UNIVERSIDADE FEDERAL DO PARANÁ

LUCAS VINICIUS HACK

MEMORIAL DE PROJETOS: TRAJETÓRIA ACADÊMICA E PROJETOS EM DESENVOLVIMENTO ÁGIL

CURITIBA

2025

LUCAS VINICIUS HACK

MEMORIAL DE PROJETOS: TRAJETÓRIA ACADÊMICA E PROJETOS EM DESENVOLVIMENTO 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

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 **LUCAS VINICIUS HACK**, intitulada: **MEMORIAL DE PROJETOS: TRAJETÓRIA ACADÊMICA E PROJETOS EM DESENVOLVIMENTO 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, 29 de Outubro de 2025.

RAFAELA MANTOVANI FONTANA
Presidente da Banca Examinadora

JAIME WOJCIECHOWSKI
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

RESUMO

Este documento apresenta a trajetória dos projetos desenvolvidos durante toda a especialização em Desenvolvimento Ágil de Software na Universidade Federal do Paraná (UFPR), com o objetivo de organizar, em ordem cronológica, os resultados finais das disciplinas do curso e demonstrar a evolução do conhecimento e sua aplicação prática no contexto do desenvolvimento ágil. Ao longo do curso, foram abordados desde os conceitos fundamentais de análise e modelagem de sistemas até práticas modernas de desenvolvimento de software, incluindo programação, banco de dados, desenvolvimento web, dispositivos móveis, DevOps e testes automatizados. Cada disciplina foi tratada com seus objetivos, relevância e conexões com temas essenciais ao desenvolvimento de um projeto de software real, proporcionando uma visão abrangente de todas as etapas, desde a concepção da ideia pelo cliente até a entrega de um sistema funcional. O memorial reúne materiais diversos, como diagramas, protótipos, cronogramas, tabelas, imagens e trechos de código, evidenciando o que foi produzido ao longo da especialização. Ao final, apresenta uma análise geral que enfatiza a importância das metodologias ágeis no cenário atual da engenharia de software e destaca a contribuição do curso tanto para o aprimoramento técnico quanto para o desenvolvimento da capacidade de análise crítica.

Palavras-chave: desenvolvimento ágil; modelagem de sistemas; testes automatizados; engenharia de software; programação.

ABSTRACT

This document presents the trajectory of the projects developed throughout the specialization in Agile Software Development at the Federal University of Paraná (UFPR), with the purpose of organizing, in chronological order, the final results of the course subjects and demonstrating the evolution of knowledge and its practical application in the context of agile development. Throughout the program, topics ranged from fundamental concepts of systems analysis and modeling to modern software development practices, including programming, databases, web development, mobile devices, DevOps, and automated testing. Each subject was addressed with its objectives, relevance, and connections to key themes in the development of a real software project, providing a comprehensive view of all stages—from the conception of the client's idea to the delivery of a functional system. The report gathers various materials such as diagrams, prototypes, schedules, tables, images, and code excerpts, showcasing what was produced during the specialization. Finally, it presents a general analysis that highlights the importance of agile methodologies in the current software engineering landscape and underscores the course's contribution to both technical improvement and the enhancement of critical analysis skills.

Keywords: agile development; systems modeling; automated testing; software engineering; programming.

SUMÁRIO

1 PARECER TÉCNICO

O presente parecer técnico tem como objetivo apresentar uma análise integrada dos projetos desenvolvidos ao longo do curso de Especialização em Desenvolvimento Ágil de Software da Universidade Federal do Paraná (UFPR). O conjunto das disciplinas, organizadas de forma progressiva e complementar, proporcionou uma visão ampla e prática sobre o ciclo de vida de desenvolvimento de software, desde a concepção e modelagem até a entrega contínua e automatizada, sempre com base nos valores e princípios do Manifesto Ágil (Beck et al., 2001).

A disciplina Métodos Ágeis de Desenvolvimento de Software - MADS foi o ponto de partida para a compreensão dos fundamentos das metodologias Scrum, Lean, Kanban e XP. Foi possível compreender a importância da colaboração e da transparência no processo de desenvolvimento, bem como o valor de ciclos curtos de entrega e feedback contínuo (Humble; Farley, 2014). Os autores das bibliografias complementavam essa visão ao reforçar que o ágil é mais do que uma metodologia: trata-se de uma cultura organizacional que estimula a inovação e a adaptação em ambientes dinâmicos.

As disciplinas de Modelagem Ágil de Software I – Visão Funcional (MAG 1) e Modelagem Ágil de Software II – Visão Estrutural (MAG 2) mostraram, na prática, como a modelagem visual pode apoiar o desenvolvimento colaborativo e evolutivo do software. O uso de diagramas UML e de casos de uso auxilia na comunicação entre os membros da equipe e promove uma visão compartilhada do sistema (Guedes, 2004) e além disso, foi ressaltado que uma boa modelagem deve ser simples, adaptável e de fácil manutenção, características fundamentais em projetos ágeis. Essas práticas foram aplicadas nas atividades da das disciplinas, conectando requisitos a componentes técnicos e garantindo coerência entre arquitetura e implementação.

Em Gerenciamento Ágil de Projetos de Software I e II (GAP 1 e GAP 2) foi possível compreender como o gerenciamento ágil contribui para manter o time organizado, sem perder a agilidade e o foco na entrega de valor. As disciplinas mostraram que a gestão enxuta (*Lean Management*) e o uso de métricas visuais, como quadros Kanban, promovem maior visibilidade e eficiência, e no dia a dia, como é possível equilibrar a autonomia das equipes com uma gestão eficiente dos processos, garantindo organização e previsibilidade sem comprometer a agilidade da

entrega dos projetos. Essa abordagem reflete os princípios do Lean Software Development, que buscam eliminar desperdícios e maximizar a entrega de valor (Poppendieck; Poppendieck, 2003).

A disciplina UX (*User Experience*) no Desenvolvimento Ágil de Software evidenciou o papel do design centrado no usuário dentro do contexto ágil, promovendo a criação de soluções alinhadas às necessidades reais das pessoas. Durante as atividades, foi desenvolvido protótipos de telas de um aplicativo. Foi estudado também o processo de desenvolvimento aproximando designers, desenvolvedores e stakeholders na construção de produtos mais intuitivos, funcionais e voltados principalmente à experiência do usuário (Goethelf, 2021).

O desenvolvimento prático de sistemas foi consolidado nas disciplinas Introdução à Programação, Aspectos Ágeis de Programação, Banco de Dados, Desenvolvimento Web I e II (WEB 1 e WEB 2) e Desenvolvimento Mobile I e II (MOB 1 e MOB 2). Nessas etapas, o uso de práticas modernas como *Test-Driven Development* (TDD) e integração contínua evidenciou a importância de se construir uma base sólida de programação, e fez conexão com as disciplinas voltadas às metodologias ágeis. A disciplina de Aspectos Ágeis de Programação aprofundou conceitos essenciais de qualidade de código, destacando a aplicação dos princípios do *Clean Code*, enfatizando a clareza, simplicidade e legibilidade como as principais características de um software limpo e sustentável. Já a disciplina de Banco de Dados teve papel essencial na organização e armazenamento das informações, por meio da modelagem de dados, foi possível entender como representar de forma clara as entidades e suas relações, facilitando a construção de sistemas bem estruturados. Além disso, o processo de normalização, mostrou-se indispensável para eliminar redundâncias e melhorar o desempenho das consultas, reforçando a importância de uma base de dados sólida para o desenvolvimento de soluções ágeis e eficientes.

Por fim, as disciplinas de Infraestrutura para Desenvolvimento e Implantação de Software (DevOps) e Testes Automatizados concluíram o ciclo de aprendizado do curso, reforçando a importância da automação e da melhoria contínua no desenvolvimento de software. A cultura DevOps foi apresentada como um elo entre desenvolvimento e operações, incentivando a integração de ferramentas, a entrega contínua e a colaboração entre as equipes. Sobre a disciplina de Testes Automatizados destacou-se a importância dos testes como parte essencial do ciclo de desenvolvimento. As atividades mostraram, na prática, como validar continuamente o

funcionamento do sistema contribui para entregas mais seguras e confiáveis. Essa abordagem reforçou a importância na garantia da qualidade e na sustentação de um processo ágil, em que cada entrega precisa manter estabilidade mesmo diante de mudanças frequentes.

2 DISCIPLINA: MADS – MÉTODOS ÁGEIS PARA DESENVOLVIMENTO DE SOFTWARE

O trabalho da disciplina Métodos Ágeis para Desenvolvimento de Software (MADS) teve como objetivo compreender, de forma prática e integrada, os fundamentos que sustentam o desenvolvimento ágil. Para isso, foi elaborado um mapa mental na ferramenta Xmind, reunindo os principais tópicos: Processo de Software, Modelos Tradicionais, Manifesto e Princípios Ágeis, *Lean Software Development*, Scrum, *Extreme Programming* (XP), Kanban e Entrega Contínua de Software.

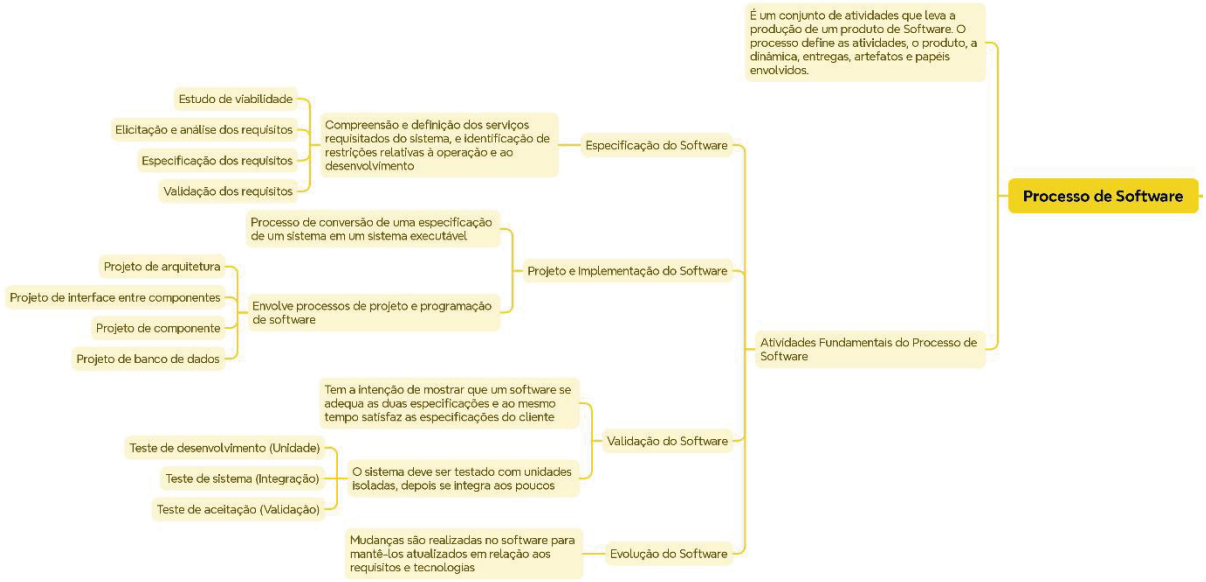
A atividade permitiu visualizar a transição dos modelos tradicionais, como o cascata, o incremental e o espiral, para abordagens mais dinâmicas e flexíveis. Essa comparação foi importante para entender o surgimento dos métodos ágeis como resposta à rigidez e à baixa adaptabilidade dos modelos clássicos. A partir desse estudo, foi possível compreender que os valores propostos pelo Manifesto Ágil (Beck et al., 2001), como colaboração, comunicação e resposta rápida a mudanças, são a base das práticas modernas de engenharia de software.

O mapa mental também apresentou os principais frameworks e práticas ágeis, como o Scrum, que organiza o trabalho em sprints e incentiva a auto-organização das equipes, e o XP, que valoriza técnicas como programação em pares, testes automatizados e refatoração contínua. Além disso, foram abordados os princípios do *Lean Software Development*, voltados à eliminação de desperdícios e entrega contínua de valor, e o uso do Kanban, que proporciona uma perspectiva visual e fluida do fluxo de trabalho.

Essa atividade foi fundamental para consolidar a base conceitual do curso, servindo como ponto de partida para as disciplinas seguintes. O entendimento dos princípios, práticas e frameworks ágeis formou a estrutura necessária para aplicar esses conceitos de maneira prática ao longo do desenvolvimento dos demais projetos.

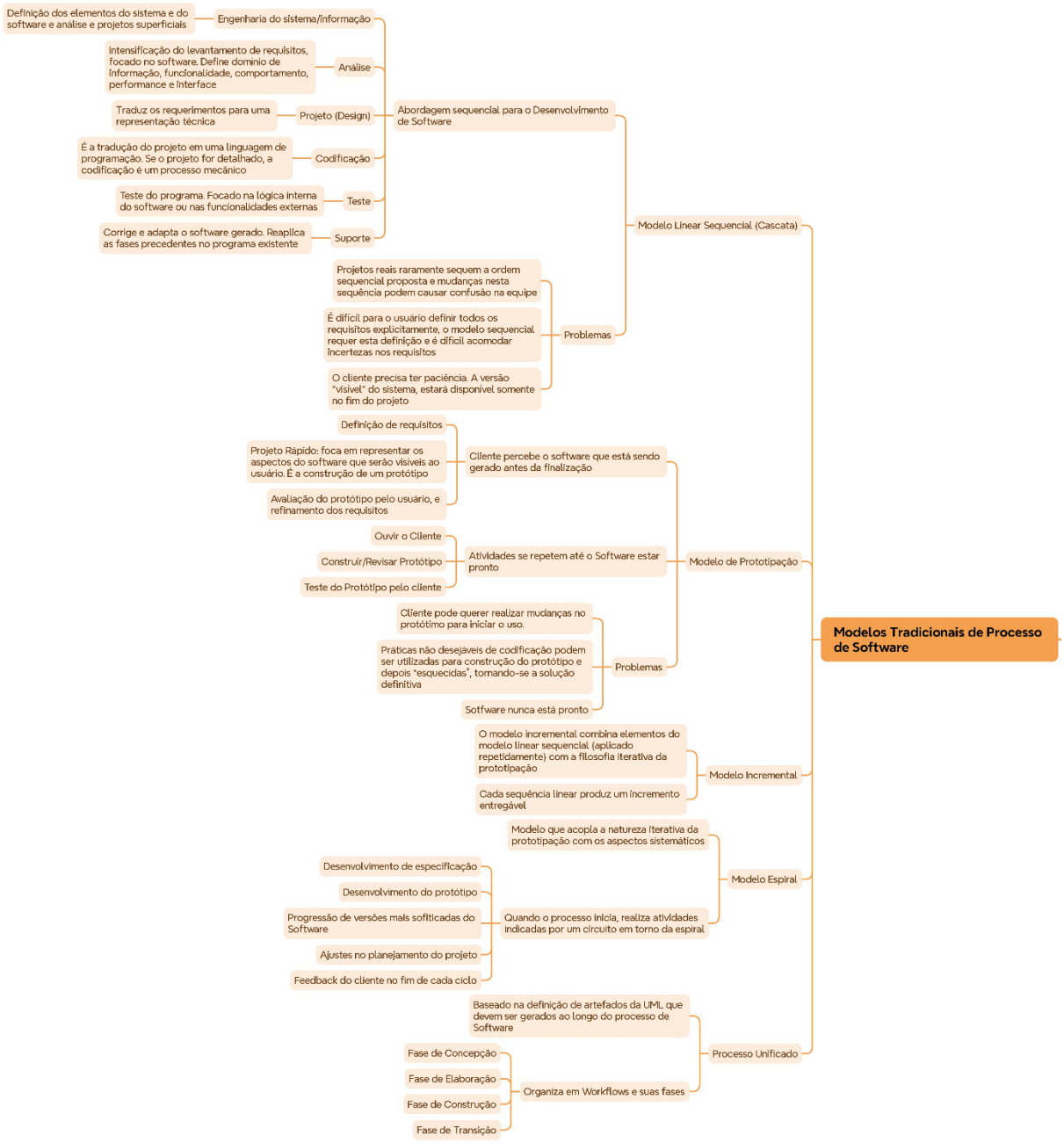
2.1 ARTEFATOS DO PROJETO

FIGURA 1 – MAPA MENTAL (PROCESSO DE SOFTWARE)



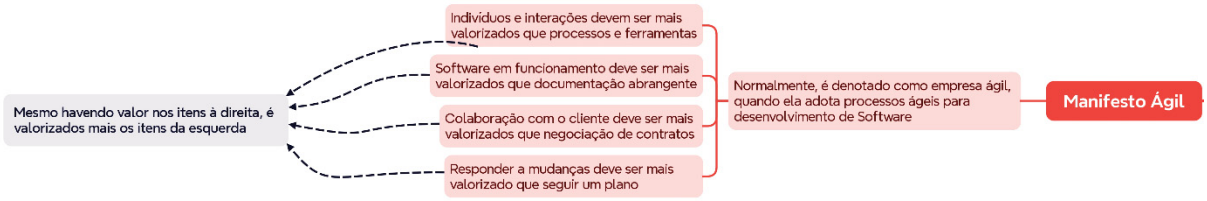
FONTE: O autor (2025)

FIGURA 2 – MAPA MENTAL (MODELOS TRADICIONAIS DE PROCESSO DE SOFTWARE)



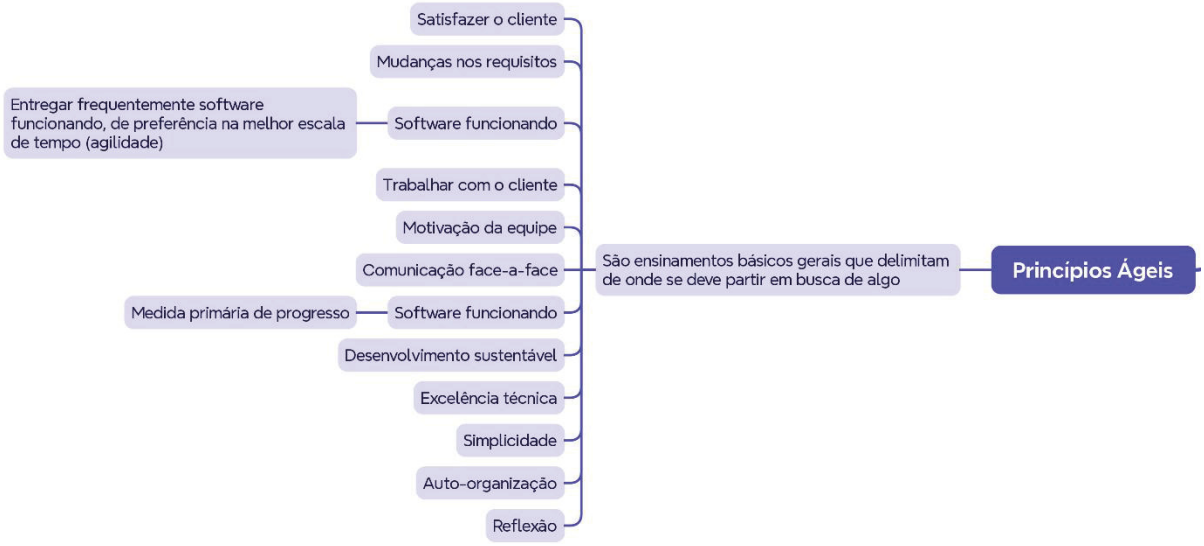
FONTE: O autor (2025)

FIGURA 3 – MAPA MENTAL (MANIFESTO ÁGIL)



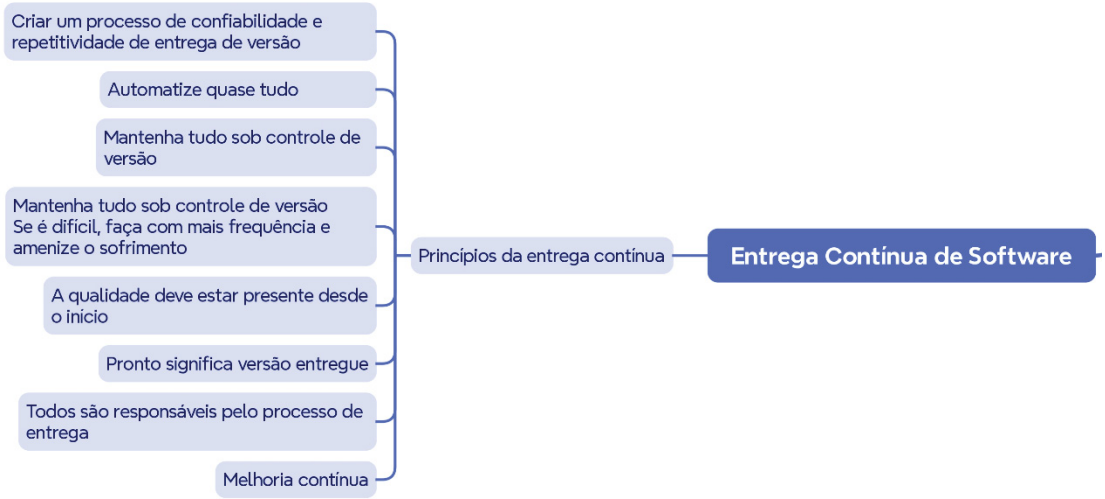
FONTE: O autor (2025)

FIGURA 4 – MAPA MENTAL (PRINCÍPIOS ÁGEIS)



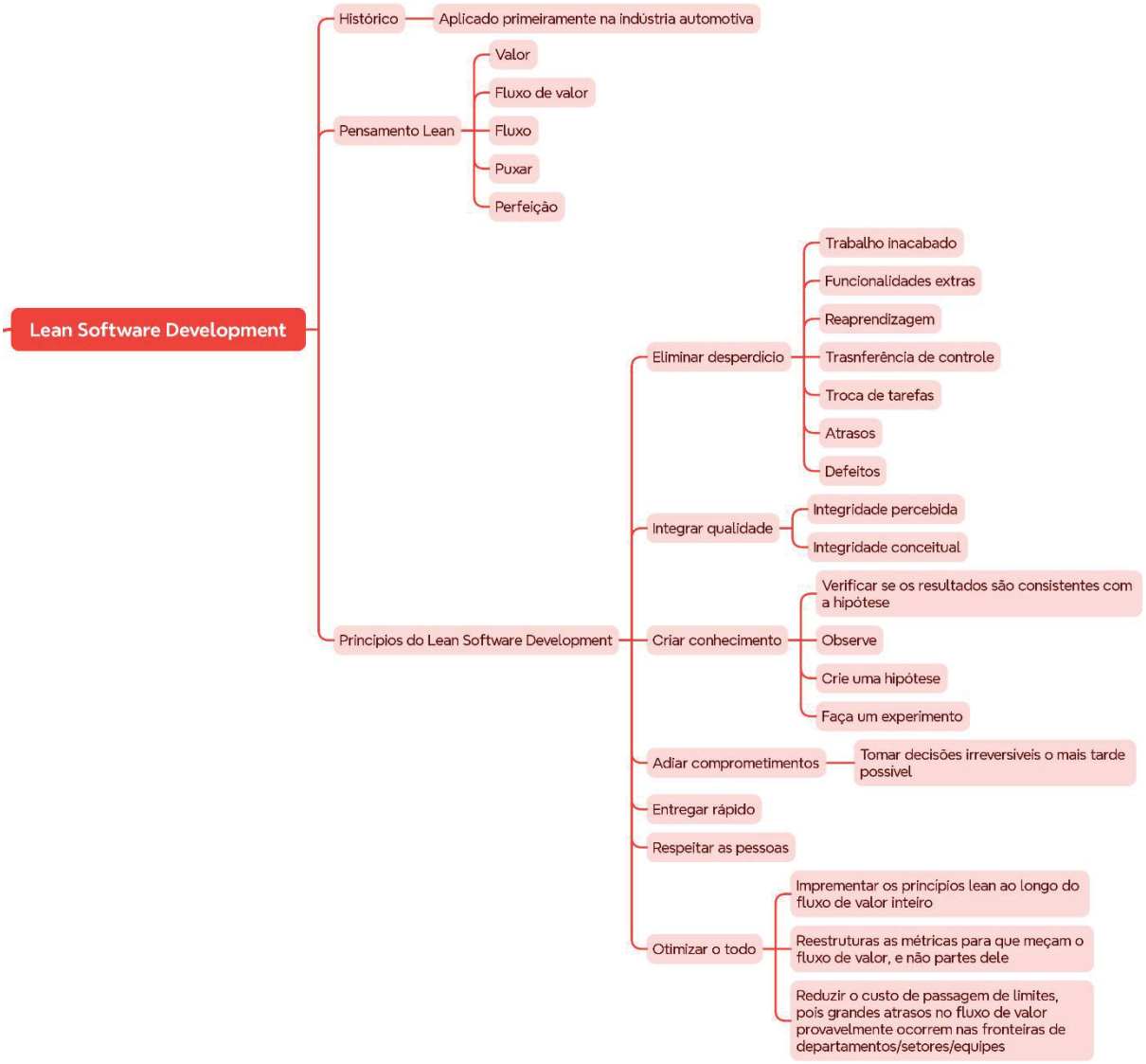
FONTE: O autor (2025)

FIGURA 5 – MAPA MENTAL (ENTREGA CONTÍNUA DE SOFTWARE)



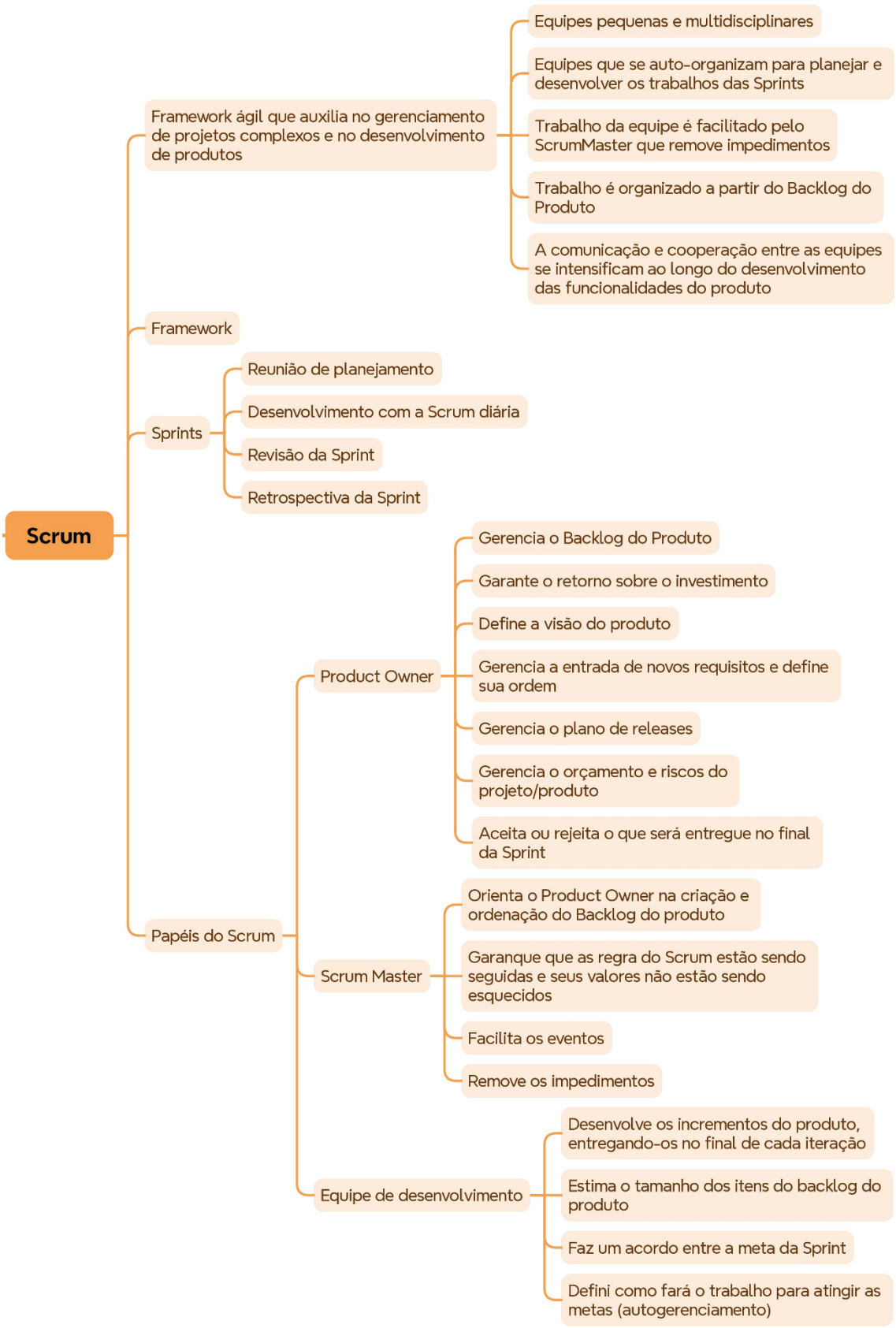
FONTE: O autor (2025)

FIGURA 6 – MAPA MENTAL (LEAN SOFTWARE DEVELOPMENT)



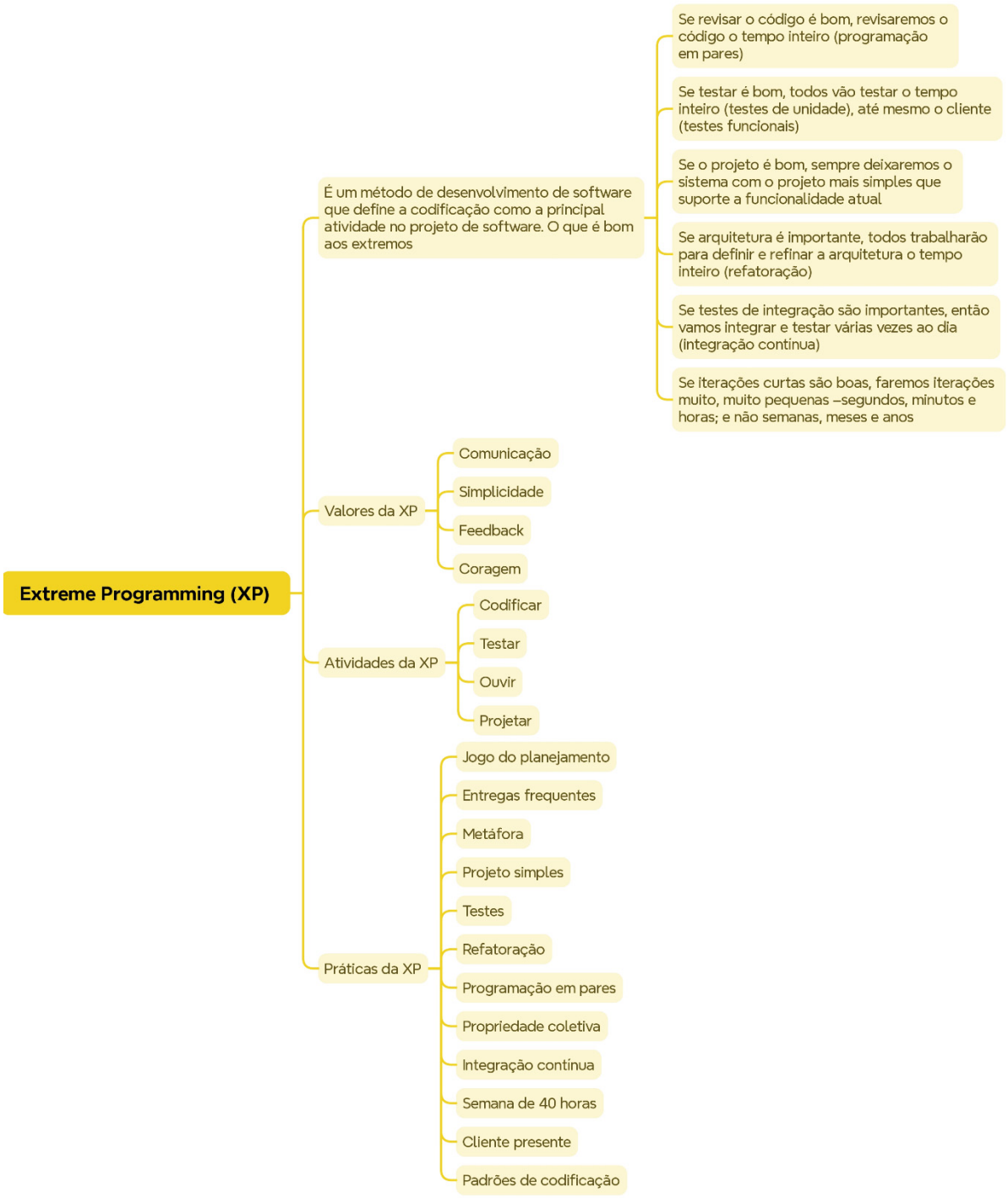
FONTE: O autor (2025)

FIGURA 7 – MAPA MENTAL (SCRUM)



FONTE: O autor (2025)

FIGURA 8 – MAPA MENTAL (EXTREME PROGRAMMING)



FONTE: O autor (2025)

FIGURA 9 – MAPA MENTAL (KANBAN)

FONTE: O autor (2025)

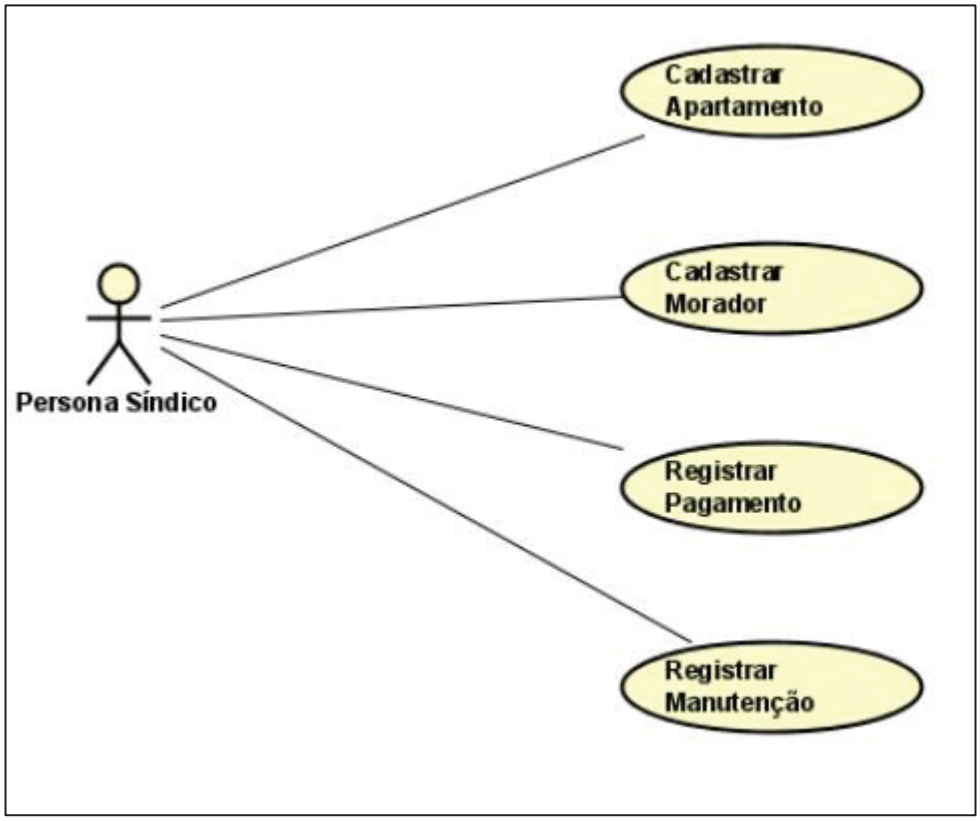
3 DISCIPLINA: MAG1 E MAG2 – MODELAGEM ÁGIL DE SOFTWARE 1 E 2

A atividade de MAG1 teve como foco a modelagem funcional de um Sistema de Gestão de Condomínio, a partir das solicitações do síndico. Foram produzidos o Diagrama de Caso de Uso Nível 1 (visão macro do escopo) e o Nível 2 (detalhamento por funcionalidades e atores), além do conjunto completo de histórias de usuário com critérios de aceitação e regras de negócio. As histórias cobriram, entre outros pontos, cadastro de apartamentos e moradores, registro de veículos e vagas, registro e validação de pagamentos e registro de manutenções do prédio. Esse pacote funcional estruturou o *backlog* do produto e deu visibilidade ao fluxo de valor para o usuário, conectando objetivos de negócio a incrementos entregáveis. Do ponto de vista ágil, a modelagem funcional deu suporte a planejamento iterativo, refinamento de *backlog* e alinhamento com as partes interessadas, reduzindo ambiguidade e aumentando previsibilidade sem atrasos no processo.

Em MAG2, o trabalho evoluiu para a modelagem estrutural do mesmo sistema, consolidando a base técnica para implementação. Foram elaborados o Diagrama de Classes (entidades como Unidade, Morador, Síndico, Boleto, com atributos e relacionamentos), o esboço das tabelas de banco de dados correspondentes e os Diagramas de Sequência para todas as histórias de usuário, como emissão de boletos e atualização de status de pagamento. A integração entre os casos de uso desenvolvidos em MAG1 e os diagramas UML elaborados em MAG2 permitiu alinhar de forma clara os requisitos funcionais à estrutura técnica do sistema, seguindo os princípios da modelagem orientada a objetos e boas práticas de documentação visual (Guedes, 2004). Essa etapa foi essencial para transformar as necessidades do usuário em componentes bem definidos, garantindo coerência entre o planejamento e a implementação.

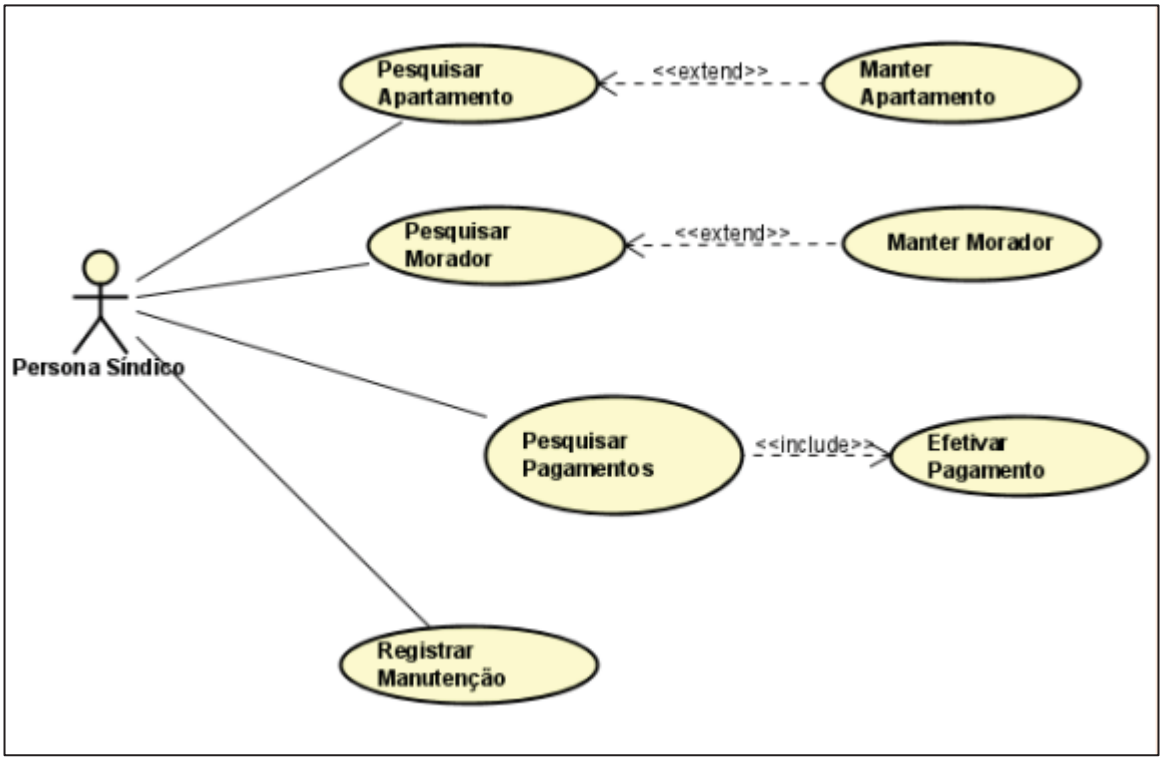
3.1 ARTEFATOS DO PROJETO

FIGURA 10 – DIAGRAMA DE CASO DE USO NÍVEL 1



FONTE: O autor (2025)

FIGURA 11 – DIAGRAMA DE CASO DE USO NÍVEL 2



FONTE: O autor (2025)

FIGURA 12 – HISTÓRIA DE USUÁRIO

HU001 – Pesquisar Apartamento

SENDO **o SÍNDICO**

QUERO **Pesquisar apartamento**

PARA **Verificar Apartamentos cadastrados**

DESENHO DA TELA

LISTA DE CRITÉRIOS DE ACEITAÇÃO

1) Deve pesquisar os apartamentos na tabela da tela

FONTE: O autor (2025)

FIGURA 13 – CRITÉRIO DE ACEITAÇÃO - DETALHAMENTO

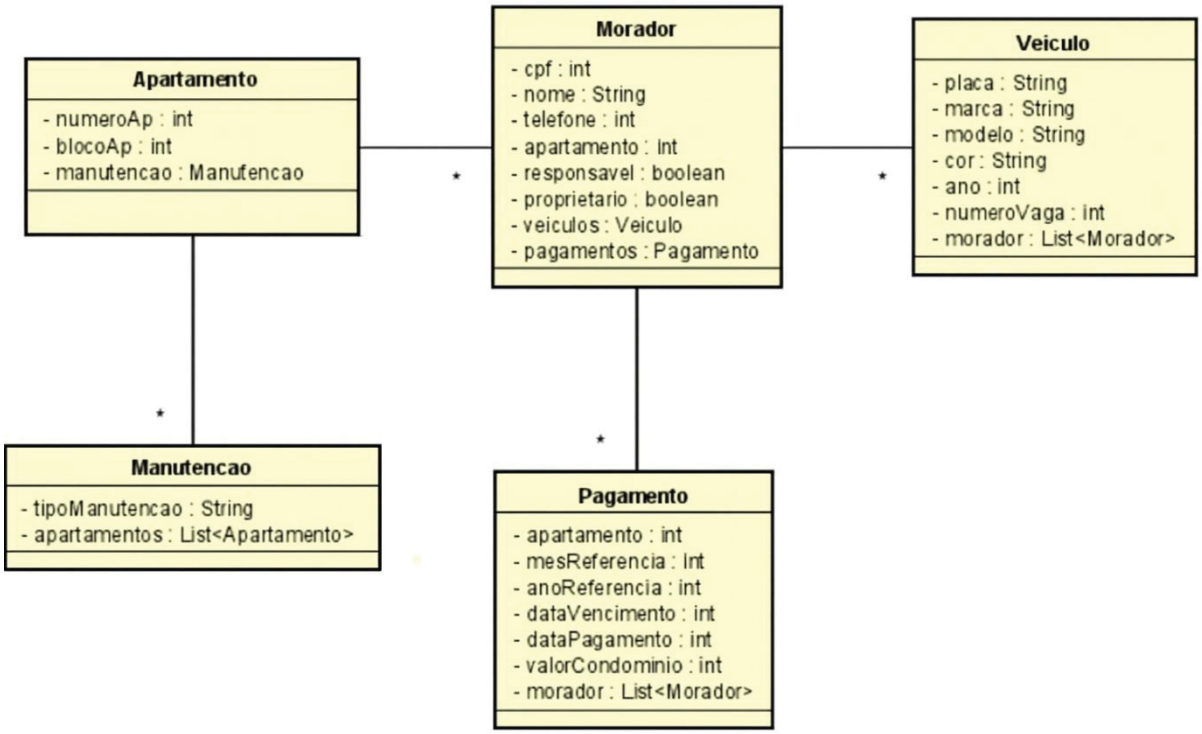
CRITÉRIOS DE ACEITAÇÃO - DETALHAMENTO

1) Deve pesquisar os apartamentos na tabela da tela

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

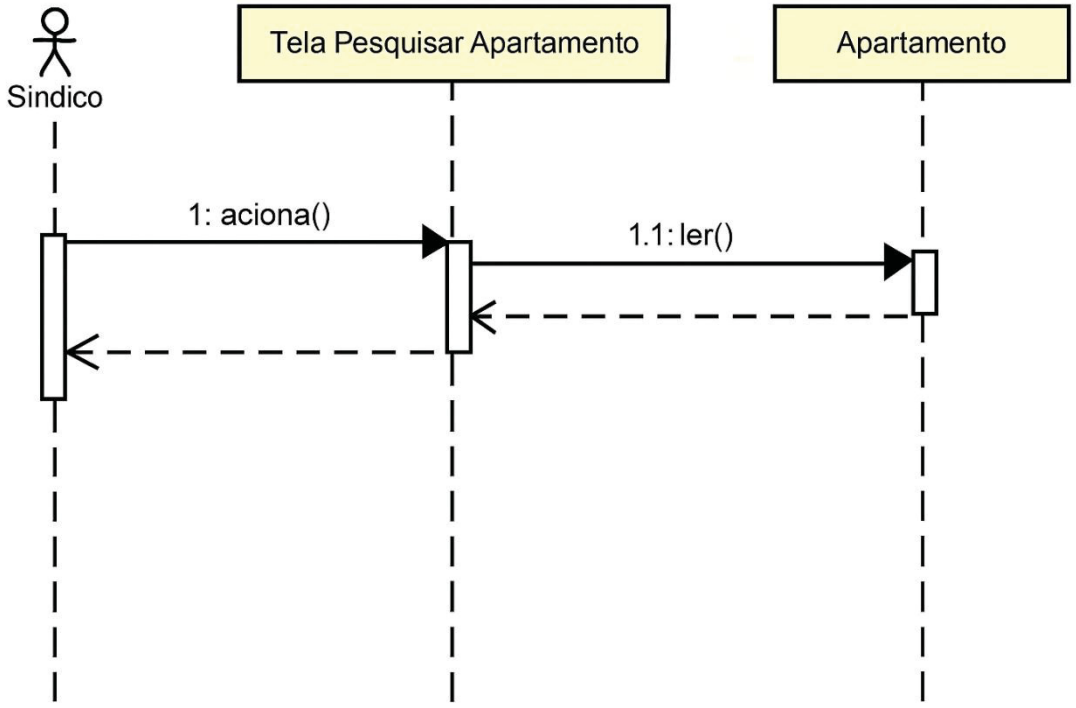
FONTE: O autor (2025)

FIGURA 14 – DIAGRAMA DE CLASSES



FONTE: O autor (2025)

FIGURA 15 – DIAGRAMA DE SEQUÊNCIA – HU001



FONTE: O autor (2025)

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

As disciplinas de Gerenciamento Ágil de Projetos de Software I e II (GAP1 e GAP2) foram essenciais para compreender, na prática, como o planejamento e a execução se complementam dentro do desenvolvimento ágil. Cada uma delas trouxe desafios diferentes, mas que se conectaram de forma direta e natural.

Em GAP1, a atividade consistiu na elaboração de um plano de release com base no sistema de gestão de condomínio desenvolvido nas disciplinas de Modelagem Ágil de Software (MAG1 e MAG2). O trabalho envolveu o cálculo da velocidade de desenvolvimento, a definição das sprints com suas respectivas datas e a organização das histórias de usuário no formato Sendo, Quero e Para. Esse processo possibilitou visualizar o ritmo de entrega, distribuir as tarefas de maneira realista e compreender como o planejamento ágil equilibra previsibilidade e adaptação. Além disso, reforçou a importância de priorizar entregas de valor e ajustar o escopo conforme o progresso do projeto, princípios fundamentais do Scrum (Cohn, 2011).

Em GAP2, o foco foi vivenciar a gestão ágil de forma mais dinâmica, por meio de uma simulação prática no KanbanBoardGame.com. Durante 35 dias de execução virtual, o desafio foi gerenciar um fluxo de tarefas, equilibrando capacidade, demanda e tempo de entrega. A experiência mostrou, na prática, como o Kanban ajuda a identificar gargalos, melhorar o fluxo de trabalho e tomar decisões baseadas em dados visuais. Ao final, a análise do Diagrama de Fluxo Cumulativo (CFD) permitiu observar a evolução do projeto e entender como pequenas melhorias contínuas impactam diretamente o desempenho e a produtividade.

Esses dois projetos se complementaram de forma muito clara: enquanto GAP1 proporcionou o aprendizado sobre como planejar, GAP2 mostrou como executar e ajustar o processo conforme o andamento do trabalho. Juntas, as disciplinas reforçaram a importância da organização, da adaptação e da transparência no desenvolvimento de software.

4.1 ARTEFATOS DO PROJETO

FIGURA 16 – PLANO DE RELEASE

Cálculo da Velocidade:

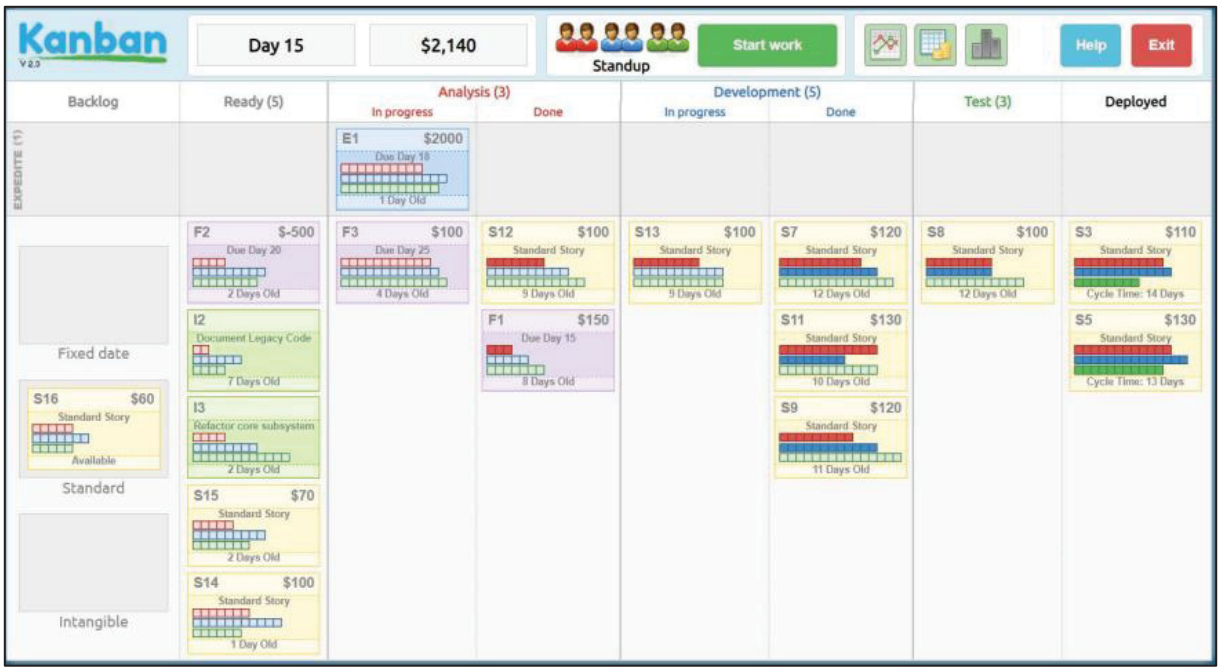
Horas disponíveis por dia:	**8h**	Tamanho da Sprint:	**2 Semanas**
Horas disponíveis por Sprint:	**80h (10 dias úteis)**	Velocidade:	**80/ 8 = 10 pontos**

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 Apartamento PARA Verificar Apartamentos cadastrados ESTIMATIVA (1)	<HU003 – Pesquisar Morador> SENDO o Síndico QUERO Pesquisar Morador PARA Cadastrar Morador ESTIMATIVA (1)	<HU005 – Pesquisar Pagamentos> SENDO o Síndico QUERO Pesquisar Pagamentos PARA Verificar registros dos Pagamentos ESTIMATIVA (1)	<HU007> SENDO o Síndico QUERO registrar uma manutenção PARA o Condomínio ESTIMATIVA (1)
<HU002 – Manter morador> SENDO o Síndico QUERO Pesquisar Apartamento PARA Cadastrar Apartamento ESTIMATIVA (2)	<HU004 – Excluir morador> SENDO o Síndico QUERO Pesquisar Morador PARA Excluir Morador ESTIMATIVA (2)	<HU006 – Efetivar Pagamento> SENDO o Síndico QUERO pesquisar pagamentos PARA incluir registro de pagamentos ESTIMATIVA (2)	

FONTE: O autor (2025)

FIGURA 17 – DIA 15 DA EXECUÇÃO

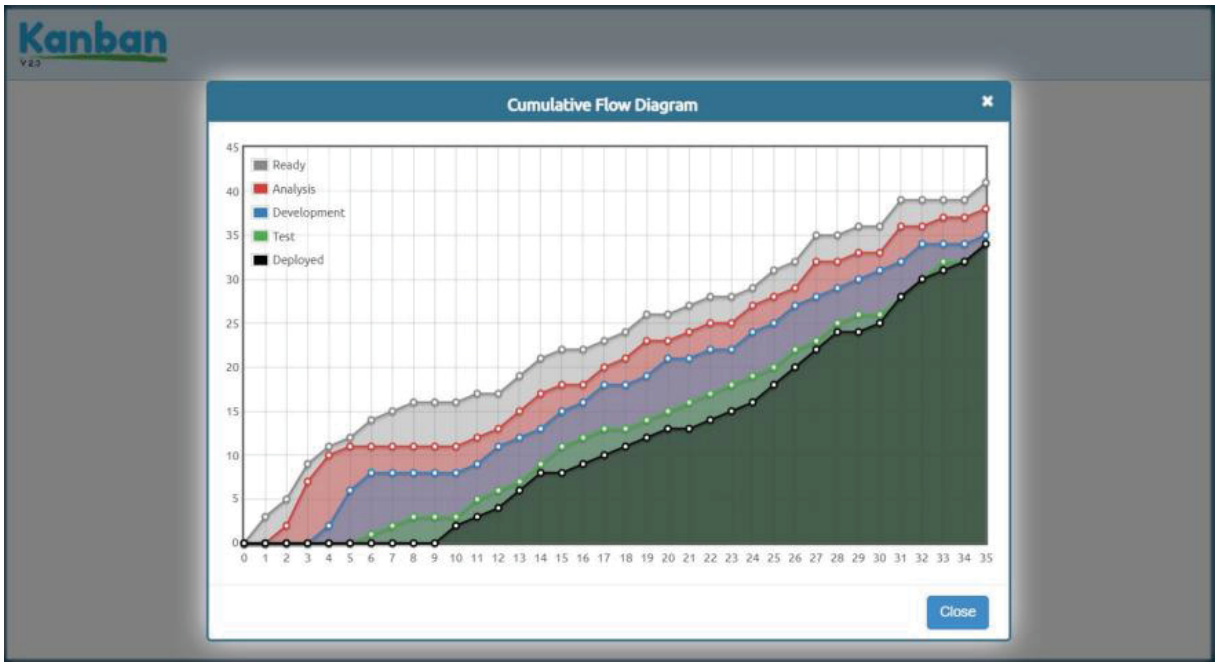


FONTE: O autor (2025)

FIGURA 18 – RESULTADO FINAL

FONTE: O autor (2025)

FIGURA 19 – CFD



FONTE: O autor (2025)

5 DISCIPLINA: INTRO – INTRODUÇÃO À PROGRAMAÇÃO

A disciplina Introdução à Programação teve como objetivo desenvolver o *back-end* de um sistema bancário simplificado em Java, responsável por gerenciar clientes, contas-correntes e investimentos. Para auxiliar o trabalho, foram fornecidos um diagrama de classes, um projeto no NetBeans com testes automatizados em JUnit e um script DDL em MySQL para criar as tabelas do banco de dados.

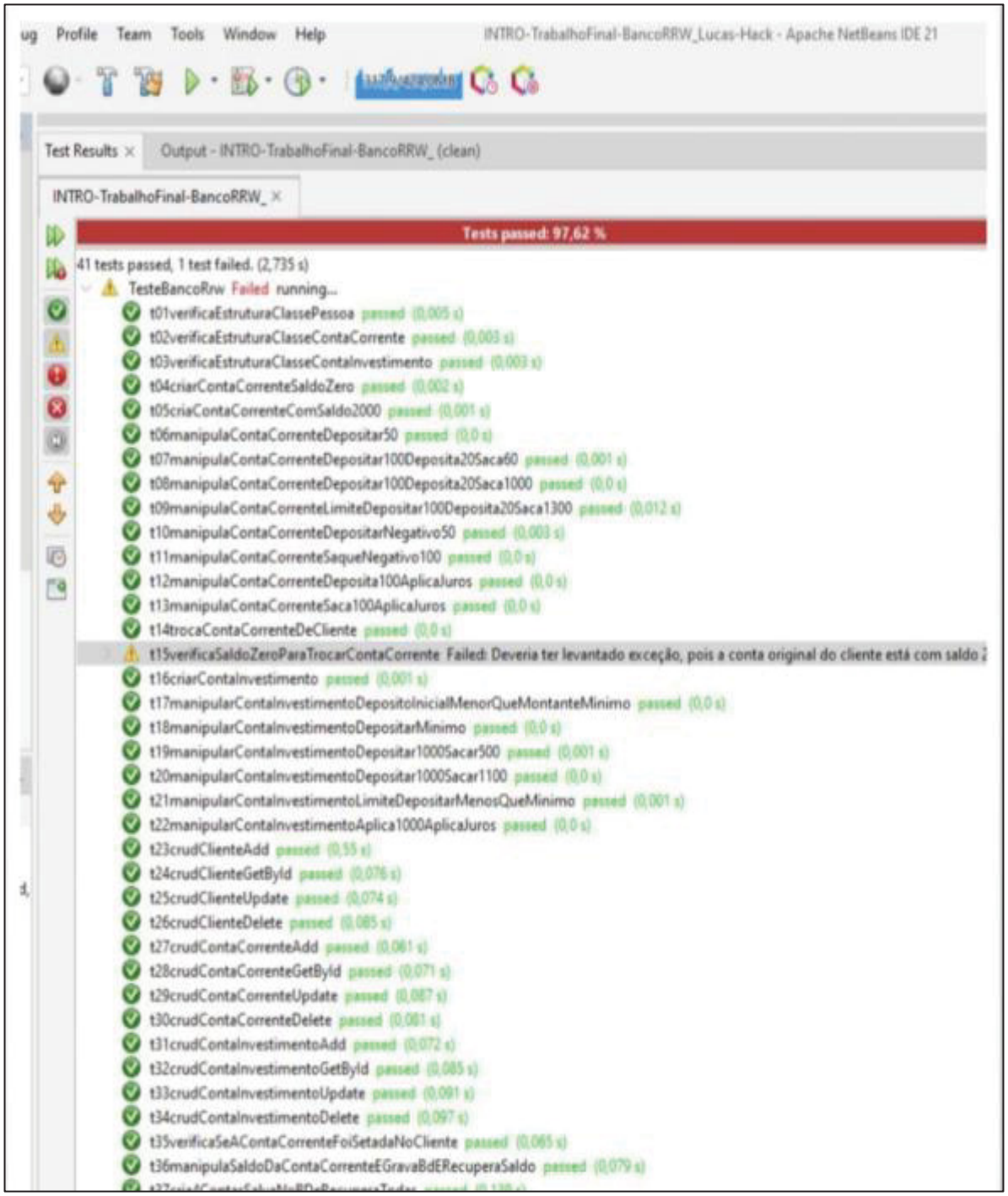
O desafio era fazer com que todos os 42 testes unitários funcionassem corretamente, garantindo o comportamento esperado do sistema. Essa atividade apresentou, na prática, o conceito de TDD (*Test-Driven Development*), uma técnica ágil que incentiva a criação de testes antes do código, ajudando a construir soluções mais seguras e organizadas (Beck, Kent, 2004).

Durante o desenvolvimento, foi possível entender melhor como os testes automatizados ajudam a detectar erros rapidamente e a melhorar a qualidade do código. Além disso, o projeto reforçou conceitos fundamentais da programação orientada a objetos, como herança, encapsulamento e polimorfismo, e mostrou a importância da integração entre o código e o banco de dados.

Essa disciplina foi fundamental para construir a base técnica do curso. Além de ensinar a lógica e a estrutura da programação, ela mostrou, na prática, como a combinação entre código limpo e testes automatizados garante sistemas mais estáveis e fáceis para implementar melhorias. O trabalho de Introdução à Programação serviu como um primeiro passo para compreender a importância da qualidade de código e do aprendizado contínuo, princípios que acompanham todo o processo de desenvolvimento ágil.

5.1 ARTEFATOS DO PROJETO

FIGURA 20 – CASOS DE TESTE



FONTE: O autor (2025)

6 DISCIPLINA: BD – BANCO DE DADOS

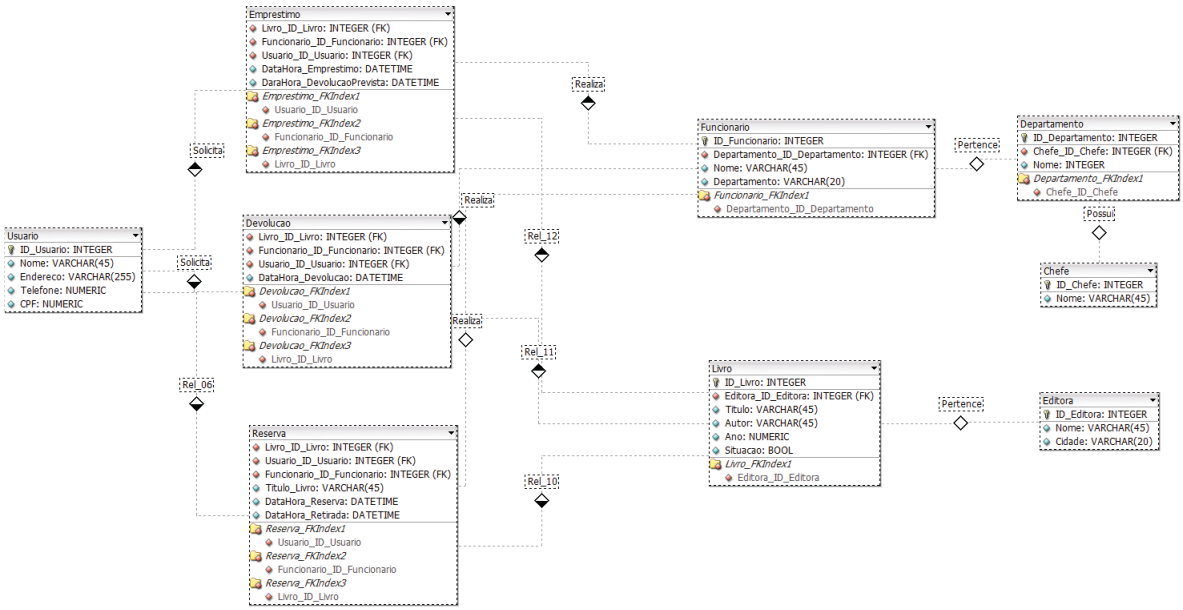
Na disciplina Banco de Dados, o trabalho foi dividido em duas etapas principais, voltadas à prática de modelagem e implementação de sistemas relacionais. Na primeira parte, o desafio foi criar o modelo de um sistema de controle de biblioteca, contemplando entidades e relacionamentos essenciais para o gerenciamento de obras, usuários, funcionários e empréstimos. Foram desenvolvidos o modelo conceitual (entidade-relacionamento) e o modelo lógico, representando tabelas, campos, chaves primárias e estrangeiras. Essa etapa permitiu compreender como organizar e relacionar informações de forma eficiente, garantindo consistência, integridade e clareza nos dados.

Na segunda parte do trabalho, o projeto se tornou mais autoral: foi desenvolvido um modelo para um sistema de controle de vacinação, escolhendo esse tema por sua relevância prática e pela variedade de relacionamentos que ele possibilita. O sistema incluiu entidades como pacientes, vacinas, enfermeiros, lotes de vacina e data de vacinação, apresentando exemplos de relacionamentos 1:1, 1:N e N:N. A partir dessa modelagem, foi gerado um *script* SQL completo, com criação das tabelas, chaves primárias e estrangeiras, restrições de integridade e inserção de dados de exemplo. Essa experiência reforçou o entendimento da normalização, da persistência de dados e da importância de um banco de dados bem estruturado como base para qualquer aplicação.

Além do aprendizado técnico, a disciplina mostrou como a modelagem de dados se conecta diretamente às etapas de desenvolvimento de software vistas em outras disciplinas, Introdução à programação e Desenvolvimento Web e Mobile. O domínio da estrutura de dados é essencial para que os sistemas desenvolvidos em ambientes ágeis sejam consistentes, escaláveis e confiáveis (Sommerville, 2011).

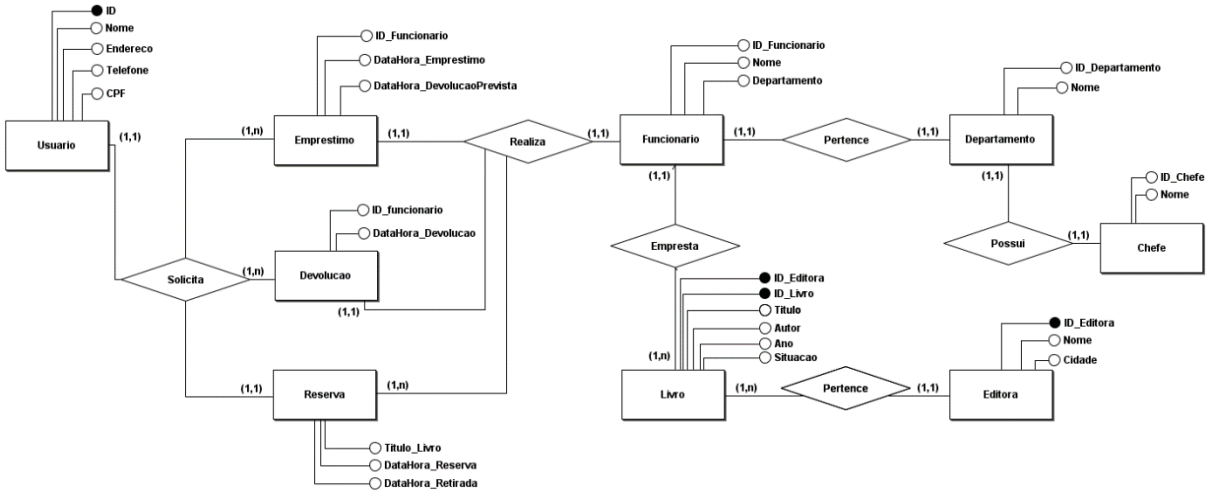
6.1 ARTEFATOS DO PROJETO

FIGURA 21 – DIAGRAMA DER BIBLIOTECA



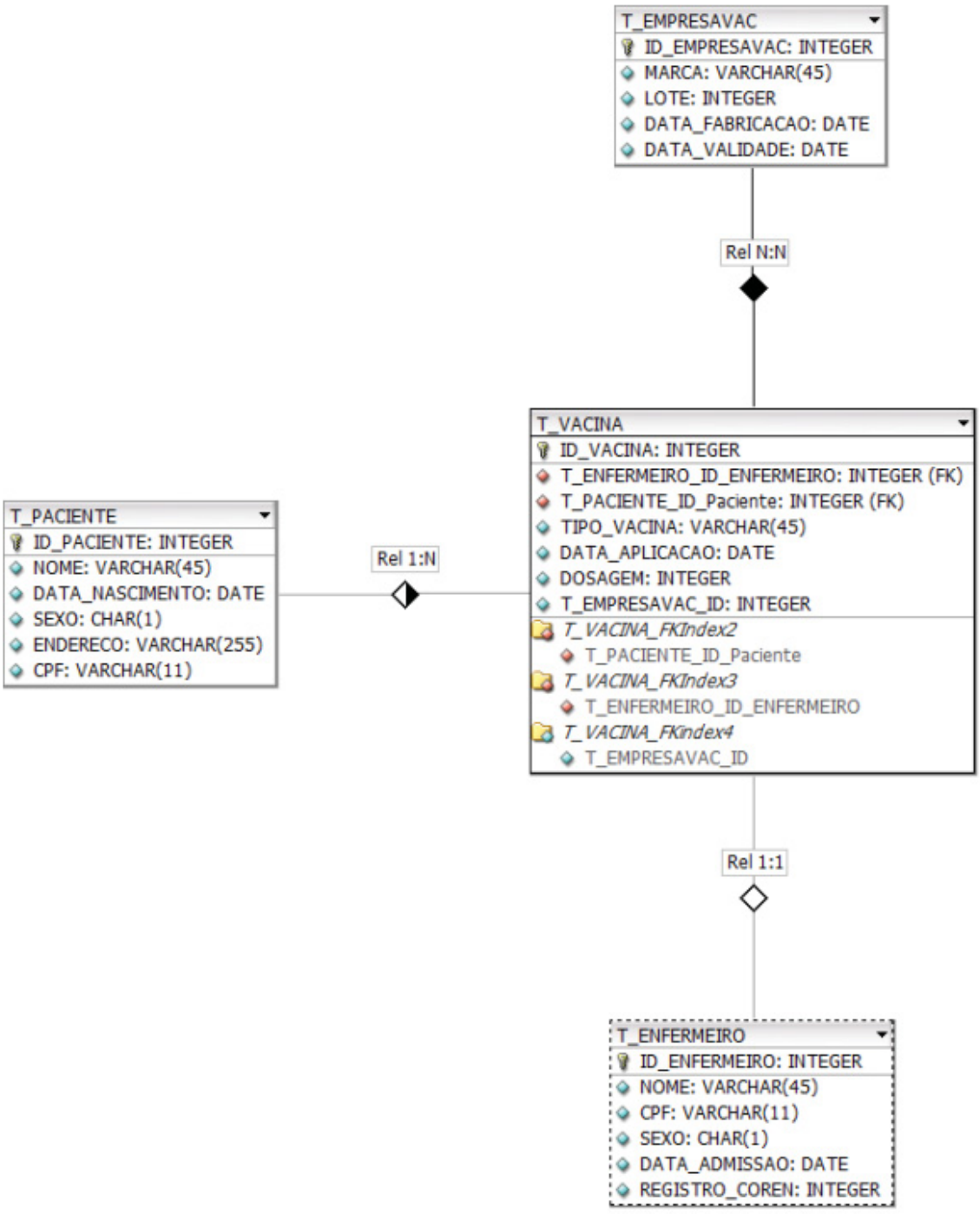
FONTE: O autor (2025)

FIGURA 22 – DIAGRAMA MER BIBLIOTECA



FONTE: O autor (2025)

FIGURA 23 – DIAGRAMA DER CONTROLE DE VACINAÇÃO



FONTE: O autor (2025)

7 DISCIPLINA: AAP – ASPECTOS ÁGEIS DE PROGRAMAÇÃO

Na disciplina Aspectos Ágeis de Programação, o projeto final teve como objetivo refatorar o algoritmo de ordenação *Bubble Sort*, aplicando os princípios de *Clean Code* para tornar o código mais legível e organizado. O desafio exigiu pelo menos três melhorias reais no código, o que envolveu renomear variáveis com nomes mais significativos, remover comentários desnecessários e dividir o código em métodos com responsabilidade única, seguindo boas práticas de clareza e simplicidade.

O conceito de *Clean Code*, amplamente difundido por Martin (2008), vai além da estética: ele representa uma forma de pensar o código como algo que deve ser fácil de entender, modificar e evoluir. Ao aplicar esses princípios, a atividade mostrou como pequenas mudanças podem trazer grande impacto na qualidade do software, tornando o trabalho do desenvolvedor mais produtivo e reduzindo a chance de erros. Além disso, um código limpo favorece a colaboração entre membros da equipe, já que todos conseguem compreender rapidamente o que foi implementado e como o sistema funciona.

Durante a disciplina, foi possível perceber como essas boas práticas se conectam com os valores do desenvolvimento ágil. Um código bem estruturado permite responder a mudanças com mais rapidez e segurança, facilita a integração contínua e reduz o retrabalho. A experiência prática com o Bubble Sort reforçou que a clareza do código é tão importante quanto a funcionalidade, e que a qualidade técnica é um pilar essencial para a agilidade no desenvolvimento.

Os aprendizados dessa disciplina tiveram reflexo direto nas demais, especialmente em Introdução à Programação, Desenvolvimento Web e Testes Automatizados, onde a manutenção e evolução de código são constantes. Com isso, Aspectos Ágeis de Programação se mostrou uma etapa importante na formação de uma mentalidade voltada à melhoria contínua.

7.1 ARTEFATOS DO PROJETO

FIGURA 24 – MÉTODO BUBBLESORT

```java
// Metódo para ordenação Bubble Sort
static void bubbleSort(int[] array, int tamanhoArray)
{
    int i, j, temp;
    boolean trocado;

    for (i = 0; i < tamanhoArray - 1; i++) {
        trocado = false;
        for (j = 0; j < tamanhoArray - i - 1; j++) {
            if (array[j] > array[j + 1]) {
                temp = array[j]; // Variável temporária para troca de ordenação
                array[j] = array[j + 1];
                array[j + 1] = temp;
                trocado = true;
            }
        }

        // Caso os dois valores sejam iguais a condicional ira finalizar
        if (trocado == false)
            break;
    }
}
```

FONTE: O autor (2025)

FIGURA 25 – MÉTODO ARRAY ORDENADO

```java
static void printArrayOrdenado(int array[], int tamanhoArray)
{
    int indice;
    for (indice = 0; indice < tamanhoArray; indice++)
        System.out.print(array[indice] + " ");
}
```

FONTE: O autor (2025)

FIGURA 26 – MÉTODO MAIN

```java
public static void main(String args[])
{
    int array[] = { 64, 34, 25, 12, 22, 11, 90 };
    int tamanhoArray = array.length;

    bubbleSort(array, tamanhoArray);
    System.out.println("Array ordenado: ");
    printArrayOrdenado(array, tamanhoArray);
}
```

FONTE: O autor (2025)

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

As disciplinas Desenvolvimento Web I e II mostraram como um sistema web moderno é construído do início ao fim, desde a interface que o usuário vê até a comunicação com o servidor. Mesmo sem a necessidade da realização do projeto prático, o conteúdo foi essencial para entender os principais conceitos e tecnologias que dão suporte ao desenvolvimento ágil de software.

Em Desenvolvimento Web I, o foco esteve nos fundamentos do *front-end*. Foram explorados conceitos de TypeScript, HTML e Bootstrap, além de temas como orientação a objetos, formulários e rotas. Essas aulas ajudaram a perceber a importância da organização do código, e da criação de componentes reutilizáveis.

Já em Desenvolvimento Web II, o aprendizado avançou para o *back-end* e para a integração entre as camadas do sistema. Foram estudadas APIs REST, Spring Boot e Spring Data JPA, entendendo como o front-end se comunica com o servidor e como os dados são armazenados e recuperados. Um dos pontos de maior destaque foi o desenvolvimento de operações CRUD, que representam a base de qualquer sistema web moderno. Essa parte do conteúdo ajudou a consolidar a compreensão de como estruturar e manipular informações de forma eficiente, integrando o front-end ao banco de dados de maneira prática e segura.

As disciplinas mostraram, na prática, como cada parte de um sistema depende da outra, e como a conexão entre design, código e dados resulta em aplicações mais eficientes. O conteúdo aprendido em Desenvolvimento Web I e II se conecta naturalmente com outras áreas do curso, como Banco de Dados.

9 DISCIPLINA: UX – UX NO DESENVOLVIMENTO ÁGIL DE SOFTWARE

Na disciplina de UX no Desenvolvimento Ágil de Software, a atividade final consistiu no desenvolvimento de telas para um aplicativo móvel de nutrição. O objetivo principal foi criar uma interface funcional e intuitiva, que ajudasse o usuário a acompanhar sua alimentação diária, calcular calorias e descobrir novas receitas saudáveis. A escolha do tema surgiu da necessidade de tornar o controle nutricional mais prático e acessível no dia a dia.

Durante o processo, foi necessário justificar a relevância do aplicativo e mostrar como ele poderia auxiliar as pessoas a manter uma dieta equilibrada. As cores, layouts e elementos gráficos foram planejados com cuidado para garantir uma navegação clara e agradável. A paleta de cores em tons de verde e branco foi escolhida por transmitir leveza, saúde e bem-estar, enquanto a tipografia simples e moderna facilitou a leitura das informações nutricionais e das receitas.

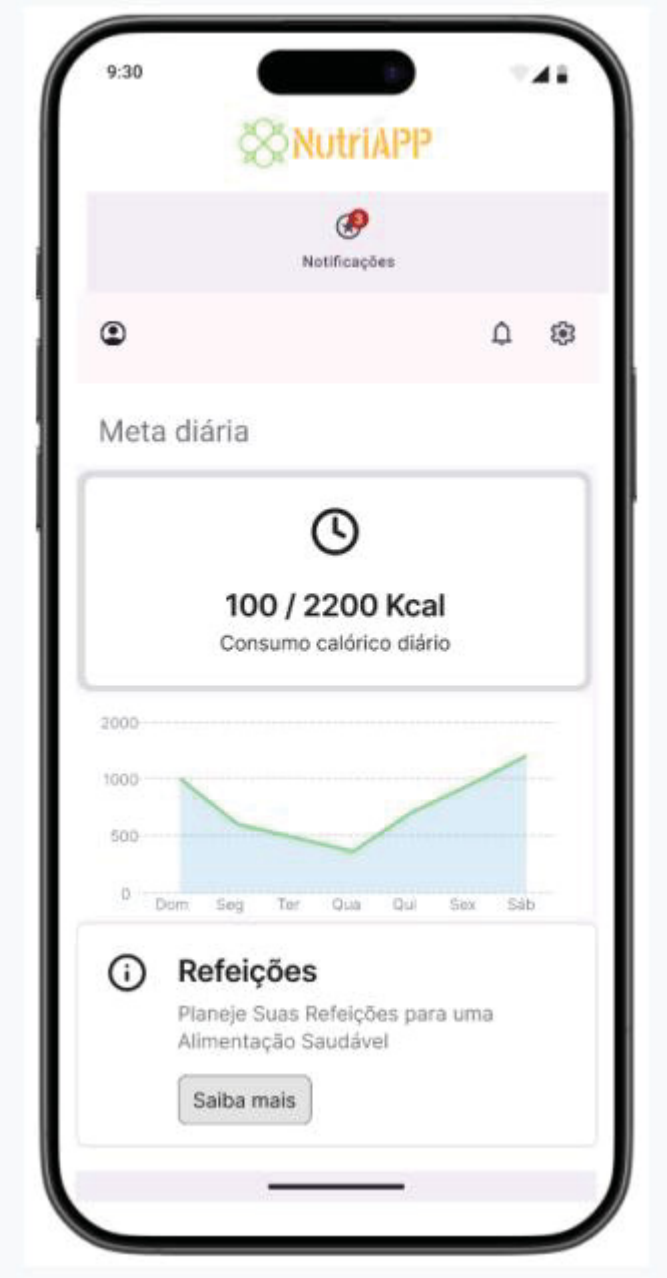
As telas foram desenvolvidas no Figma, aplicando os conceitos aprendidos sobre grids e layouts, tipografia, cores e ícones, e princípios de design visual. A ferramenta permitiu criar protótipos rápidos, o que possibilitou ajustar o design conforme se espera de um aplicativo dessa categoria.

Também foram considerados os princípios de acessibilidade, garantindo que o aplicativo fosse inclusivo e fácil de usar por qualquer pessoa (Grant, 2019). Além disso, buscou-se explorar o papel das emoções no design, criando uma experiência que transmitisse motivação e satisfação ao acompanhar os resultados da alimentação saudável.

A disciplina de UX foi essencial para compreender como a experiência do usuário influencia diretamente o sucesso de um produto digital.

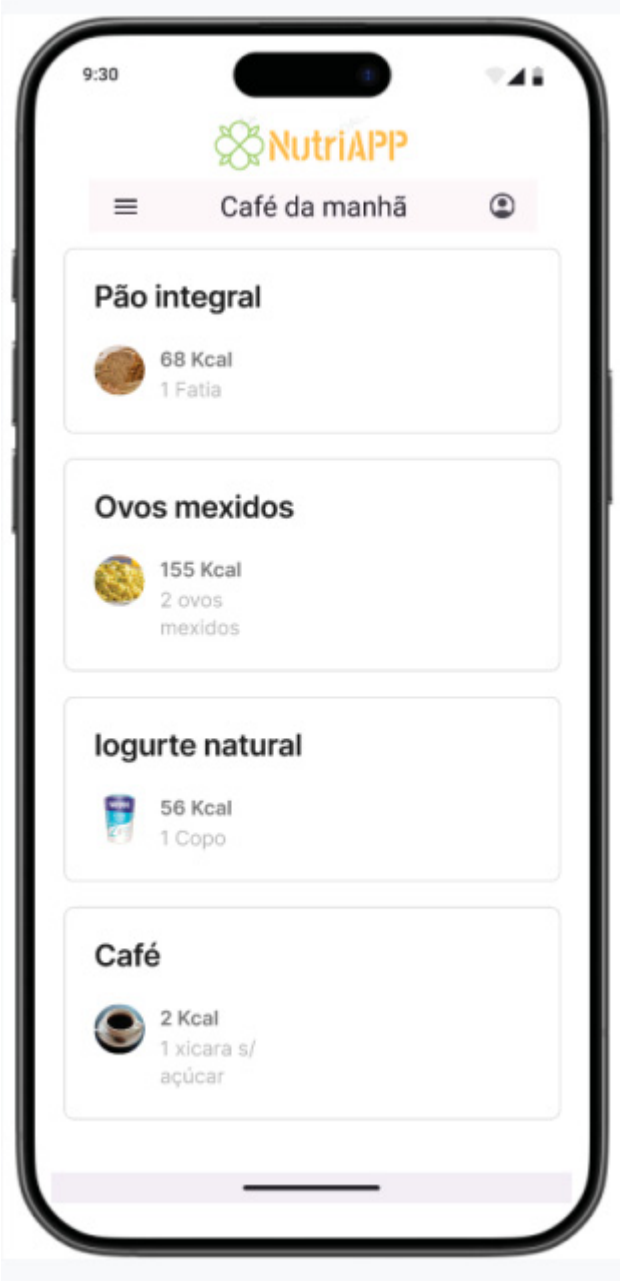
9.1 ARTEFATOS DO PROJETO

FIGURA 27 – TELA DE META E GRÁFICO DIÁRIO



FONTE: O autor (2025)

FIGURA 28 – TELA DE REFEIÇÕES E CALORIAS



FONTE: O autor (2025)

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

Em Desenvolvimento Mobile I, o aprendizado começou pelos fundamentos da plataforma Android, passando pela criação de interfaces e pela interação entre telas. Foram estudados conceitos como *Activities, Intents, Layouts* e os principais componentes de interface (GUI), com exemplos práticos e simples, como Frase do Dia, CambioApp e TipCalc. Esses exercícios ajudaram a entender como um bom design e uma navegação bem planejada tornam a experiência do usuário mais fluida e agradável. Também foi possível conhecer diferentes formas de desenvolver aplicativos, desde o desenvolvimento nativo até abordagens híbridas, o que ampliou a visão sobre as possibilidades.

Em Desenvolvimento Mobile II, o foco passou a ser a integração entre aplicativos e serviços externos, com destaque para o consumo de APIs e a comunicação entre diferentes telas (MultiActivities). A disciplina mostrou como estruturar aplicativos que oferecem respostas rápidas, interações dinâmicas e feedback constante. Também foi reforçada a importância de manter o código organizado e modular, o que facilita ajustes e evoluções sem comprometer o sistema.

Mesmo sem a obrigatoriedade de entrega dos projetos finais dessas disciplinas, o conteúdo possibilitou compreender o papel essencial dos aplicativos móveis tanto na rotina das pessoas quanto nas estratégias das empresas. O aprendizado evidenciou que desenvolver para o ambiente mobile vai muito além do domínio técnico de programação: envolve preocupação com a experiência do usuário, atenção às regras de negócio e capacidade de transformar as necessidades em soluções eficazes e duráveis.

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

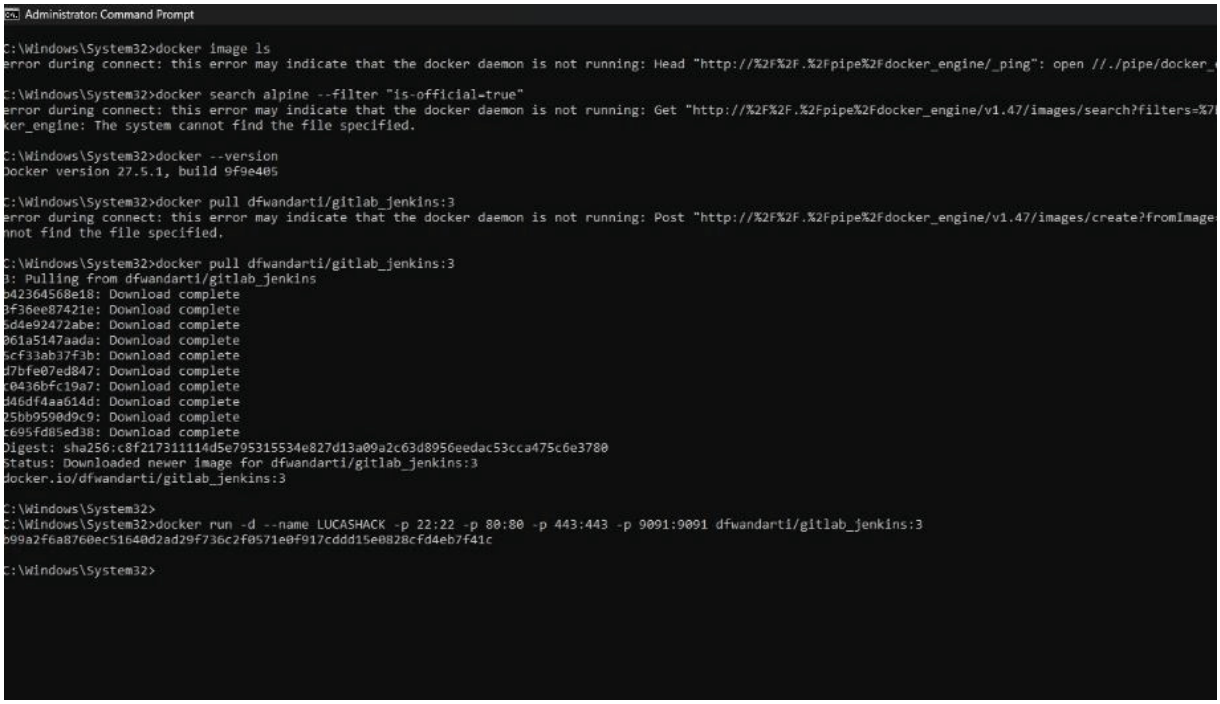
A disciplina de Infraestrutura para Desenvolvimento e Implantação de Software (DevOps) apresentou, de forma prática, os conceitos que unem desenvolvimento e operações em um ambiente integrado e automatizado. Teve como foco o uso de Docker e Git, simulando um cenário real de versionamento e implantação de software.

O trabalho proposto consistiu em criar um container Docker a partir de uma imagem pré-definida, configurando portas e acessos, e em seguida navegar dentro desse ambiente para localizar credenciais que permitissem o acesso ao GitLab. Depois, foi necessário executar comandos como *commit* e *push*, garantindo o versionamento correto do código. Um ponto interessante da atividade foi a utilização do Docker tanto pelo terminal quanto por interface visual, o que possibilitou uma compreensão mais completa da ferramenta e demonstrou sua versatilidade no dia a dia do desenvolvedor.

A experiência apresentou como a automação e a integração contínua são fundamentais para manter a agilidade no desenvolvimento de software. O DevOps, ao aproximar as áreas de desenvolvimento e infraestrutura, promove a importância de construir ambientes sustentáveis e automatizados, que reduzem falhas e tornam o processo de entrega mais fluido e confiável.

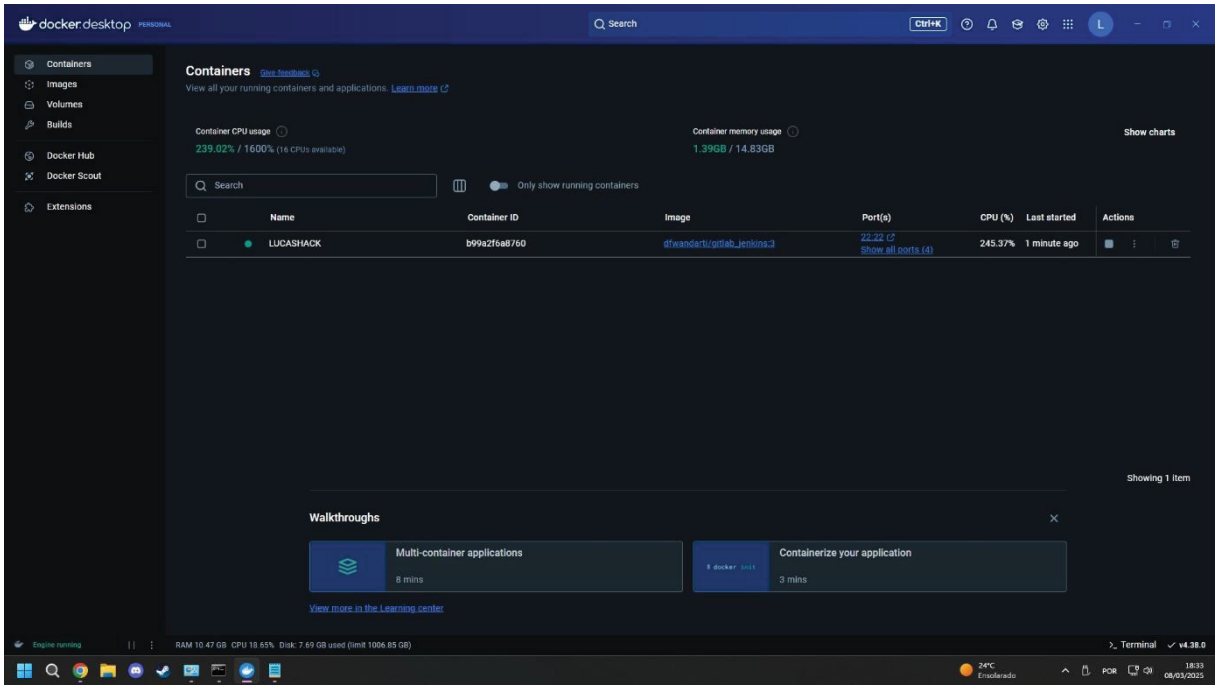
11.1 ARTEFATOS DO PROJETO

FIGURA 29 – TELA DO PROMPT



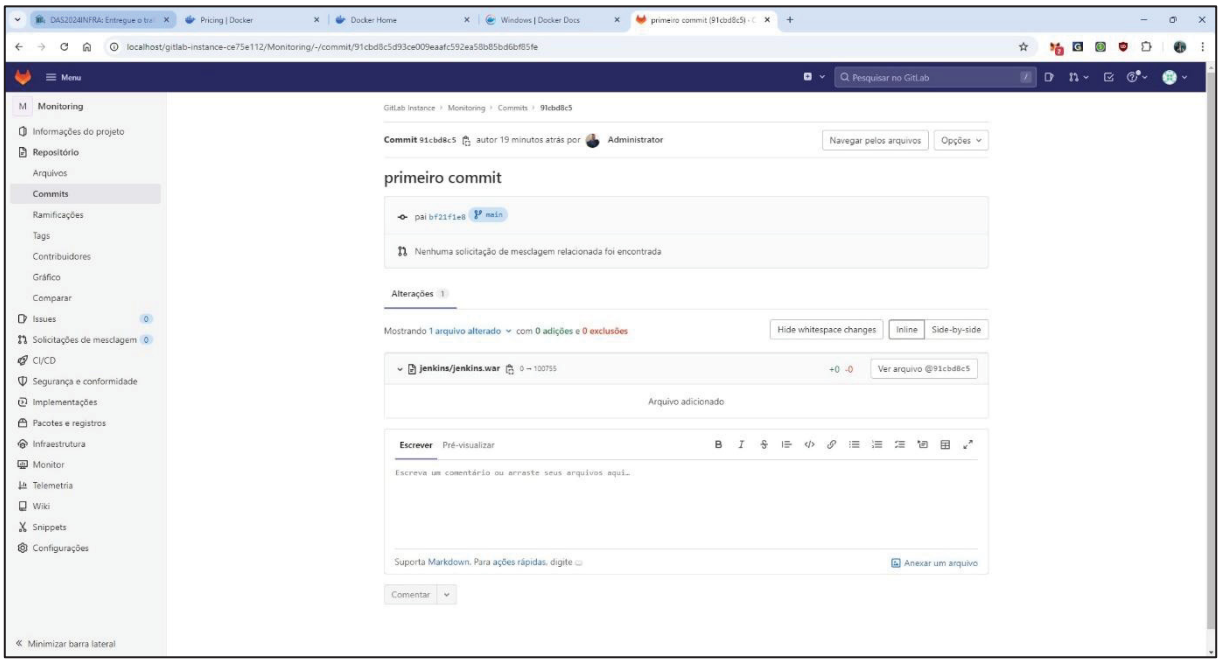
FONTE: O autor (2025)

FIGURA 30 – TELA DO CONTAINER



FONTE: O autor (2025)

FIGURA 31 – TELA DE COMMIT



FONTE: O autor (2025)

12 DISCIPLINA: TEST – TESTES AUTOMATIZADOS

Na disciplina Testes Automatizados, o objetivo foi compreender, de forma prática, o papel dos testes no desenvolvimento de software e sua importância dentro do contexto ágil. O trabalho final consistiu em criar um script em Java, utilizando o Playwright, para automatizar uma tarefa simples no site pt.anotepad.com: abrir a página, criar uma nova nota com o título “Entrega trabalho TEST DAS 2024” e registrar o nome e a matrícula do aluno.

A atividade demonstrou como os testes automatizados podem validar o comportamento de um sistema de forma precisa e repetitiva, garantindo que as funcionalidades continuem operando corretamente após alterações no código. Além disso, evidenciou a importância de integrar os testes ao processo de desenvolvimento, permitindo detectar erros com antecedência e manter a estabilidade do sistema. No contexto ágil, essa prática é essencial para sustentar entregas contínuas e de qualidade, fortalecendo a confiança da equipe em cada nova versão do produto.

12.1 ARTEFATOS DO PROJETO

FIGURA 32 – TELA DO CÓDIGO

Main.java × pom.xml (TrabFinal)

```
1  package org.example;
2
3  import com.microsoft.playwright.*;
4  import com.microsoft.playwright.options.AriaRole;
5
6  public class Main {
7      public static void main(String[] args) {
8
9          /*  "Entrega trabalho TEST DAS 2024"
10             "Lucas Vinicius Hack"
11             "40001016398E1"                */
12
13         try (Playwright playwright = Playwright.create()) {
14             Browser browser = playwright.chromium().launch(new BrowserType.LaunchOptions().setHeadless(false));
15             Page page = browser.newPage();
16
17             //Locator inputTitulo = page.getByLabel("edit_title);
18
19             page.navigate( url: "https://pt.anotepad.com/");
20
21             page.getByPlaceholder( text: "Título da Nota").click();
22             page.getByPlaceholder( text: "Título da Nota").fill( value: "Entrega trabalho Test DAS 2024");
23
24             page.getByPlaceholder( text: "Conteúdo da Nota").click();
25             page.getByPlaceholder( text: "Conteúdo da Nota").fill( value: "Lucas Vinicius Hack / 40001016398E1");
26
27             //browser.close();
28         }
29     }
30 }
```

FONTE: O autor (2025)

13 CONCLUSÃO

O curso de Especialização em Desenvolvimento Ágil de Software da Universidade Federal do Paraná proporcionou uma formação completa, unindo teoria e prática para desenvolver uma visão ampla sobre o ciclo de vida de software. Ao longo das disciplinas, foi possível compreender como cada etapa, desde a modelagem e programação até a entrega contínua e os testes automatizados, contribui para o desenvolvimento de soluções mais eficientes, colaborativas e centradas nas necessidades do usuário.

Os projetos desenvolvidos mostraram, na prática, o impacto das metodologias ágeis no cotidiano do desenvolvedor. Disciplinas como Modelagem Ágil de Software, Gerenciamento de Projetos, DevOps e Testes Automatizados evidenciaram que a agilidade não está apenas nos processos, mas também na mentalidade de melhoria contínua e colaboração entre equipes. A integração entre áreas técnicas e de design, por meio de UX e Desenvolvimento Web e Mobile, reforçou a importância da experiência do usuário como fator determinante para o sucesso de qualquer produto digital.

A jornada acadêmica permitiu compreender que o verdadeiro valor das metodologias de desenvolvimento ágil está na capacidade de adaptação às mudanças e na entrega constante de valor. No entanto, a aplicação prática dessas metodologias ainda enfrenta desafios. Entre eles, a resistência cultural à mudança, a dificuldade de equilibrar prazos com qualidade, a necessidade de maturidade das equipes multidisciplinares e a implantação consistente da automação em todo o ciclo de desenvolvimento.

Conclui-se que a especialização foi fundamental para consolidar uma visão crítica e moderna sobre o desenvolvimento de software, preparando o profissional para atuar em ambientes dinâmicos, tanto na parte técnica, quanto na gestão. Mais do que dominar ferramentas e frameworks, o aprendizado obtido reforçou que o sucesso no desenvolvimento ágil depende de pessoas comprometidas, comunicação clara e da busca constante por excelência organizacional.

REFERÊNCIAS

BECK, K; et al. ***Manifesto para o Desenvolvimento Ágil de Software***. 2001.

BECK, K. ***Programação extrema explicada: acolha as mudanças***. Porto Alegre: Bookman, 2004.

COHN, M. ***Desenvolvimento de software com Scrum: aplicando métodos ágeis com sucesso***. Porto Alegre: Bookman, 2011.

GUEDES, G. ***UML 2: uma abordagem prática***. São Paulo: Novatec, 2004.

GOETHELF, J. ***Lean UX: Designing Great Products with Agile Teams***. O'Reilly Media, 2021.

GRANT, W. ***UX Design: Guia Definitivo com as Melhores Práticas de UX*****. Novatec**, 2019

HUMBLE, J; FARLEY, D. ***Entrega Contínua: como entregar software de forma rápida e confiável***. Porto Alegre: Bookman, 2014.

POPPENDIECK, M; POPPENDIECK, T. ***Lean Software Development: An Agile Toolkit***. Addison Wesley, 2003.

SOMMERVILLE, I. ***Engenharia de Software***. 9ª ed. São Paulo: Pearson Prentice Hall, 2011.