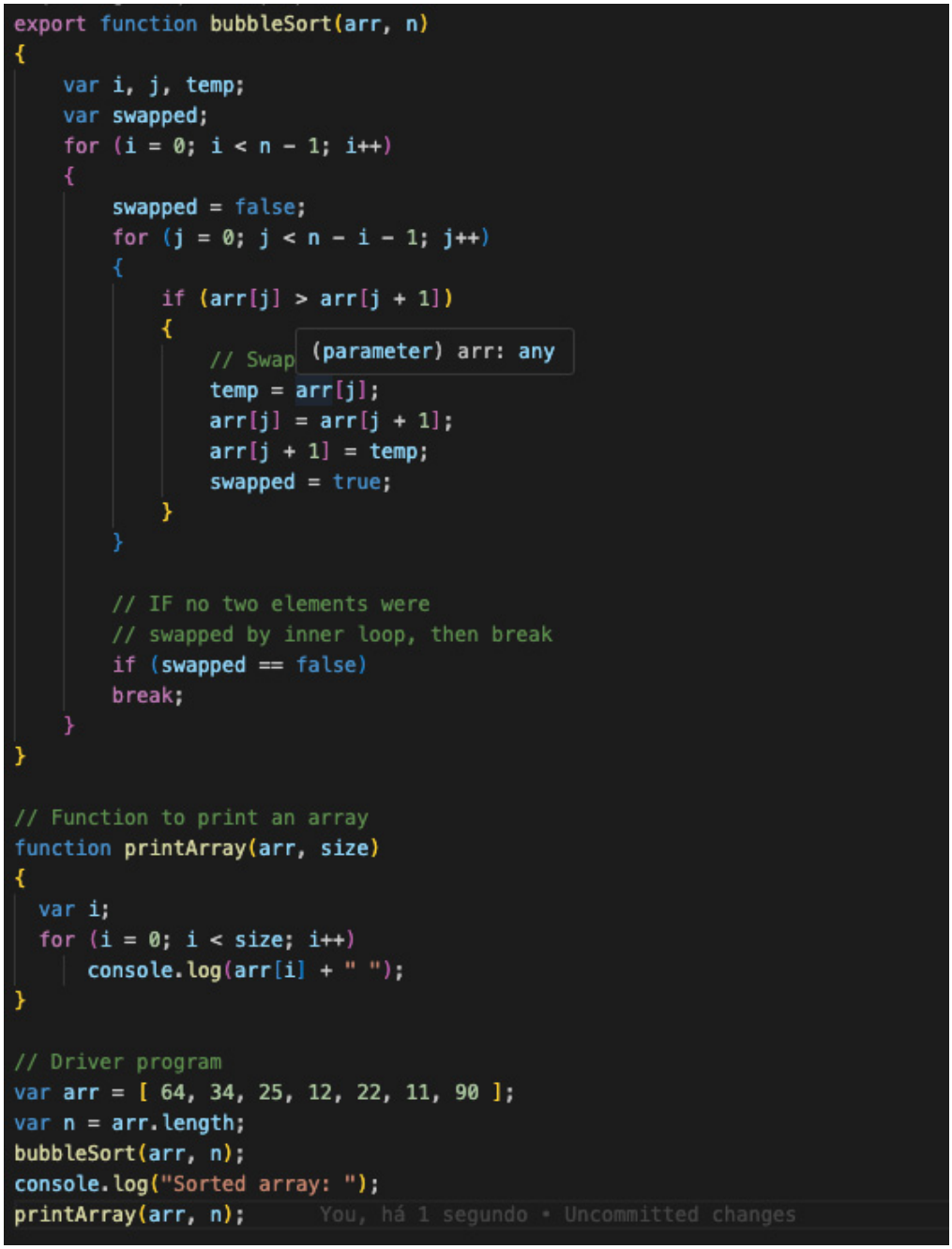
A disciplina de Aspectos Ágeis de Programação tem como objetivo aprimorar a qualidade do código e a eficiência do desenvolvimento de software por meio de boas práticas ágeis. Conceitos como *Clean Code*, *Pair Programming*, Refatoração, TDD, BDD e *Clean Architecture* foram explorados para garantir que o código produzido seja mais legível, sustentável e fácil de manter (Martin, 2009). A ênfase na escrita de código limpo permitiu entender a importância de nomes descritivos, estrutura organizada e redução de complexidade desnecessária, promovendo um desenvolvimento mais ágil e eficiente.

Ao longo da disciplina, a prática de *Pair Programming* foi introduzida como uma técnica colaborativa, reforçando a troca de conhecimento entre desenvolvedores e melhorando a qualidade do código desde as primeiras etapas (Martin, 2011). O conceito de Refatoração se mostrou essencial para aprimorar trechos de código sem alterar seu comportamento, tornando-os mais simples e eficientes (Fowler, 2020).

Como atividade prática, foi realizada a refatoração de uma função de ordenação, aplicando os conceitos de *Clean Code p*ara melhorar sua organização, clareza e eficiência. Esse exercício possibilitou a aplicação direta dos conhecimentos adquiridos, demonstrando como pequenas mudanças na estrutura do código podem torná-lo mais compreensível e fácil de manter. Com isso, a disciplina reforçou a importância de práticas ágeis no dia a dia do desenvolvimento de software, garantindo código de alta qualidade e alinhado aos princípios da engenharia de software moderna.

7.1 ARTEFATOS DO PROJETO

FIGURA 16 - CÓDIGO INICIAL



```
export function bubbleSort(arr, n)
{
    var i, j, temp;
    var swapped;
    for (i = 0; i < n - 1; i++)
    {
        swapped = false;
        for (j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                // Swap (parameter) arr: any
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
                swapped = true;
            }
        }

        // IF no two elements were
        // swapped by inner loop, then break
        if (swapped == false)
        break;
    }
}

// Function to print an array
function printArray(arr, size)
{
  var i;
  for (i = 0; i < size; i++)
      console.log(arr[i] + " ");
}

// Driver program
var arr = [ 64, 34, 25, 12, 22, 11, 90 ];
var n = arr.length;
bubbleSort(arr, n);
console.log("Sorted array: ");
printArray(arr, n);      You, há 1 segundo • Uncommitted changes
```

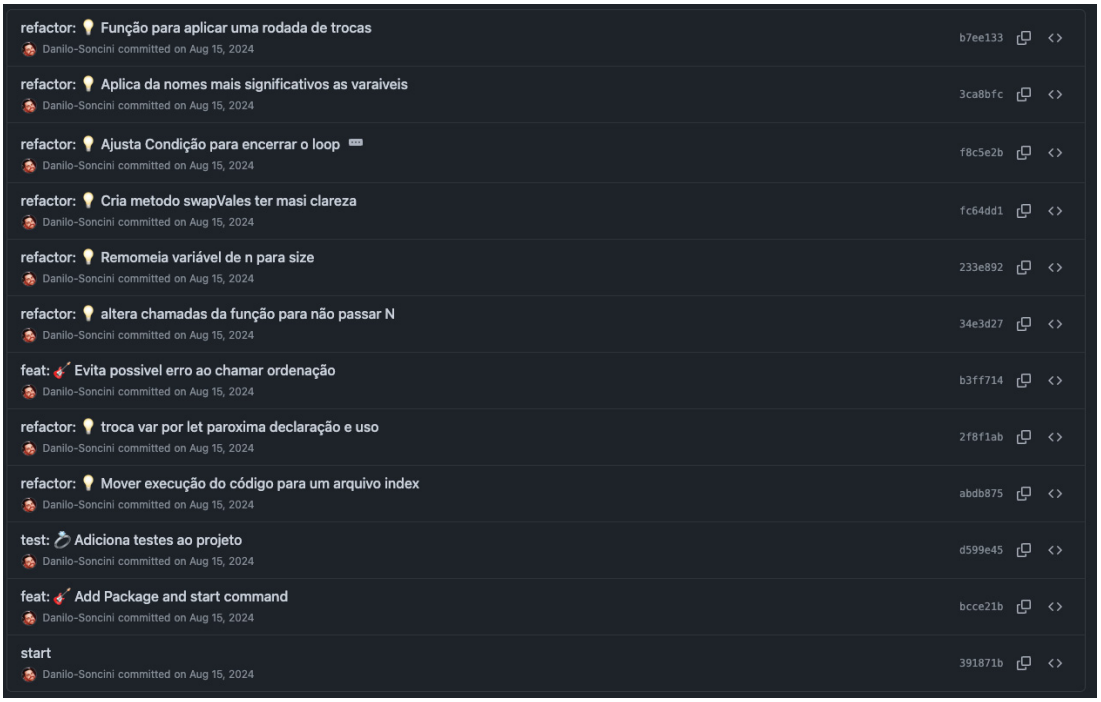
FONTE: O autor (2025)

FIGURA 17 - CÓDIGO FINAL

```
export function bubbleSort(numbers){
    for (let i = 0; i < numbers.length - 1; i++) {
        if (!swapValues(numbers, i - 1)) break;
    }
}

function swapValues(numbers, alreadySorted) {
    let swapped = false;
    for (let key = 0; key < numbers.length - alreadySorted; key++) {
        if (isNextValueGreaterThanCurrent(numbers, key)) {
            swapWithNextValue(numbers, key);
            swapped = true;
        }
    }       You, há 7 meses • start
    return swapped;
}

function isNextValueGreaterThanCurrent(numbers, key) {
    return numbers[key] > numbers[key + 1];
}

function swapWithNextValue(numbers, key) {
    const temp = numbers[key];
    numbers[key] = numbers[key + 1];
    numbers[key + 1] = temp;
}
```

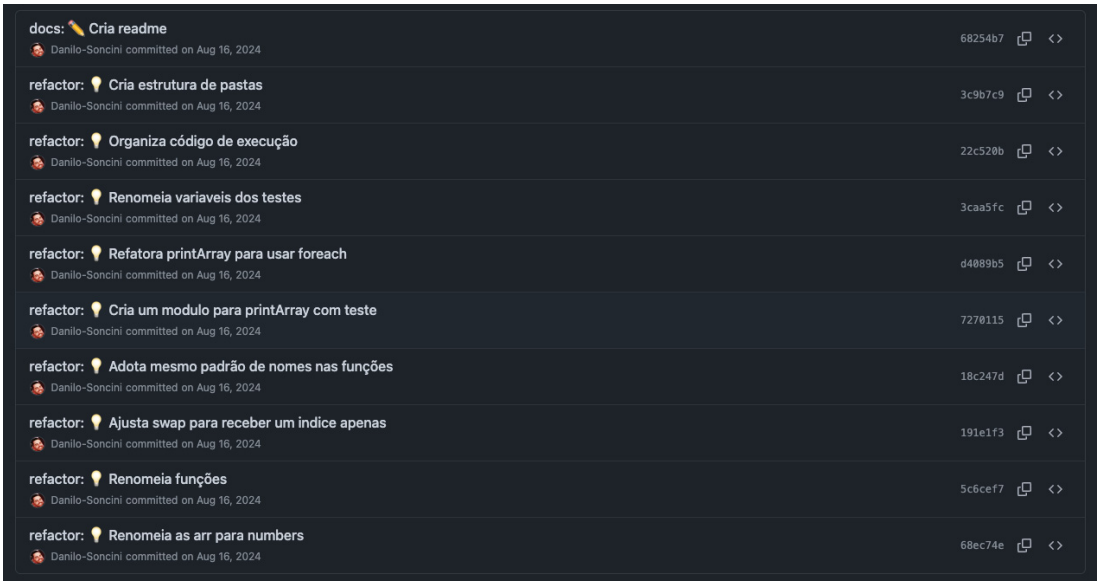
FONTE: O autor (2025)

FIGURA 19 - SEQUÊNCIA DE COMMITS UTILIZADOS NA REFATORAÇÃO



FONTE: O autor (2025)

FIGURA 20 - SEQUÊNCIA DE COMMITS UTILIZADOS NA REFATORAÇÃO



FONTE: O autor (2025)

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

A disciplina de Desenvolvimento Web teve como foco a construção de aplicações modernas utilizando o framework Angular e a linguagem *TypeScript*, explorando os principais conceitos e práticas para o desenvolvimento de sistemas dinâmicos e interativos. Desde os primeiros momentos, foi introduzido o conceito de *Single Page Application* (SPA), uma abordagem que permite que a aplicação funcione de forma mais fluida e responsiva, carregando apenas os dados necessários sem a necessidade de recarregar toda a página. Essa arquitetura melhora a experiência do usuário e otimiza o desempenho das aplicações web.

Foram explorados conceitos como componentes, diretivas, serviços e roteamento, demonstrando como a modularização do código e a reutilização de componentes auxiliam no desenvolvimento ágil. O *TypeScript*, por sua vez, trouxe tipagem estática e recursos avançados que melhoram a qualidade do código, tornando-o mais seguro e fácil de manter.

Um dos temas aprofundados foi a implementação de formulários e validações, garantindo que os dados inseridos pelos usuários estivessem corretos e em conformidade com as regras de validações personalizadas para melhorar a usabilidade e evitar erros de entrada. Esse aprendizado mostrou como os frameworks web contribuem para o desenvolvimento ágil, facilitando a criação de interfaces robustas e interativas de forma eficiente.

A disciplina abordou a construção de um backend utilizando Java, permitindo a integração entre o frontend desenvolvido no Angular e um servidor que gerencia os dados e as regras de negócio. Foram explorados conceitos como APIs REST, consumo de endpoints e comunicação entre frontend e backend, garantindo que os alunos tivessem uma visão completa do desenvolvimento full-stack.

9 DISCIPLINA: UX – UX NO DESENVOLVIMENTO ÁGIL DE SOFTWARE

A disciplina de UX *User Experience* agregou um novo olhar ao desenvolvimento de software, trazendo a importância de colocar o usuário no centro do processo de criação. Ao longo da disciplina, foi reforçado que um sistema bem projetado não se trata apenas de funcionalidade, mas também de como ele se encaixa na realidade e nos desafios do usuário. Para isso, foram aplicadas técnicas como a criação de personas e o mapeamento da jornada do usuário, permitindo uma compreensão mais profunda das necessidades, dores e expectativas de quem irá utilizar a aplicação.

Outro ponto central foi a busca por feedbacks rápidos, uma prática essencial para validar as decisões de design e garantir que o sistema evolua de forma alinhada às necessidades reais dos usuários. Além disso, a consistência na interface, na navegação e no uso foi enfatizada como um fator essencial para criar experiências intuitivas e eficientes. O aprendizado reforçou que manter padrões visuais e interativos melhora significativamente a usabilidade e reduz a curva de aprendizado dos usuários.

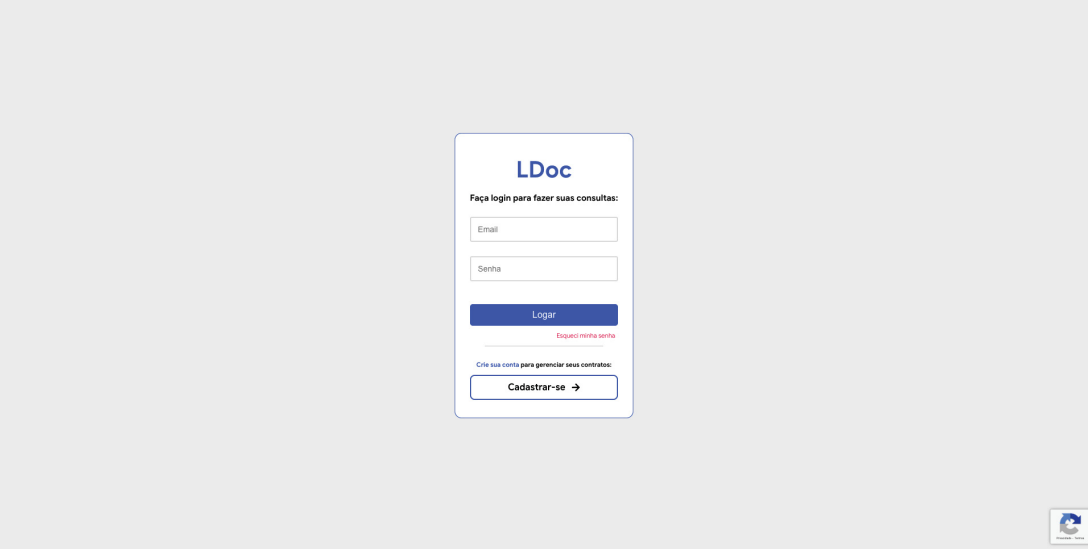
A responsividade e adaptabilidade também foram temas recorrentes, garantindo que as interfaces sejam acessíveis e funcionais em diferentes dispositivos e tamanhos de tela. A disciplina destacou que, em um cenário onde o acesso móvel é predominante, projetar para diferentes contextos de uso é uma necessidade fundamental.

Os conceitos de design foram explorados, abordando o uso estratégico de cores, tipografia e acessibilidade, tornando as interfaces mais agradáveis e inclusivas. Foi enfatizada a importância de projetar sistemas que não apenas sejam visualmente atraentes, mas que também ofereçam uma experiência fluida para todos os usuários, incluindo aqueles com limitações visuais ou motoras.

No trabalho da disciplina pudemos trabalhar alguns conceitos de design e prototipação e trabalhar na necessidade do usuário.

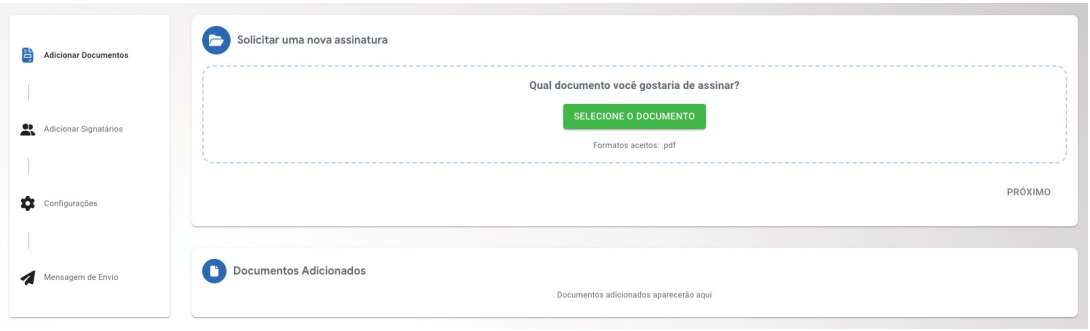
9.1 ARTEFATOS DO PROJETO

FIGURA 25 - TELA DE LOGIN



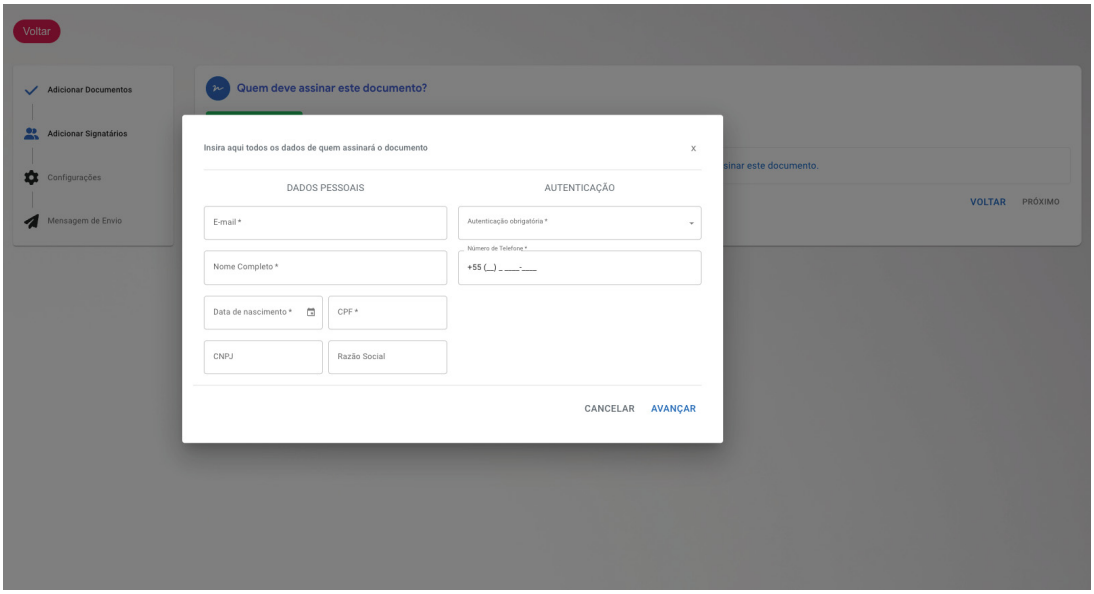
FONTE: O autor (2025)

FIGURA 26 - FLUXO CADASTRO DO DOCUMENTO



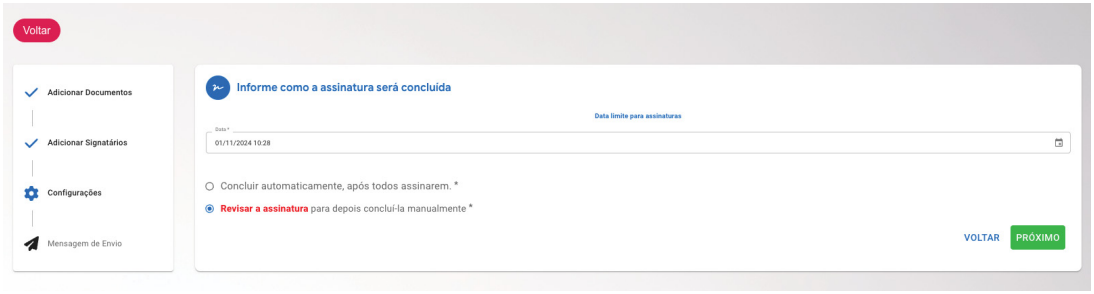
FONTE: O autor (2025)

FIGURA 27 - FORMULÁRIO DE CADASTRO DE ASSINANTE



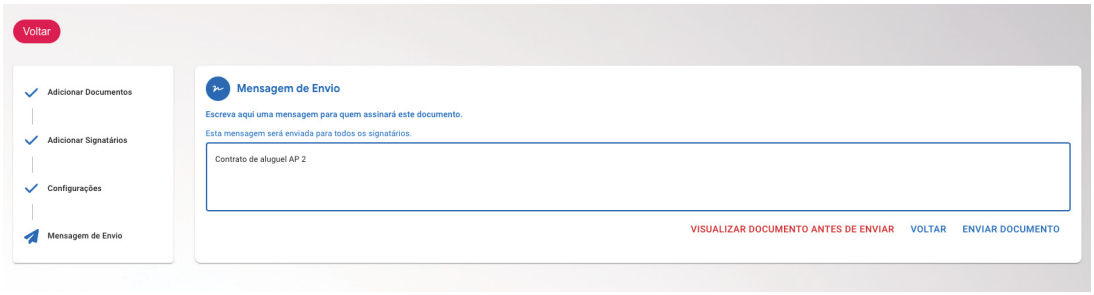
FONTE: O autor (2025)

FIGURA 28- ESCOLHA DE COMO A ASSINATURA SER FINALIZADA



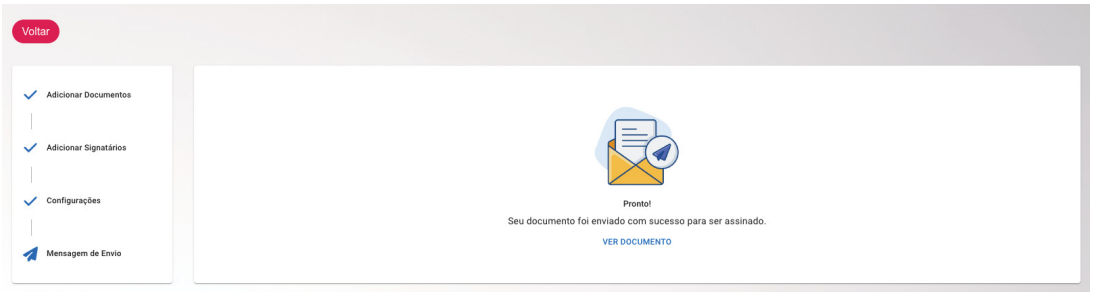
FONTE: O autor (2025)

FIGURA 29 - FORMULÁRIO MENSAGEM PARA SOLICITAÇÃO DA ASSINATURA



FONTE: O autor (2025)

FIGURA 30 - CONFIRMAÇÃO DE ENVIOS

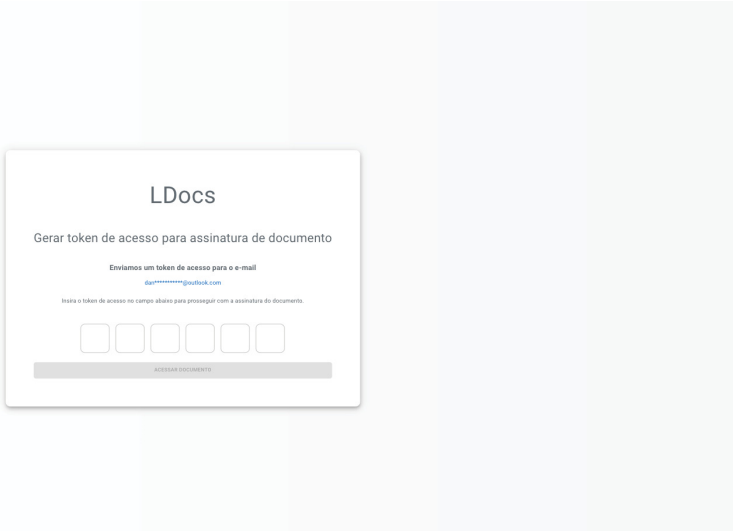


FONTE: O autor (2025)

FIGURA 31 - ENVIO DE TOKEN PARA CONFIRMAÇÃO

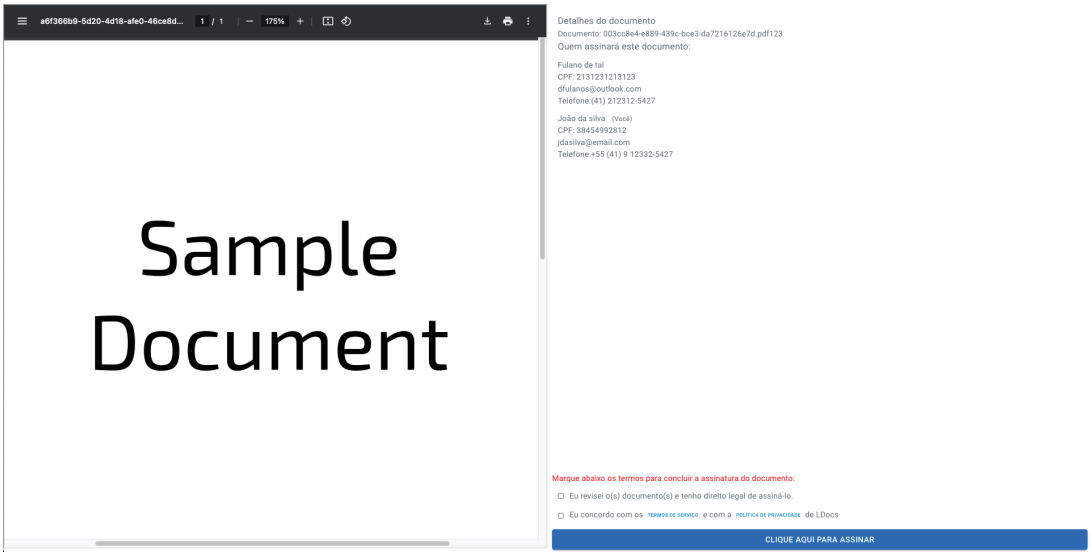
FONTE: O autor (2025)

FIGURA 32 - PREENCHIMENTO DO TOKEN



FONTE: O autor (2025)

FIGURA 33 - CONFIRMAÇÃO DA ASSINATURA



FONTE: O autor (2025)

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

A disciplina de Desenvolvimento Mobile teve como objetivo explorar o ecossistema do mercado mobile e capacitar os alunos para o desenvolvimento de aplicativos Android utilizando o Android Studio. O curso iniciou com uma visão geral do setor, apresentando as tendências e desafios do desenvolvimento para dispositivos móveis, destacando a importância da experiência do usuário e da performance das aplicações nesse ambiente altamente dinâmico.

O aprendizado seguiu com um aprofundamento nos conceitos fundamentais do Android, com foco na construção da interface por meio das Views e no funcionamento do empilhamento de Views dentro da estrutura do sistema. Através de aulas práticas, foram exploradas técnicas essenciais para a criação de interfaces responsivas e interativas, permitindo um melhor entendimento sobre a navegação entre telas e a organização dos elementos visuais em uma aplicação mobile.

Com o avanço da disciplina, os estudos se concentraram no reuso de Views utilizando RecyclerView, uma ferramenta essencial para lidar com listas de grande volume de dados de forma eficiente. Além disso, a disciplina abordou a integração com bancos de dados locais e remotos, bem como a comunicação com APIs, permitindo que os aplicativos pudessem consumir e armazenar dados de maneira estruturada. Um dos grandes desafios enfrentados foi a programação assíncrona no ambiente Android, uma necessidade fundamental para garantir uma experiência fluida ao usuário, evitando travamentos e melhorando o desempenho da aplicação.

A experiência da disciplina proporcionou uma continuidade natural ao conhecimento adquirido ao longo do curso, mas agora aplicado ao desenvolvimento mobile. A transição dos conceitos já estudados para o contexto de aplicativos Android reforçou a importância de práticas ágeis, do reuso de código e da otimização do desempenho, preparando os alunos para enfrentar os desafios do desenvolvimento de aplicações para dispositivos móveis.

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

A disciplina de *DevOps* apresentou uma visão abrangente do ciclo de vida do desenvolvimento de software, destacando a importância da integração entre desenvolvimento e operações para entregar software de forma mais eficiente e confiável. O conceito de *DevOps* foi introduzido como um conjunto de práticas que busca automatizar e aprimorar os processos de desenvolvimento, teste e entrega contínua, garantindo maior agilidade e qualidade nos projetos.

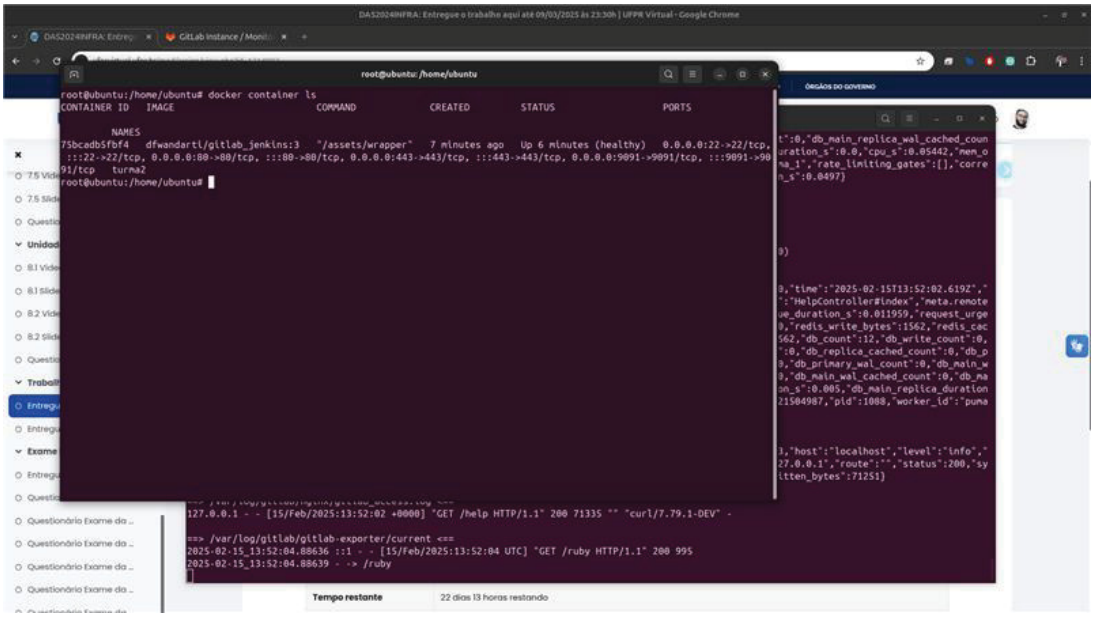
Um dos primeiros temas abordados foi a gestão de versionamento com Git e GitHub, enfatizando seu papel essencial na colaboração entre equipes, rastreamento de mudanças e controle do código-fonte. Em seguida, exploramos a importância das métricas, que permitem monitorar o desempenho dos processos e tomar decisões mais embasadas para otimizar o fluxo de desenvolvimento.

Outro ponto-chave da disciplina foi a conteinerização com Docker, que possibilita criar ambientes isolados e padronizados, facilitando a replicação e a portabilidade das aplicações. Também foram abordados conceitos fundamentais sobre pipelines de CI/CD *Continuous Integration/Continuous Deployment*, demonstrando como automatizar a construção, os testes e a entrega de software. Além disso, tivemos um primeiro contato com Kubernetes, um orquestrador de containers que permite escalar e gerenciar aplicações distribuídas de forma eficiente. O tema da observabilidade também foi explorado, reforçando a necessidade de monitoramento contínuo para identificar falhas rapidamente e garantir a estabilidade dos sistemas.

A disciplina foi dinâmica e focada em apresentar as bases de cada conceito, oferecendo aos alunos a oportunidade de aprofundamento nos temas de maior interesse. O trabalho prático teve como foco o uso do Docker e Git, permitindo aplicar os conhecimentos adquiridos na prática e compreender melhor os benefícios dessas ferramentas dentro do fluxo *DevOps*. Com isso, a disciplina proporcionou uma base sólida para entender os desafios e as soluções modernas na automação e entrega contínua de software.

11.1 ARTEFATOS DO PROJETO

FIGURA 34 - PRINT DA CRIAÇÃO DO DOCKER



FONTE: O autor (2025)

12 DISCIPLINA: TEST – TESTES AUTOMATIZADOS

A disciplina de Testes de Software foi objetiva e direta, oferecendo uma visão clara sobre a importância dos testes no desenvolvimento de software e os diferentes níveis de testes que garantem a qualidade e a confiabilidade das aplicações. Desde o início, foi enfatizado que testar não é apenas uma etapa adicional do processo, mas sim uma prática essencial para evitar falhas e garantir que o código esteja sempre funcionando conforme o esperado.

O foco principal da disciplina esteve nos testes unitários dentro do ecossistema Java, explorando como estruturar códigos testáveis e reforçando a conexão com os conceitos abordados na disciplina de Aspectos Ágeis de Programação, como TDD (*Test-Driven Development)*. Durante as aulas práticas, os alunos aprenderam a criar e executar testes unitários de forma eficiente, garantindo que cada componente do software fosse validado de maneira isolada, facilitando a manutenção e a evolução do sistema.

Além dos testes unitários, a disciplina também abordou a importância de testes mais abrangentes, culminando na prática de testes End-to-End (E2E) utilizando a ferramenta Playwright. Essa abordagem permitiu validar o funcionamento completo da aplicação, simulando a interação real do usuário e garantindo que todos os componentes trabalhassem corretamente juntos.

O trabalho final da disciplina consistiu na implementação de um teste utilizando o Playwright, consolidando os conceitos aprendidos e permitindo que os alunos experimentassem na prática os desafios e benefícios da automação de testes. Dessa forma, a disciplina proporcionou uma base essencial para a aplicação de testes dentro do ciclo de desenvolvimento, reforçando a necessidade de garantir qualidade, confiabilidade e segurança no código desde as fases iniciais do projeto.

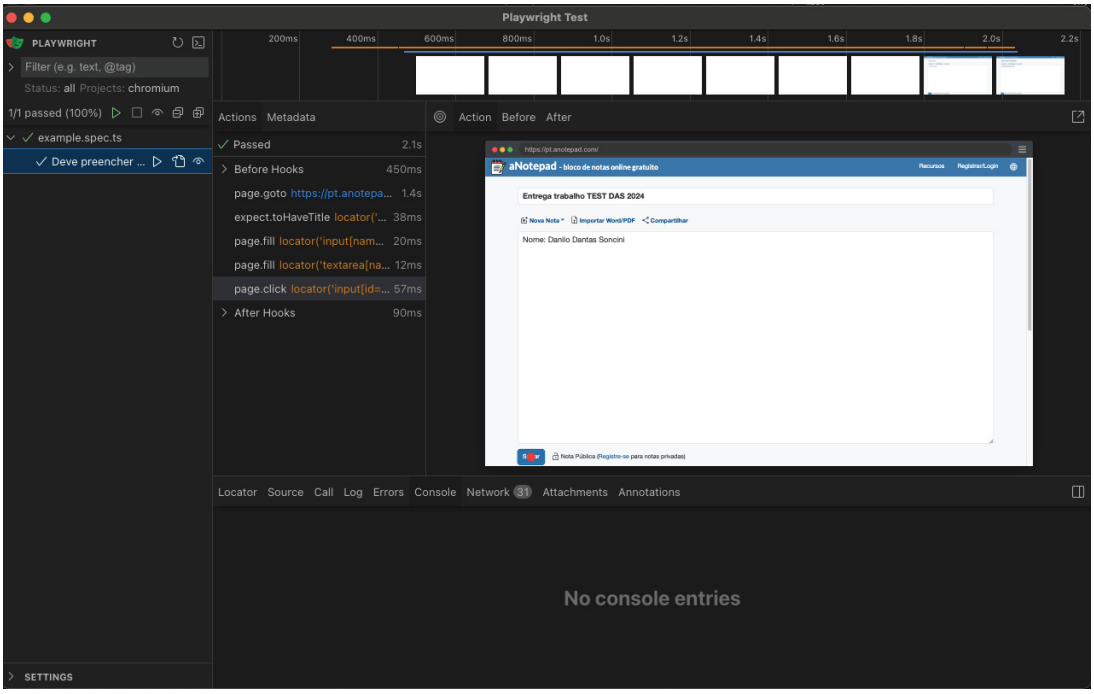
12.1 ARTEFATOS DO PROJETO

FIGURA 35 - ARQUIVO DE TESTE

```
import { test, expect } from '@playwright/test';

test('Deve preencher os dados do trabalho', async ({ page }) => {
 await page.goto('https://pt.anotepad.com/', { waitUntil: 'domcontentloaded' });

 await expect(page).toHaveTitle(/Bloco de Notas Online - Crie e Compartilhe Notas
Online/);

 await page.fill('input[name="notetitle"]', 'Entrega trabalho TEST DAS 2024');

 await page.fill('textarea[name="notecontent"]', 'Nome: Danilo Dantas Soncini');

 await page.click('input[id="btnSaveNote"]');
});
```

FONTE: O autor (2025)

FIGURA 36 - RESULTADO DO TESTE



FONTE: O autor (2025)

13 CONCLUSÃO

A jornada da pós-graduação em Desenvolvimento Ágil de Software proporcionou uma experiência transformadora, consolidando o aprendizado sobre as metodologias ágeis e suas aplicações práticas. O curso, estruturado em módulos que abordaram os principais conceitos do desenvolvimento ágil, desde o planejamento estratégico até a entrega de software, permitiu uma imersão profunda nos diversos aspectos da construção de sistemas robustos e eficientes.

A cada disciplina, novos desafios surgiram, e a aplicação prática dos conhecimentos adquiridos em projetos práticos tornou a aprendizagem mais engajadora e significativa. Destaca-se a importância do trabalho em equipe, da comunicação eficaz, da adaptabilidade às mudanças e da busca constante por aprimorar os processos e a qualidade do código.

A experiência prática, com a utilização de ferramentas como *Scrum*, Kanban, UML, TDD, Git, Docker e Kubernetes, proporcionou a oportunidade de vivenciar as melhores práticas ágeis, vivenciando como a aplicação de cada técnica impacta positivamente o desenvolvimento de software.

A pós-graduação em Desenvolvimento Ágil de Software forneceu as ferramentas e a base para a construção de uma carreira de sucesso no desenvolvimento de software, preparando profissionais capazes de se adaptar às demandas do mercado e entregar soluções inovadoras e de alta qualidade.

REFERÊNCIAS

AMBLER, S. **Agile Modeling: Effective Practices for Extreme Programming and the Unified Process**. New York: John Wiley & Sons, 2002.

ANDERSON, D. **Kanban: Successful Evolutionary Change for Your Technology Business**. Sequim: Blue Hole Press, 2010.

BECK, K. **Test-Driven Development: by Example**. Boston: Addison-Wesley, 2002.

BECK, K. **Programação Extrema (XP) explicada: acolha as mudanças**. Porto Alegre: Bookman, 2004

DEITEL, P; DEITEL, H. **Java: como programar. 10. ed**. São Paulo: Pearson, 2016.

FOWLER, M. **Refatoração: Aperfeiçoando o Design de Códigos Existentes. 2. ed**. São Paulo: Novatec, 2020.

GOTHELF, J; SEIDEN, J. **Lean UX: Applying Lean Principles to Improve User Experience. 2. ed**. Sebastopol: O'Reilly Media, 2013.

HUMBLE, J; FARLEY, D. **Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation**. Boston: Addison-Wesley, 2011.

MARTIN, R. **Código Limpo: Habilidades Práticas do Agile Software**. São Paulo: Alta Books, 2009.

MARTIN, R. **O Codificador Limpo: Um Código de Conduta para Programadores Profissionais**. São Paulo: Alta Books, 2011.

PRESSMAN, R.; MAXIM, B. **Engenharia de Software: uma abordagem profissional. 9. ed**. Porto Alegre: AMGH, 2021.

RIGBY, D. **Ágil do Jeito Certo: Transformação sem Caos**. São Paulo: Benvirá, 2020.

SCHWABER, K; SUTHERLAND, J. **The Scrum Guide. 2020**. Disponível em: https://scrumguides.org. Acesso em: 17 out. 2025.