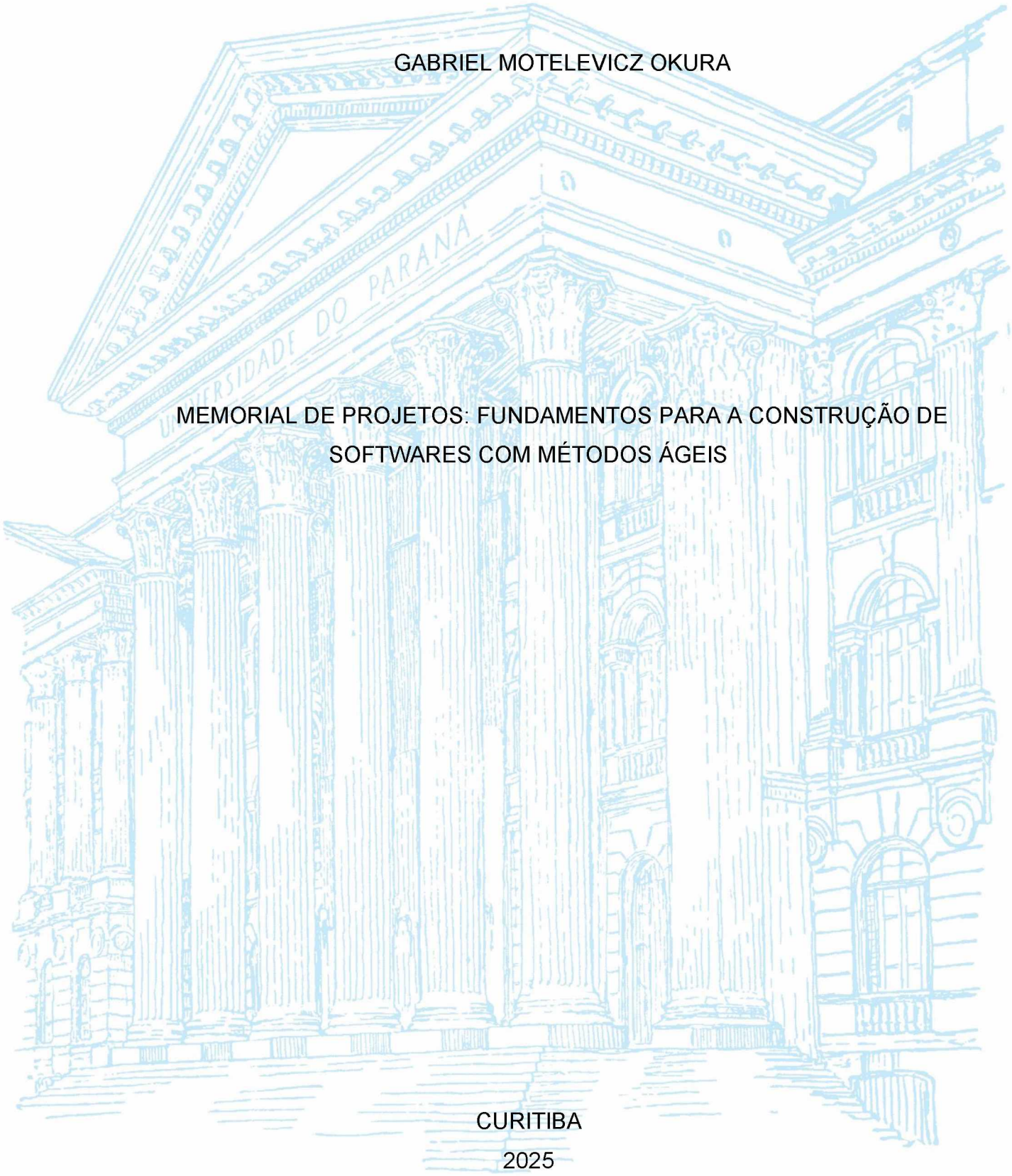
UNIVERSIDADE FEDERAL DO PARANÁ

GABRIEL MOTELEVICZ OKURA



MEMORIAL DE PROJETOS: FUNDAMENTOS PARA A CONSTRUÇÃO DE SOFTWARES COM MÉTODOS ÁGEIS

CURITIBA

2025

GABRIEL MOTELEVICZ OKURA

MEMORIAL DE PROJETOS: FUNDAMENTOS PARA CONSTRUÇÃO DE SOFTWARES COM MÉTODOS ÁGEIS

Memorial de Projetos apresentado ao curso de Especialização 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

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 **GABRIEL MOTELEVICZ OKURA**, intitulada: **MEMORIAL DE PROJETOS: FUNDAMENTOS PARA A CONSTRUÇÃO DE SOFTWARES COM MÉTODOS ÁGEIS**, 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, 03 de Novembro de 2025.

JAIME WOJCIECHOWSKI
Presidente da Banca Examinadora

RAFAELA MAMED VANI FONTANA
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

RESUMO

Este memorial de projetos apresenta uma análise integrada das atividades desenvolvidas ao longo da Especialização em Desenvolvimento Ágil de Software da Universidade Federal do Paraná, com foco na aplicação prática dos princípios e métodos do desenvolvimento ágil. Tem como objetivo demonstrar, por meio da integração entre modelagem, codificação, gestão, design e implantação contínua, como o aprendizado em diferentes disciplinas contribuiu para a formação de uma visão sistêmica do ciclo de vida do software moderno. Também evidencia a evolução das competências técnicas e metodológicas adquiridas no curso e a aplicação de ferramentas como Scrum, XP (*Extreme Programming*), *Kanban* e DevOps em projetos reais, resultando em entregas incrementais, adaptativas e colaborativas. A metodologia adotada baseou-se em estudos de caso práticos desenvolvidos em cada disciplina, com ênfase em experimentação, iteração contínua e integração entre áreas como modelagem ágil (MAG1 e MAG2), gerenciamento de projetos (GAP1 e GAP2), desenvolvimento web e mobile, testes automatizados e infraestrutura DevOps. Essa abordagem multidisciplinar possibilitou compreender o software não apenas como produto técnico, mas como um sistema vivo e evolutivo, sustentado por valores como comunicação, colaboração e entrega contínua de valor. Como resultado, observou-se que a junção entre as disciplinas técnicas com as práticas ágeis promoveu uma cultura de aprendizado constante. Característica fundamental para o profissional que entrega soluções de alta qualidade com foco no usuário e adaptabilidade às mudanças.

Palavras-chave: desenvolvimento ágil; scrum; integração contínua; entrega contínua; programação.

ABSTRACT

This project memorial presents an integrated analysis of the activities developed throughout the Specialization in Agile Software Development at the Federal University of Paraná, focusing on the practical application of agile principles and methods. Its objective is to demonstrate, through the integration of modeling, coding, management, design, and continuous deployment, how learning across different disciplines contributed to the development of a systemic understanding of the modern software life cycle. It also highlights the evolution of the technical and methodological competencies acquired during the course and the application of frameworks such as Scrum, XP (Extreme Programming), Kanban, and DevOps in real projects, resulting in incremental, adaptive, and collaborative deliveries. The adopted methodology was based on practical case studies developed in each discipline, emphasizing experimentation, continuous iteration, and integration among areas such as agile modeling (MAG1 and MAG2), project management (GAP1 and GAP2), web and mobile development, automated testing, and DevOps infrastructure. This multidisciplinary approach enabled an understanding of software not merely as a technical product, but as a living and evolving system supported by values such as communication, collaboration, and continuous value delivery. As a result, it was observed that combining technical disciplines with agile practices fostered a culture of continuous learning — a fundamental characteristic for professionals who deliver high-quality solutions focused on user needs and adaptability to change.

Keywords: agile development; scrum; continuous Integration; continuous delivery; programming.

SUMÁRIO

1 PARECER TÉCNICO

O presente trabalho tem como objetivo apresentar, de forma integrada e reflexiva, os principais projetos desenvolvidos ao longo das disciplinas do curso de Pós-graduação em Desenvolvimento Ágil de Software da Universidade Federal do Paraná. Cada projeto contribuiu de maneira incremental para a consolidação de competências técnicas e metodológicas que, em conjunto, representam um ciclo completo de engenharia de software moderno — desde a concepção e modelagem até a implantação contínua e o monitoramento de qualidade.

A formação iniciou-se com a disciplina MADS – Métodos Ágeis para Desenvolvimento de Software, que introduziu os fundamentos teóricos e práticos da agilidade, por meio da aplicação do *framework* Scrum e das metodologias Lean e Kanban (Pressman; Maxim, 2021; Schwaber; Sutherland, 2020). O projeto de MADS, consistiu na elaboração de um mapa mental que sintetiza os princípios e práticas ágeis, servindo de base conceitual para todas as demais disciplinas. Essa atividade permitiu compreender que o valor entregue ao cliente e a adaptação contínua são elementos centrais de todo processo de desenvolvimento moderno (Beck et al., 2001; Highsmith, 2010).

Na sequência, as disciplinas MAG1 e MAG2 – Modelagem Ágil de Software reforçaram a importância da comunicação visual e da modelagem iterativa, por meio do uso da UML (*Unified Modeling Language*) para representar casos de uso, diagramas de classes e sequência. Esses artefatos alinharam o design técnico às histórias de usuário e ao *backlog* definido em MADS, garantindo coerência entre requisitos e arquitetura. Como destacam Larman (2005) e Sommerville (2019), a modelagem em ambientes ágeis não se opõe à leveza documental, mas a complementa, fornecendo uma visão compartilhada e evolutiva do sistema.

O aprendizado em GAP1 e GAP2 – Gerenciamento Ágil de Projetos de Software consolidou as bases de planejamento, monitoramento e controle de projetos iterativos, utilizando métricas como *lead time*, *velocity* e *throughput* (Cohn, 2011; Anderson, 2010). A aplicação prática ocorreu com a elaboração de um plano de lançamento e simulação de fluxo de trabalho no KanbanBoardGame, o que evidenciou como o gerenciamento de valor e a gestão de riscos se integram ao ciclo técnico estudado em disciplinas anteriores.

O desenvolvimento técnico foi aprofundado nas disciplinas de Introdução à Programação e Banco de Dados. O primeiro projeto envolveu a criação de um sistema bancário simplificado com 42 testes automatizados em *JUnit*, validando regras de negócio e promovendo o pensamento orientado a testes — conceito basilar para práticas como *Test Driven Development* (TDD) (Beck, 2000). Já em Banco de Dados, os modelos entidade-relacionamento e lógicos consolidaram o entendimento sobre persistência e integridade dos dados, permitindo a conexão futura entre backend, frontend e infraestrutura de sistemas ágeis.

Na disciplina AAP – Aspectos Ágeis de Programação, o foco deslocou-se para o código limpo, refatoração e desenvolvimento orientado a testes, práticas centrais do *Extreme Programming* (XP) (Fowler, 2004). O projeto de refatoração proposto exemplificou como a qualidade técnica e a modularidade do código impactam diretamente na capacidade da equipe de responder rapidamente às mudanças de requisitos, promovendo sustentabilidade do produto.

As disciplinas de Desenvolvimento Web (WEB1 e WEB2) e UX – UX no Desenvolvimento Ágil de Software ampliaram o escopo do desenvolvimento para o contato direto com o usuário e a entrega de valor perceptível. Em WEB, foram desenvolvidas aplicações completas com Angular e Spring Boot, simulando ciclos reais de integração contínua. Em *User Experience* (UX), o desenvolvimento de personas e protótipos de alta fidelidade permitiu alinhar o design à jornada do usuário, reforçando o papel da empatia e da iteração no ciclo ágil (Norman, 2013; Gothelf; Seiden, 2016).

A etapa de Desenvolvimento Mobile (MOB1 e MOB2) consolidou a integração multidisciplinar entre design, backend, banco de dados e infraestrutura. O desenvolvimento de um aplicativo completo exemplificou o ciclo de feedback rápido e a adaptação contínua do produto — valores essenciais do Manifesto Ágil (Beck et al., 2001).

Por fim, as disciplinas INFRA – DevOps e TEST – Testes Automatizados encerraram o ciclo com a automação de pipelines e a implementação de testes *end-to-end* (E2E) com Playwright. Essa combinação garantiu que a integração contínua a entrega contínua (CI/CD) fossem aplicadas de ponta a ponta, unindo teoria e prática da engenharia de software moderna (Kim et al., 2016; Fowler, 2006).

Todos os desafios citados compõem uma base sólida para o desenvolvimento de software. Conhecimento que vem sendo refinado desde os primeiros programas e

se fazem presentes atualmente. Visando aperfeiçoar suas habilidades e garantir qualidade nas entregas no ambiente profissional, o autor utiliza dessa base técnica e teórica como objetivos a serem implementados e discutidos.

2 DISCIPLINA: MADS – MÉTODOS ÁGEIS PARA DESENVOLVIMENTO DE SOFTWARE

O projeto desenvolvido na disciplina de Métodos Ágeis para Desenvolvimento de Software (MADS) teve como principal objetivo aplicar ferramentas e práticas ágeis na criação de um produto de software funcional, com entregas incrementais e foco em valor para o cliente. A proposta foi experimentar, em ambiente educacional, a dinâmica real de equipes de desenvolvimento que operam sob princípios de agilidade, enfatizando a colaboração, adaptação contínua e entrega frequente de valor.

De acordo com Pressman e Maxim (2021), o desenvolvimento ágil surgiu como resposta à rigidez dos modelos tradicionais de ciclo de vida, como o modelo cascata, promovendo flexibilidade e colaboração contínua entre cliente e equipe. O Manifesto Ágil (Beck et al., 2001) estabelece que indivíduos e interações são mais valiosos que processos e ferramentas, e que responder a mudanças deve ter prioridade sobre seguir um plano preestabelecido.

Dentro desse contexto, o Scrum foi adotado como estrutura central para organizar o trabalho da equipe. Segundo Schwaber e Sutherland (2020), o Scrum se baseia em iterações curtas (sprints) de até um mês, nas quais a equipe entrega incrementos potencialmente utilizáveis do produto, reforçando o ciclo de inspeção e adaptação. Os papéis definidos — Product Owner, Scrum Master e Equipe de Desenvolvimento — foram fundamentais para promover a auto-organização e a transparência das entregas.

O projeto de MADS serviu de base conceitual e operacional para diversas disciplinas subsequentes. As práticas de planejamento iterativo, definição de backlog e gestão de fluxo foram aplicadas e refinadas em GAP1 e GAP2 (Gerenciamento Ágil de Projetos), enquanto a estrutura de artefatos e priorização de histórias influenciou diretamente os processos de modelagem (MAG1 e MAG2) e implementação (AAP e WEB 1 e 2).

A experiência prática proporcionou uma compreensão aprofundada sobre a importância da colaboração, comunicação efetiva e transparência no ambiente de desenvolvimento ágil.

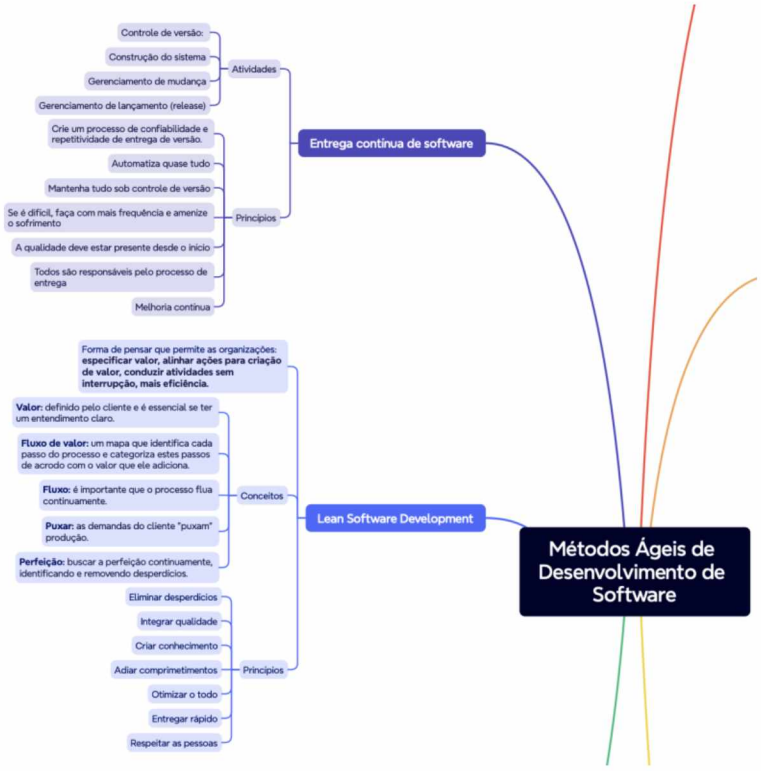
Entre os principais aprendizados, destaca-se a percepção de que o valor entregue ao cliente deve guiar todas as decisões de projeto, e que a adaptação

contínua é essencial para lidar com a complexidade e incerteza típicas do desenvolvimento de software (Highsmith, 2010).

O artefato produzido nesta disciplina foi um mapa mental representando os conceitos fundamentais de processos de software, princípios ágeis e métodos ágeis de desenvolvimento, conforme estabelecido no objetivo da atividade. O mapa sintetiza, de forma hierárquica e visual, a relação entre os modelos tradicionais de processo, as metodologias ágeis e os fundamentos teóricos que sustentam a prática do desenvolvimento ágil de software.

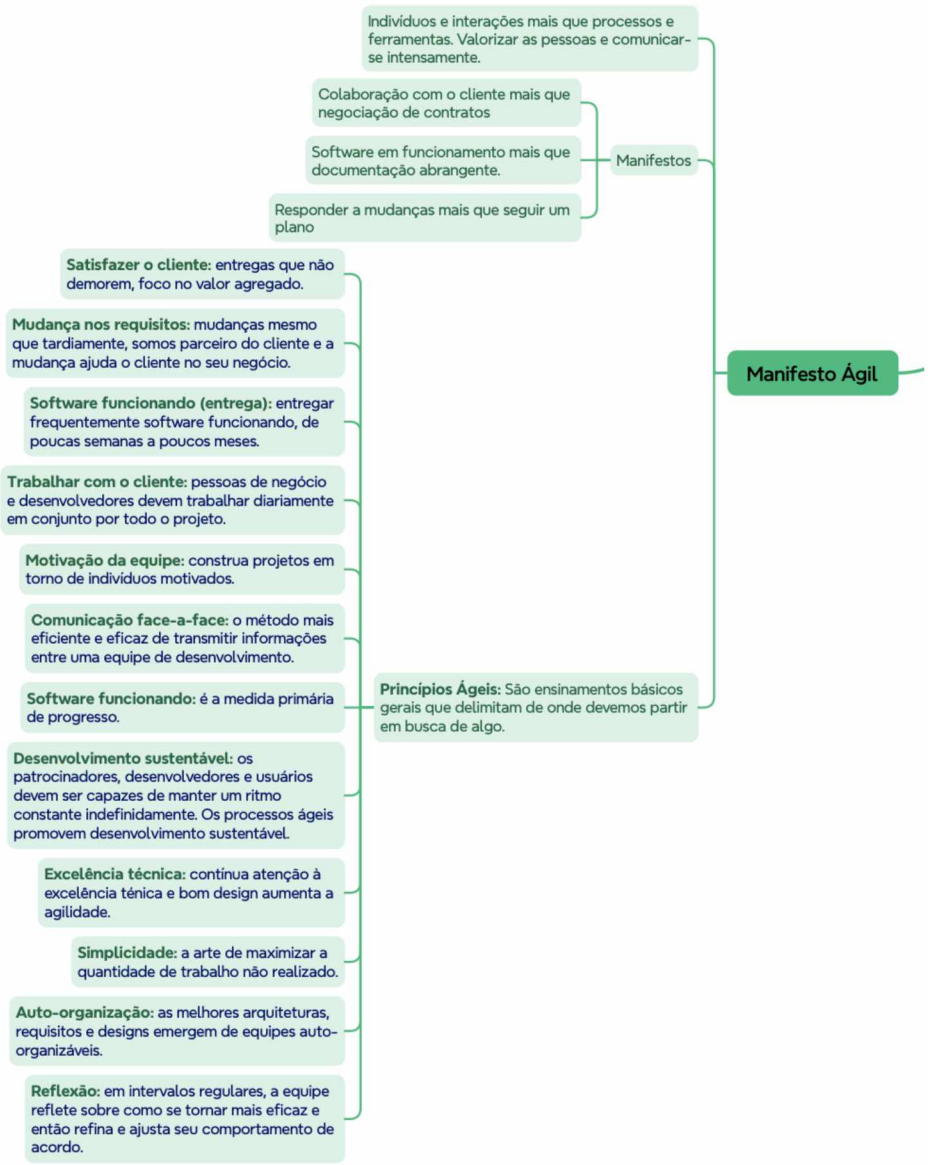
Para melhorar a visualização, o mapa foi dividido em 4 partes (FIGURA 1, FIGURA 2, FIGURA 3, FIGURA 4).

FIGURA 1 – MAPA MENTAL: ENTREGA CONTÍNUA E LEAN SOFTWARE DEVELOPMENT



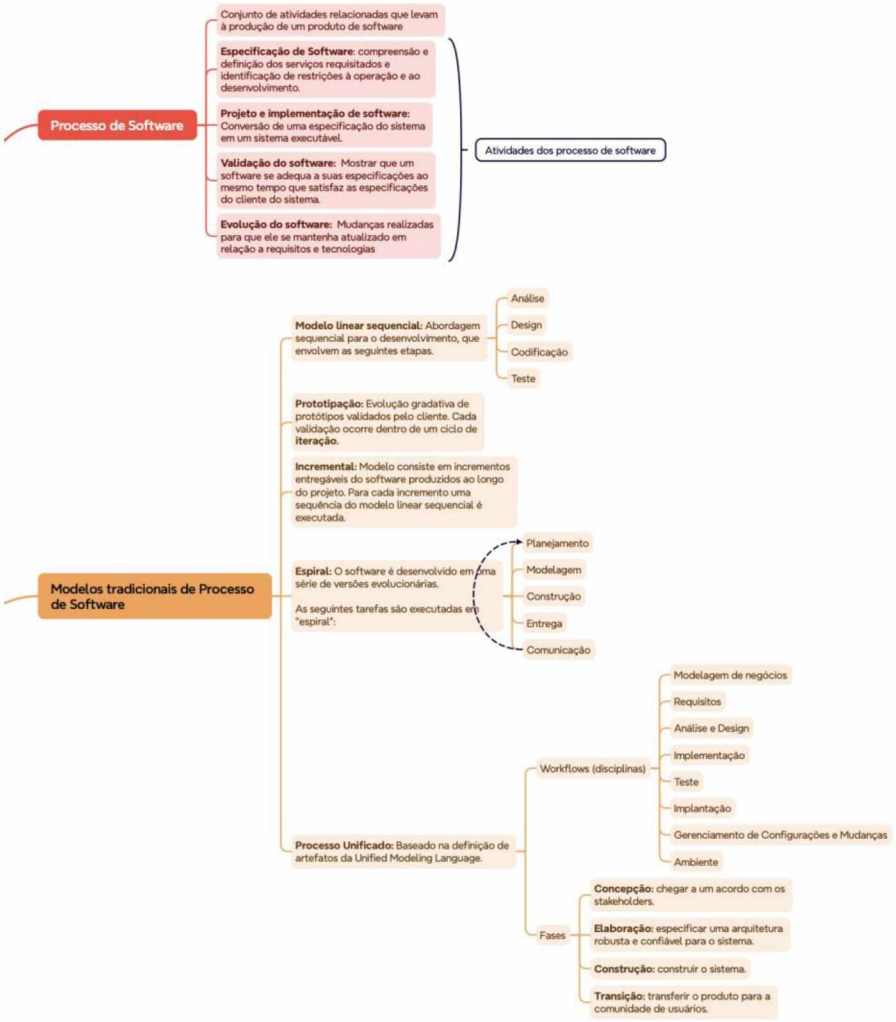
FONTE: O Autor (2025)

FIGURA 2 – MAPA MENTAL: MANIFESTO ÁGIL



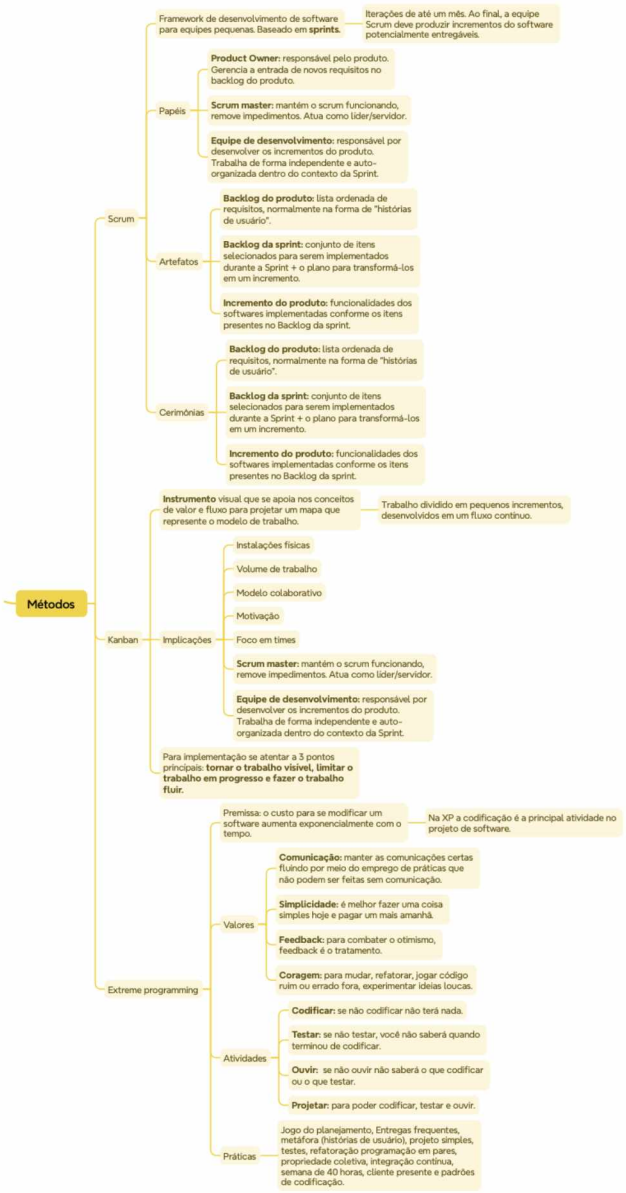
FONTE: Os Autor (2025)

FIGURA 3 – MAPA MENTAL: PROCESSO DE SOFTWARE E MODELOS TRADICIONAIS DE PROCESSO DE SOFTWARE



FONTE: O Autor (2025)

FIGURA 4 – MAPA MENTAL: MÉTODOS



FONTE: O Autor (2025)

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

O projeto desenvolvido nas disciplinas MAG1 e MAG2 – Modelagem Ágil de Software teve como principal objetivo aplicar os fundamentos da UML (*Unified Modeling Language*) dentro de um contexto ágil de desenvolvimento, promovendo a compreensão visual e incremental do sistema a partir de artefatos que refletem o comportamento real do software. As atividades envolveram a construção de diagramas estruturais e comportamentais, como diagramas de classes, atividades, transição de estados, sequência e casos de uso, utilizando ferramentas como o Astah UML.

Esses diagramas foram elaborados de maneira iterativa, alinhados à filosofia ágil de entregas incrementais e validação contínua de requisitos. **N**a primeira etapa (MAG1), o foco esteve na visão estrutural do sistema, com a identificação de objetos, classes, atributos e métodos, bem como o entendimento dos relacionamentos — associação, agregação, composição, herança e polimorfismo — que dão suporte à arquitetura do software. Essa estrutura é a base para garantir coerência entre requisitos e design técnico, reduzindo retrabalhos e facilitando a integração entre equipes.

Já na segunda etapa, o trabalho aprofundou a visão dinâmica e comportamental, explorando diagramas de sequência, atividades e transição de estados, representando o fluxo de eventos entre objetos e usuários em cada caso de uso. A prática de modelagem por meio desses diagramas reforçou a importância da comunicação visual no contexto ágil — permitindo que desenvolvedores, analistas e stakeholders compartilhassem um entendimento comum sobre o comportamento esperado do sistema e suas interações.

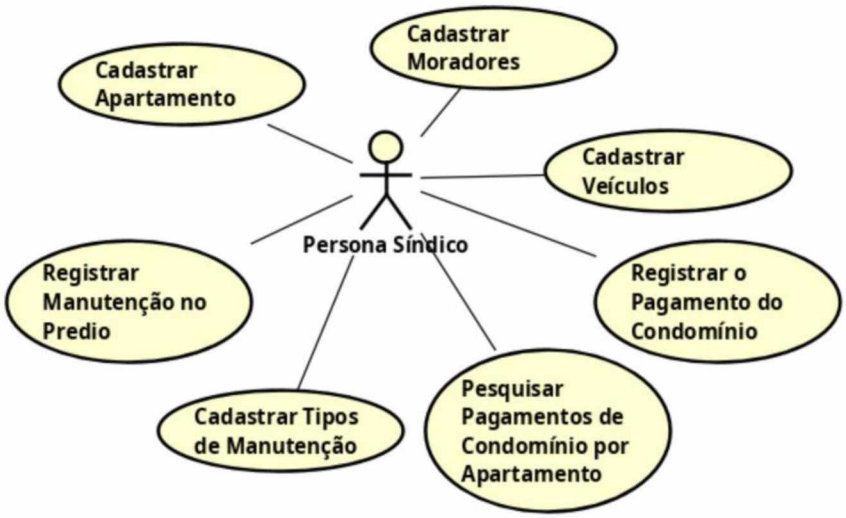
Aprender os fundamentos da modelagem de software nos permite avançar de forma mais assertiva nas etapas de desenvolvimento desses sistemas, diretamente ligados aos que construímos nas disciplinas seguintes do curso.

Os artefatos desenvolvidos no trabalho final de MAG2 tiveram como objetivo representar, de forma visual e iterativa, a estrutura e o comportamento de um sistema de gestão condominial, utilizando práticas de modelagem ágil para integrar requisitos, design e documentação leve. O conjunto de artefatos reflete o ciclo completo de modelagem, desde a definição dos casos de uso até os diagramas de classe e

sequência, contemplando também histórias de usuário e regras de negócio que alinham o sistema ao contexto funcional do produto.

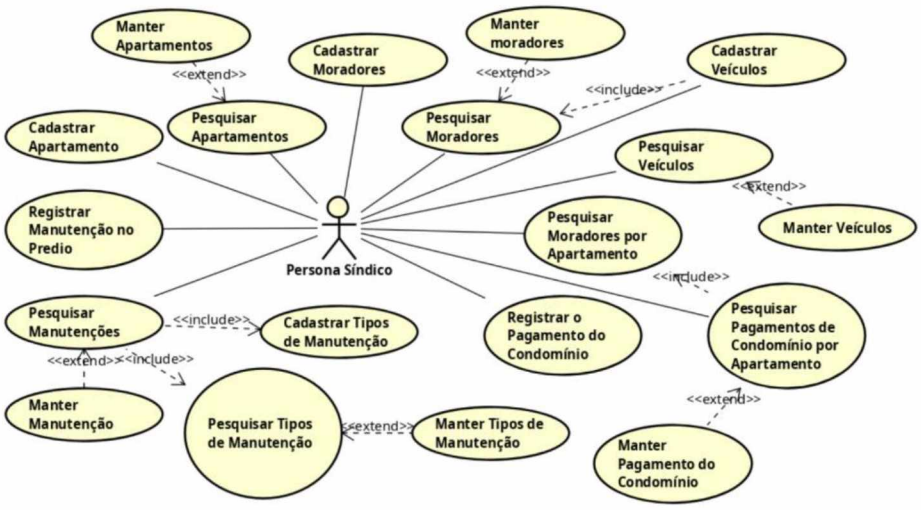
Nas imagens FIGURA 5 e FIGURA 6, encontram-se os casos de uso desenvolvidos para o trabalho. Já na FIGURA 7, encontra-se um exemplo de diagrama de sequência.

FIGURA 5 – CASO DE USO DE NÍVEL 1



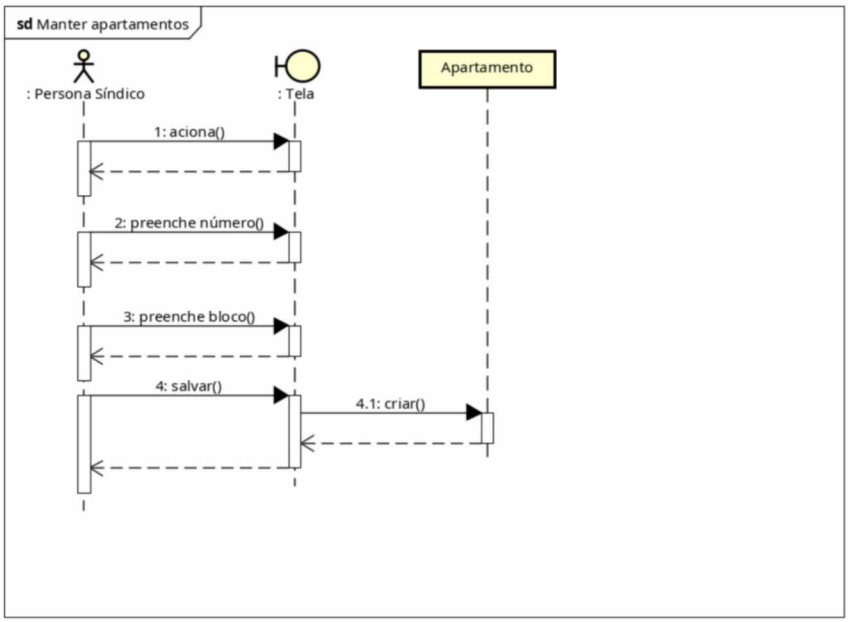
FONTE: O Autor (2025)

FIGURA 6 – CASO DE USO DE NÍVEL 2



FONTE: O Autor (2025)

FIGURA 7 – EXEMPLO DE DIAGRAMA DE SEQUÊNCIA



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 apresentaram elementos centrais tanto para o gerenciamento quanto para a entrega de software de alta performance. O foco foi em desenvolver competências voltadas à planejamento, acompanhamento e controle de projetos ágeis, integrando práticas de ferramentas como Scrum, Kanban e XP (*Extreme Programming*) com fundamentos clássicos de gestão de projetos propostos pelo PMBOK 7ª edição.

No primeiro módulo (GAP1), o aprendizado concentrou-se no papel do gerente ágil e na estruturação do projeto em ciclos iterativos e incrementais. Foram abordados tópicos como formação de equipes auto-organizadas, planejamento adaptativo, identificação de stakeholders e gestão de riscos e incertezas.

Já no segundo módulo, GAP 2, direcionamo-nos para a gestão da qualidade, medição e melhoria contínua dentro do contexto ágil. Foram explorados conceitos como integração de testes ao processo de desenvolvimento, automação de testes, métricas de desempenho, dívida técnica e refatoração — com base nos trabalhos de Cohn (2011), Fowler (2004) e Prikladnicki, Willi e Milani (2014).

Além disso, foram introduzidas as métricas ágeis como instrumentos de acompanhamento do desempenho da equipe e da entrega de valor, incluindo velocidade, *lead time*, *cycle time*, *throughput* e *WIP*, reforçando o uso de indicadores simples, visuais e colaborativos. Essas ferramentas tornam a tomada de decisão orientada mais precisa e guiada por dados, e o gerenciamento mais preditivo e alinhado aos princípios Lean de eliminação de desperdícios e fluxo contínuo.

O trabalho da disciplina consistiu na elaboração de um plano de lançamento de funcionalidade para um sistema, também desenvolvido durantes as disciplinas do curso. Na FIGURA 8 pode-se observar como ficou o calendário para esse exercício.

FIGURA 8 – PLANO DE LANÇAMENTO PARA GAP1

Cálculo da Velocidade:

Horas disponíveis por dia:	8 horas	Tamanho da Sprint:	5 pontos
Horas disponíveis por Sprint:	8 horas * 5 dias = 40 horas	Velocidade:	40 horas por semana / 8 horas (ponto) = 5

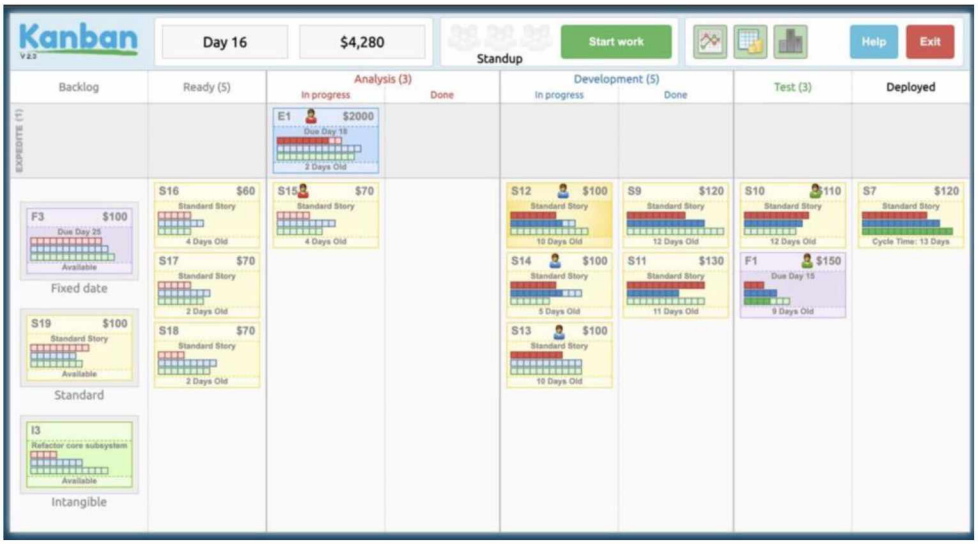
Plano de Release:

Iteração/Sprint 1	Iteração/Sprint 2	Iteração/Sprint 3	Iteração/Sprint 4
Data Início: 03/06/2024	Data Início: 10/06/2024	Data Início: 17/06/2024	Data Início: 24/06/2024
Data Fim: 07/06/2024	Data Fim: 14//06/2024	Data Fim: 21/06/2024	Data Fim: 28/06/2024
<HU001 - Pesquisar Apartamentos> **SENDO** o Síndico **QUERO** Pesquisar os apartamentos **PARA** Fazer manutenção nos seus dados **ESTIMATIVA (2)**	<HU003 - Pesquisar Moradores> **SENDO** o Síndico **QUERO** Pesquisar os moradores **PARA** Fazer manutenção nos seus dados **ESTIMATIVA (2)**	<HU005 - Pesquisar Veículos> **SENDO** o Síndico **QUERO** Pesquisar os veículos **PARA** Fazer manutenção nos seus dados **ESTIMATIVA (1)**	<HU007 - Registrar o Pagamento do Condomínio> **SENDO** o Síndico **QUERO** Registrar um pagamento **PARA** manter os dados do condomínio atualizados **ESTIMATIVA (2)**
<HU002 - Manter Apartamentos> **SENDO** o Síndico **QUERO** Manter os dados dos apartamentos **PARA** que seus dados fiquem atualizados **ESTIMATIVA (3)**	<HU004 - Manter Moradores> **SENDO** o Síndico **QUERO** Manter os dados dos moradores **PARA** que seus dados fiquem atualizados **ESTIMATIVA (3)**	<HU006 - Manter Veículos> **SENDO** o Síndico **QUERO** Manter os dados dos veículos **PARA** que seus dados fiquem atualizados **ESTIMATIVA (2)**	<HU008 - Registrar as manutenções do prédio> **SENDO** o Síndico **QUERO** Registrar uma manutenção para o prédio **PARA** manter os dados do condomínio atualizados **ESTIMATIVA (1)**

FONTE: O Autor (2025)

Por fim, também se realizou uma simulação de gerenciamento de projetos através do sistema KANBANBOARDGAME. Nele foi possível entender melhor como as ferramentas estudadas ajudam a lidar com a alta dinamicidade do mundo real. O jogo proporciona cenários complexos de mudança de equipes, escopo e prazo. Um exemplo pode ser encontrado na FIGURA 9.

FIGURA 9 – SIMULAÇÃO PARA GAP2



FONTE: O Autor (2025)

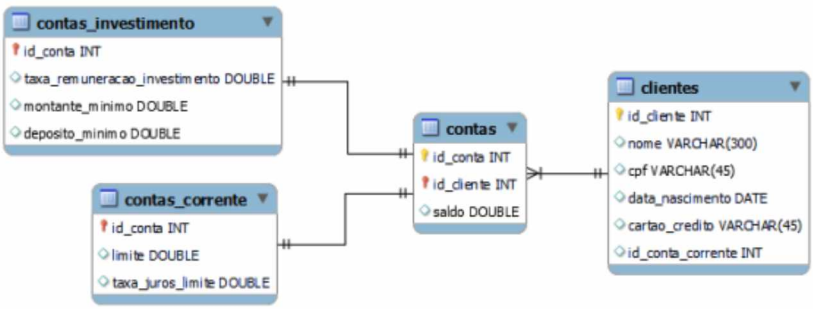
5 DISCIPLINA: INTRO – INTRODUÇÃO À PROGRAMAÇÃO

Compreender os fundamentos da programação é o primeiro passo para entregar software de qualidade e capaz de evoluir rapidamente com segurança. Os princípios aprendidos em Introdução à Programação fortalecem o pensamento lógico e analítico necessário à modelagem e implementação de sistemas. A clareza na estruturação dos algoritmos também apoia diretamente as práticas de modelagem ágil (MAG1 e MAG2). Entender praticamente como se dá o trabalho técnico do desenvolvedor também é extremamente importante.

Entre todas as atividades realizadas durante a disciplina, algumas envolveram a construção de algoritmos sequenciais, condicionais e iterativos, o uso de variáveis, funções e estruturas de dados, além de exercícios voltados à manipulação de listas, dicionários e arquivos. Essa base técnica permitiu consolidar a lógica de programação necessária para etapas posteriores do curso, como aspectos ágeis de programação (AAP) e desenvolvimento web (WEB1 e WEB2).

Em relação ao trabalho de INTRO, o artefato desenvolvido consistiu na implementação do backend de um sistema bancário simplificado, voltado ao controle de cadastro de clientes, contas correntes e contas de investimento, com persistência em banco de dados MySQL e validação através de testes automatizados em JUnit. O projeto disponibilizou 42 casos de teste automatizados utilizando o framework JUnit, com o objetivo de validar todas as regras de negócio e garantir o correto funcionamento do sistema. Na FIGURA 10 é possível observar o diagrama de classes que serviu como base para o sistema.

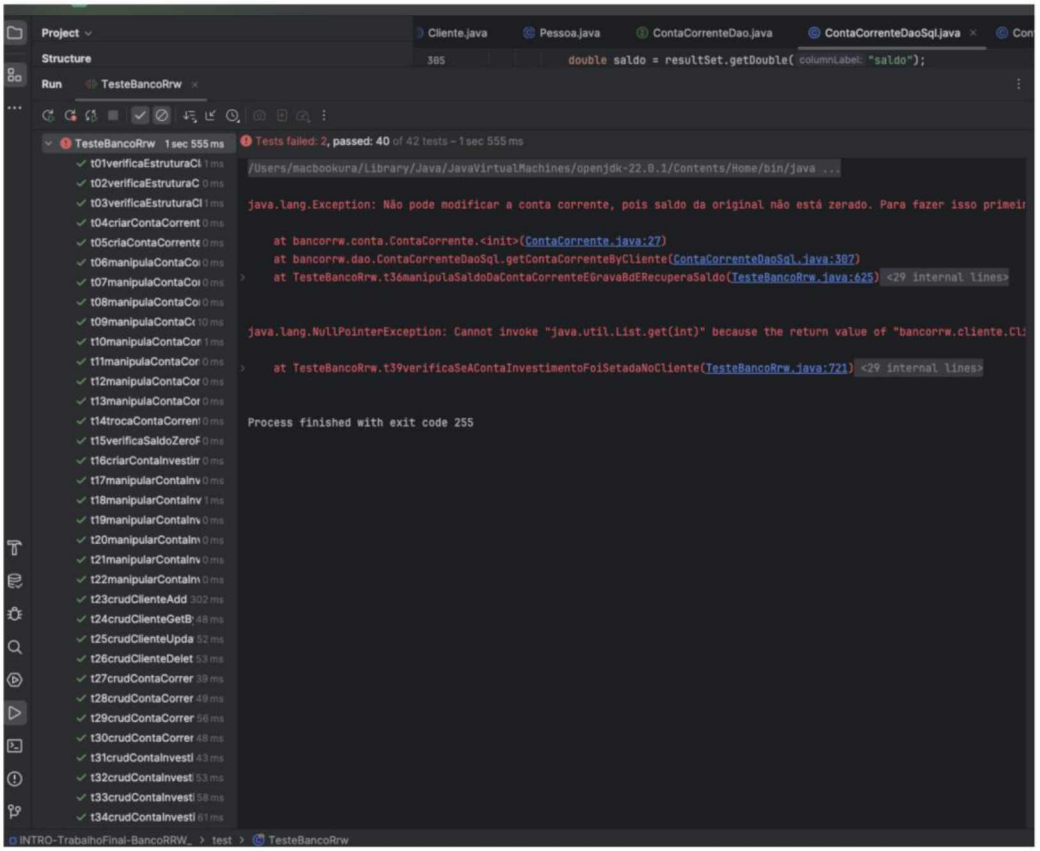
FIGURA 10 – DIAGRAMA DE CLASSES DO SISTEMA BANCÁRIO SIMPLIFICADO



FONTE: O Autor (2025)

Já na FIGURA 11, observa-se os resultados obtidos pelo aluno. Com 95% dos casos de testes executados com sucesso.

FIGURA 11 – RESULTADOS PARA O TRABALHO DE INTRODUÇÃO À PROGRAMAÇÃO



FONTE: O Autor (2025)

6 DISCIPLINA: BD – BANCO DE DADOS

O projeto desenvolvido na disciplina Banco de Dados teve como objetivo principal compreender e aplicar os conceitos de modelagem lógica e física de dados, explorando a estrutura, a integridade e o desempenho de sistemas de armazenamento de informações. A proposta foi desenvolver um modelo relacional completo, desde a definição de entidades, atributos e relacionamentos até a criação de consultas SQL para manipulação de dados. Essa construção visou fornecer a base técnica essencial para o desenvolvimento de sistemas consistentes, escaláveis e aderentes às boas práticas de engenharia de software.

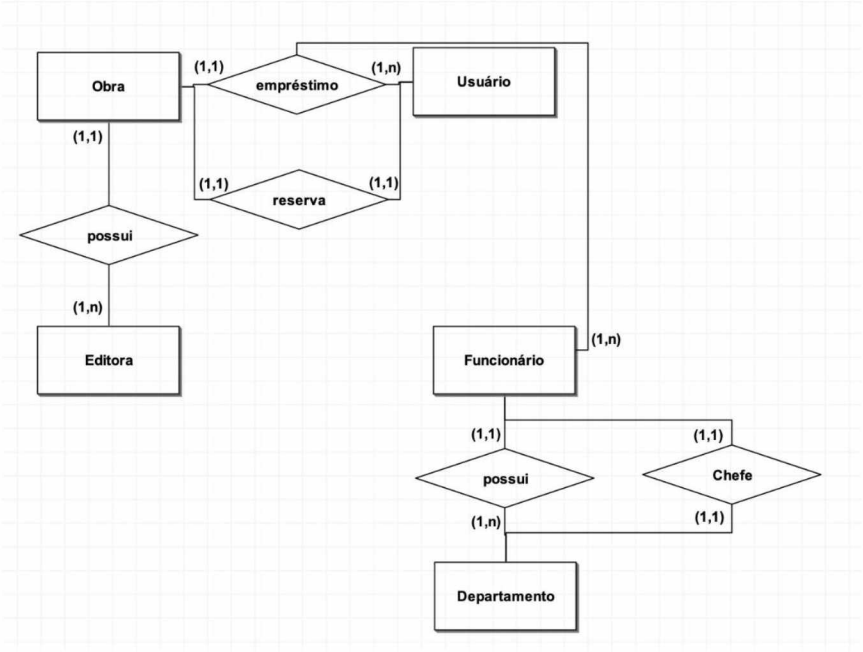
A disciplina abordou as etapas fundamentais da modelagem de dados, destacando o Modelo Entidade-Relacionamento (MER) como ferramenta de representação conceitual e o modelo relacional como base para implementação física. A partir dos conteúdos trabalhados nos slides, foi possível compreender a importância das chaves primárias e estrangeiras, da normalização e das restrições de integridade como elementos estruturantes para garantir a coerência e eliminar redundâncias nos bancos de dados. Esses princípios foram aplicados na criação de um sistema que simulava o funcionamento de um ambiente real, onde o controle e a relação entre tabelas refletiam processos do mundo corporativo, como cadastro de clientes, produtos e transações financeiras.

A execução do projeto envolveu o uso do MySQL, ferramenta que permitiu traduzir o modelo conceitual em código SQL, implementando comandos *CREATE*, *INSERT*, *UPDATE* e *SELECT* com ênfase em eficiência e clareza. Essa prática possibilitou a compreensão da importância do SQL estruturado e otimizado, e relacioná-la ao papel dos bancos de dados em sistemas modernos que demandam agilidade e consistência na entrega de informações.

Para o trabalho da disciplina, foram desenvolvidos artefatos em duas etapas: a primeira, voltada à modelagem de um sistema de controle de biblioteca, e a segunda, voltada ao desenvolvimento de um modelo de controle de compras online, com script de criação das tabelas e inserção de dados.

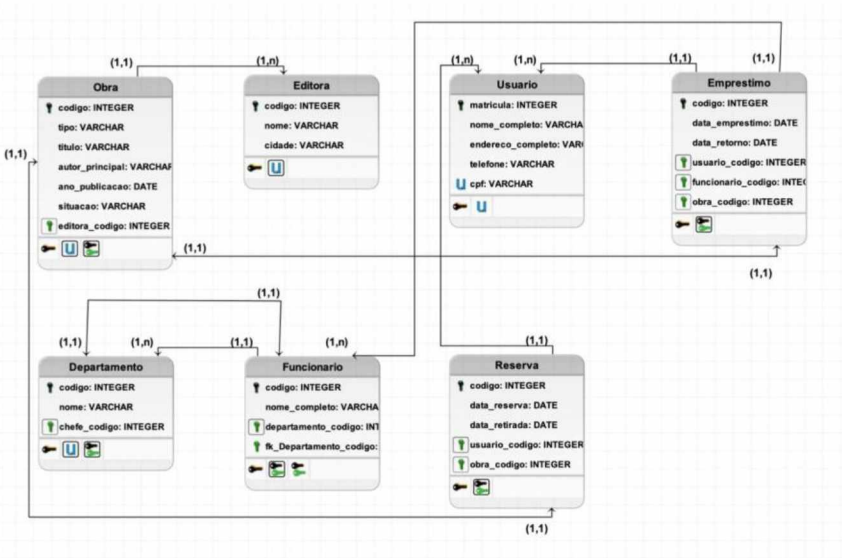
Na primeira parte do projeto, foi desenvolvido um modelo entidade-relacionamento (MER) representando o funcionamento de uma biblioteca (FIGURA 12). E um diagrama conceitual com os relacionamentos do exercício, FIGURA 13.

FIGURA 12 – MODELO ENTIDADE RELACIONAMENTO



FONTE: O Autor (2025)

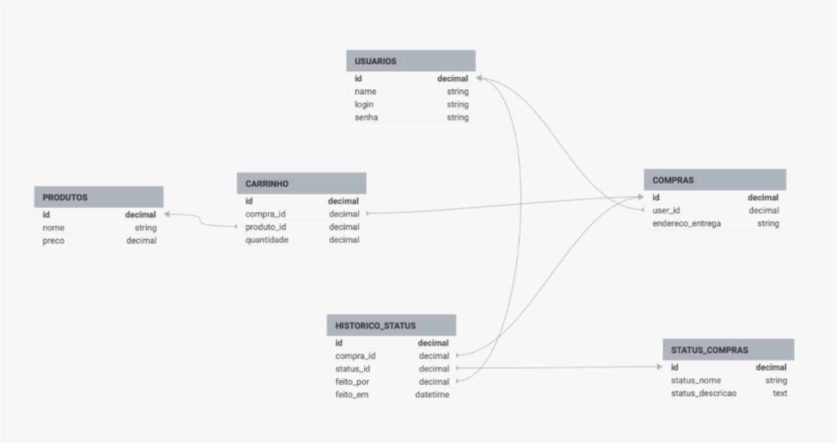
FIGURA 13 – DIAGRAMA CONCEITUAL



FONTE: O Autor (2025)

Na segunda parte do trabalho, foi elaborado um modelo de banco de dados para controle de compras online, simulando o funcionamento de um *e-commerce*. O modelo lógico foi desenvolvido no DB Designer (FIGURA 14). Além disso, também foi enviado tabelas com exemplos de dados populados, simulando o banco de dados (FIGURA 15).

FIGURA 14 – MODELO LÓGICO



FONTE: O Autor (2025)

FIGURA 15 – EXEMPLO DE BANCO DE DADOS PARA O TRABALHO DA DISCIPLINA

USUÁRIOS			
id	nome	login	senha
1	Gabriel	gab123	CRIPTOGRAFADA
2	Jorge	jorge123	CRIPTOGRAFADA
3	Lucas	lucas123	CRIPTOGRAFADA
4	Adriano	adriano123	CRIPTOGRAFADA
5	Lionel	lionel123	CRIPTOGRAFADA

PRODUTOS		
id	nome	preco
1	Bicicleta	1000
2	Playstation 5	3500
3	Xbox 720	3000
4	Macbook Pro	12000
5	iPhone	5000

CARRINHO			
id	compra_id	produto_id	quantidade
1	1	2	10
2	2	3	20
3	3	2	30
4	4	4	40
5	5	1	50

COMPRAS		
id	user_id	endereco_entrega
1	1	UFPR Campus Politécnico
2	1	UFPR Campus Politécnico
3	1	UFPR Campus Politécnico
4	1	UFPR Campus Politécnico
5	1	UFPR Campus Politécnico

STATUS_COMPRAS		
id	status_nome	status_descricao
1	Aguardando pagamento	Aguardando confirmação do pagamento
2	Preparando	Preparando o produto para envio
3	Enviando	Produto na transportadora
4	Enviado	Enviado para o endereço de entrega
5	Entregue	Produto entregue

HISTORICO_STATUS				
id	compra_id	status_id	feito_por	feito_em
1	1	1	1	06/07/2024
2	1	2	1	06/07/2024
3	1	3	1	06/07/2024
4	1	4	1	06/07/2024
5	1	5	1	06/07/2024

FONTE: O Autor (2025)

7 DISCIPLINA: AAP – ASPECTOS ÁGEIS DE PROGRAMAÇÃO

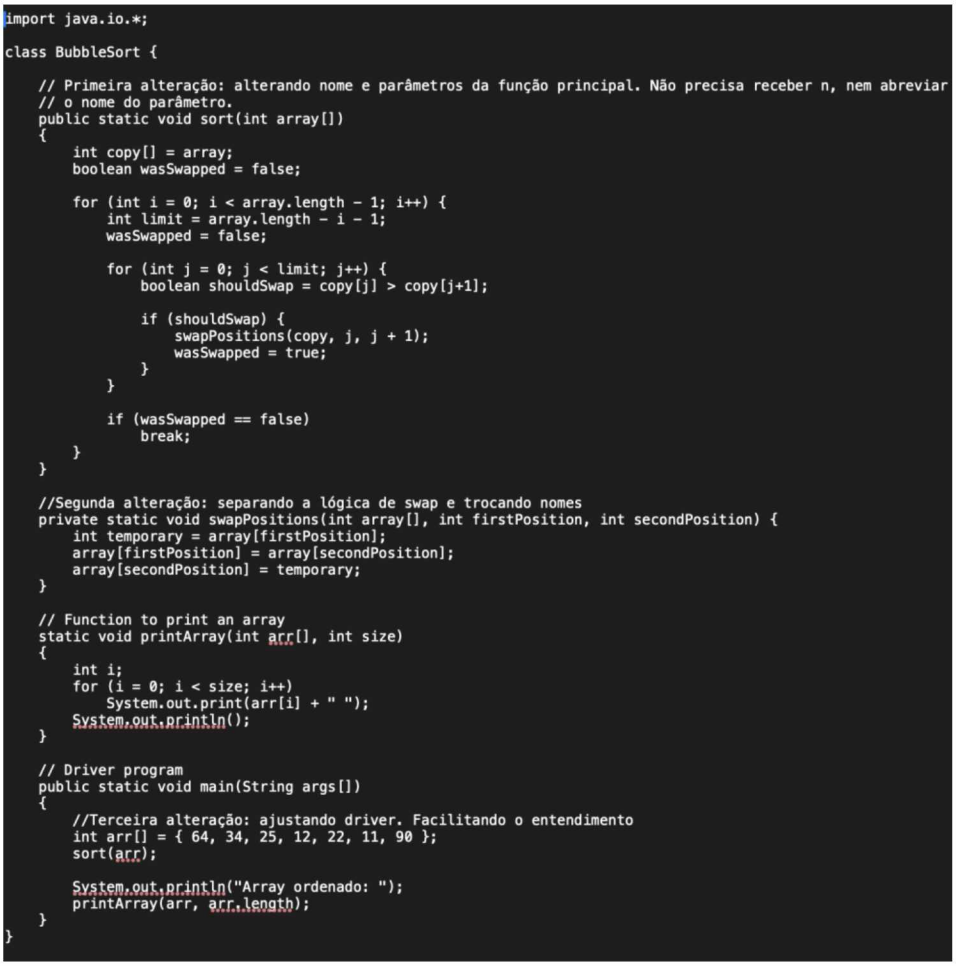
O estudo dos Aspectos Ágeis de Programação (AAP) fundamenta-se em princípios que transformaram a engenharia de software. A partir do Manifesto Ágil (Beck et al., 2001), surgiram práticas que valorizam indivíduos e interações mais do que processos e ferramentas, e software em funcionamento mais do que documentação extensiva. Nesse contexto, a disciplina aprofunda-se nas técnicas que materializam esses valores dentro do código, como *Test Driven Development* (TDD), refatoração contínua e programação pareada, pilares do *Extreme Programming* (XP) (Beck, 2000).

De acordo com Fowler (2004), a refatoração é o processo sistemático de aprimorar o design interno do código sem alterar seu comportamento externo, o que mantém a base técnica saudável e flexível para mudanças. Também é enfatizado a importância de pequenos ciclos de *feedback* e da automação de testes para garantir estabilidade e evolução sustentável.

Essa disciplina estabelece uma ponte entre a teoria dos métodos ágeis e a prática da engenharia de software moderna. Ao trabalhar com código limpo, versionado e testável, pode-se compreender como a qualidade técnica é condição essencial para a agilidade organizacional, conforme defendido por Pressman & Maxim (2021). Além disso, ela dialoga diretamente com outras disciplinas do curso. Em Introdução à Programação, os fundamentos de lógica e estrutura de código oferecem a base para aplicar testes automatizados e refatoração. Em Banco de Dados, as práticas ágeis apoiam a integração contínua entre camadas de persistência e aplicação. Com GAP e MADS, há uma convergência natural, pois o gerenciamento ágil de projetos depende da eficiência técnica que AAP proporciona. Por fim, as disciplinas WEB e INFRA (DevOps) se beneficiam das competências em automação e entrega contínua desenvolvidas aqui, completando o ciclo de desenvolvimento ágil de ponta a ponta.

O projeto proposto para a conclusão dos estudos consistiu em refatorar um código fornecido pelo professor utilizando os conceitos de código limpo reforçados na disciplina (FIGURA 16).

FIGURA 16 – CÓDIGO REFATORADO COM CONCEITOS DE CÓDIGO LIMPO



```
import java.io.*;

class BubbleSort {

    // Primeira alteração: alterando nome e parâmetros da função principal. Não precisa receber n, nem abreviar
    // o nome do parâmetro.
    public static void sort(int array[])
    {
        int copy[] = array;
        boolean wasSwapped = false;

        for (int i = 0; i < array.length - 1; i++) {
            int limit = array.length - i - 1;
            wasSwapped = false;

            for (int j = 0; j < limit; j++) {
                boolean shouldSwap = copy[j] > copy[j+1];

                if (shouldSwap) {
                    swapPositions(copy, j, j + 1);
                    wasSwapped = true;
                }
            }

            if (wasSwapped == false)
                break;
        }
    }

    //Segunda alteração: separando a lógica de swap e trocando nomes
    private static void swapPositions(int array[], int firstPosition, int secondPosition) {
        int temporary = array[firstPosition];
        array[firstPosition] = array[secondPosition];
        array[secondPosition] = temporary;
    }

    // Function to print an array
    static void printArray(int arr[], int size)
    {
        int i;
        for (i = 0; i < size; i++)
            System.out.print(arr[i] + " ");
        System.out.println();
    }

    // Driver program
    public static void main(String args[])
    {
        //Terceira alteração: ajustando driver. Facilitando o entendimento
        int arr[] = { 64, 34, 25, 12, 22, 11, 90 };
        sort(arr);

        System.out.println("Array ordenado: ");
        printArray(arr, arr.length);
    }
}
```

FONTE: O Autor (2025)

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

O conjunto de disciplinas Desenvolvimento Web 1 e 2 tiveram como propósito introduzir e aprofundar as práticas e tecnologias que sustentam o ecossistema da web moderna. **N**o contexto da Especialização em Desenvolvimento Ágil de Software, essas disciplinas cumprem um papel essencial: proporcionar a vivência prática da criação de aplicações completas, conectando os conceitos de front-end, back-end e integração com bancos de dados — pilares fundamentais para o desenvolvimento ágil e colaborativo.

Inicialmente, em Desenvolvimento Web 1, o foco recai sobre os fundamentos da construção de interfaces dinâmicas e responsivas. São abordados os princípios de HTML, CSS e JavaScript, evoluindo para frameworks modernos como Angular e Bootstrap, que promovem produtividade e organização de código por meio da modularização e reutilização de componentes.

Em Desenvolvimento Web 2, o aprendizado avança para a criação de sistemas mais robustos e integrados. Os conteúdos incluem criação e validação de formulários, máscaras e pipes personalizados, integração com APIs REST, autenticação e controle de acesso, e implementação de CRUDs completos utilizando Spring Boot e Angular. Essas práticas reforçam a importância da comunicação entre front-end e back-end, simulando o ambiente real de times ágeis, nos quais a entrega contínua e a integração constante são cruciais para o sucesso do produto. O uso de ferramentas como ng-bootstrap, Spring Data JPA e programação reativa mostra o alinhamento com tendências atuais da indústria, privilegiando escalabilidade, manutenção e qualidade de código. Essa integração multidisciplinar é o que torna o aprendizado significativo e prepara o desenvolvedor para atuar em equipes reais de desenvolvimento ágil.

9 DISCIPLINA: UX – UX NO DESENVOLVIMENTO ÁGIL DE SOFTWARE

O conceito de experiência do usuário (UX) ultrapassa o campo estético ou visual e se concentra na percepção, emoção e satisfação do usuário durante o uso de um produto digital, conectando diretamente o design à entrega de valor e à iteração contínua.

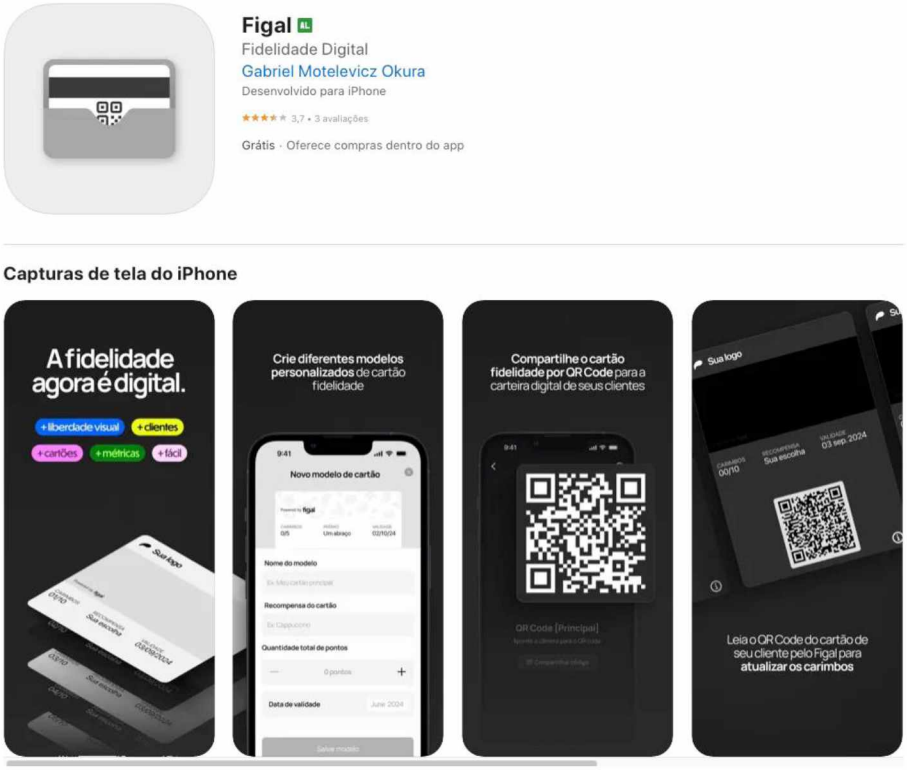
De acordo com **N**orman (2013) e Garrett (2011), uma boa experiência surge da convergência entre usabilidade, utilidade e emoção. Dentro desse contexto, o processo de UX é apresentado como um ciclo iterativo composto por descoberta, ideação, prototipação, testes e refinamento, alinhando-se à filosofia ágil de aprendizado contínuo. Para reduzir o tempo entre as etapas de concepção e validação algumas ferramentas como o *Design Thinking* e as abordagens Lean UX e *Design Sprint*, são utilizadas.

A disciplina também contribui para a melhora da colaboração multidisciplinar em equipes de desenvolvimento, tento em vista que o design é responsável por amarrar todos os conceitos, entender as reais dores dos usuários e entregar, com a melhor experiência, a solução. **N**esse papel, o designer deixa de atuar isoladamente e passa a trabalhar de forma integrada com desenvolvedores, *product owners* e *stakeholders*, validando hipóteses e decisões de design a cada sprint. Mitigando riscos de retrabalho e assegurando que o produto gere valor percebido e mensurável.

Durante o estudo, a aplicação prática do conteúdo se deu por meio do desenvolvimento de personas, jornadas do usuário e protótipos de baixa e alta fidelidade, elaborados em ferramentas como Figma e Miro, além de testes de usabilidade realizados com base em métricas qualitativas e quantitativas. Esses artefatos permitiram avaliar fluxos, reduzir fricções e ajustar funcionalidades antes da implementação, reforçando a ideia de que falhar cedo é aprender rápido, princípio fundamental do ágil e do design iterativo.

Como trabalho final, apresentou-se o aplicativo Figal desenvolvido pelos estudantes e que contempla todos os conteúdos abordados na disciplina. **N**a FIGURA 18 encontra-se uma prévia do aplicativo disponibilizado na loja oficial.

FIGURA 17– PRÉVIA DO APLICATIVO FIGAL



FONTE: O Autor (2025)

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

As disciplinas Desenvolvimento Mobile 1 e 2 tiveram como propósito capacitar o aluno na concepção, desenvolvimento e implantação de aplicações móveis multiplataforma, abordando desde os fundamentos da programação mobile até práticas avançadas de integração e publicação. Esses componentes curriculares representam uma etapa essencial da formação em Desenvolvimento Ágil de Software, pois unem a agilidade de entrega à necessidade contemporânea de aplicações escaláveis, responsivas e centradas no usuário.

Em Desenvolvimento Mobile 1, o foco recai sobre os conceitos introdutórios e arquiteturas de aplicativos híbridos e nativos, com ênfase na utilização de frameworks como Ionic, Angular e Capacitor. São trabalhados temas como estrutura de componentes, roteamento, armazenamento local e integração com APIs RESTful, criando uma base sólida para o desenvolvimento ágil de protótipos funcionais. Essa primeira etapa estimula o aprendizado incremental, permitindo que o aluno compreenda o ciclo de vida de um aplicativo e aplique práticas ágeis como entregas iterativas e validação contínua.

Na sequência, Desenvolvimento Mobile 2 amplia o escopo técnico e conceitual do curso, introduzindo tópicos como integração com serviços externos, notificações *push*, autenticação com Firebase, persistência em tempo real e publicação em lojas digitais. Fomos desafiados a desenvolver um aplicativo completo, com interface moderna e conectividade dinâmica. **N**essa etapa, o processo ágil se manifesta pela entrega de incrementos contínuos e pela aplicação de boas práticas de versionamento, testes e refatoração, garantindo qualidade técnica e aderência às necessidades do usuário final.

A disciplina destaca, ainda, a importância de princípios de UX e UI Design, integrando conceitos vistos anteriormente em UX no Desenvolvimento Ágil e Desenvolvimento Web. A construção de interfaces adaptadas ao contexto mobile requer a aplicação de heurísticas de usabilidade (**N**ielsen, 1994) e princípios de design responsivo, que garantem uma experiência fluida em diferentes dispositivos. Assim, pode-se aprender a alinhar viabilidade técnica, valor de negócio e experiência do usuário — elementos essenciais para produtos digitais de sucesso.

Do ponto de vista da metodologia ágil, o desenvolvimento mobile exemplifica perfeitamente o ciclo de feedback rápido e adaptação constante. A evolução do aplicativo ocorre em pequenos sprints, nos quais cada incremento é testado, revisado e integrado continuamente, reforçando os valores do Manifesto Ágil (Beck et al., 2001). Além disso, práticas derivadas do *Extreme Programming* (XP), como refatoração e integração contínua, garantem estabilidade e sustentabilidade ao código ao longo do tempo (Beck, 2000).

Por sua natureza multidisciplinar, Desenvolvimento Mobile 1 e 2 conecta-se diretamente com outras áreas do curso. A disciplina se relaciona com Banco de Dados, na implementação e consumo de APIs e serviços; com AAP (Aspectos Ágeis de Programação), pela aplicação de testes e integração; e com INFRA (DevOps), ao abordar o ciclo de *build*, *deploy* e publicação contínua. Ao mesmo tempo, complementa os conhecimentos de UX e Web, permitindo que o aluno desenvolva produtos completos, desde a concepção visual até a entrega funcional.

Em síntese, o aprendizado em Desenvolvimento Mobile 1 e 2 consolida o domínio de tecnologias modernas e práticas ágeis, promovendo a formação de um desenvolvedor capaz de atuar de ponta a ponta no ciclo de vida de um software. Essa integração entre teoria, prática e método reflete o verdadeiro espírito do desenvolvimento ágil: entregar valor continuamente com qualidade, adaptabilidade e foco no usuário.

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

A transformação digital e a crescente complexidade dos sistemas de software têm impulsionado a consolidação de práticas que aproximam desenvolvimento e operações. **N**esse cenário, a abordagem DevOps emerge como um paradigma essencial para a entrega contínua de valor e a redução do tempo de ciclo de desenvolvimento. Mais do que uma metodologia técnica, DevOps representa uma mudança cultural e organizacional, voltada à colaboração entre equipes e à automação de processos em todas as fases do ciclo de vida do software (Kim et al., 2016).

A adoção dessa metodologia implica compreender o software não como produto estático, mas como sistema adaptativo em evolução contínua. Essa perspectiva se alinha aos princípios da engenharia de software moderna, que privilegiam a adaptabilidade e a previsibilidade operacional (Pressman; Maxim, 2021). Em ambientes ágeis, a infraestrutura deixa de ser apenas suporte técnico e passa a exercer papel estratégico: ela sustenta a agilidade técnica, viabilizando o fluxo de integração entre as disciplinas de Testes Automatizados, Desenvolvimento Web, Desenvolvimento Mobile e Aspectos Ágeis de Programação. Ao mesmo tempo, conecta-se a Gerenciamento Ágil de Projetos (GAP) e Métodos Ágeis (MADS), fornecendo a base tecnológica para a execução e o monitoramento de releases contínuos. O foco desloca-se do produto estático para o fluxo contínuo de entrega e feedback, garantindo que cada alteração no código seja testada, integrada e disponibilizada de forma previsível e sustentável.

Para tornar todas essas metodologias, (integração e entrega contínua), possíveis e eficientes, ferramentas como Jenkins, GitHub Actions e GitLab CI foram desenvolvidas e automatizam as etapas de build, test e deploy.

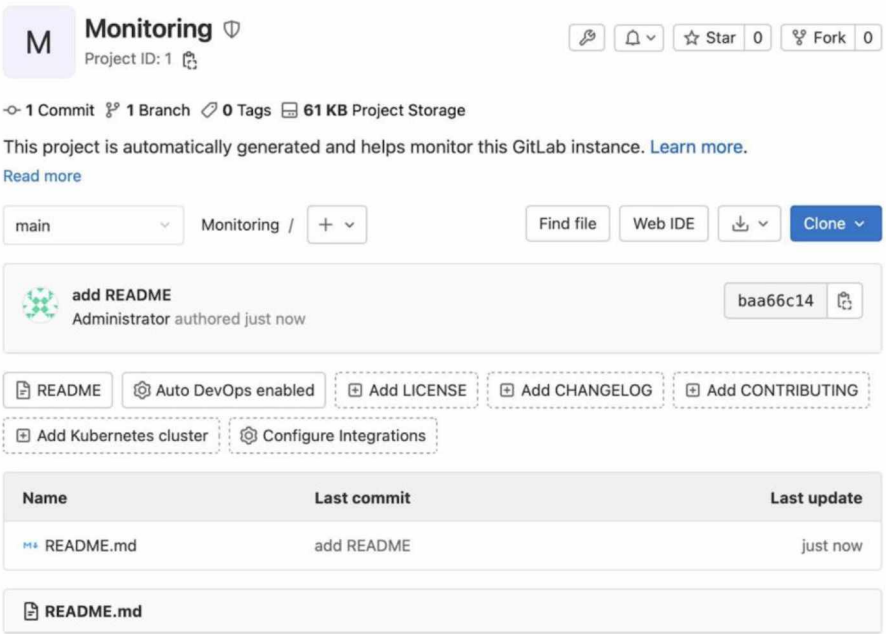
Unindo a teoria à prática, o trabalho da disciplina envolveu a utilização dessas ferramentas citadas, que permitem a criação automatizada de ambientes de desenvolvimento, a gestão de versionamento de código e a execução de pipelines de integração, traduzindo os princípios de colaboração e agilidade técnica em um ambiente controlado e reprodutível. Os resultados podem ser observados nas

FIGURAS 18 e 19.

FIGURA 18 – COMANDO PARA EXECUTAR O DOCKER

FONTE: O Autor (2025)

FIGURA 19 – CAPTURA DE TELA DO REPOSITÓRIO



FONTE: O Autor (2025)

12 DISCIPLINA: TEST – TESTES AUTOMATIZADOS

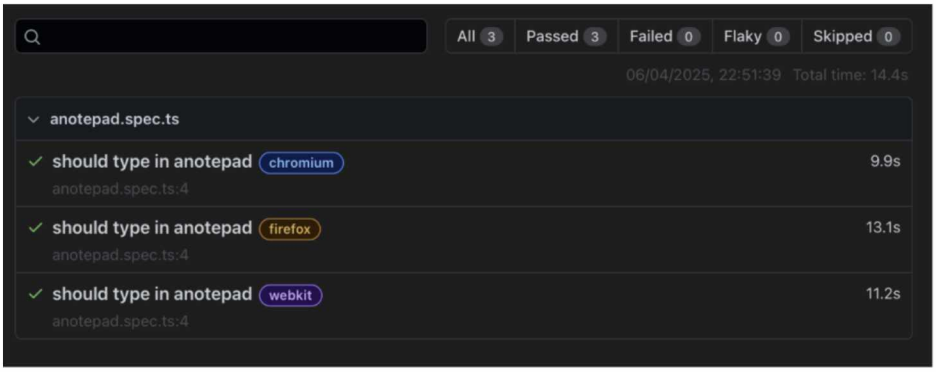
A automatização de testes de software constitui um dos pilares da engenharia ágil, sendo um elemento determinante para a garantia da qualidade contínua e da confiabilidade evolutiva dos sistemas. A literatura de engenharia de software enfatiza que a qualidade não é um atributo final do produto, mas uma característica construída iterativamente durante o ciclo de desenvolvimento (Pressman; Maxim, 2021). Nesse contexto, os testes automatizados emergem como instrumento essencial de verificação e validação, permitindo que os incrementos de software sejam validados de forma rápida, sistemática e reprodutível, conforme os princípios de entrega contínua e melhoria incremental defendidos pelo Manifesto Ágil (Beck et al., 2001).

A aplicação de testes automatizados reflete a transição de um paradigma reativo, no qual o teste é uma etapa posterior à codificação, para um paradigma preventivo, em que a qualidade é promovida desde o início do desenvolvimento. Esse deslocamento teórico é sustentado por metodologias como o *Test Driven Development* (TDD) e o *Behavior Driven Development* (BDD), amplamente difundidos no contexto do *Extreme Programming* (XP) (Beck, 2000). Nesse contexto, os testes assumem uma função estratégica ao promover feedback contínuo e resposta rápida a mudanças.

Beck (2000) e Fowler (2004) defendem que a automação de testes é um dos mecanismos que sustentam a agilidade técnica, pois garante que a refatoração e a evolução do sistema ocorram sem comprometimento da estabilidade. Essa prática é potencializada pela integração de testes a pipelines de Integração Contínua e Entrega Contínua (CI/CD), os quais automatizam a execução de testes a cada novo iteração do projeto e/ou produto. Assim, cada incremento de código é submetido a uma validação sistemática, assegurando o cumprimento dos critérios de qualidade previamente estabelecidos e minimizando riscos de regressão funcional.

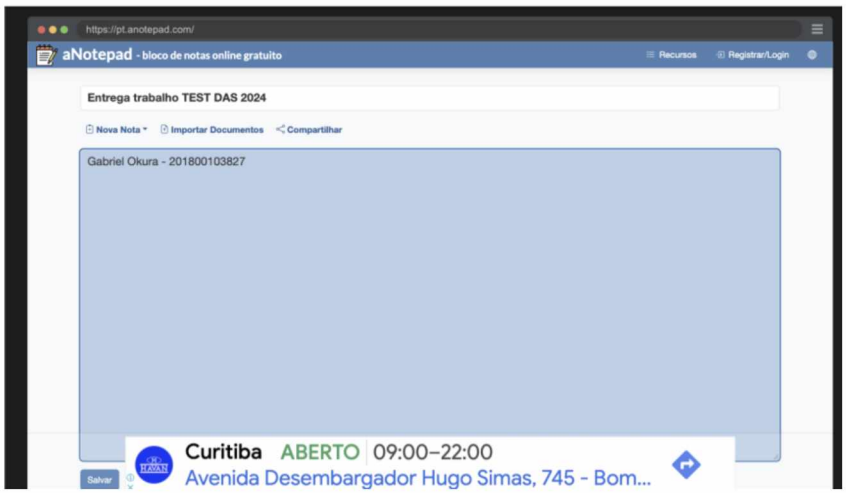
Por fim, o trabalho da disciplina consistiu no desenvolvimento de testes de integração, conhecidos como *end-to-end* (E2E), utilizando o *framework* Playwright. Os resultados podem ser encontrados nas FIGURAS 20 e 21 a seguir.

FIGURA 20 – CAPTURE DE TELA COM OS TESTES EXECUTADOS



FONTE: O Autor (2025)

FIGURA 21 – RESULTADO DOS TESTES DE INTEGRAÇÃO NO FRONT-END



FONTE: O Autor (2025)

13 CONCLUSÃO

Desde a concepção do produto até a sua entrega contínua, a criação desse material e o estudo durante o curso permitiram entender o quão complexo o trabalho pode se tornar. E são nesses momentos que as metodologias ágeis se demonstram extremamente importantes. São elas que garantem agilidade e qualidade para as entregas. As disciplinas cursadas apresentaram uma progressão lógica e integrada com esse pensamento: partiram da base teórica dos métodos ágeis (MADS), avançaram pela modelagem iterativa (MAG1 e MAG2) e pela gestão colaborativa de projetos (GAP1 e GAP2), culminando na aplicação técnica dos conceitos por meio de programação, testes automatizados, infraestrutura e DevOps.

Ao longo do curso, observou-se que a essência do desenvolvimento ágil não está apenas na utilização de ferramentas como Scrum, Kanban ou XP, mas na mentalidade de melhoria contínua, na colaboração entre equipes multidisciplinares e na entrega de valor incremental ao cliente. Projetos como o desenvolvimento de aplicações web e mobile, o design de experiência do usuário (UX) e a automação de pipelines DevOps demonstraram, de maneira prática, que a agilidade técnica é inseparável da agilidade organizacional — ambas sustentadas por comunicação eficaz, autonomia das equipes e integração de ferramentas.

Entretanto, a adoção prática de metodologias ágeis ainda apresenta desafios significativos. Muitas organizações ainda mantêm estruturas hierárquicas rígidas, que dificultam a autonomia das equipes e a rápida adaptação a mudanças. Percebe-se que quanto maior a empresa, mais rígida a sua estrutura e engessados os seus processos. Aqui observa-se que, caso essas metologias já não estejam implementadas na empresa, o processo de convencimento é 1demorado e difícil. Outro desafio encontrado é técnico, relacionado à necessidade de maturidade em automação, testes e integração contínua, para que a agilidade não se restrinja ao planejamento, mas alcance também a entrega de software de qualidade. Muitas vezes os conhecimentos necessários estão hiper concentrados em equipes especificas, dificultando o compartilhamento.

Por fim, há também um desafio organizacional, que envolve alinhar métricas de negócio com indicadores de valor real entregue ao usuário. Em outras palavras, como convencer os chefes de que gastar tempo com metodologias, processos e

ferramentas vai se pagar no futuro e superando a visão limitada de produtividade por volume de tarefas. O presente estudo serve como um excelente reforço para aplicação dos conhecimentos adquiridos em ambientes profissionais e educacionais.

REFERÊNCIAS

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

BECK, K. **Extreme Programming Explained: Embrace Change**. 2. ed. Boston: Addison-Wesley, 2000.

BECK, K. et al. **Manifesto Ágil para Desenvolvimento de Software**. 2001. Disponível em: https://agilemanifesto.org.

COHN, M. **User Stories Applied: For Agile Software Development**. Boston: Addison-Wesley, 2004.

COHN, M. **Desenvolvimento de Software com Scrum: Aplicando Métodos Ágeis com Sucesso**. Porto Alegre: Bookman, 2011.

FOWLER, M. **Refactoring: Improving the Design of Existing Code**. Boston: Addison-Wesley, 2004.

FOWLER, M. **Continuous Integration: Improving Software Quality and Reducing Risk**. Boston: Addison-Wesley, 2006.

GARRETT, J. J. **The Elements of User Experience**. 2. ed. Berkeley: New Riders, 2011.

GOTHELF, J.; SEIDEN, J. Lean UX: **Applying Lean Principles to Improve User Experience**. 2. ed. Sebastopol: O'Reilly Media, 2016.

HIGHSMITH, J. **Agile Project Management: Creating Innovative Products**. 2. ed. Boston: Addison-Wesley, 2010.

KIM, G.; HUMBLE, J.; DEBOIS, P.; WILLIS, J. **The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations**. Portland: IT Revolution Press, 2016.

LARMAN, C. **Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development**. 3. ed. Upper Saddle River: Prentice Hall, 2005.

NIELSEN, J. **Usability Engineering**. San Francisco: Morgan Kaufmann, 1994.

NORMAN, D. A. **O Design do Dia a Dia**. Rio de Janeiro: Rocco, 2013.

PRIKLADNICKI, R.; WILLI, R.; MILANI, F. (Orgs.). **Métodos Ágeis para Desenvolvimento de Software**. Porto Alegre: Bookman, 2014.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de Software: Uma Abordagem Profissional**. 9. ed. Porto Alegre: AMGH, 2021.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide**. 2020.

SOMMERVILLE, I. **Engenharia de Software**. 10. ed. São Paulo: Pearson, 2019.