

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DE SÃO PAULO

GUILHERME VIEIRA DA SILVA

**Implementação da metodologia GTD em
software de organização acadêmica**

2025

GUILHERME VIEIRA DA SILVA

Implementação da metodologia GTD em software de organização acadêmica

Relatório Técnico apresentado
ao Instituto Federal de Educação,
Ciência e Tecnologia de São Paulo,
como parte dos requisitos para a
obtenção do grau de Tecnólogo
em Análise e Desenvolvimento de
Sistemas.

Orientador: Prof. Me. Giovani Fon-
seca Ravagnani Disperati

2025

*“Por falta de um prego,
perdeu-se uma ferradura. Por
falta de uma ferradura,
perdeu-se um cavalo. Por falta
de um cavalo, perdeu-se um
cavaleiro. Por falta de um
cavaleiro, perdeu-se uma
batalha. E assim, um reino foi
perdido. Tudo por falta de um
prego.”.*

George Herbert

LISTA DE ILUSTRAÇÕES

Figura 1 – Tela de login	20
Figura 2 – Mensagem de credenciais inválidas	21
Figura 3 – Tela para envio de e-mail de recuperação de senha	22
Figura 4 – Tela para redefinição de senha	22
Figura 5 – Tela de cadastro de novo usuário	23
Figura 6 – Tela de cadastro de nova tarefa ou projeto	24
Figura 7 – Tela de cadastro de nova ação	24
Figura 8 – Tela de edição de tarefa	25
Figura 9 – Tela de confirmação de exclusão	26
Figura 10 – Tela de confirmação de finalização de tarefa	26
Figura 11 – Tela de busca de tarefas por contexto	27
Figura 12 – Tela de sincronização de e-mail com serviço do Moodle	28
Figura 13 – Tela de segmentação de tarefas do Moodle	29
Figura 14 – Tela de resposta em caso de indisponibilidade da API externa	29
Figura 15 – Tela de Projeto	31
Figura 16 – Visão em calendário	32
Figura 17 – Edição de tarefa a partir da tela de calendário	32
Figura 18 – Tela de configuração de notificações	33
Figura 19 – Tela de notificações após login	34
Figura 20 – Tela com gráficos de atividade dos últimos 7 dias	35

SUMÁRIO

	Sumário	5
1	INTRODUÇÃO	7
1.1	Objetivos	8
1.1.1	Objetivo Geral	8
1.1.2	Objetivos Específicos	8
1.2	O método GTD	8
1.2.1	Capturar	9
1.2.2	Esclarecer	9
1.2.3	Organizar	10
1.2.4	Refletir	10
1.2.5	Engajar	10
2	MÉTODO DE DESENVOLVIMENTO	12
2.1	Visão Geral do Projeto	12
2.2	Requisitos	13
2.3	Processo de Desenvolvimento de Software	13
2.4	Tecnologias	14
2.4.1	Git e GitHub	14
2.4.2	Eclipse IDE	14
2.4.3	Postman	15
2.4.4	Java	15
2.4.5	Spring Boot	15
2.4.6	Spring Data JPA	15
2.4.7	JWT	16
2.4.8	Maven	16
2.4.9	Swagger	16
2.4.10	MySQL	17
2.4.11	MySQL Workbench	17
2.4.12	Flyway	17
2.4.13	Bootstrap	17
2.4.14	AJAX e jQuery	17
2.4.15	Toast UI Calendar	18
2.4.16	Chart.js	18
2.4.17	Moodle	18
2.4.18	Docker	18

3	RESULTADOS E DISCUSSÃO	20
3.1	Tela de Login e Autenticação	20
3.2	Telas de Tarefas	23
3.3	Integração com o Moodle	28
3.4	Organização de listas de Projetos	30
3.5	Visão de calendário mensal	31
3.6	Notificações	33
3.7	Dashboard	34
4	CONCLUSÕES	36
	REFERÊNCIAS	38
	APÊNDICES	41
	APÊNDICE A – MODELO DE REQUISITOS DO SISTEMA E DIA- GRAMA DE CASOS DE USO	42
A.1	Épico 1 – Login e Autenticação	42
A.2	Épico 2 – Modificação de Atividades	43
A.3	Épico 3 – Integração com Moodle	45
A.4	Épico 4 – Gestão de Projetos	47
A.5	Épico 5 – Visualização de Tarefas no Calendário	48
A.6	Épico 6 – Relatório e Notificação	49
A.7	Diagramas de Casos de Uso da UML	51
	APÊNDICE B – MODELO ESTRUTURAL DO SISTEMA: DIAGRAMA DE CLASSES DA UML	54
	APÊNDICE C – MODELO COMPORTAMENTAL DO SISTEMA: DI- AGRAMA DE SEQUÊNCIA DA UML	55
C.1	Login e Autorização com JWT	55
C.2	Requisição de tarefas	55
	APÊNDICE D – MODELO DA ARQUITETURA DO SISTEMA: DIA- GRAMA DE COMPONENTES DA UML	56
	APÊNDICE E – MODELO DE DADOS DO SISTEMA: DIAGRAMA ENTIDADE-RELACIONAMENTO	57

1 INTRODUÇÃO

Em um contexto globalizado, no qual a informação é acessada instantaneamente e os indivíduos estão expostos a uma sobrecarga contínua de estímulos digitais, intensifica-se a busca por técnicas que promovam a concentração nas atividades essenciais, como a execução eficaz de tarefas (SÁ et al., 2018).

Ao considerar o cenário educacional e o crescimento exponencial do ensino remoto — uma modalidade que exige elevado nível de autogerenciamento por parte dos estudantes —, observa-se a necessidade de discutir a organização e a produtividade no contexto acadêmico. Nos últimos dez anos, a educação a distância teve um aumento de 474% no país (INEP, 2022), ao passo que a procrastinação, fenômeno recorrente entre alunos de todas as idades, tem se consolidado como objeto frequente de estudo devido às suas causas e impactos negativos no desempenho acadêmico (DW, 2024).

Além do ensino a distância, destaca-se uma questão social significativa: em 2022, cerca de 8 milhões de jovens brasileiros, entre 15 e 29 anos, conciliavam simultaneamente trabalho e estudos (PINUSA, 2023). Nesse cenário de múltiplas demandas e tempo escasso, a gestão eficiente de tarefas e prazos torna-se um fator essencial para um rendimento satisfatório.

Nesse contexto, o norte-americano David Allen desenvolveu um método amplamente reconhecido, denominado Getting Things Done (GTD), que tem como objetivo principal aprimorar o foco e a produtividade individual por meio da organização sistemática de compromissos e tarefas mentais:

O método consiste basicamente em capturar todas as “coisas” que rondam a mente, sejam “coisas” a serem feitas ou algo que seja considerado útil, colocar essas “coisas” num sistema de confiança capaz de tirá-las da cabeça, se habituar a tomar decisões a respeito dessas capturas realizadas recorrendo a um sistema de “próximas ações” para executá-las em momentos oportunos, e por fim, organizar e coordenar todo o material capturado relacionando os níveis de comprometimento que eles implicam (SÁ et al., 2018, p. 75).

Embora existam adaptações do método GTD em diversas ferramentas de produtividade disponíveis no mercado (ATLASSIAN, 2017), não se identificam soluções que o implementem de forma nativa, especialmente com foco no contexto educacional. Diante dessa lacuna, este trabalho propõe o desenvolvimento de um software baseado no método GTD, voltado a estudantes, que centralize a captura, o processamento e a organização de tarefas acadêmicas, com o objetivo de promover maior eficiência e rendimento.

A escolha deste tema justifica-se pela crescente demanda por soluções que ajudem estudantes a gerenciar melhor suas atividades acadêmicas e pessoais, considerando o aumento das exigências e a escassez de tempo. A implementação de um sistema que facilite a organização do cotidiano estudantil pode trazer impactos significativos no aumento da produtividade e no desempenho acadêmico, proporcionando aos usuários ferramentas que promovem maior controle e foco.

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver uma aplicação web que com base na metodologia GTD, possibilite ao usuário o cadastro, organização e acompanhamento de seus afazeres acadêmicos, bem como acesso a um relatório de conclusões, notificações quando próximo de suas entregas previamente definidas e integração de tarefas com a plataforma do Moodle, visando melhorar a produtividade e o rendimento acadêmico.

1.1.2 Objetivos Específicos

- Realizar um estudo sobre a metodologia GTD e analisar a viabilidade de sua implementação em um sistema computacional.
- Investigar as tecnologias apropriadas para o desenvolvimento de uma aplicação web, com foco em suas vantagens e limitações.
- Examinar as técnicas e ferramentas necessárias para a modelagem e arquitetura do sistema, garantindo sua escalabilidade e eficiência.
- Desenvolver a aplicação, implementando suas funcionalidades principais de acordo com os requisitos definidos.

1.2 O método GTD

Com os inúmeros compromissos que assumimos em diversos âmbitos da vida, é comum que nos esqueçamos de responsabilidades, percamos objetos e, principalmente, tempo. A desorganização e a ineficiência prejudicam nossa produtividade de maneiras variadas (AUSTIN, 2024). O método Getting Things Done (GTD), desenvolvido por David Allen, propõe proporcionar um maior senso de organização, economia de tempo e, conseqüentemente, maior produtividade (SÁ et al., 2018). Como afirma a autora Thais Godinho: "Ter esse controle do tempo é o que dá a sensação de dever cumprido e vida coerente, pois podemos fazer ajustes aqui e ali. É comum perceber que passamos

os dias deixando de lado o que é importante [...]"(GODINHO, 2014, p. 44, apud SÁ et al., 2018, p. 78).

O método GTD é estruturado em cinco etapas principais, que podem ser descritas da seguinte forma:

1.2.1 Capturar

A etapa de captura envolve reunir tarefas, ideias, compromissos, e-mails e qualquer outra informação que possa estar presente na mente em um local confiável, denominado caixa de entrada. O objetivo desta etapa é transferir as ideias para esse espaço, sem qualquer critério avaliativo, para posteriormente processá-las. Isso permite "esvaziar" a mente e organizar melhor as atividades (SÁ et al., 2018).

1.2.2 Esclarecer

A etapa de esclarecimento é fundamental para definir o significado de cada item capturado e determinar as ações necessárias. É recomendável esvaziar a caixa de entrada regularmente (a cada um ou dois dias), a fim de evitar a necessidade de revisitar constantemente as tarefas em busca de itens urgentes (SÁ et al., 2018). Aqui, o questionamento central é: O que precisa ser feito? Se não houver necessidade de ação, os itens são classificados de acordo com um dos seguintes destinos:

- Lixo: Itens que não demandam ação, não são úteis como material de referência e não precisam ser arquivados para futura consulta.
- Incubação: Quando se percebe que algo deve ser feito, mas não agora. O item é retirado da mente, mas será revisitado em um momento mais oportuno. Exemplos incluem planos de longo prazo, como um curso de especialização.
- Referência: Itens que não exigem ação imediata, mas são úteis como fontes de informação, como apostilas e guias práticos.

Se o item exigir ação, ele pode ser classificado em uma das seguintes categorias:

- Fazer: A ação demanda de 2 a 10 minutos para ser concluída e deve ser realizada imediatamente, mesmo que não seja urgente. Exemplos incluem responder a um e-mail ou conferir rapidamente a devolutiva de uma atividade.
- Delegar: Quando a ação pode ser realizada por outra pessoa. O item é atribuído a alguém adequado para a tarefa, e seu progresso deve ser acompanhado.

- **Adiar:** Itens que não podem ser delegados e que demandam mais tempo para serem realizados, mas que podem ser completados em um único ato. Quando uma tarefa demanda mais de uma ação para ser finalizada, ela é considerada um "Projeto" e deve ser regularmente revisada quanto às suas próximas ações.

1.2.3 Organizar

A organização envolve estruturar as tarefas e informações de forma lógica, permitindo uma visão clara do que precisa ser feito e quando. O processo inclui, por exemplo, organizar as referências por ordem alfabética, conforme sugerido pelo autor, e criar listas com as próximas ações de projetos e tarefas isoladas. Tarefas com prazos específicos devem ser adicionadas ao calendário, enquanto tarefas delegadas a terceiros são incluídas na lista de "aguardando resposta" (GODINHO, 2017). Além disso, é importante criar etiquetas que identifiquem as tarefas de acordo com o contexto, como "quando estiver no computador", "quando estiver na faculdade", garantindo que a organização seja otimizada para cada situação específica.

1.2.4 Refletir

É necessário realizar uma revisão periódica das listas e do calendário, ajustando as informações conforme necessário. Este processo é fundamental para garantir que o sistema permaneça dinâmico e alinhado às mudanças nas necessidades individuais, bem como para monitorar o progresso das tarefas.

1.2.5 Engajar

A etapa de engajamento é quando a execução real das tarefas acontece. Neste ponto, diversas técnicas podem ser empregadas para determinar as prioridades, considerando o contexto do usuário, como local, tempo e energia disponíveis para realizar as tarefas. O ciclo envolve a definição do trabalho a ser feito nas etapas anteriores, a execução das tarefas definidas e a adaptação aos imprevistos. Essa última etapa se torna mais eficaz à medida que as tarefas são bem organizadas, permitindo um maior controle sobre o que precisa ser feito (SÁ et al., 2018).

Com a definição do método GTD bem esclarecida, é possível aplicar seus princípios a um sistema projetado especificamente para estudantes. O sistema proposto deve ser capaz de receber as entradas do usuário (Etapa Capturar), guiá-lo no processo de esclarecimento e organização das tarefas, atribuindo etiquetas de contexto, prazo e categoria, conforme as etapas de Esclarecer e Organizar. Além disso, deve permitir a criação de listas de Projetos e fornecer uma interface amigável para facilitar a revisão

periódica (Etapa Refletir), capacitando o usuário a navegar entre as etapas do método diariamente e selecionar tarefas conforme seu contexto.

2 MÉTODO DE DESENVOLVIMENTO

2.1 Visão Geral do Projeto

Em um cenário cada vez mais interconectado, caracterizado pela troca constante de informações entre múltiplas plataformas, torna-se essencial a utilização de protocolos e métodos que promovam a integração eficiente entre dados e serviços.

Dentre esses métodos, destacam-se os serviços web desenvolvidos segundo o padrão arquitetural REST, comumente denominados Web APIs ou simplesmente APIs REST. Esse modelo foi proposto por Roy Fielding, um dos autores do protocolo HTTP, no início dos anos 2000, como parte de sua tese de doutorado. O REST define um conjunto de princípios e restrições para o desenvolvimento de aplicações web baseadas no protocolo HTTP. Quando todas essas restrições são seguidas corretamente, diz-se que a API é RESTful (FERREIRA; KNOP, 2016).

A arquitetura REST baseia-se no paradigma cliente-servidor, promovendo a separação de responsabilidades entre o cliente (interface) e o servidor (lógica de negócio), o que facilita a escalabilidade do sistema, pois os serviços podem ser consumidos por diferentes plataformas de forma independente (FERREIRA; KNOP, 2016).

Um de seus princípios fundamentais é a interação sem estado (stateless), ou seja, cada requisição enviada ao servidor deve conter todas as informações necessárias para seu processamento, sem depender de qualquer contexto armazenado previamente no servidor. Isso contribui para o aumento da confiabilidade, escalabilidade e visibilidade do sistema (FIELDING, 2000).

A REST também estabelece o uso de uma interface uniforme, baseada nos recursos do protocolo HTTP, como os métodos (GET, POST, PUT, DELETE) e os códigos de status, permitindo uma comunicação padronizada e desacoplada entre cliente e servidor (FERREIRA; KNOP, 2016). Além disso, inclui outras restrições importantes, como:

Arquitetura em camadas, que organiza os componentes em diferentes níveis hierárquicos;

Código sob demanda, permitindo que funcionalidades adicionais sejam enviadas ao cliente conforme necessário, promovendo a extensibilidade (FIELDING, 2000).

Neste projeto, esses conceitos serão aplicados no desenvolvimento de uma API REST utilizando a linguagem Java com o framework Spring Boot, com o objetivo de construir um sistema moderno, eficiente e robusto, alinhado às boas práticas do

desenvolvimento web contemporâneo.

2.2 Requisitos

VALENTE (2020) argumenta que de nada adianta um sistema possuir o melhor design, utilizar as linguagens mais modernas do mercado, adotar processos de desenvolvimento bem estruturados e apresentar alta cobertura de testes, se não for capaz de atender efetivamente às necessidades do usuário final. O foco deve estar em resolver os problemas reais do público-alvo.

Com esse objetivo, definem-se os requisitos funcionais e não funcionais da aplicação. Os requisitos funcionais descrevem as funcionalidades que o sistema deve oferecer, ou seja, o que ele deve fazer. Já os requisitos não funcionais estabelecem as restrições e condições de qualidade sob as quais essas funcionalidades devem operar, envolvendo aspectos como desempenho, confiabilidade, segurança e usabilidade (VALENTE, 2020).

Com base nesse entendimento, foram elicítadas, para este projeto, histórias de usuário acompanhadas de suas respectivas definições de pronto, critérios de aceitação e requisitos funcionais, cuja descrição completa encontra-se no Apêndice deste trabalho.

2.3 Processo de Desenvolvimento de Software

No contexto atual, caracterizado por rápidas mudanças de requisitos e alto dinamismo, torna-se essencial adotar processos de desenvolvimento de software que sejam adaptáveis. Nesse cenário, os métodos ágeis foram concebidos para possibilitar a entrega rápida e contínua de software funcional, por meio de incrementos sucessivos e interações frequentes com o usuário (SOMMERVILLE, 2011).

Diferentemente dos modelos tradicionais, os processos ágeis priorizam menos a documentação extensiva — embora ela ainda tenha seu papel — e adotam um design incremental, que evolui ao longo do tempo, em vez de ser completamente definido nas etapas iniciais. Além disso, esses processos valorizam a capacidade de responder a mudanças mais do que a aderência a um plano fixo (VALENTE, 2020).

Neste projeto, adotou-se o princípio de Baby Steps, oriundo da metodologia Extreme Programming (XP), uma das abordagens mais influentes dentro do desenvolvimento ágil. Esse princípio defende que é preferível realizar pequenas entregas frequentes, com código validado, do que implementar grandes funcionalidades de forma isolada e arriscada. Integrações diárias e refatorações constantes são incentivadas, uma vez que facilitam a manutenção da qualidade do código e reduzem os riscos de

falhas acumuladas (VALENTE, 2020).

Além disso, foi aplicada a prática do Design Incremental, também definida pelo XP. Ao contrário de abordagens prescritivas como o modelo cascata (Waterfall), o design do sistema é tratado como uma atividade contínua, ajustada conforme novas funcionalidades são implementadas. O foco recai sobre "fazer a coisa mais simples que funcione", seguido por sucessivos refinamentos à medida que a complexidade do sistema evolui (VALENTE, 2020).

Ainda que este seja um projeto individual — situação em que a aplicação rigorosa de processos formais tende a ser menos crítica (VALENTE, 2020) —, optou-se pela adoção de princípios ágeis com o objetivo de promover produtividade, flexibilidade e qualidade no desenvolvimento da aplicação.

2.4 Tecnologias

Este projeto utiliza um conjunto de tecnologias escolhidas com base em critérios como confiabilidade, comunidade ativa, documentação, curva de aprendizado e aderência às boas práticas de desenvolvimento web. A seguir, descrevem-se as principais ferramentas adotadas e as respectivas justificativas de uso.

2.4.1 Git e GitHub

O Git é um sistema de controle de versão distribuído que armazena o histórico de alterações em arquivos monitorados, permitindo reverter estados anteriores, comparar mudanças e rastrear modificações feitas por diferentes usuários. É uma ferramenta rápida e eficiente para projetos de diferentes escalas (FERNANDA; LOUZADA, 2024).

O GitHub, por sua vez, é uma plataforma que utiliza o Git como sistema de controle de versão, servindo como repositório remoto para projetos, simplificando o armazenamento e compartilhamento de código. Oferece uma interface gráfica e serviços adicionais que facilitam o controle de versões e o rastreamento de mudanças (FERNANDA; LOUZADA, 2024). Foram utilizados durante o desenvolvimento para manutenção do histórico de código e acesso remoto facilitado.

2.4.2 Eclipse IDE

O Eclipse é um ambiente de desenvolvimento integrado (IDE) de código aberto, mantido pela Eclipse Foundation desde 2004 (CORTES, 2023). Trata-se de uma plataforma extensível, amplamente utilizada no desenvolvimento de aplicações Java, com suporte nativo ao Java EE (Enterprise Edition), facilitando a criação de aplicações web robustas.

Sua adoção justifica-se pela compatibilidade com o ecossistema Java Spring, integração com Maven e pelos recursos que agilizam o desenvolvimento, como depuração e gerenciamento de dependências.

2.4.3 Postman

O Postman é uma ferramenta amplamente utilizada para o desenvolvimento de APIs, permitindo facilitar testes de requisições HTTP e documentação em uma interface gráfica intuitiva. Suporta diversos métodos HTTP (como GET, POST, PUT, DELETE, PATCH) e diferentes formatos de corpo de requisição, além de simplificar a autenticação e organizar testes em coleções (GEEKS, 2025).

2.4.4 Java

O Java é uma linguagem orientada a objetos criada pela Sun Microsystems na década de 1990, posteriormente adquirida pela Oracle. Destaca-se pela robustez, segurança e portabilidade e extensa documentação (MELO, 2021). Segundo pesquisas do StackOverflow (2024), o Java é a sétima linguagem mais utilizada globalmente, representando 30% dos usos. Foi usada como linguagem principal de back-end da aplicação, integrando o banco de dados às tecnologias de front-end.

2.4.5 Spring Boot

Spring Boot é uma extensão do Spring Framework voltada à simplificação do desenvolvimento de aplicações Java, especialmente web e baseadas em microsserviços (IBM, 2025). Seu objetivo é reduzir a complexidade de configuração e acelerar a criação de aplicações prontas para produção.

Dentre suas principais características, destacam-se a autoconfiguração, que ajusta automaticamente componentes com base nas dependências, e a abordagem opinativa, que adota convenções e pacotes pré-configurados (como os Spring Starters) para facilitar o uso de bibliotecas comuns. Essas vantagens tornam o desenvolvimento mais ágil e padronizado, sendo um dos motivos de sua adoção.

2.4.6 Spring Data JPA

O Spring Data JPA é um módulo do Spring que simplifica a persistência de dados em aplicações Java, reduzindo a quantidade de código necessário para operações no banco de dados (DEV MEDIA, 2025).

Baseado na especificação JPA 2 (Java Persistence API), ele funciona como uma abstração sobre frameworks ORM, como o Hibernate, permitindo que os desen-

volvedores se concentrem na lógica da aplicação, em vez de na implementação da persistência.

2.4.7 JWT

JWT (JSON Web Token) é um padrão aberto (RFC 7519) utilizado para autenticação segura e compacta em serviços web, permitindo a transmissão de informações entre partes em formato JSON, que são assinadas digitalmente para garantir integridade e autenticidade (AUTH0, 2025).

No contexto do Spring Security, o token JWT é gerado após a autenticação do usuário e contém informações como a identificação do usuário e o tempo de expiração. No presente projeto, os tokens são assinados com uma chave secreta utilizando o algoritmo HMAC, o que possibilita a verificação de sua validade pelo servidor sem a necessidade de manutenção de estado no backend.

O token é armazenado no navegador em um cookie com a flag `HttpOnly`, o que impede que scripts do lado do cliente acessem o JWT, mitigando riscos como ataques XSS (MOZILLA, 2025). Essa abordagem oferece um mecanismo de autenticação seguro, eficiente e escalável, mantendo o servidor stateless e evitando a exposição desnecessária do token no cliente.

2.4.8 Maven

Maven é uma ferramenta de automação de compilação amplamente utilizada em projetos Java, responsável por gerenciar o ciclo de vida do software e suas dependências (APACHE, 2025). Baseia-se no Project Object Model (POM), um arquivo XML que define a estrutura do projeto, bibliotecas externas, plugins e configurações.

Foi usado neste projeto por automatizar tarefas como compilação, empacotamento e gestão de dependências, garantindo padronização, repetibilidade e maior produtividade no desenvolvimento.

2.4.9 Swagger

Swagger é uma especificação amplamente adotada para descrever APIs REST. Ela define um contrato com os detalhes dos endpoints, parâmetros, formatos de dados e respostas da API (SANTOS et al., 2020). No projeto, foi usada a abordagem bottom-up, onde a documentação é gerada a partir de anotações no código Java com Spring. A ferramenta Swagger UI foi integrada à aplicação para gerar automaticamente uma interface interativa que facilita o entendimento e o teste dos serviços expostos pela API.

2.4.10 MySQL

O MySQL é o sistema de gerenciamento de banco de dados relacional mais utilizado no mundo, segundo pesquisa do StackOverflow (2024). Trata-se de uma aplicação de código aberto, rápida e altamente compatível com as principais linguagens de programação, como o Java (ORACLE, 2024). Sua compatibilidade com as tecnologias do projeto fez do MySQL a escolha para o banco de dados.

2.4.11 MySQL Workbench

O MySQL Workbench é uma ferramenta visual unificada para gerenciar bancos de dados MySQL, oferecendo funcionalidades como edição e execução de código SQL, administração de usuários e servidores, modelagem de bases e backup (MYSQL, 2024), sendo essas ferramentas essenciais para as rotinas de desenvolvimento.

2.4.12 Flyway

Flyway é uma ferramenta de versionamento de banco de dados SQL, cujo objetivo é garantir que a aplicação esteja sempre conectada à versão correta do banco, independentemente do ambiente de implantação. Atua de forma semelhante ao Git, mas aplicada à estrutura e dados do banco, assegurando uma sequência consistente de migrações que leva sempre ao mesmo estado final (SERICATI, 2021).

Entre suas principais características, destacam-se o uso de scripts SQL versionados, o controle de versões por meio da tabela interna `flyway_schema_history` e a verificação de integridade via checksum, que impede alterações em migrações já aplicadas. Foi adotado no projeto para facilitar o controle das alterações no banco de dados e manutenção de dados para testes.

2.4.13 Bootstrap

Bootstrap é um framework front-end de código aberto que facilita a criação de interfaces responsivas e compatíveis com diferentes tamanhos de tela (LIMA, 2023). Oferece uma biblioteca de componentes prontos, como botões, menus e formulários, além de estilos CSS pré-definidos e personalizáveis. Foi escolhido neste trabalho para acelerar o desenvolvimento da interface gráfica, garantir responsividade e manter a padronização visual da aplicação.

2.4.14 AJAX e jQuery

O AJAX (Asynchronous JavaScript and XML) é uma técnica que permite atualizar páginas web de forma assíncrona, ou seja, sem recarregar a página inteira. Isso

proporciona uma experiência de usuário mais fluida, com tempos de resposta mais rápidos e maior eficiência na utilização da rede (PANDE, 2014).

Já o jQuery é uma biblioteca JavaScript que facilita a interação com elementos HTML, manipulação de eventos e realização de chamadas AJAX, oferecendo uma sintaxe concisa e funções prontas para uso (PANDE, 2014). A combinação de AJAX com jQuery foi escolhida para melhorar a fluidez e responsividade da interface, além de otimizar a comunicação com o servidor.

2.4.15 Toast UI Calendar

O Toast UI Calendar é uma biblioteca JavaScript de código aberto, voltada para o gerenciamento visual de agendas e eventos. Ela oferece múltiplas visualizações de calendário, permitindo ao usuário ajustar a exibição conforme a necessidade (UI, 2025).

Com foco em usabilidade e configuração simplificada, a ferramenta proporciona uma experiência fluida e eficiente na implementação de calendários, sendo utilizada neste projeto para oferecer uma visão mensal das tarefas agendadas pelo usuário.

2.4.16 Chart.js

O Chart.js é uma biblioteca open-source de visualização de dados em JavaScript, lançada em 2013. Com mais de 60 mil estrelas no GitHub e aproximadamente 2,4 milhões de downloads semanais via npm (CHARTJS.ORG, 2025), é uma das bibliotecas de gráficos mais populares do ecossistema JavaScript.

Sua grande comunidade e documentação robusta a tornam uma escolha sólida, sendo integrada na aplicação para o dashboard semanal de conclusões do usuário.

2.4.17 Moodle

O Moodle é um Sistema de Gestão de Aprendizagem (LMS) que oferece uma plataforma robusta, segura e integrada para a criação de ambientes de aprendizagem personalizados (MOODLE, 2024).

Disponível como software livre e de código aberto, permite a personalização por meio de plugins desenvolvidos por sua vasta comunidade global. Neste projeto, foi utilizado como uma API externa de tarefas, com o objetivo de demonstrar a integração entre APIs.

2.4.18 Docker

O Docker é uma plataforma de código aberto que permite empacotar aplicações e todas as suas dependências em contêineres padronizados e executáveis, garantindo

que elas sejam executadas de forma consistente em qualquer ambiente. Ele simplifica o processo de criação, implantação e gerenciamento de aplicações, tornando-se uma ferramenta amplamente utilizada para testes e integração contínua (SUSNJARA; SMALLEY, 2025).

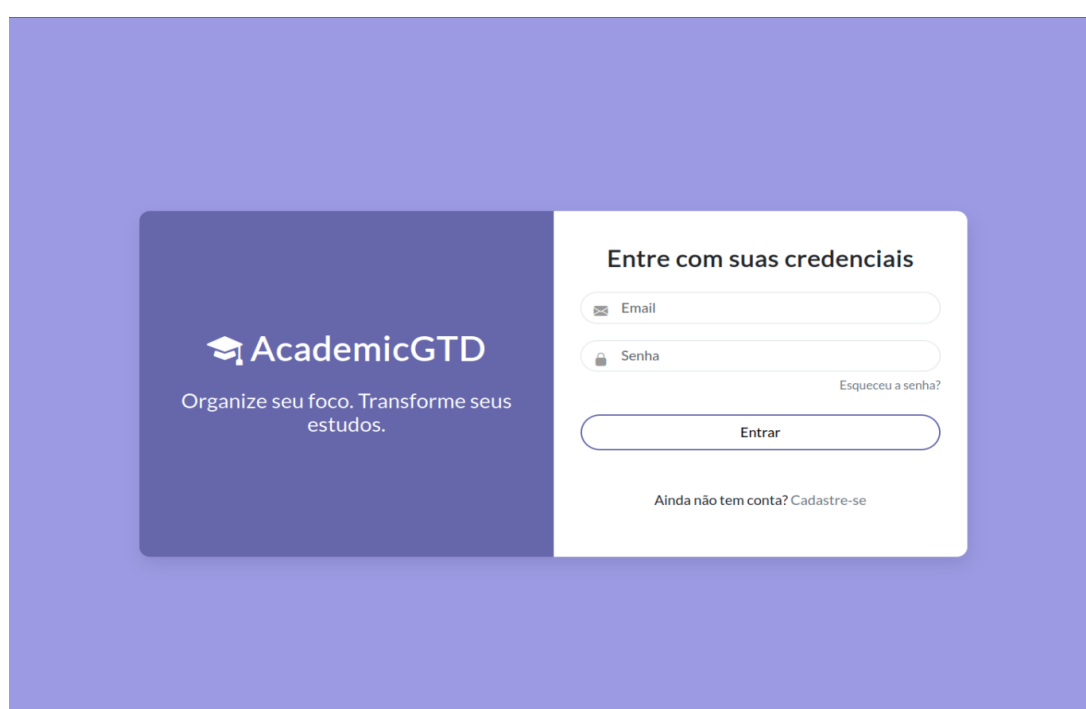
No projeto, o Docker foi utilizado para subir uma instância de teste do Moodle por meio do Docker Compose, facilitando a execução local da aplicação em ambiente isolado e configurado automaticamente, com todos os serviços necessários definidos em um único arquivo YAML. Essa estratégia viabilizou uma validação eficiente da integração entre o sistema GTD e a API do Moodle, conforme os requisitos do sistema.

3 RESULTADOS E DISCUSSÃO

Neste capítulo são apresentadas as telas da versão atual do sistema, juntamente com as funcionalidades implementadas em conformidade com os Requisitos Funcionais (RFx.x), descritos integralmente no Apêndice A.

3.1 Tela de Login e Autenticação

Figura 1 – Tela de login



Fonte: Elaborado pelo autor.

A tela exibida na Figura 1 atende ao RF1.1, que prevê a exibição de uma tela de login com campos para e-mail e senha ao iniciar o sistema. Também contempla o RF1.2, referente à validação das credenciais fornecidas, permitindo o acesso somente se forem válidas.

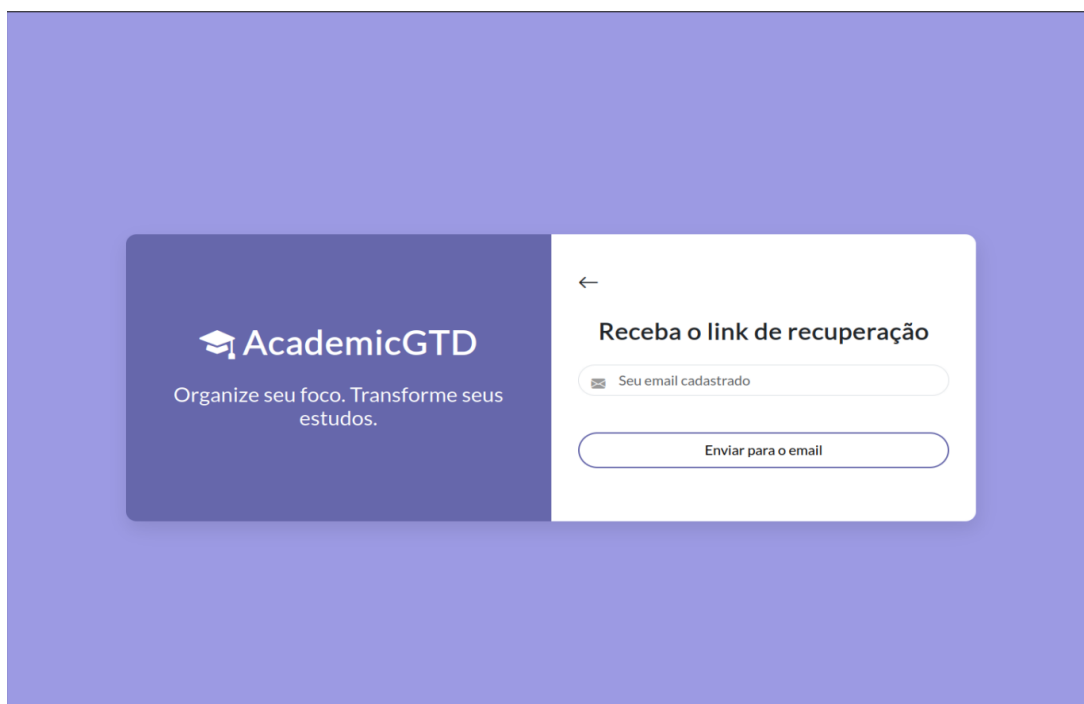
Figura 2 – Mensagem de credenciais inválidas



Fonte: Elaborado pelo autor.

A tela exibida na Figura 2 atende ao RF1.3, que prevê a exibição de mensagem de erro quando o e-mail ou a senha fornecidos estiverem incorretos. Para isso, o sistema adiciona um alerta visual em destaque, exibido em um fundo vermelho na tela de login, informando o usuário sobre a falha na autenticação.

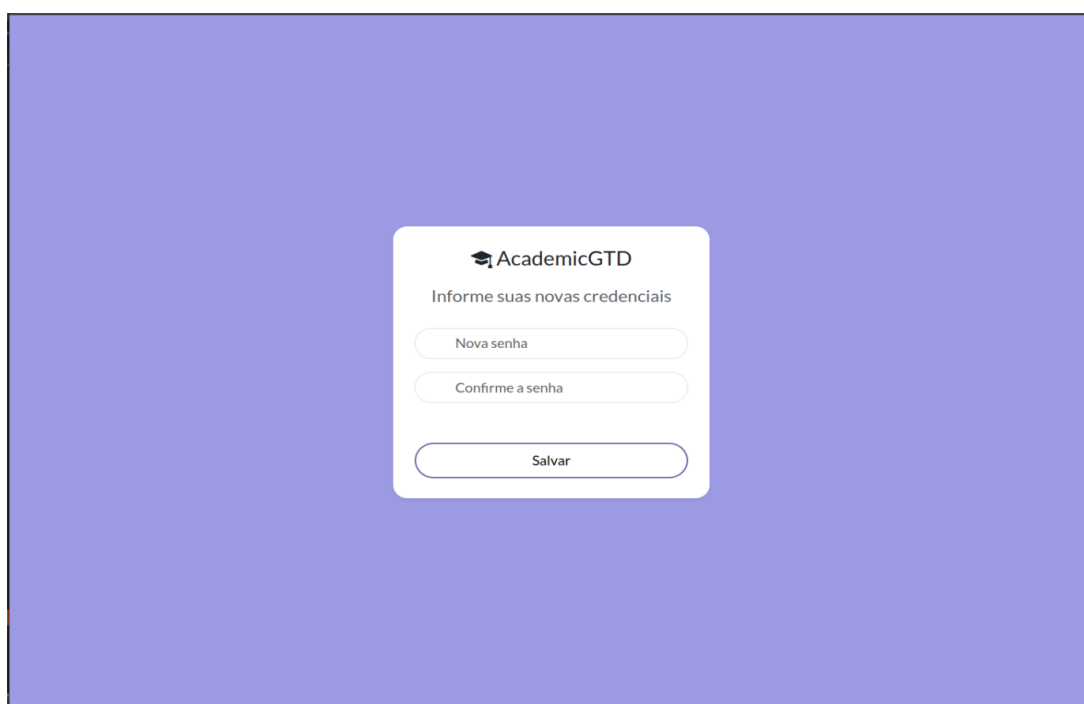
Figura 3 – Tela para envio de e-mail de recuperação de senha



A interface de usuário para a recuperação de senha, exibida em um fundo roxo. O formulário é dividido em duas seções: uma lateral esquerda com o logo 'AcademicGTD' e o slogan 'Organize seu foco. Transforme seus estudos.', e uma lateral direita com o título 'Receba o link de recuperação'. Na seção direita, há um campo de entrada para o e-mail cadastrado, precedido por um ícone de envelope, e um botão 'Enviar para o e-mail'.

Fonte: Elaborado pelo autor.

Figura 4 – Tela para redefinição de senha



A interface de usuário para a redefinição de senha, exibida em um fundo roxo. O formulário centralizado contém o logo 'AcademicGTD' e o título 'Informe suas novas credenciais'. Abaixo, há dois campos de entrada: 'Nova senha' e 'Confirme a senha', seguidos por um botão 'Salvar'.

Fonte: Elaborado pelo autor.

As Figuras 3 e 4 atendem ao requisito RF1.4, que possibilita ao usuário recuperar sua senha em caso de esquecimento por meio da opção “Esqueci minha senha”. A Figura 3 apresenta a tela de recuperação, onde o usuário insere seu e-mail cadastrado para receber o link de redefinição de senha. Já a Figura 4 exibe a página acessada pelo link enviado ao e-mail, contendo dois campos para a inserção e confirmação da nova senha.

Figura 5 – Tela de cadastro de novo usuário

A imagem mostra a interface de usuário para o cadastro de uma nova conta no sistema AcademicGTD. O fundo é uma cor sólida de roxo claro. No centro, há um formulário branco com uma borda arredondada. À esquerda do formulário, sobre um retângulo de cor roxa mais escura, está o logo do AcademicGTD (um ícone de graduação) e o slogan "Organize seu foco. Transforme seus estudos.". À direita, o formulário contém o título "Crie sua conta" com o subtítulo "com email e senha". Abaixo disso, há quatro campos de entrada com ícones de usuário, envelope, cadeado e cadeado, respectivamente, rotulados "Nome", "Email", "Senha" e "Confirme a senha". No final do formulário, há um botão arredondado com o texto "Cadastre-se".

Fonte: Elaborado pelo autor.

A tela exibida na Figura 5 atende aos requisitos RF1.5 e RF1.6, relacionados ao cadastro de novos usuários no sistema. Essa interface, semelhante à tela de login, apresenta campos para nome, e-mail, senha e confirmação de senha, possibilitando que novos usuários realizem seu registro por meio da opção “Cadastre-se”. Além disso, o sistema garante a segurança dos dados armazenando as senhas de forma criptografada no banco de dados, conforme exigido pelo RF1.6.

3.2 Telas de Tarefas

Figura 6 – Tela de cadastro de nova tarefa ou projeto

AcademicGTD

Tarefas Pendentes Dashboard Calendário

Caixa de Entrada

Capture 3

Categorias

Quando Puder 4

Agendados 4

Aguardando 2

Arquivo 5

Outros

Projetos

Do Moodle

Concluídas

Contexto

Filtre por contexto

Capture uma nova tarefa...

Adicionar

Please fill out this field.

Caixa de Entrada

☐ Ler material sobre ferramentas de observabilidade Caixa de Entrada

☐ Assistir à videoaula sobre criptografia e autenticação Caixa de Entrada

☐ escrever testes unitários Projeto Caixa de Entrada

Fonte: Elaborado pelo autor.

Figura 7 – Tela de cadastro de nova ação

AcademicGTD

Tarefas Pendentes Dashboard Calendário

Trabalho Avaliativo 01

Adicionar

Caixa de Entrada

Capture 3

Categorias

Quando Puder 4

Agendados 4

Aguardando 2

Arquivo 5

Outros

Projetos

Do Moodle

Concluídas

Contexto

Filtre por contexto

Como deseja organizar esta ação?

Organizar como Projeto

Adicionar como Tarefa única

☐ Ler material sobre ferramentas de observabilidade Caixa de Entrada

☐ Assistir à videoaula sobre criptografia e autenticação Caixa de Entrada

☐ escrever testes unitários Projeto Caixa de Entrada

Fonte: Elaborado pelo autor.

A figura 6 exibe a tela inicial da aplicação que conta com um menu lateral esquerdo com segmentações de tarefas e uma barra de navegação superior, permitindo a visualização em uma interface estruturada, conforme o RF2.2. Além disso, a tela possui o campo "Capture uma nova tarefa...", que ao ser enviado direciona para a tela da figura 7, onde o usuário pode organizar a ação como tarefa única ou projeto, em conformidade com os requisitos RF2.1 e RF4.1.

Figura 8 – Tela de edição de tarefa

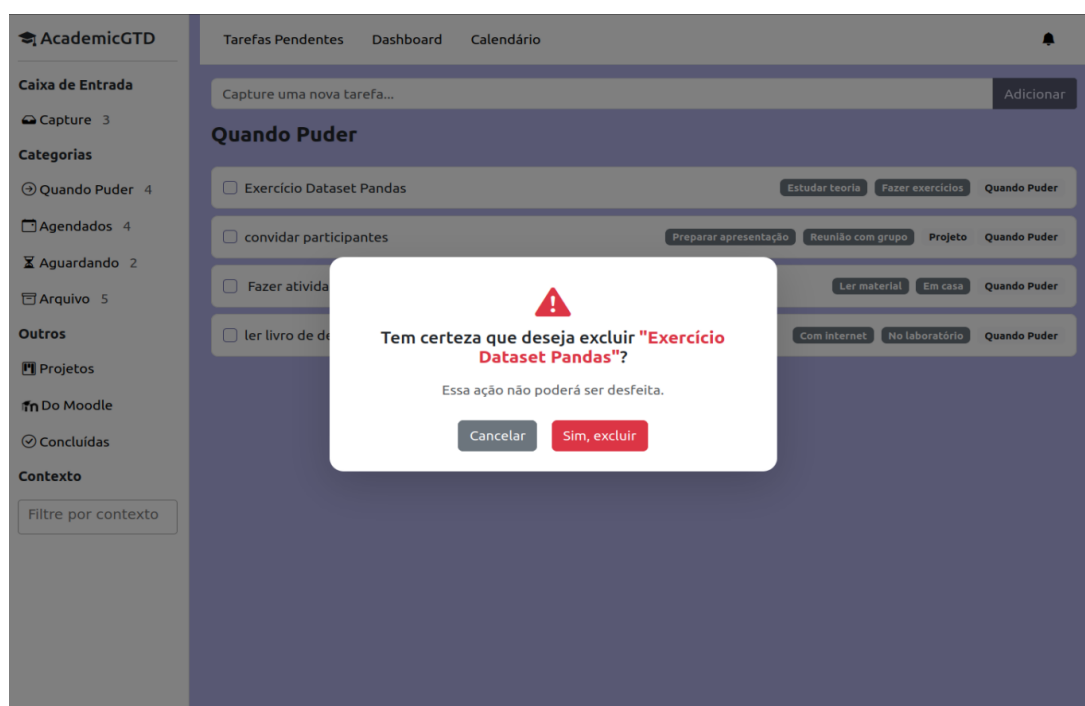
A captura de tela mostra a interface de edição de uma tarefa no aplicativo AcademicGTD. O formulário principal, intitulado "Assistir à videoaula sobre listas e filas", contém os seguintes campos e controles:

- Descrição:** Campo de texto com o valor "Assistir à videoaula sobre listas e filas".
- É prioridade?:** Botão de alternância (toggle) configurado para "Não".
- Prazo:** Campo de data com o valor "23/05/2025".
- Tempo Estimado:** Menu suspenso com o valor selecionado "30 min".
- Assunto:** Campo de texto com o valor "Estrutura de Dados 1".
- Pode ser delegado?:** Botão de alternância (toggle) configurado para "Não".
- Adicione contexto:** Campo de texto com tags selecionadas: "Celular", "Em casa" e "Durante o transporte".
- Associado ao projeto:** Menu suspenso com o valor "Sem projeto associado".
- Categoria:** Menu suspenso com o valor "Agendado".

Na base do formulário, há dois botões de ação: "Excluir Tarefa" (em vermelho) e "Salvar Alterações" (em cinza). O menu lateral à esquerda mostra a estrutura do aplicativo, incluindo "Caixa de Entrada", "Capturar", "Categorias" (com filtros como "Quando Puder", "Agendados", "Aguardando", "Arquivo") e "Outros" (com filtros como "Projetos", "Do Moodle", "Concluídas").

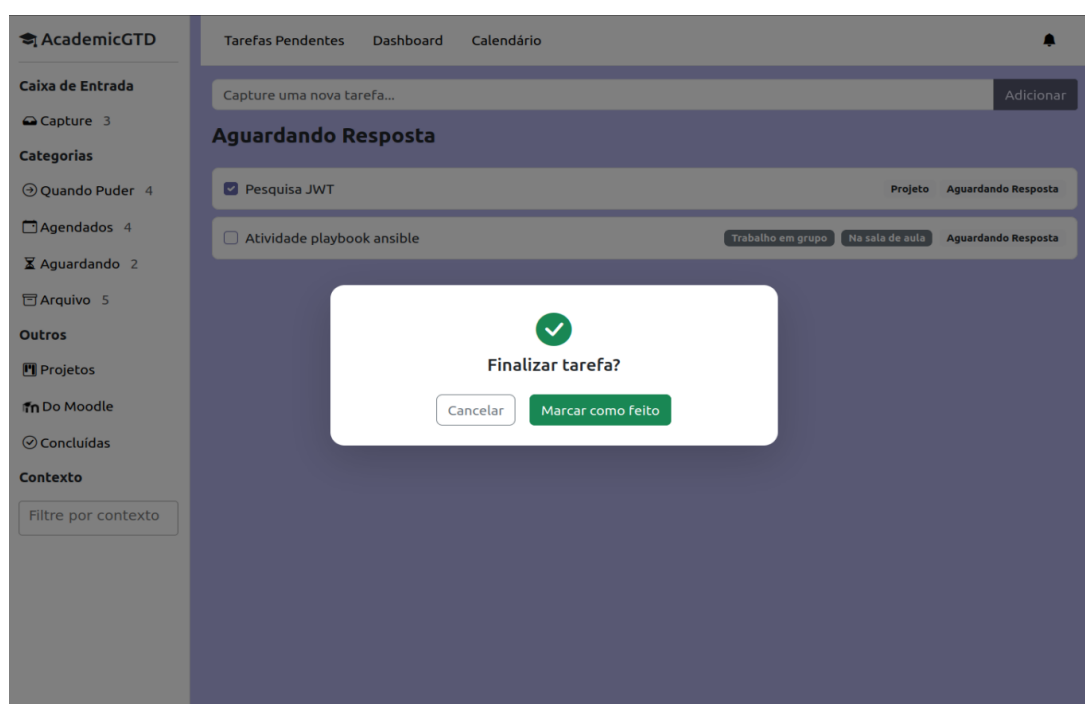
Fonte: Elaborado pelo autor.

Figura 9 – Tela de confirmação de exclusão



Fonte: Elaborado pelo autor.

Figura 10 – Tela de confirmação de finalização de tarefa



Fonte: Elaborado pelo autor.

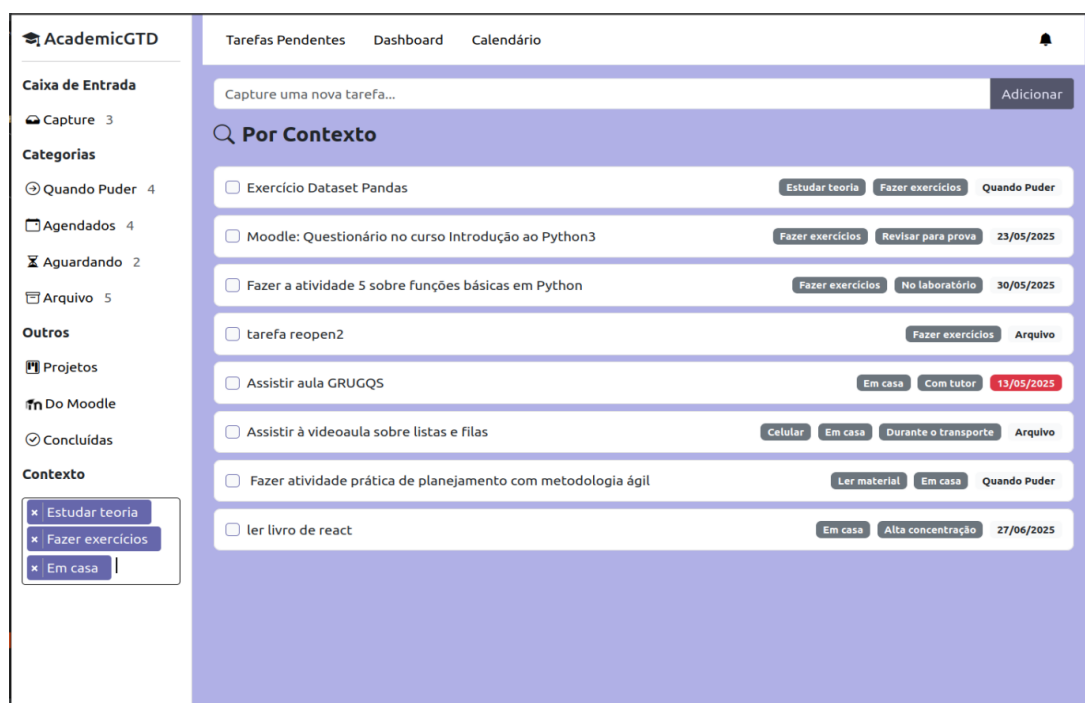
As Figuras 8, 9 e 10 ilustram as funcionalidades relacionadas ao gerenciamento de tarefas previamente cadastradas no sistema. A Figura 8 apresenta o modal de edição de tarefas, que permite alterar informações como descrição, prioridade e prazo, atendendo ao RF2.3. Essa funcionalidade também contempla o RF2.6, ao oferecer a categorização automática das tarefas com base nos dados inseridos, e o RF2.7, ao permitir a associação de rótulos de contexto pelo usuário.

Além disso, essa tela permite que o usuário associe uma tarefa a um projeto já existente ou altere essa associação posteriormente, conforme previsto nos RF4.4 e RF4.5, que tratam respectivamente da vinculação e desvinculação de tarefas a projetos a qualquer momento do seu ciclo de vida.

A Figura 9 mostra o modal de confirmação para exclusão de tarefas, exigido pelo RF2.4, assegurando que o usuário confirme a ação antes de sua execução. Já a Figura 10 exhibe o modal de confirmação para marcação de tarefa como concluída, implementando o RF2.5.

Todas essas funcionalidades fazem parte das operações de atualização e exclusão do ciclo CRUD, em conformidade com o RF2.9, que exige que o sistema mantenha um banco de dados capaz de realizar tais operações.

Figura 11 – Tela de busca de tarefas por contexto

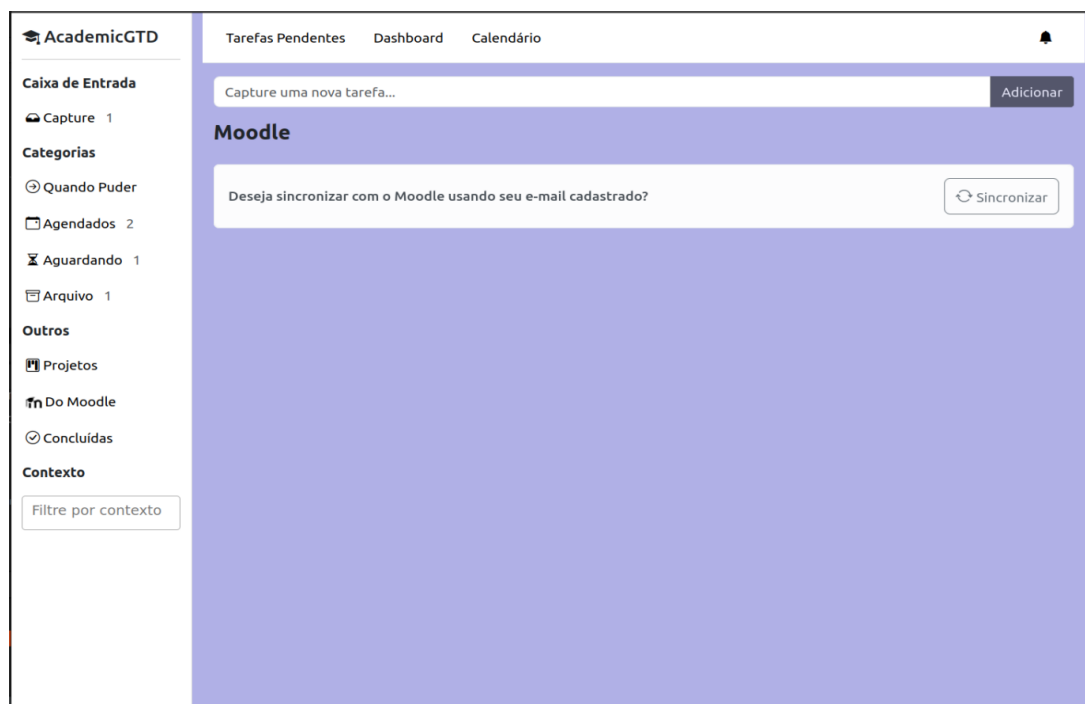


Fonte: Elaborado pelo autor.

A figura 11 mostra a implementação do requisito RF2.8, que define que o usuário deve poder filtrar suas tarefas com base em rótulos de contexto previamente atribuídos a elas, por meio da inserção de tags no menu lateral.

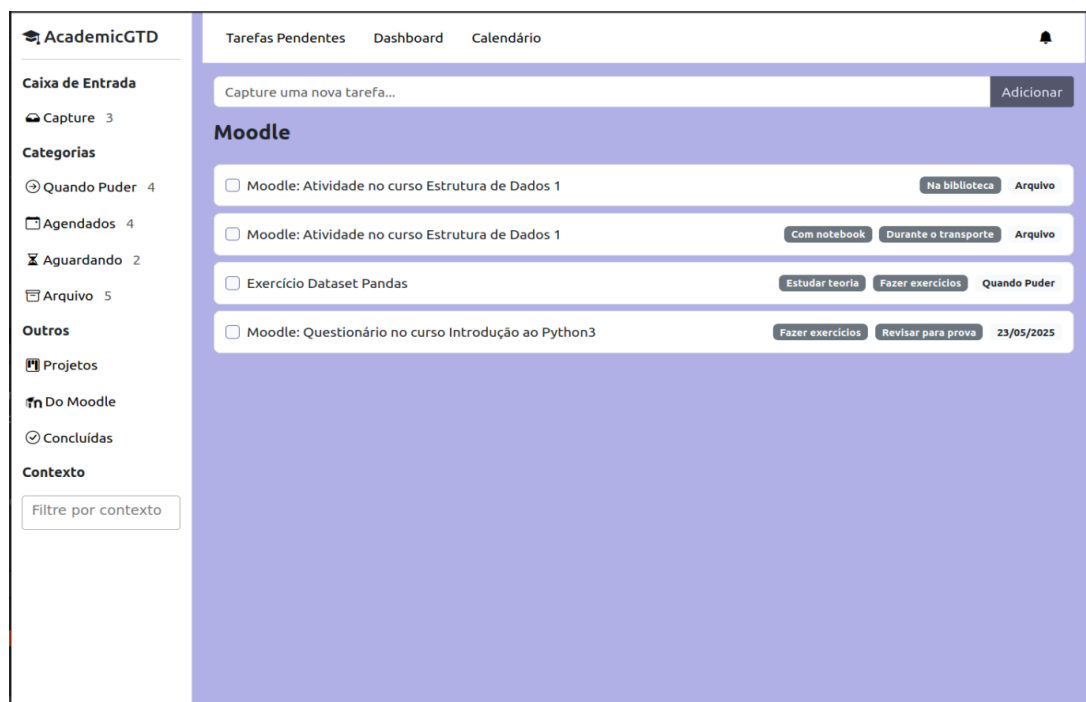
3.3 Integração com o Moodle

Figura 12 – Tela de sincronização de e-mail com serviço do Moodle



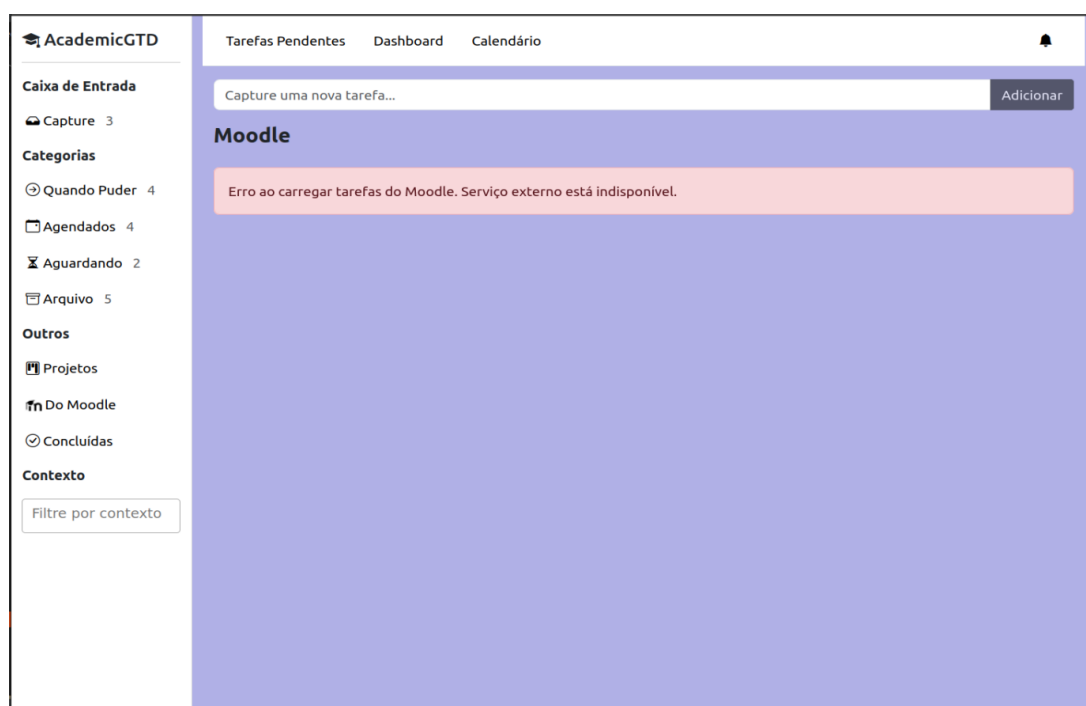
Fonte: Elaborado pelo autor.

Figura 13 – Tela de segmentação de tarefas do Moodle



Fonte: Elaborado pelo autor.

Figura 14 – Tela de resposta em caso de indisponibilidade da API externa



Fonte: Elaborado pelo autor.

As Figuras 12, 13 e 14 apresentam a interface de sincronização com o Moodle, responsável por integrar tarefas acadêmicas ao ambiente GTD.

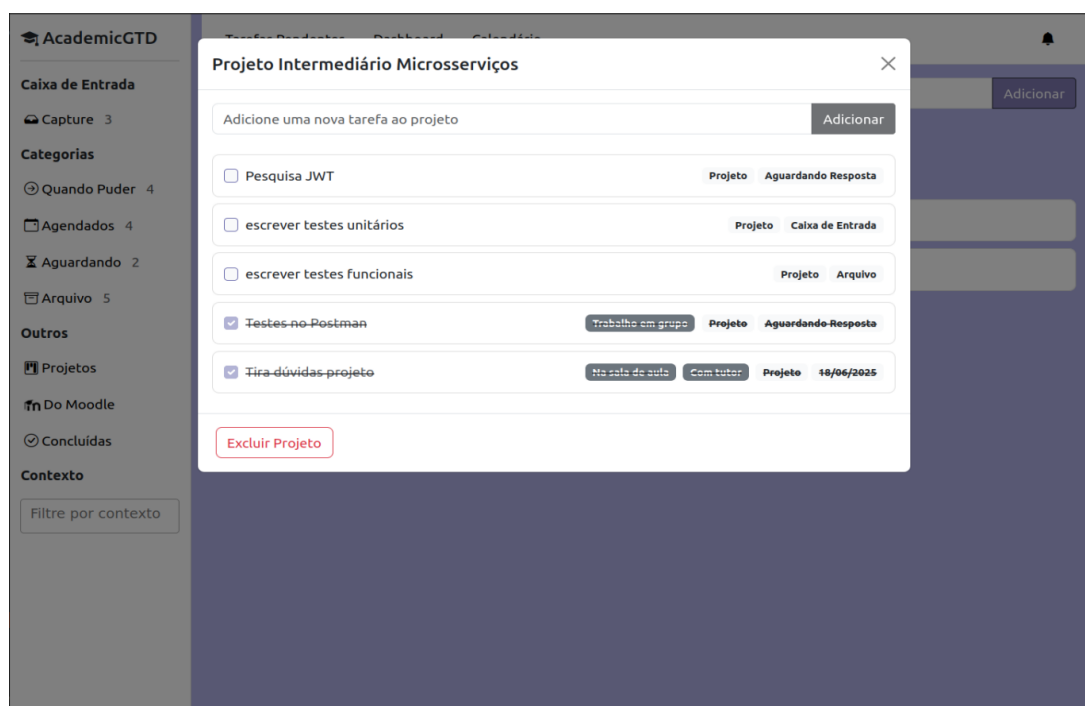
A Figura 12 mostra a tela inicial da funcionalidade, onde o sistema solicita ao usuário a autorização para sincronizar sua conta Moodle com base no e-mail já cadastrado no GTD, conforme definido no RF3.1. Após a autorização, o sistema utiliza um token de acesso à API do Moodle para identificar o ID do usuário e permitir futuras sincronizações (RF3.2).

A Figura 13 apresenta o estado da mesma tela após uma sincronização bem-sucedida. Nela, são exibidas as tarefas importadas do Moodle, todas não concluídas, em conformidade com o RF3.3. Essas tarefas se tornam parte do ambiente e podem ser classificadas normalmente de acordo com critérios do modelo GTD — como prioridade, contexto, prazo e delegação — atendendo ao RF3.4, e exibidas de forma visualmente destacada, conforme o RF3.9. O sistema também garante que tarefas duplicadas não sejam importadas (RF3.6) e realiza atualizações automáticas de status, sincronizando as tarefas concluídas no Moodle com o ambiente GTD (RF3.5). Além disso, está prevista a sincronização periódica em segundo plano, como descrito no RF3.7.

Por fim, a Figura 14 ilustra um cenário em que ocorre falha na comunicação com a API externa do Moodle. A interface exibe uma mensagem de erro clara, informando a indisponibilidade do serviço, em conformidade com o RF3.8, que exige tratamento seguro e compreensível de falhas externas.

3.4 Organização de listas de Projetos

Figura 15 – Tela de Projeto



Fonte: Elaborado pelo autor.

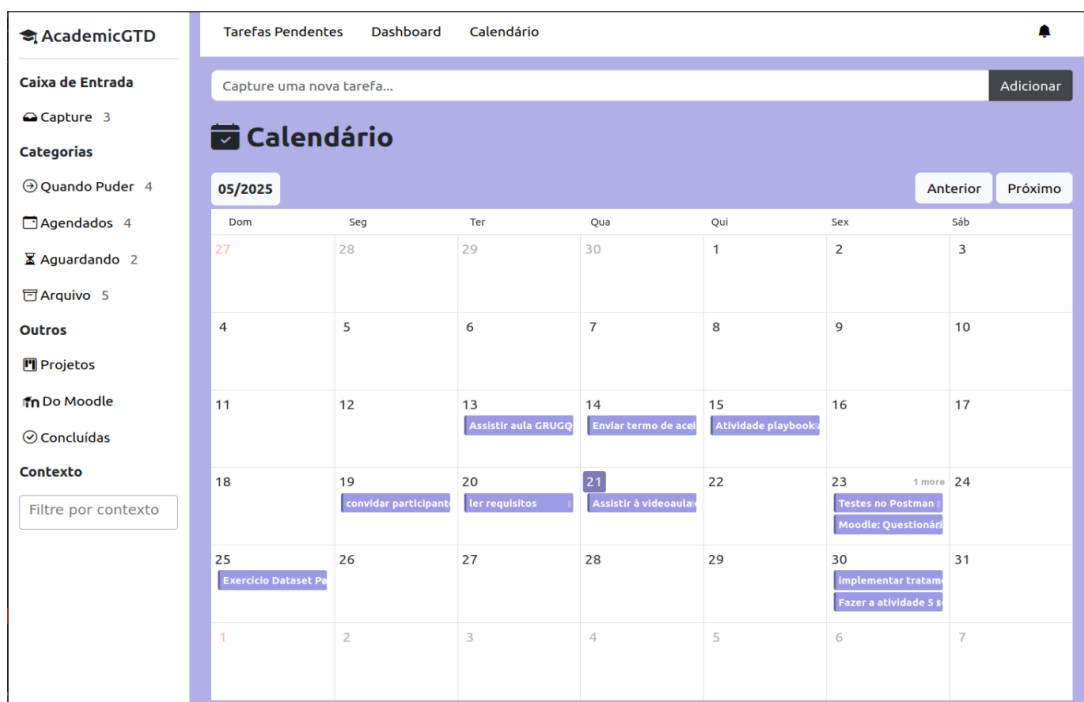
A Figura 15 apresenta o modal de gerenciamento de projetos. Ao clicar no botão "Projetos", localizado no menu lateral esquerdo, o sistema exibe uma lista de projetos. Selecionando um deles, o modal correspondente é aberto, exibindo informações detalhadas e permitindo sua edição, conforme previsto no RF4.2.

A parte superior do modal contém um campo de entrada que possibilita ao usuário adicionar uma nova tarefa diretamente vinculada ao projeto selecionado, implementando o RF4.4, que trata da associação de tarefas a projetos. Logo abaixo, é exibida uma lista de tarefas já associadas ao projeto, permitindo sua seleção para edição, conforme funcionalidades abordadas anteriormente.

No canto inferior esquerdo do modal, há um botão de exclusão de projeto, que aciona uma confirmação antes da execução da ação, em conformidade com o RF4.3. Todas as alterações realizadas — sejam de conteúdo do projeto ou de suas associações com tarefas — são refletidas de forma consistente no banco de dados, atendendo ao RF4.6. Quando todas as tarefas associadas a um projeto são marcadas como concluídas, o projeto também é marcado como concluído.

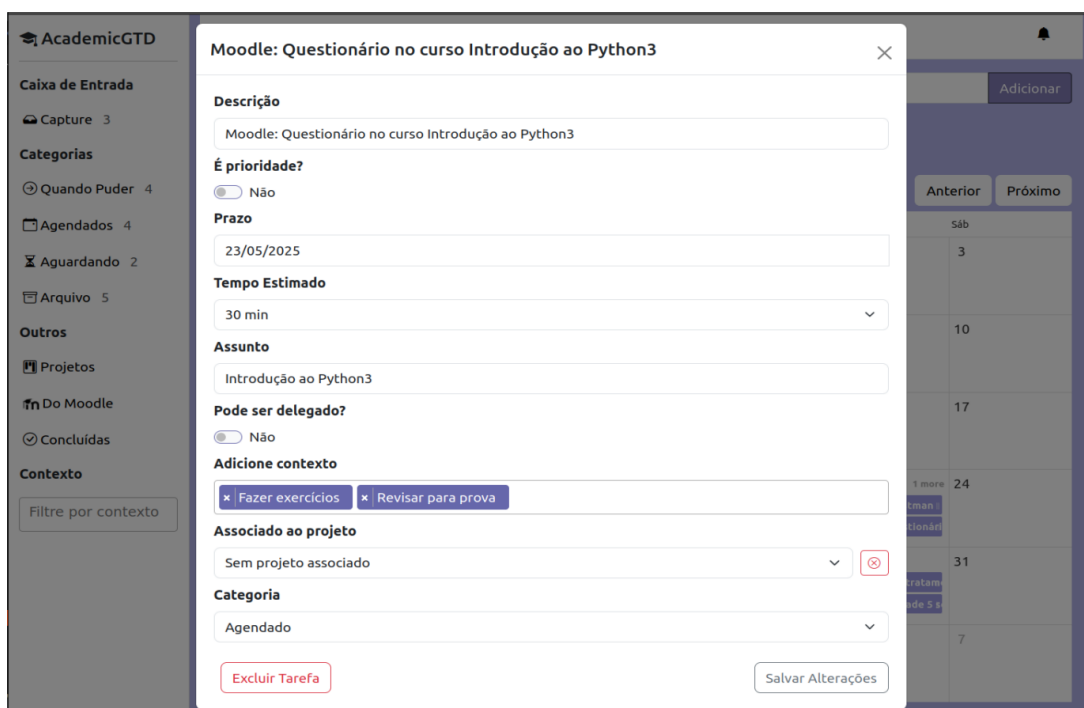
3.5 Visão de calendário mensal

Figura 16 – Visão em calendário



Fonte: Elaborado pelo autor.

Figura 17 – Edição de tarefa a partir da tela de calendário



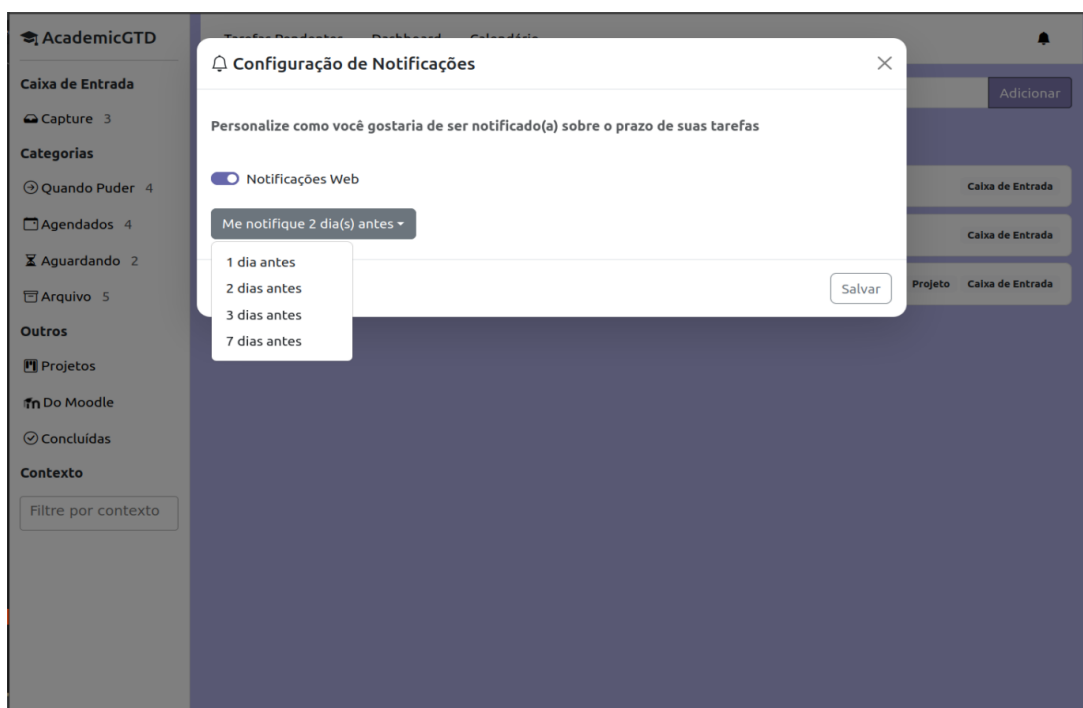
Fonte: Elaborado pelo autor.

As Figuras 16 e 17 apresentam as funcionalidades relacionadas à visualização e edição de tarefas por meio da interface de calendário. A Figura 16 implementa o RF5.1, disponibilizando uma visualização mensal em formato de calendário, e o RF5.2, exibindo em cada dia as tarefas que possuem data atribuída correspondente. A navegação entre meses, por meio dos botões de avanço e retorno, atende ao RF5.3.

A Figura 17 apresenta o modal de edição de tarefas aberto a partir do clique em uma tarefa no calendário, conforme o RF5.4. As alterações realizadas por essa interface são corretamente refletidas no banco de dados, conforme exigido pelo RF5.5.

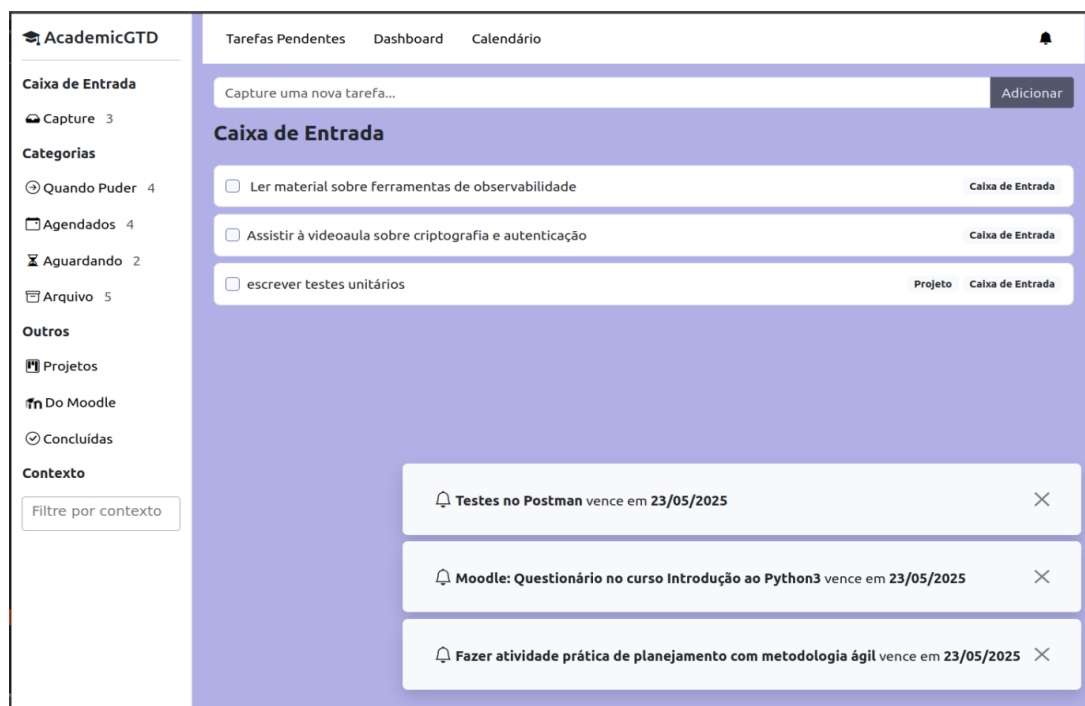
3.6 Notificações

Figura 18 – Tela de configuração de notificações



Fonte: Elaborado pelo autor.

Figura 19 – Tela de notificações após login



Fonte: Elaborado pelo autor.

As Figuras 18 e 19 ilustram as funcionalidades de configuração e exibição de notificações no sistema. A Figura 18 apresenta o modal de configuração, onde o usuário pode ativar ou desativar as notificações por meio de um switch, selecionar o período de antecedência desejado para o recebimento das notificações, e salvar suas preferências, atendendo ao RF6.2.

A Figura 19 mostra a tela de entrada padrão do sistema com notificações exibidas na forma de bootstrap toasts no canto inferior direito, conforme o RF6.3. Essas notificações são geradas com base nas tarefas que possuem prazo definido, em conformidade com o RF6.1.

3.7 Dashboard

Figura 20 – Tela com gráficos de atividade dos últimos 7 dias



Fonte: Elaborado pelo autor.

A Figura 20 apresenta o dashboard de relatórios que sintetiza as informações das tarefas concluídas nos últimos sete dias. O painel exibe quatro gráficos: um gráfico de barras mostrando a quantidade de tarefas por assunto; um gráfico de anel ilustrando a distribuição por prioridade; um gráfico de setores representando o tempo estimado das tarefas; e um gráfico de linha demonstrando a evolução diária das tarefas concluídas. Além disso, é exibida a média diária de tarefas concluídas no período, em formato numérico com duas casas decimais.

Essa visualização atende aos requisitos RF6.4, RF6.5, RF6.6 e RF6.7, ao gerar automaticamente um relatório com dados estatísticos, médias e gráficos de desempenho detalhados. O sistema assegura ainda a consistência dos dados apresentados, conforme o RF6.8, refletindo corretamente o histórico armazenado no banco de dados.

4 CONCLUSÕES

O desenvolvimento da aplicação web proposta teve como base a metodologia GTD (Getting Things Done), visando oferecer aos usuários uma plataforma voltada à organização de suas atividades acadêmicas com foco em produtividade. Ao longo do projeto, foi possível implementar uma solução que contempla desde o cadastro de tarefas e projetos até sua visualização em diferentes formatos como listas, segmentações contextuais e visão em calendário. A integração com o Moodle, a emissão de notificações automáticas e a apresentação de gráficos de desempenho complementam o escopo proposto, reforçando o alinhamento com os objetivos iniciais.

Durante a construção do sistema, os objetivos específicos foram tratados de maneira incremental, com decisões técnicas e de arquitetura sendo validadas ao longo da implementação. O estudo sobre a metodologia GTD mostrou-se essencial para adequar os fluxos de tarefas à realidade do público-alvo, exigindo flexibilidade na modelagem das entidades e nas interações com o usuário.

No decorrer do projeto, destacaram-se alguns desafios significativos, entre eles a implementação de autenticação via Spring Security, que demandou um estudo aprofundado sobre tokens e chamadas seguras devido a complexidade do módulo. Ademais, a adaptação da metodologia GTD para um sistema digital também exigiu múltiplos refinamentos, principalmente na integração entre o front-end e a API, que embora prevista como uma etapa direta, apresentou complexidade adicional devido a diferenças entre o comportamento esperado dos componentes do lado cliente e o fluxo real de dados, exigindo refatorações pontuais e ajustes.

Como lição aprendida, destaca-se a importância de considerar, desde o início, a usabilidade como parte da arquitetura do sistema, principalmente na integração entre front-end e back-end. As dificuldades enfrentadas nessa comunicação evidenciaram que a experiência do usuário está diretamente ligada ao modo como os dados são trafegados, processados e exibidos, exigindo atenção ao consumo assíncrono, tratamento de erros e coerência entre camadas. Além disso, a implementação de autenticação com Spring Security permitiu consolidar conhecimentos sobre segurança em aplicações web, ressaltando a importância da correta configuração de fluxos de autenticação, uso de JWT e proteção de recursos sensíveis. Essas experiências reforçam a necessidade de uma abordagem integrada entre usabilidade, segurança e arquitetura durante todo o ciclo de desenvolvimento.

Como trabalhos futuros, recomenda-se a ampliação da aplicação para contemplar recursos colaborativos entre usuários, integração com outras plataformas

acadêmicas além do Moodle, suporte a dispositivos móveis com funcionalidades específicas, e a realização de testes com usuários em ambientes reais de uso para avaliar a usabilidade e o impacto da ferramenta na produtividade acadêmica.

REFERÊNCIAS

- APACHE, M. P. **Introduction**. 2025. Disponível em: <<https://maven.apache.org/what-is-maven.html>>. 16
- ATLASSIAN. **The gtd approach to maximizing productivity with Trello**. 2017. Disponível em: <<https://www.atlassian.com/blog/productivity/gtd-getting-things-done-maximizing-productivity-trello>>. 7
- AUSTIN, D. **Ser organizado pode melhorar a sua saúde mental – saiba como!** 2024. Disponível em: <<https://www.nationalgeographicbrasil.com/ciencia/2024/01/ser-organizado-pode-melhorar-a-sua-saude-mental-saiba-como>>. 8
- AUTH0. **Introduction to JSON Web Tokens**. 2025. Disponível em: <<https://jwt.io/introduction>>. 16
- CHARTJS.ORG. **Chart.js**. 2025. Disponível em: <<https://www.chartjs.org/docs/latest/>>. 18
- CORTES, A. **IDE Eclipse: o que é e sua importância para desenvolvedores**. 2023. Disponível em: <<https://www.remessaonline.com.br/blog/ide-eclipse/>>. 14
- DEVMEDIA. **Persistência com Spring Data JPA**. 2025. Disponível em: <<https://www.devmedia.com.br/persistencia-com-spring-data-jpa/24390>>. 15
- DW. **Procrastinação: por que deixamos para depois?** 2024. Disponível em: <<https://g1.globo.com/saude/noticia/2024/03/19/procrastinacao-por-que-deixamos-para-depois.ghtml>>. 7
- FERNANDA, C.; LOUZADA, V. **O que é Git e Github: os primeiros passos nessas ferramentas**. 2024. Disponível em: <https://www.alura.com.br/artigos/o-que-e-git-github?srsId=AfmBOool1x_C-ZHyq414-Oz7EehAymfvOSXUvSV2zxSvjAojiyizcLXw>. 14
- FERREIRA, W. O.; KNOP, I. O. Estruturação de aplicações distribuídas com a arquitetura rest. **Caderno de Estudos em Sistemas de Informação**, v. 3, 2016. 12
- FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**. Tese (Doutorado) — University of California, 2000. 12
- GEEKS, G. F. **Introduction to Postman for API Development**. 2025. Disponível em: <<https://www.geeksforgeeks.org/introduction-postman-api-development/>>. 15
- GODINHO, T. **Como usar GTD em 11 minutos**. 2017. Disponível em: <<https://www.youtube.com/watch?v=OYQfpQTer2A>>. 10
- IBM. **O que é Java Spring Boot?** 2025. Disponível em: <<https://www.ibm.com/br-pt/topics/java-spring-boot>>. 15
- INEP. **Ensino a distância cresce 474% em uma década**. 2022. Disponível em: <<https://www.gov.br/inep/pt-br/assuntos/noticias/censo-da-educacao-superior/ensino-a-distancia-cresce-474-em-uma-decada>>. 7

- LIMA, G. **Bootstrap: O que é, Documentação, como e quando usar**. 2023. Disponível em: <https://www.alura.com.br/artigos/bootstrap?srsId=AfmBOopoXCukcKbpKanqSk4GrM2kliLxw6bzQLhOFKqoAY8iWJd_1fwu>. 17
- MELO, D. **O que é Java? [Guia para iniciantes]**. 2021. Disponível em: <<https://tecnoblog.net/responde/o-que-e-java-guia-para-iniciantes/>>. 15
- MOODLE, H. **About Moodle**. 2024. Disponível em: <https://docs.moodle.org/500/en/About_Moodle>. 18
- MOZILLA. **Cookies HTTP**. 2025. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Guides/Cookies>>. 16
- MYSQL. **MySQL Workbench**. 2024. Disponível em: <<https://www.mysql.com/products/workbench/>>. 17
- ORACLE. **O que é o MySQL?** 2024. Disponível em: <<https://www.oracle.com/br/mysql/what-is-mysql/>>. 17
- PANDE, A. K. **jQuery 2 Recipes: A Problem-Solution Approach**. [S.l.]: Apress, 2014. 18
- PINUSA, S. **Jovens que estudam e trabalham falam sobre a rotina para conciliar a escola e o emprego: 'A gente precisa fazer esse sacrifício'**. 2023. Disponível em: <<https://g1.globo.com/ce/ceara/educacao/noticia/2023/11/17/jovens-que-estudam-e-trabalham-falam-sobre-a-rotina-para-conciliar-a-escola-e-o-emprego-a-gente-precisa-fazer-esse-sacrificio.ghtml>>. 7
- SANTOS, J. S. d. et al. Analysis of tools for rest contract specification in swagger/openapi. In: **Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 2**. [S.l.: s.n.], 2020. p. 201–208. 16
- SERICATI, K. **What is Flyway? Everything you need to know to get started with Flyway**. 2021. Disponível em: <<https://medium.com/@astontechnologies/what-is-flyway-5199d2278a06>>. 17
- SOMMERVILLE, I. **Engenharia de Software**. [S.l.]: 6 Edição. São Paulo. Editora Pearson Prentice Hall, 2011. 13
- STACKOVERFLOW. **Most Popular Technologies 2024**. 2024. Disponível em: <<https://survey.stackoverflow.co/2024/technology#most-popular-technologies-language>>. 15, 17
- SUSNJARA, S.; SMALLEY, I. **What is Docker?** 2025. Disponível em: <<https://www.ibm.com/think/topics/docker>>. 19
- SÁ, M. A. D. et al. O método getting things done (gtd) e as ferramentas de gerenciamento de tempo e produtividade. **NAVUS - Revista de Gestão e Tecnologia**, v. 8, p. 72–87, 2018. 7, 8, 9, 10
- UI, T. **Toast UI Calendar**. 2025. Disponível em: <<https://ui.toast.com/>>. 18

VALENTE, M. T. **Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade**. [S.l.]: Editora: Independente, 2020. 13, 14

Apêndices

APÊNDICE A – MODELO DE REQUISITOS DO SISTEMA E DIAGRAMA DE CASOS DE USO

A.1 Épico 1 – Login e Autenticação

O sistema deve permitir que os usuários se autentiquem utilizando credenciais próprias, garantindo acesso individualizado a seus perfis de tarefas e quadros de organização pessoal.

Histórias de usuário seguindo o modelo de Cohn

- Eu, como usuário do sistema, gostaria de me autenticar utilizando meu e-mail e senha, a fim de acessar de forma segura minha área pessoal na aplicação.

Contexto

Para garantir a segurança e a privacidade dos dados, o sistema deverá exigir autenticação por meio de e-mail e senha. Cada usuário será responsável por definir sua própria senha no momento do cadastro. O acesso à área pessoal estará condicionado à autenticação válida.

Critérios de aceitação de cada história

Ao iniciar o aplicativo, deverá ser exibida uma tela de login contendo um formulário com os seguintes elementos:

- Campo para e-mail do usuário;
- Campo para senha;
- Opção “Esqueci minha senha” para recuperação de acesso;
- Opção “Cadastre-se” para novos usuários.

O sistema deverá contar com um banco de dados contendo as informações de autenticação, garantindo a criptografia das senhas conforme boas práticas de segurança.

Definição de pronto

- Tela de login implementada, com conformidade aos princípios de acessibilidade;
- Banco de dados funcional, com senhas armazenadas de forma criptografada e protegida segundo padrões de segurança.

Requisitos funcionais

- **RF1.1:** O sistema deve exibir uma tela de login com campos para e-mail e senha ao ser iniciado.
- **RF1.2:** O sistema deve validar as credenciais fornecidas e permitir o acesso apenas se forem válidas.
- **RF1.3:** O sistema deve exibir mensagem de erro caso o e-mail ou a senha estejam incorretos.
- **RF1.4:** O sistema deve permitir que o usuário recupere sua senha por meio da opção “Esqueci minha senha”.
- **RF1.5:** O sistema deve disponibilizar a opção “Cadastre-se” para novos usuários.
- **RF1.6:** O sistema deve armazenar senhas de forma criptografada no banco de dados.

A.2 Épico 2 – Modificação de Atividades

O sistema deve permitir o cadastro, visualização, edição e exclusão de tarefas, possibilitando ao usuário controle sobre suas atividades de forma estruturada e bem segmentada.

Histórias de usuário seguindo o modelo de Cohn

- Eu, como usuário do sistema, gostaria de cadastrar todas as tarefas relacionadas aos meus estudos, a fim de mantê-las organizadas em uma plataforma confiável.
- Eu, como usuário do sistema, gostaria de visualizar minhas tarefas de maneira estruturada, a fim de obter controle visual sobre os próximos compromissos.
- Eu, como usuário do sistema, gostaria de filtrar minhas tarefas com base em rótulos previamente definidos, a fim de facilitar a organização e o planejamento das atividades.

- Eu, como usuário do sistema, gostaria de editar ou excluir tarefas já criadas, atualizando suas informações ou removendo-as, a fim de manter meu planejamento alinhado com minha rotina de estudos.

Contexto

O usuário deve ter acesso a um ambiente que permita o registro e a atualização de tarefas, bem como sua visualização segmentada por categorias. O sistema deve implementar mecanismos de organização baseados na metodologia Getting Things Done (GTD), possibilitando a classificação automática das tarefas conforme os dados fornecidos pelo usuário.

Critérios de aceitação de cada história

- Tela contendo uma caixa de entrada, onde o usuário poderá registrar novas tarefas de forma rápida para posterior classificação;
- Funcionalidade de categorização das tarefas com base em parâmetros como prioridade, prazo, assunto e delegação, gerando listas organizadas conforme os critérios definidos;
- Sistema de filtragem que permite exibir tarefas com base em rótulos previamente definidos de contexto, definindo o tipo de ação;
- Funcionalidades de edição e exclusão de tarefas, incluindo a opção de marcar tarefas como concluídas.

Definição de pronto

- Telas finalizadas, acessíveis e em conformidade com os requisitos funcionais;
- Banco de dados operacional, suportando operações de criação, leitura, atualização e exclusão de tarefas (CRUD).

Requisitos funcionais

- **RF2.1:** O sistema deve permitir o cadastro de novas tarefas por meio de uma caixa de entrada acessível ao usuário.
- **RF2.2:** O sistema deve permitir a visualização de todas as tarefas cadastradas em uma interface estruturada.

- **RF2.3:** O sistema deve possibilitar a edição das informações de uma tarefa previamente cadastrada.
- **RF2.4:** O sistema deve permitir a exclusão de tarefas, com opção de confirmação por parte do usuário.
- **RF2.5:** O sistema deve permitir marcar tarefas como concluídas.
- **RF2.6:** O sistema deve oferecer funcionalidade de categorização automática das tarefas com base nos dados inseridos pelo usuário (como prioridade, prazo, assunto e delegação).
- **RF2.7:** O sistema deve permitir que o usuário associe rótulos de contexto às tarefas cadastradas.
- **RF2.8:** O sistema deve permitir a filtragem das tarefas com base nos rótulos previamente definidos.
- **RF2.9:** O sistema deve manter um banco de dados capaz de executar operações de criação, leitura, atualização e exclusão (CRUD) de tarefas.

A.3 Épico 3 – Integração com Moodle

O sistema deve permitir a integração com a plataforma educacional Moodle para importar automaticamente tarefas acadêmicas vinculadas aos usuários, mantendo-as sincronizadas com o sistema GTD e otimizando o processo de organização de estudos.

Histórias de usuário seguindo o modelo de Cohn

- Eu, como usuário, gostaria de conectar minha conta Moodle ao sistema GTD, a fim de visualizar automaticamente minhas atividades acadêmicas no ambiente de organização.
- Eu, como usuário, gostaria que minhas atividades do Moodle fossem importadas como tarefas classificáveis, a fim de integrá-las ao meu fluxo GTD.
- Eu, como usuário, gostaria que as tarefas concluídas no Moodle fossem marcadas automaticamente como concluídas no sistema, a fim de manter a consistência e evitar retrabalho.

Contexto

A integração com o Moodle deve permitir que o sistema GTD acesse atividades acadêmicas do usuário por meio de sua conta institucional. O serviço busca, filtra

e transforma atividades pendentes do Moodle em tarefas GTD, garantindo que não sejam duplicadas e estejam disponíveis para classificação. Além disso, atividades já concluídas no Moodle devem refletir automaticamente no status da tarefa no GTD. A sincronização é realizada periodicamente e deve ser tolerante a falhas na comunicação com a API externa.

CrITÉRIOS de aceitação de cada história

- O sistema deve buscar o usuário no Moodle com base no e-mail cadastrado no GTD;
- Atividades não concluídas no Moodle devem ser importadas como novas tarefas;
- Tarefas concluídas no Moodle devem atualizar automaticamente o status da respectiva tarefa no GTD;
- A sincronização deve evitar a duplicação de tarefas;
- Deve existir uma interface que exiba claramente as tarefas sincronizadas com origem externa.

Definição de pronto

- Integração funcional com autenticação via token de API fornecido pelo Moodle;
- Sincronização robusta, resiliente a falhas de comunicação com a API externa;
- Dados das tarefas importadas corretamente refletidos no sistema, com status sincronizado;
- Interface dedicada às tarefas integradas e garantia de não duplicação de registros.

Requisitos funcionais

- **RF3.1:** O sistema deve permitir que o usuário conecte sua conta Moodle por meio de autenticação via token.
- **RF3.2:** O sistema deve buscar automaticamente atividades acadêmicas vinculadas ao e-mail do usuário no Moodle.
- **RF3.3:** O sistema deve importar tarefas não concluídas do Moodle como novas tarefas no ambiente GTD.
- **RF3.4:** O sistema deve classificar as tarefas importadas de forma compatível com o modelo GTD.

- **RF3.5:** O sistema deve atualizar automaticamente o status de uma tarefa no GTD quando ela for concluída no Moodle.
- **RF3.6:** O sistema deve evitar a duplicação de tarefas importadas durante o processo de sincronização.
- **RF3.7:** O sistema deve realizar a sincronização periódica com o Moodle, mesmo em segundo plano.
- **RF3.8:** O sistema deve ser resiliente a falhas na comunicação com a API externa do Moodle, garantindo que erros sejam tratados de forma segura e clara ao usuário.
- **RF3.9:** O sistema deve disponibilizar uma interface dedicada que permita ao usuário visualizar claramente quais tarefas têm origem no Moodle.

A.4 Épico 4 – Gestão de Projetos

O sistema deverá permitir a criação, edição e exclusão de projetos, viabilizando o agrupamento de tarefas relacionadas sob uma mesma iniciativa, com o objetivo de melhorar a organização, o acompanhamento e a contextualização das atividades do usuário.

Histórias de usuário seguindo o modelo de Cohn

- Eu, como usuário, gostaria de criar e editar projetos, a fim de agrupar tarefas com temáticas ou objetivos semelhantes.

Contexto

O usuário deve ser capaz de estruturar projetos como contêineres lógicos para múltiplas tarefas. Cada projeto representa uma área temática, um objetivo específico ou uma iniciativa contínua, permitindo a associação de tarefas relacionadas para facilitar o planejamento e a execução.

Critérios de aceitação de cada história

- Funcionalidade que permita a criação, edição e exclusão de projetos;
- Capacidade de associar e desassociar tarefas a projetos, conforme critério do usuário.

Definição de pronto

- Telas concluídas, com navegação acessível e responsiva;
- Criação, edição e exclusão de projetos funcionando conforme especificado;
- Associação de tarefas a projetos executada corretamente e refletida no banco de dados.

Requisitos funcionais

- **RF4.1:** O sistema deve permitir a criação de projetos pelo usuário.
- **RF4.2:** O sistema deve permitir a edição das informações de um projeto existente.
- **RF4.3:** O sistema deve permitir a exclusão de projetos, com confirmação da ação pelo usuário.
- **RF4.4:** O sistema deve permitir a associação de tarefas a um projeto no momento do cadastro ou posteriormente.
- **RF4.5:** O sistema deve permitir a desassociação de tarefas de um projeto a qualquer momento.
- **RF4.6:** O sistema deve refletir corretamente no banco de dados as alterações feitas nos projetos e suas associações com tarefas.

A.5 Épico 5 – Visualização de Tarefas no Calendário

O sistema deverá disponibilizar uma interface em formato de calendário, permitindo ao usuário visualizar as tarefas com data definida ao longo do tempo, facilitando o planejamento e a gestão de prazos.

Histórias de usuário seguindo o modelo de Cohn

- Eu, como usuário, gostaria de visualizar em formato de calendário as tarefas com data atribuída, a fim de otimizar minha organização e acompanhar prazos de forma clara.

Contexto

O sistema deve fornecer uma visualização mensal em formato de calendário, exibindo todas as tarefas com datas atribuídas. A partir dessa visualização, o usuário

poderá navegar entre os meses e acessar opções de edição das tarefas diretamente na interface do calendário.

Critérios de aceitação de cada história

- Funcionalidade que apresenta um calendário mensal com a listagem visual de todas as tarefas agendadas;
- Permitir a navegação entre diferentes meses;
- Habilitar a edição das tarefas diretamente a partir do calendário.

Definição de pronto

- Telas finalizadas, acessíveis e responsivas;
- Tarefas exibidas corretamente no calendário de acordo com suas datas atribuídas;
- Funcionalidade de edição integrada à visualização do calendário.

Requisitos funcionais

- **RF5.1:** O sistema deve disponibilizar uma visualização mensal em formato de calendário.
- **RF5.2:** O sistema deve exibir, em cada dia do calendário, todas as tarefas que possuem data atribuída correspondente.
- **RF5.3:** O sistema deve permitir a navegação entre diferentes meses no calendário.
- **RF5.4:** O sistema deve permitir a edição das informações de uma tarefa diretamente a partir da interface do calendário.
- **RF5.5:** O sistema deve garantir que alterações feitas nas tarefas via calendário sejam refletidas corretamente no banco de dados.

A.6 Épico 6 – Relatório e Notificação

O sistema deverá notificar os usuários sobre tarefas com prazos definidos e disponibilizar relatórios semanais com dados estatísticos sobre a conclusão de atividades, com o objetivo de auxiliar no acompanhamento do desempenho e manutenção do ritmo de estudos.

Histórias de usuário seguindo o modelo de Cohn

- Eu, como usuário, gostaria de receber notificações para as tarefas com prazo definido, a fim de manter-me em dia com minhas entregas.
- Eu, como usuário, gostaria de acessar um relatório com dados relevantes sobre o andamento dos meus estudos, a fim de me manter motivado e continuar progredindo.

Contexto

O sistema deve ser capaz de identificar tarefas com prazos definidos e emitir notificações ao usuário conforme as preferências configuradas. Além disso, deverá gerar automaticamente um relatório com a média de tarefas concluídas nos últimos sete dias, incluindo indicadores visuais e segmentações por categoria.

Critérios de aceitação de cada história

- O sistema deve exibir notificações ao usuário no momento do login, desde que a tarefa possua prazo definido e o usuário tenha habilitado a opção de ser notificado com antecedência de 1, 2, 3 dias ou uma semana;
- O relatório deve ser gerado com base nas tarefas marcadas como concluídas, considerando os últimos sete dias, e apresentar:
 - Média semanal de tarefas concluídas;
 - Gráficos de desempenho;
 - Distribuição das tarefas concluídas por rótulo de assunto, prioridade e tempo estimado.

Definição de pronto

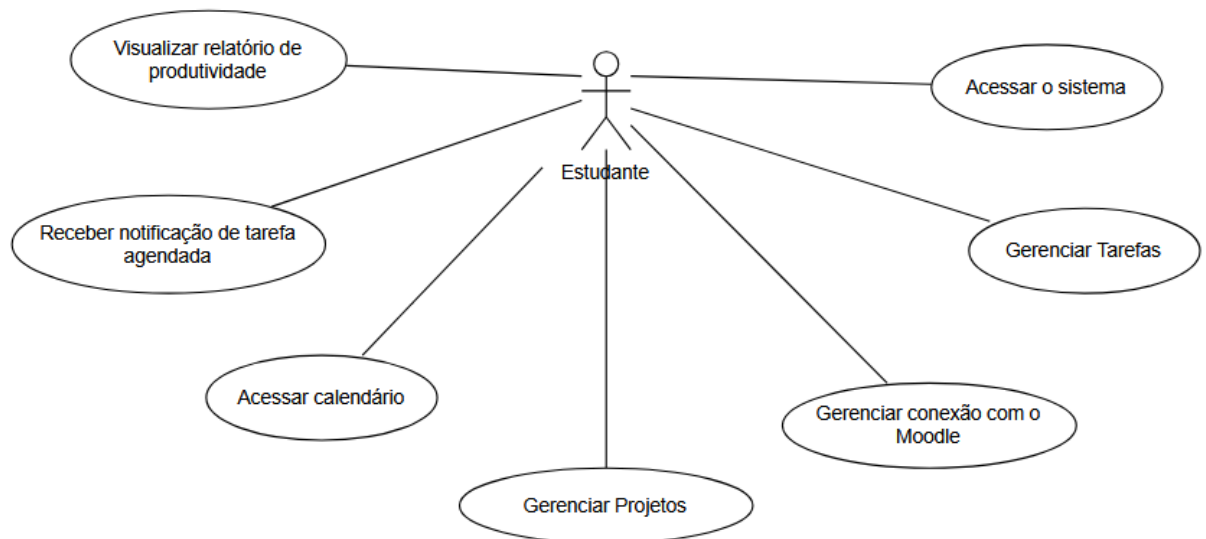
- Telas finalizadas, acessíveis e em conformidade com os requisitos de usabilidade;
- Notificações funcionais emitidas conforme configurações do usuário;
- Relatório gerado corretamente com dados consistentes e visualizações adequadas.

Requisitos funcionais

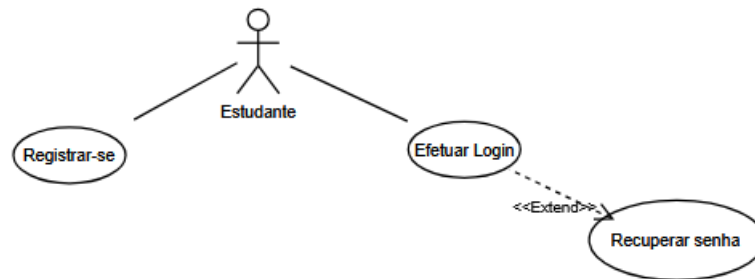
- **RF6.1:** O sistema deve identificar tarefas com prazo definido e armazenar essa informação para uso em notificações.

- **RF6.2:** O sistema deve permitir que o usuário configure preferências de notificação para tarefas com prazo, incluindo opções de antecedência (1, 2, 3 dias ou 1 semana).
- **RF6.3:** O sistema deve exibir notificações no momento do login, conforme as preferências do usuário e os prazos das tarefas.
- **RF6.4:** O sistema deve gerar automaticamente um relatório baseado nas tarefas concluídas nos últimos sete dias.
- **RF6.5:** O relatório deve apresentar a média de tarefas concluídas na semana.
- **RF6.6:** O relatório deve conter gráficos de desempenho visualizando o progresso do usuário.
- **RF6.7:** O relatório deve exibir a distribuição das tarefas concluídas por rótulo de assunto, prioridade e tempo estimado.
- **RF6.8:** O sistema deve garantir a consistência dos dados exibidos no relatório com o histórico de tarefas registradas no banco de dados.

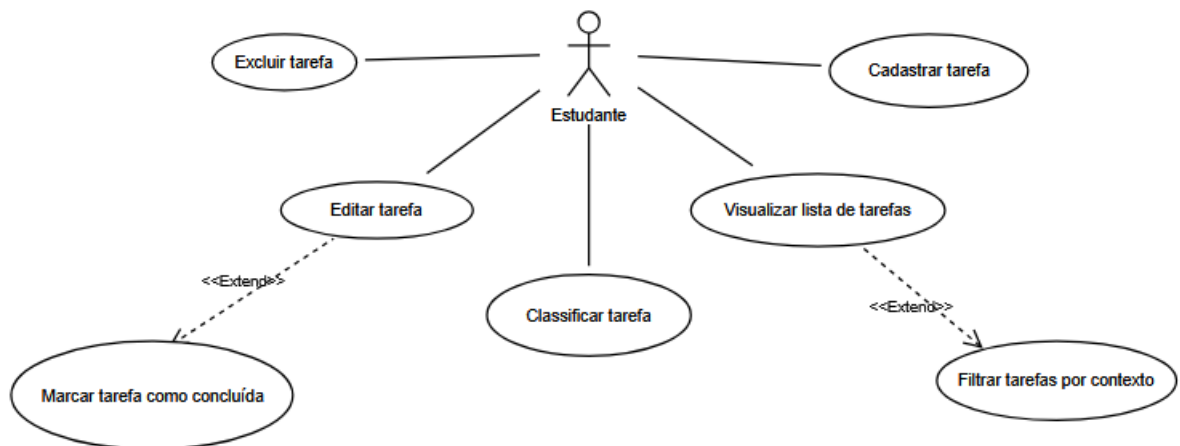
A.7 Diagramas de Casos de Uso da UML



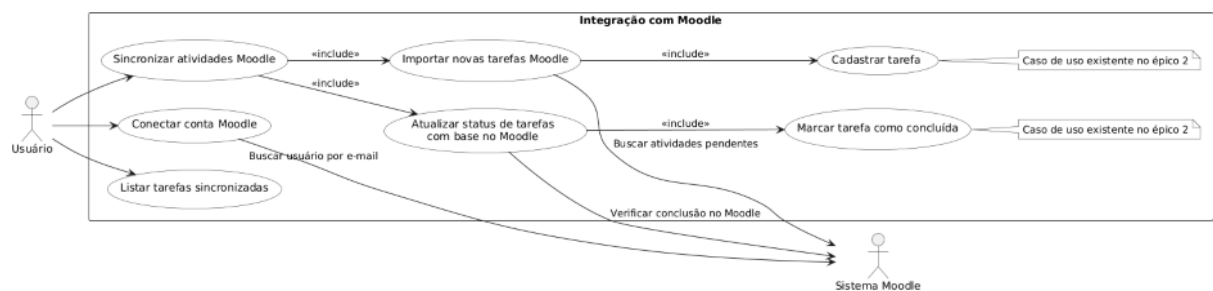
Casos de Uso Épico 1 – Login e Autenticação



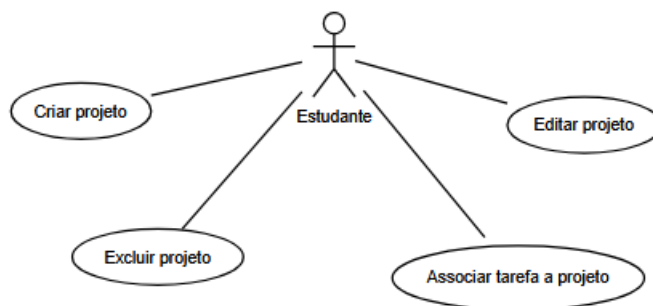
Casos de Uso Épico 2 – Modificação de Atividades



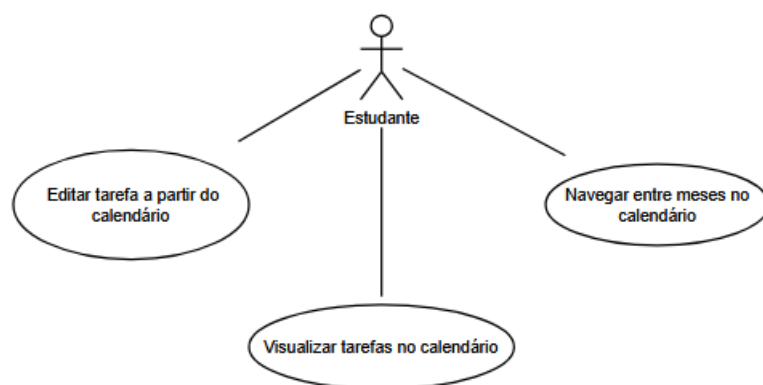
Casos de Uso Épico 3 – Integração com Moodle



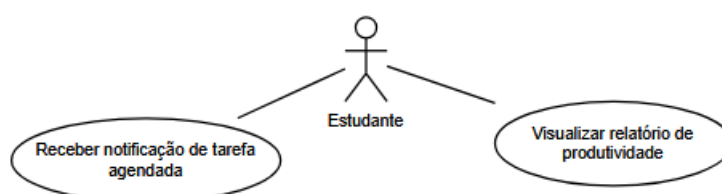
Casos de Uso Épico 4 – Gestão de Projetos



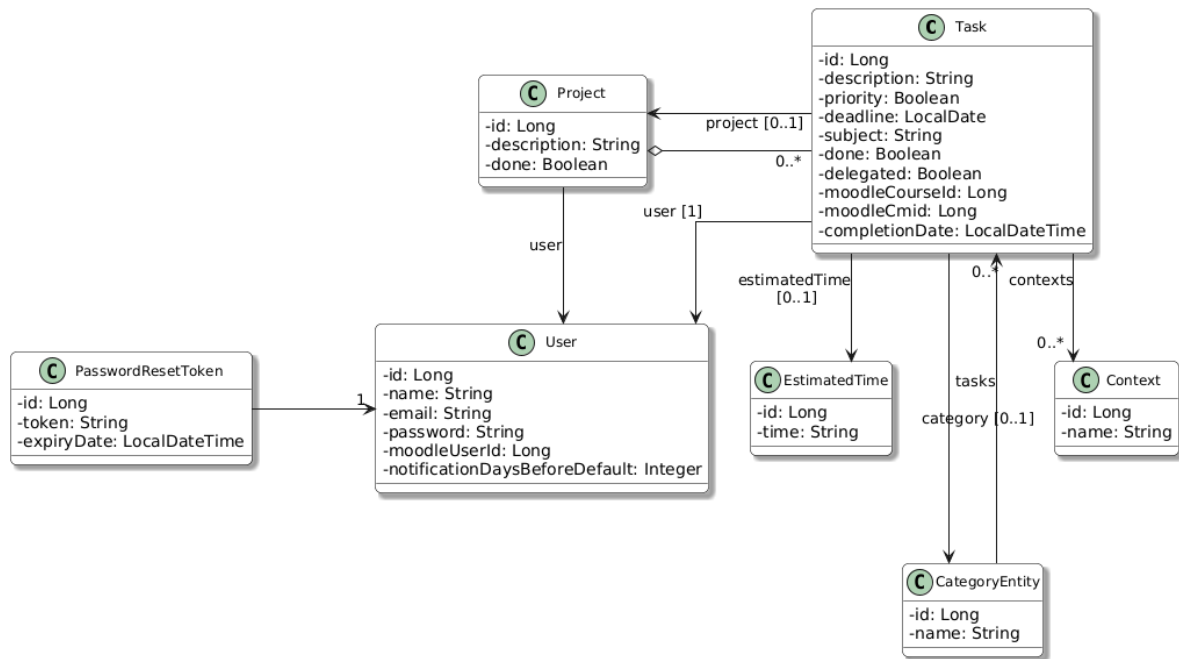
Casos de Uso Épico 5 – Visualização de Tarefas no Calendário



Casos de Uso Épico 6 – Relatório e Notificação

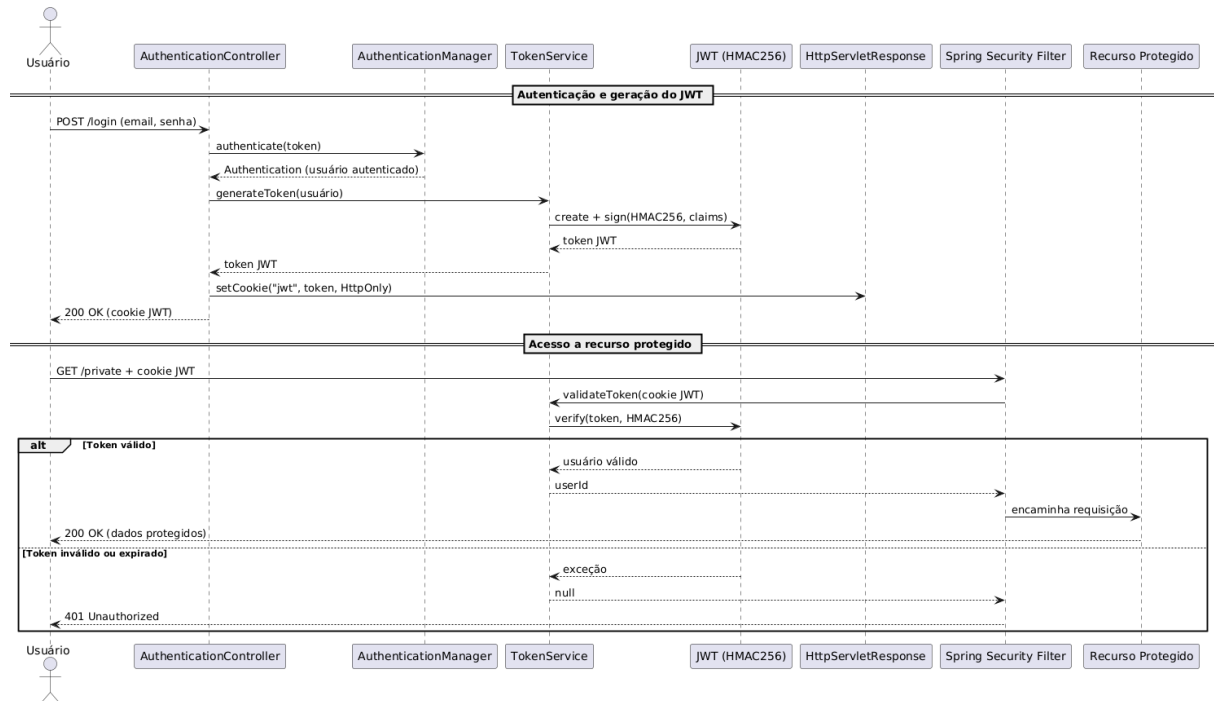


APÊNDICE B – MODELO ESTRUTURAL DO SISTEMA: DIAGRAMA DE CLASSES DA UML

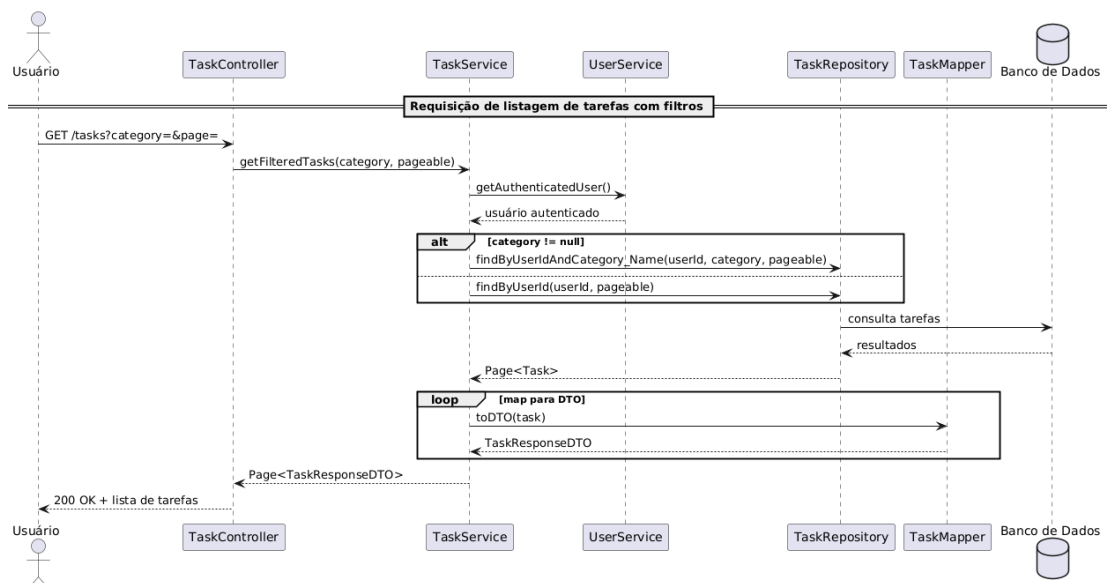


APÊNDICE C – MODELO COMPORTAMENTAL DO SISTEMA: DIAGRAMA DE SEQUÊNCIA DA UML

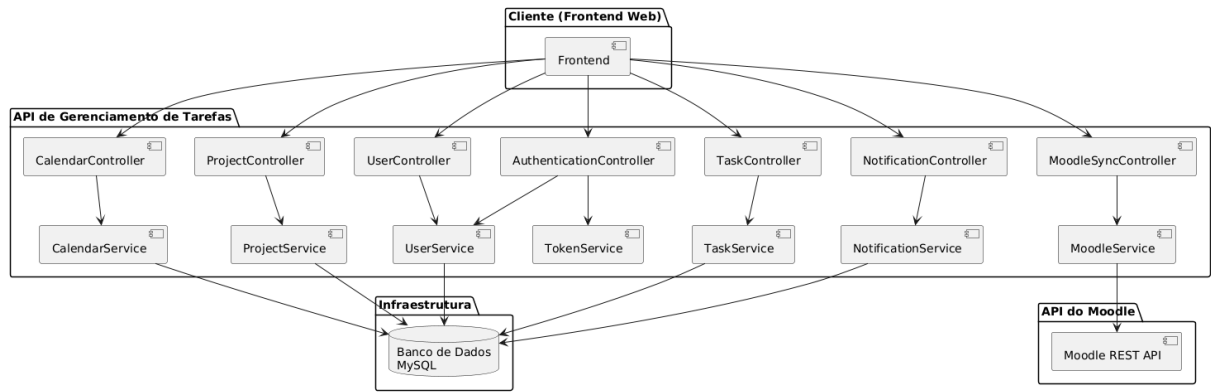
C.1 Login e Autorização com JWT



C.2 Requisição de tarefas



APÊNDICE D – MODELO DA ARQUITETURA DO SISTEMA: DIAGRAMA DE COMPONENTES DA UML



APÊNDICE E – MODELO DE DADOS DO SISTEMA: DIAGRAMA ENTIDADE-RELACIONAMENTO

