UNIVERSIDADE PAULISTA

ICET - INSTITUTO DE CIÊNCIAS EXATAS E TECNOLOGIA

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROJETO INTEGRADO MULTIDISCIPLINAR PIM II

Desenvolvimento de um Sistema Acadêmico Colaborativo com Apoio de IA

Nome	**R.A**
Camille Vitória dos Reis Silva	H665442
Fernanda Leticia do Prado Andrade	R8674A8
Keyla de Souza Fróes	F362FI3
Nicole Aparecida Gomes	H610010
Nicole Grassini Gonçalves	H6611J0
Sabrina Helen dos Santos Reis Ferreira	R263JE1

SÃO JOSÉ DOS CAMPOS – SP

NOVEMBRO / 2025

	RA
Camille Vitória dos Reis Silva	H665442
Fernanda Leticia do Prado Andrade	R8674A8
Keyla de Souza Fróes	F362FI3
Nicole Aparecida Gomes	H610010
Nicole Grassini Gonçalves	H6611J0
Sabrina Helen dos Santos Reis Ferreira	R263JE1

Desenvolvimento de um Sistema Acadêmico Colaborativo com Apoio de IA

Projeto Integrado Multidisciplinar (PIM) desenvolvido como exigência parcial dos requisitos obrigatórios à aprovação semestral no Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da UNIP (Universidade Paulista), orientado pelo corpo docente do curso.

São José dos Campos – SP Novembro / 2025

RESUMO

Este trabalho tem como objetivo automatizar os processos entre alunos e a instituição educacional por meio de um Sistema Acadêmico Colaborativo apoiado por Inteligência Artificial (IA). A proposta busca eliminar procedimentos manuais e suscetíveis a erros, promovendo maior eficiência administrativa e contribuindo para a preservação ambiental, com a redução do consumo de papel e de recursos naturais. Para o desenvolvimento do sistema, foi aplicada a metodologia da engenharia de software, fundamentada em seus princípios e boas práticas. A coleta de dados ocorreu por meio da observação de processos administrativos em instituições de ensino. O projeto foi estruturado em etapas, desde o levantamento de requisitos até a implementação do software, desenvolvido em linguagem C para o registro de arquivos em formato CSV e com interface construída em Python. Os resultados foram analisados buscando identificar padrões de uso do sistema e a diminuição do consumo de matéria-prima. Conclui-se que o sistema contribui para a redução de custos institucionais e para a conscientização ambiental de alunos e colaboradores.

Palavras-Chave: Automação; Preservação ambiental; Educacional; Redução de consumo.

Sumário

1. Introdução

O avanço das tecnologias digitais tem transformado a maneira como as instituições de ensino organizam e gerenciam suas informações. Apesar da modernização crescente, muitas ainda enfrentam dificuldades para controlar dados de alunos e atividades de forma integrada, utilizando planilhas e registros manuais que comprometem a eficiência e a segurança das informações. Esse cenário evidencia a necessidade de ferramentas tecnológicas que centralizem os processos administrativos e pedagógicos, promovendo maior agilidade, precisão e colaboração no ambiente escolar.

Com base nessa realidade, este projeto propõe o desenvolvimento de um Sistema Acadêmico Colaborativo com apoio de Inteligência Artificial (IA). A plataforma visa unificar o gerenciamento de aulas e atividades em um ambiente digital interativo e acessível, substituindo métodos manuais por processos automatizados. Além disso, o uso da IA como ferramenta de apoio técnico aos desenvolvedores busca aprimorar o processo de codificação e testes, auxiliando na geração de código, detecção de falhas e otimização de rotinas. Essa abordagem pretende demonstrar como a aplicação prática da IA pode aumentar a produtividade da equipe e melhorar a qualidade final do software.

A metodologia adotada é de caráter aplicado e tecnológico, fundamentada em princípios da engenharia de software e nas metodologias ágeis, especialmente o framework Scrum, que permite um desenvolvimento iterativo e colaborativo. O sistema será implementado com linguagens como Python e C, utilizando conceitos de modelagem UML e redes de computadores para garantir eficiência e funcionamento em rede local. A pesquisa também adota práticas sustentáveis, substituindo relatórios impressos por registros digitais, contribuindo para a modernização e a responsabilidade ambiental no contexto educacional.

Em síntese, o trabalho busca integrar inovação tecnológica, metodologias modernas e Inteligência Artificial na criação de um sistema acadêmico funcional e colaborativo. Espera-se que o resultado contribua para o aprimoramento da gestão escolar e sirva de base para futuras aplicações da IA na área de Análise e Desenvolvimento de Sistemas.

1.1 Objetivo Geral

Desenvolver e implementar um sistema acadêmico colaborativo que integre o gerenciamento de alunos, aulas e atividades em uma plataforma digital unificada, utilizando princípios de engenharia de software ágil e técnicas de Inteligência Artificial como apoio ao desenvolvimento, testes e otimização de código. Além de oferecer uma ferramenta funcional e acessível, o projeto busca explorar o papel da IA na automação de tarefas e na melhoria da produtividade de equipes de software, aplicando linguagens de programação como Python e C, modelagem UML e conceitos de redes de computadores para garantir desempenho, segurança e escalabilidade. O sistema também visa promover práticas sustentáveis, substituindo processos manuais por soluções digitais e reduzindo o uso de papel em ambientes escolares.

1.2 Objetivos Específicos

- Aplicar metodologias ágeis de engenharia de software para o planejamento e acompanhamento das etapas de desenvolvimento do projeto.
- Utilizar a Inteligência Artificial como suporte aos desenvolvedores na geração, revisão e otimização do código-fonte da plataforma.
- Implementar algoritmos e estruturas de dados em Python para funcionalidades de busca, ordenação e geração de relatórios.
- Desenvolver partes fundamentais em linguagem C para compreender e aplicar conceitos de sistemas de baixo nível e integração com hardware.
- Modelar e documentar o sistema por meio de diagramas UML (casos de uso, classes e sequência).
- Aplicar conceitos de redes de computadores e sistemas distribuídos para garantir o funcionamento do sistema em rede local.
- Pesquisar e aplicar tecnologias emergentes que contribuam para a eficiência e inovação do projeto.
- Adotar práticas sustentáveis, como a substituição por registros digitais.

1.3 Contextualização do Caso

Atualmente, muitas instituições de ensino enfrentam dificuldades no gerenciamento de informações acadêmicas de forma integrada e eficiente. A ausência

de sistemas centralizados leva ao uso de planilhas, e-mails e mensagens em aplicativos, o que causa falhas na comunicação, perda de dados e falta de padronização nos processos. Esse cenário compromete tanto a gestão administrativa quanto o acompanhamento pedagógico, gerando atrasos e retrabalho.

Com o intuito de solucionar essas limitações, o presente projeto propõe o desenvolvimento de um Sistema Acadêmico Colaborativo, voltado à centralização de dados e à automação de rotinas escolares. A plataforma permitirá o cadastro de alunos, o registro de atividades e a consulta de informações em rede local, garantindo praticidade e segurança no acesso. Durante o processo de desenvolvimento, a Inteligência Artificial foi utilizada como ferramenta de apoio técnico aos desenvolvedores, auxiliando na geração de código, análise de erros e otimização de algoritmos. Essa abordagem visa aprimorar a produtividade da equipe e a qualidade final do software, aliando inovação tecnológica, eficiência operacional e responsabilidade ambiental.

2. Programação Estruturada em C

O sistema desenvolvido em linguagem C tem como finalidade aplicar, de forma prática e fundamentada, os princípios da programação estruturada, conforme apresentados por Luis Damas (2020) em “Linguagem C”. O programa implementa um processo de criação e manipulação de arquivos no formato CSV, os quais funcionam como um banco de dados textual. Esses arquivos são posteriormente utilizados em uma interface desenvolvida em Python, que fará a leitura e inserção de novos dados. Essa integração evidencia o papel da linguagem C como base estruturante para sistemas modulares, eficientes e interoperáveis. O código utiliza uma estrutura de dados (struct) denominada “Usuario”, que agrupa informações referentes ao tipo de conta, e-mail e senha de cada usuário. Essa prática reflete o conceito de estruturas compostas descrito por Damas, permitindo armazenar diferentes tipos de dados de forma unificada e organizada. Segundo Forbellone e Eberspächer (2015), o uso de estruturas é fundamental para o desenvolvimento de programas mais expressivos e de fácil manutenção, uma vez que promove clareza e reduz redundâncias no código. Além disso, a função “salvar_usuario_csv” é responsável por gravar os dados de cada usuário em um arquivo. Essa separação entre a função e o escopo principal (main) demonstra a aplicação do princípio de modularização, que busca dividir o programa em partes independentes e reutilizáveis.

Como destacam Deitel (2017), a divisão funcional de um código favorece a legibilidade e o reuso, princípios essenciais da programação estruturada. Durante a execução, o sistema percorre um vetor de usuários através de um laço de repetição for, gravando cada registro no arquivo CSV com a função fprintf. Essa estrutura de repetição elimina tarefas manuais e garante a escalabilidade do sistema, conforme orientações de Damas (2020) e Tanenbaum (2013), que reforçam a importância dos laços de controle no desenvolvimento de algoritmos eficientes. Além disso, o uso de estruturas condicionais (if) assegura o tratamento de possíveis erros de execução, como a falha na abertura de arquivos, promovendo maior segurança e previsibilidade no fluxo do programa — um exemplo direto do princípio de decisão e controle de fluxo abordado nos capítulos iniciais de Damas.

Outro ponto relevante é o uso das funções padrão da biblioteca <stdio.h>, como fopen, fprintf e fclose, que realizam a leitura e escrita de arquivos, cumprindo os requisitos de persistência e manipulação de dados externos. A criação dos arquivos “usuario_dados.csv” e “dados_academicos.csv” garante a integridade das informações e serve de base para futuras implementações, onde pela interface em Python poderá inserir novos registros e realizar operações de consulta e edição. Esse mecanismo de troca de dados entre linguagens exemplifica um modelo simples de banco de dados integrado, discutido por Silberschatz, Korth e Sudarshan (2020), no qual a camada de persistência é independente da camada de aplicação.

A futura adição de novas estruturas de decisão, como autenticação de usuários e validação de entrada, ampliará ainda mais a robustez e a aplicabilidade do sistema.

3. Engenharia de Software Ágil

A Engenharia de Software (ES) é uma disciplina fundamental que integra o processo de desenvolvimento e de documentação de sistemas, aplicando uma abordagem sistemática, disciplinada e quantificável (IEEE Computer Society, apud MAITINO NETO, 2016). Seu principal objetivo é a produção de software de alta qualidade, a um custo razoável, que seja confiável, eficiente, manutenível e rápido de construir, atendendo às necessidades do usuário (PRESSMAN, 2010; SOMMERVILLE, 2011; LAPLANTE, 2007, apud MAITINO NETO, 2016). Essa disciplina abrange todo o ciclo de vida do software, desde a concepção e especificação dos requisitos até a manutenção e evolução do sistema (DIO, [s.d.]). Ela utiliza teorias, técnicas e ferramentas da Ciência da Computação para guiar a produção de software de forma eficiente e com qualidade (UNICESUMAR, [s.d.]).

3.1 Abordagem Sistemática

São utilizadas as mesmas sistemáticas existentes em outras áreas da engenharia, buscando a adoção de uma abordagem organizada e o uso de ferramentas e técnicas apropriadas (IFRN, apud PRESSMAN, 2006). O processo de desenvolvimento, guiado pela disciplina, envolve uma série de atividades interrelacionadas no projeto, como:

- **Definição de Requisitos:** Para entender as necessidades do cliente e dos stakeholders, foram estabelecidos os Requisitos Funcionais (gerenciar usuários, logar, cadastrar tarefa, consultar notas, consultar presença, consultar tarefa, registrar notas, registrar presença) e os Requisitos Não Funcionais (consulta em menos de 2 segundos, autenticação robusta e criptografia de dados, disponibilidade mínima de 99% durante o período letivo, suporta vários usuários sem perda de desempenho e interface intuitiva), para especificar o que o sistema deve fazer.
- **Projeto (Design):** Define a arquitetura do software, seus componentes e interfaces. Pensando na arquitetura que melhor atende às necessidades, os requisitos de hardware utilizados foram: Memória RAM: 8 GB, Processador: Intel Core i5-12400, 2.50 GHz, Sistema Operacional: Windows 10, Rede: Via cabo de rede e a Comunicação via pasta compartilhada na rede.
- **Implementação (Codificação):** Desenvolver o código-fonte.
- **Testes:** Garantir a qualidade do software, identificando e corrigindo falhas (bugs).
- **Manutenção e Evolução:** Realizar atualizações, correções e melhorias no software existente.

3.2 Disciplina e Gerenciamento

Quando se trata desta área, ela se dedica ao gerenciamento de projetos de software, tratando o produto (intangível e flexível) e o processo de desenvolvimento, muitas vezes com baixa padronização, de forma controlada.

- **Modelos de Processo:** Utiliza diversos paradigmas (como o Clássico, Orientado a Objetos, ou metodologias Ágeis como o Scrum, que foi utilizado no projeto) para estruturar o trabalho, garantindo a coordenação da equipe e a gestão do processo de desenvolvimento (THOMÉ, 1998; IFFar, [s.d.]).

3.2.1 Caracterização da Engenharia de Software na Documentação de Sistemas

A documentação é um componente essencial do software, juntamente com o programa e a configuração associada (SOMMERVILLE, 2007, apud EDUCAPES, [s.d.]). Deste modo, ela assegura que a documentação seja precisa, útil e atualizada,

servindo como uma descrição exata do sistema de software (ZUP INNOVATION, [s.d.]).

A Engenharia de Software diferencia os tipos de documentação com base no público-alvo e no foco da informação (SYDLE ONE, [s.d.]; ZUP INNOVATION, [s.d.]). Com base nos tipos de Documentação, foi utilizada a Técnica/de Arquitetura, visando atingir o objetivo do projeto:

- **Foco:** Detalhes do design, implementação, estrutura e lógica interna do código.
- **Público-Alvo:** Desenvolvedores, Engenheiros de Software.

Além desses artefatos, foram elaboradas especificações de requisitos funcionais e não funcionais, que descrevem detalhadamente o comportamento esperado do sistema e as restrições técnicas envolvidas. Essa documentação garante a rastreabilidade entre as necessidades do usuário e as funcionalidades implementadas, servindo como base para as demais etapas de desenvolvimento e testes.

3.2.2 Diagrama de Fluxo de Dados

O Diagrama de Fluxo de Dados (DFD) representa o fluxo de informações do Sistema de Gestão Acadêmica. O objetivo é descrever de forma visual e estruturada como os dados circulam entre usuários, processos e bancos de dados do sistema.

O modelo apresentado reflete um sistema que gerencia o acesso de usuários (login), o registro de notas, tarefas e presenças, bem como a interação entre estudantes e professores. As entidades externas representam os agentes que interagem com o sistema, enviando e recebendo informações.

O DFD foi desenvolvido utilizando ferramentas de modelagem de software, seguindo a notação tradicional de processos, fluxos e armazenamentos de dados. Essa representação facilita o entendimento entre a equipe técnica e os stakeholders, garantindo clareza na comunicação e reduzindo ambiguidades no desenvolvimento do sistema.

Além dos fundamentos apresentados, também foi aplicado no projeto o método ágil Scrum, conforme os requisitos. Foi estabelecida a estrutura do desenvolvimento

em sprints quinzenais, utilizando um backlog de produto para organizar as funcionalidades e acompanhar o progresso.

Durante cada sprint, foram realizados encontros para revisar o andamento, atualizar o quadro de tarefas e ajustar prioridades com base nas entregas concluídas.

No levantamento de requisitos, foram definidos tanto os requisitos funcionais quanto os não funcionais, documentados de forma rastreável. Essa prática assegurou que cada requisito tivesse origem, implementação e validação claramente identificadas no ciclo de vida do sistema. Os artefatos gerados incluíram:

- Documento de Requisitos (funcionais e não funcionais) revisado a cada sprint;
- Diagrama de Fluxo de Dados (DFD) para representar o fluxo de informações entre os módulos;
- Relatórios de progresso, com validação de backlog e evolução das tarefas.

Assim, a disciplina foi aplicada não apenas como teoria, mas como uma abordagem prática para gerenciar a construção do sistema, garantindo transparência, qualidade e entrega contínua de valor.

4. Estrutura em Python

O Sistema Acadêmico é uma aplicação desenvolvida em Python utilizando a biblioteca Tkinter para a interface gráfica. Ele tem como objetivo gerenciar usuários e dados acadêmicos, permitindo diferentes níveis de acesso: Administrador: Gerencia usuários (adicionar, editar e excluir) com acesso à senha master. Professor: Lança presenças, notas e tarefas. Aluno: Visualiza suas notas, presenças e tarefas. O sistema utiliza arquivos de texto (usuario_dados.csv e dados_academicos.csv) para armazenamento contínuo de informações.

Arquivos Utilizados

usuario_dados.csv	Armazena os usuários cadastrados (email, senha, tipo).
dados_academicos.csv	Armazena registros de dados acadêmicos no formato: (tipo, aluno, disciplina, informação)

Cada arquivo de dados foi gerado a partir de outra plataforma de desenvolvimento, sendo ela, a Linguagem C. O projeto segue a arquitetura de interface por janelas, com telas específicas para cada tipo de usuário.

5. Análise e Projeto de Sistemas

A disciplina de Análise e Projeto de Sistemas contempla as etapas iniciais mais importantes para que o software atenda a todos os requisitos e funcionalidades que são necessários para suprir as expectativas do cliente desde antes do início do projeto até a entrega do produto final. Tendo como principais medidas em seu processo a adoção de métodos que possibilitam a compreensão, modelagem e a representação do funcionamento do produto antes da implementação.

Na etapa da análise, o objetivo é identificar e definir o que o sistema deverá executar. E o projeto, como ele será estruturado para fazer com que os requisitos que foram definidos na Engenharia de Software sejam cumpridos, segundo Ian Sommerville (2011).

No entanto, o projeto foi construído com base na metodologia UML (Unified Modeling Language), que possui um conjunto de diagramas já estabelecidos para representar de forma visual os componentes, interações e as ações do sistema, de forma que facilite a compreensão dos desenvolvedores, usuários e stakeholders sobre a plataforma.

De acordo com Booch, Rumbaugh e Jacobson (2006), a UML é uma linguagem frequentemente utilizada na modelagem dos sistemas orientados a objetos, já que ela torna possível ter uma pré-visualização dos diagramas com suas funcionalidades. Considerando isso, os diagramas que auxiliaram na documentação e comunicação entre o time de desenvolvimento, e utilizados na estruturação do trabalho foram:

- **Diagrama de Casos de uso:** possibilita a visualização dos requisitos funcionais e as interações com os atores que terão contato com o sistema;
- **Diagrama de Classes:** representa as classes, atributos, os métodos e relações entre elas;
- **Diagrama de Sequência:** detalhamento sobre as interações entre objetos em uma sequência cronológica específica.

As decisões técnicas adotadas, como a escolha da UML e dos tipos de diagramas, foram utilizadas pela capacidade de padronizar a comunicação entre as partes envolvidas, reduzindo ambiguidades e melhorando a clareza dos requisitos.

5.1 Diagrama de Casos de Uso

O diagrama de casos de uso foi desenvolvido para representar as interações entre os atores e o sistema. Os atores representados no diagrama foram apresentados na disciplina de Engenharia de Software. Com isso, cada caso de uso foi detalhado de acordo com os requisitos funcionais e possuem as interações que foram escritas usando o formato tabular:

RF1 - Gerenciar usuário

Administrador	**Sistema**
Acessa o sistema e insere a senha master	Valida a senha master e faz a liberação da área de gerenciamento de usuários
Seleciona a opção desejada (cadastrar, editar ou remover)	Solicita as informações necessárias (e-mail, tipo de usuário e senha)
Confirma a operação	Salva os dados e exibe a mensagem de sucesso

RF2 - Logar

Administrador, Professor e Estudante	**Sistema**
Acessa a tela de login	Exibe os campos de e-mail e senha
Insere e-mail e senha	Valida as credenciais
	Se validas, exibe a página inicial. Caso contrário, informa que a senha está errada

RF3 - Registrar Presença

Professor	**Sistema**
Seleciona a disciplina	Exibição do nome dos alunos
Marca a presença ou ausência do aluno no dia	Registra as presenças e faltas
Confirma a operação	Salva a operação e exibe a mensagem de confirmação

RF4 - Registrar Notas

Professor	**Sistema**
Seleciona a disciplina	Exibe a lista de alunos e o campo para as notas
Insere as notas dos alunos	Armazena os dados
Confirma a operação	Salva a operação e exibe a mensagem de confirmação

RF5 - Cadastrar tarefa

Professor	**Sistema**
Seleciona a disciplina	Exibe a opção de adicionar tarefa
Cadastra a nova tarefa	Exibe os campos de descrição da tarefa e data de entrega
Preenche e confirma a operação	Salva a operação e publica a atividade

RF6 - Entregar tarefa

Estudante	**Sistema**
Acessa a área de tarefas	Exibe as tarefas disponíveis e a data de entrega
Seleciona a tarefa desejada	Exibe a descrição de cada uma

RF7 - Consultar Notas

Estudante	**Sistema**
Acessa a área de notas	Exibe as notas de cada disciplina

RF8 - Consultar Presença

Estudante	**Sistema**
Seleciona a disciplina	Exibe a presença do dia

6. Redes de Computadores e Sistemas Distribuídos

As redes de computadores são estruturas essenciais para a conectividade e o compartilhamento de informações entre dispositivos, permitindo que dados, arquivos e serviços sejam acessados de forma rápida e confiável. A abordagem do sistema favorece a flexibilidade na manutenção e atualização do código, além de permiti-lo funcionar de maneira integrada a outros serviços e ferramentas de rede. Para garantir uma execução eficiente e segura, a estrutura proposta considera princípios fundamentais das redes de computadores, como o uso de protocolos padronizados, endereçamento lógico e segmentação de rede, que asseguram a comunicação entre diferentes dispositivos e usuários dentro de um ambiente acadêmico.

No caso do sistema acadêmico, a comunicação entre usuários e dados pode ocorrer em uma rede local (LAN) ou via internet, exigindo a compreensão de como os pacotes trafegam, são roteados e recebidos corretamente. O gerenciamento dos usuários e das informações de cada perfil é sustentado por camadas de software que seguem a lógica de modularidade e encapsulamento, permitindo que cada parte do sistema se comunique sem comprometer o todo — princípio também presente nos sistemas distribuídos.

Por fim, os conceitos como endereçamento IP, sub-redes, VLANs e virtualização orientam o planejamento da infraestrutura lógica que dá suporte à aplicação, garantindo desempenho e segurança na comunicação. Já as noções de sistemas distribuídos, microsserviços e web services inspiram a organização modular do sistema acadêmico, permitindo que cada função do mesmo seja tratada como um serviço independente. Assim, o projeto reflete a aplicação prática dos fundamentos de redes e sistemas distribuídos, assegurando escalabilidade, confiabilidade e integração tecnológica no ambiente educacional.

Para complementar a aplicação prática dos conceitos de redes e sistemas distribuídos, foi elaborado um layout lógico da rede utilizada no projeto. A configuração adotada segue o modelo cliente-servidor simples, com dois computadores conectados em rede local (LAN) por meio de um switch, utilizando endereçamento IP estático.

7. Educação Ambiental

A Educação Ambiental (EA) constitui um processo educativo permanente que busca desenvolver nos indivíduos e nas comunidades uma percepção crítica e responsável acerca das relações entre o ser humano e o meio ambiente. Conforme estabelece a Política Nacional de Educação Ambiental (Lei nº 9.795/1999), a EA tem como objetivo promover a construção de valores, conhecimentos e atitudes voltados à conservação do meio ambiente e à sustentabilidade da vida no planeta.

Os sistemas computacionais têm papel significativo na ampliação do alcance da Educação Ambiental. Plataformas digitais, bancos de dados e sistemas de gestão acadêmica permitem que informações ambientais sejam compartilhadas de maneira rápida, acessível e eficiente. Segundo Santos e Pires (2021), a tecnologia da informação é uma aliada da sustentabilidade, pois possibilita o monitoramento de ações ambientais em tempo real e o incentivo ao engajamento coletivo.

Além disso, o uso de ambientes virtuais de aprendizagem (AVA) e sistemas integrados de gestão favorece a redução do uso de papel, a desburocratização de processos e o acesso democrático ao conhecimento ambiental. Tais ferramentas não apenas otimizam a gestão institucional, mas também reforçam o compromisso das organizações com a preservação do meio ambiente e com os princípios da sustentabilidade (DIAS, 2015).

A incorporação de práticas sustentáveis nos sistemas de gestão acadêmica reflete o compromisso das instituições com a responsabilidade socioambiental. Dentre os aspectos mais relevantes, destacam-se:

- A digitalização de processos administrativos, que reduz o consumo de papel e de energia elétrica;
- A divulgação de campanhas ambientais e projetos ecológicos por meio de plataformas digitais;
- O incentivo à participação da comunidade acadêmica em ações de conscientização ambiental.

Essas medidas estão alinhadas à visão de Jacobi (2003), que defende a importância da participação coletiva e da educação contínua como instrumentos de transformação ecológica e social. Assim, o uso consciente da tecnologia contribui diretamente para a difusão da Educação Ambiental dentro das instituições de ensino.

Ações Voltadas à Educação Ambiental:

- **Programa de Coleta Seletiva e Reciclagem:** A implantação de programas de coleta seletiva e reciclagem nas instituições educacionais é uma ação de grande relevância ambiental e pedagógica, pois contribui para a redução de resíduos e incentiva a responsabilidade compartilhada entre alunos, professores e colaboradores. Segundo Jacobi (2003), a gestão participativa dos resíduos sólidos favorece atitudes éticas e sustentáveis, fortalecendo o senso de cidadania ambiental.
- **Campanhas Digitais de Sensibilização Ambiental:** As campanhas digitais educativas, divulgadas em sistemas acadêmicos, portais e redes sociais, abordam temas como economia de energia, uso racional da água e preservação da biodiversidade. Essas ações promovem a conscientização coletiva e ampliam o alcance das práticas sustentáveis. Conforme Dias (2015), o uso de tecnologias digitais na Educação Ambiental estimula o engajamento social e fortalece a disseminação de valores sustentáveis.

Além das ações já descritas, o projeto também incorpora práticas de sustentabilidade digital, como a substituição total de relatórios impressos por documentos eletrônicos, evitando o uso de papel, já que o sistema desenvolvido reforça esses valores ao permitir que professores e alunos troquem informações, notas e atividades em formato digital, eliminando a necessidade de registros físicos.

8. Inteligência Artificial

A Inteligência Artificial (IA) é um ramo da ciência da computação que busca desenvolver sistemas capazes de simular o raciocínio e o aprendizado humano. Segundo Russell e Norvig (2021), a IA tem como objetivo permitir que os computadores realizem tarefas que exigem inteligência, como reconhecimento de padrões, tomada de decisão e aprendizado com base em dados. No contexto do desenvolvimento de software, a IA é fundamental para melhorar o desempenho, a eficiência e a precisão das aplicações, proporcionando soluções mais rápidas e assertivas durante a construção de sistemas computacionais.

8.1 Aplicação da IA no Projeto

No projeto desenvolvido, a Inteligência Artificial não foi implementada na plataforma em si, mas sim utilizada como ferramenta de apoio ao processo de desenvolvimento. Ela serviu como um recurso auxiliar para os desenvolvedores, facilitando a escrita e otimização de códigos, a análise de possíveis erros e a compreensão de conceitos técnicos. Por meio de ferramentas baseadas em IA, os integrantes da equipe puderam acelerar etapas de codificação, obter sugestões de soluções e melhorar a organização lógica do sistema, garantindo um desenvolvimento mais eficiente e estruturado.

8.2 Processo de Definição e Implementação

Durante o processo de criação da plataforma, os desenvolvedores fizeram uso de ferramentas de Inteligência Artificial para auxiliar em diversas etapas do projeto, especialmente na implementação das linguagens Python e C. Essas ferramentas possibilitaram:

- A geração e correção automática de trechos de código;
- A identificação de falhas lógicas e sintáticas;
- O apoio na documentação técnica e nas boas práticas de programação;
- A otimização do tempo de desenvolvimento com respostas e exemplos rápidos.

Com isso, a IA contribuiu de forma significativa para que o grupo agilizasse o processo de desenvolvimento e garantisse maior qualidade no resultado final da plataforma.

9. Pesquisa, Tecnologia e Inovação

No cenário atual, a pesquisa, a tecnologia e a inovação são essenciais para o desenvolvimento de sistemas informatizados. A pesquisa tecnológica permite identificar tendências, compreender problemas existentes e gerar conhecimento aplicado que serve como base para soluções eficientes. Já a inovação transforma esse conhecimento em produtos e processos diferenciados, enquanto a tecnologia fornece os recursos e ferramentas necessários para implementar soluções seguras, escaláveis e eficientes (TIDD; BESSANT, 2018; SCHUMPETER, 1934).

O surgimento de tecnologias emergentes, como inteligência artificial, computação em nuvem, big data e blockchain, reforça a necessidade de atualização constante e da integração de soluções inovadoras no desenvolvimento de software (RUSSELL; NORVIG, 2021; MARR, 2018).

9.1 Processo de adoção de tecnologias inovadoras

A adoção de tecnologias inovadoras envolve três etapas principais:

- **Pesquisa e análise:** identificação de soluções tecnológicas que atendam às necessidades do sistema, considerando viabilidade técnica, custos e impactos sociais.
- **Experimentação e prototipagem:** desenvolvimento de módulos piloto ou protótipos para testar a eficácia da tecnologia antes da implementação completa.
- **Integração e escalabilidade:** aplicação das soluções em todo o sistema, garantindo interoperabilidade entre módulos e conformidade com padrões de qualidade e segurança (CHESBROUGH, 2003).

No contexto do sistema de gestão escolar, a substituição de processos em papel por módulos digitais, como registro de presença nas aulas que contribui para a sustentabilidade ambiental e facilita o acompanhamento do desempenho estudantil (UNESCO, 2020).

9.2 Importância para o desenvolvimento de software

Integrar pesquisa, tecnologia e inovação no desenvolvimento do sistema proposto traz diversos benefícios:

- **Automatização de processos:** reduz erros e economiza o tempo dos professores na gestão dos alunos e atividades.
- **Sustentabilidade:** diminui o uso de papel, promovendo práticas ambientais responsáveis.
- **Acesso remoto e colaboração:** módulos distribuídos em rede permitem acesso seguro e em tempo real para professores e alunos.
- **Adaptação a novas demandas:** a pesquisa contínua possibilita incorporar novas funcionalidades ou tecnologias emergentes de forma ágil (PRESSMAN, 2020).

9.3 Soluções tecnológicas emergentes

Para incentivar a adoção de soluções tecnológicas emergentes, é recomendado:

- Realizar pesquisas qualitativas com professores e alunos para identificar necessidades reais e personalizar os módulos do sistema.
- Antes de implementar qualquer função nova, é importante conversar com os usuários do sistema (professores e alunos) para entender o que realmente precisam. Isso ajuda a personalizar os módulos (lançamento de presença, notas e visualização de atividades, de seus dados e informações acadêmicas) de forma que realmente facilite o trabalho deles.
- Promover treinamentos e workshops sobre o uso de novas ferramentas digitais.
- Mesmo que o sistema seja bem feito, os professores precisam aprender a usálo corretamente. Pequenos cursos ou workshops ajudam os usuários a se familiarizar com as novas funções, evitando erros e aumentando a adoção da tecnologia. Treinamentos fornecidos aos alunos de novas tecnologias também são uma solução emergente para o uso consciente da tecnologia em seus estudos.

- Criar um plano de atualização contínua do sistema, permitindo a incorporação gradual de tecnologias emergentes de forma sustentável.
- É importante estabelecer um plano de atualização contínua do sistema, que permita a incorporação gradual de novas tecnologias de forma ecológica. Como a tecnologia evolui rapidamente, o sistema precisa ser revisado e aprimorado regularmente, isso envolve planejar cuidadosamente a introdução de novos módulos ou melhorias, garantindo que o sistema permaneça moderno, funcional e alinhado às necessidades dos usuários.

10. Desenvolvimento do Projeto

10.1 Caracterização do ambiente de estudo

O sistema foi desenvolvido com base nas disciplinas do semestre, com o objetivo de implementar soluções sustentáveis e eficientes que atendessem aos requisitos funcionais e não funcionais definidos no projeto. A proposta consistiu na criação de um Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial, voltado à gestão de alunos, professores e atividades, buscando eficiência, digitalização e sustentabilidade.

A aplicação foi construída utilizando as linguagens C e Python, integradas por meio de arquivos CSV, que funcionam como banco de dados, garantindo leveza e compatibilidade. A linguagem C foi escolhida pela eficiência e controle de memória, permitindo o uso de estruturas de decisão, repetição e funções modulares, aplicando princípios da programação estruturada.

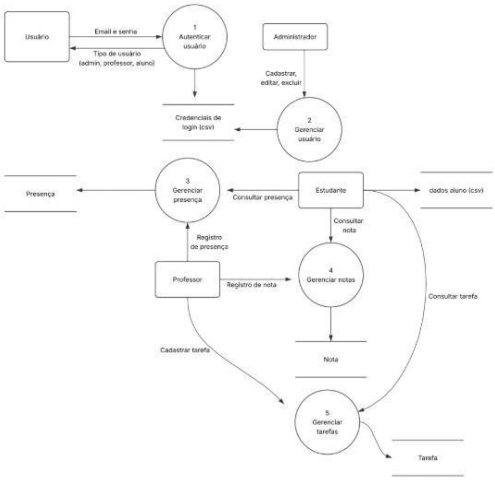
Também foram explorados conceitos da disciplina de Estrutura de Dados em C, para armazenar registros de usuários e dados acadêmicos, proporcionando maior desempenho e estabilidade.

O ambiente técnico contou com Visual Studio Code, biblioteca Tkinter e configuração em rede local (LAN) com computadores conectados por switch e endereçamento IP estático, simulando uma arquitetura cliente-servidor. O uso de ferramentas gratuitas e armazenamento digital reforçou o compromisso com a viabilidade econômica e a educação ambiental, eliminando impressões e incentivando práticas sustentáveis.

10.2 Desenvolvimento

A Engenharia de Software Ágil orientou o desenvolvimento com o uso do Scrum, organizando as tarefas em sprints quinzenais. O Diagrama de Fluxo de Dados (DFD) representou o fluxo das informações entre usuários e módulos, evidenciando a comunicação entre as camadas do sistema.

Figura 1: Diagrama de Fluxo de Dados

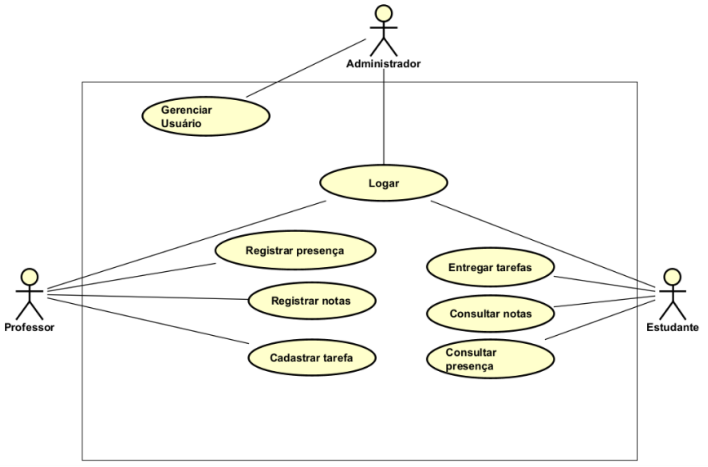


Fonte: As autoras (2025)

Essa modelagem foi essencial para demonstrar a comunicação entre as camadas do sistema e orientar o processo de codificação, mantendo coerência entre os requisitos levantados e as funcionalidades implementadas.

Na disciplina de Análise e Projeto de Sistemas, foram elaborados os diagramas UML, que auxiliaram na documentação e na comunicação entre o time de desenvolvimento. O Diagrama de Casos de Uso representou as interações entre os atores e o sistema, detalhando as ações de cada tipo de usuário, como o gerenciamento de contas, o registro de presenças, o lançamento de notas e a consulta de atividades.

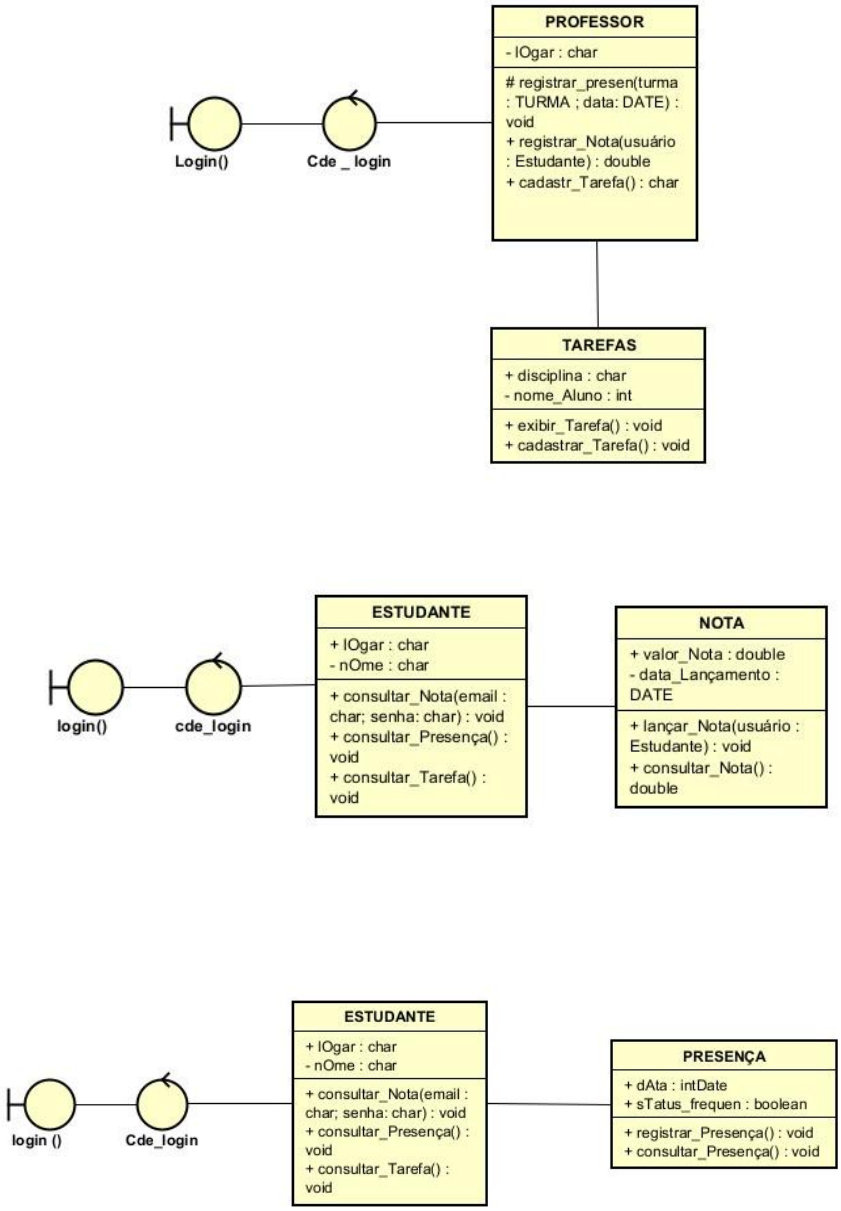
Figura 2: Diagrama de Casos de Uso

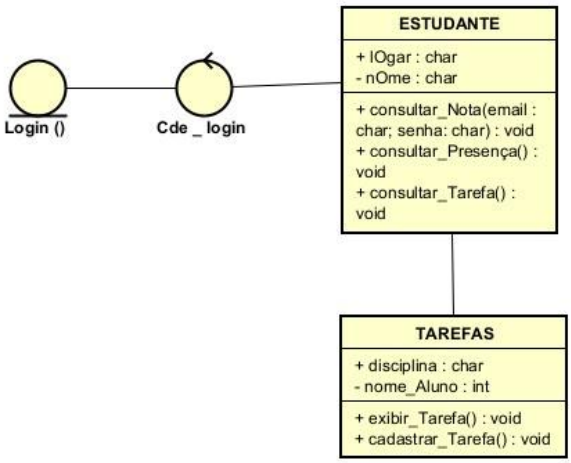


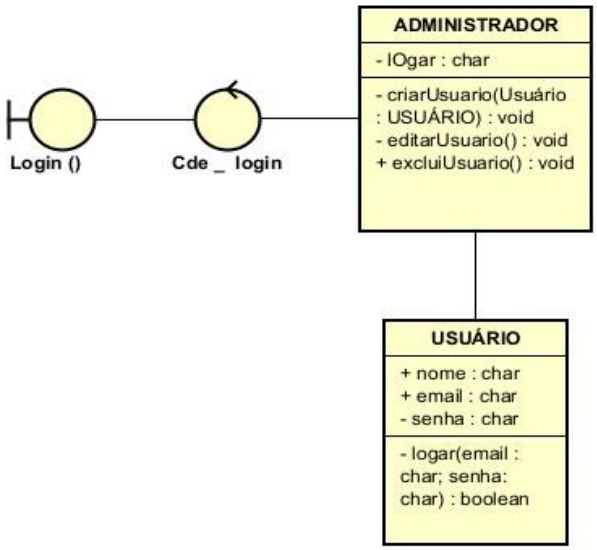
Fonte: As autoras (2025)

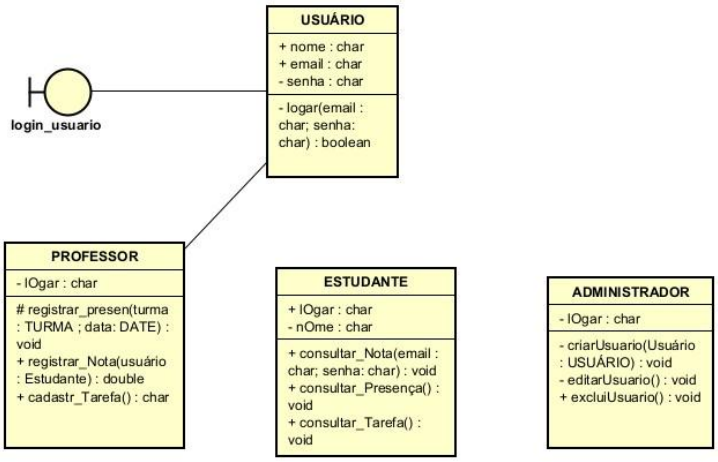
Em complemento, o Diagrama de Classes ilustrou a estrutura estática do sistema de gestão acadêmica, definindo classes, atributos e métodos, e estabelecendo os relacionamentos entre Administrador, Professor, Estudante, Nota, Presença e Tarefa. Essa representação foi fundamental para compreender a arquitetura orientada a objetos e servir como base para a implementação da aplicação.

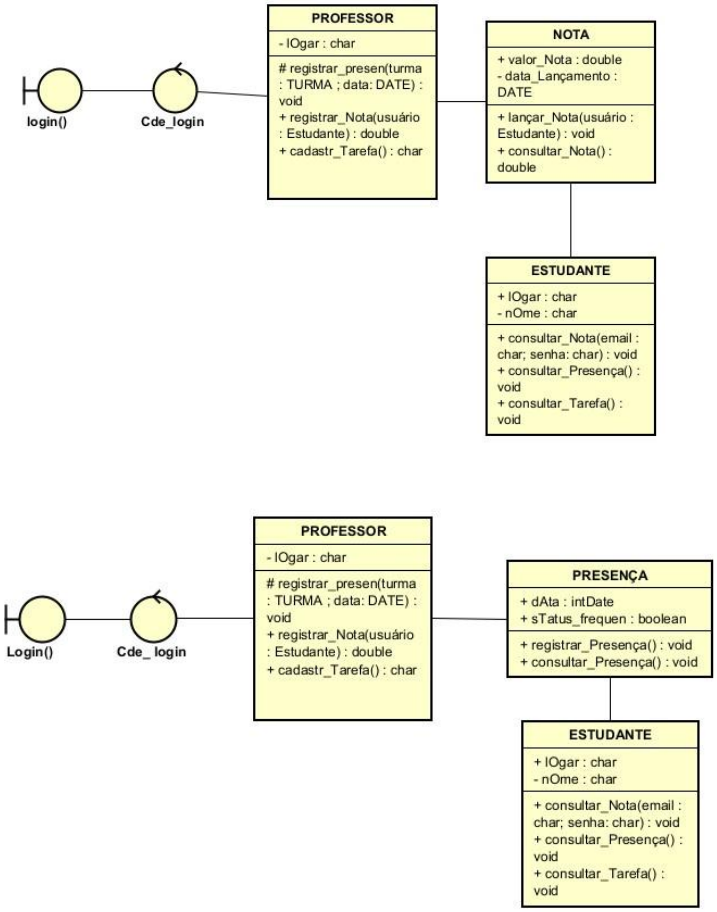
Figura 3 a 10: Diagrama de Classes




ESTUDANTE
+ lOgar : char
- nOme : char
+ consultar_Nota(email : char; senha: char) : void
+ consultar_Presença() : void
+ consultar_Tarefa() : void
Login ()
Cde _ login
TAREFAS
+ disciplina : char
- nome_Aluno : int
+ exibir_Tarefa() : void
+ cadastrar_Tarefa() : void


ADMINISTRADOR
- lOgar : char
- criarUsuario(Usuário : USUÁRIO) : void
- editarUsuario() : void
+ excluiUsuario() : void
Login ()
Cde _ login
USUÁRIO
+ nome : char
+ email : char
- senha : char
- logar(email : char; senha: char) : boolean

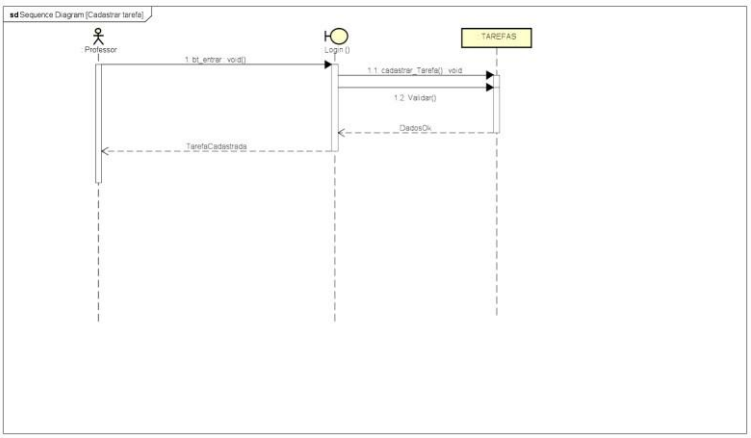

USUÁRIO
+ nome : char
+ email : char
- senha : char
- logar(email : char; senha: char) : boolean
login_usuario
PROFESSOR
- lOgar : char
registrar_presen(turma : TURMA ; data: DATE) : void
+ registrar_Nota(usuário : Estudante) : double
+ cadastr_Tarefa() : char
ESTUDANTE
+ lOgar : char
- nOme : char
+ consultar_Nota(email : char; senha: char) : void
+ consultar_Presença() : void
+ consultar_Tarefa() : void
ADMINISTRADOR
- lOgar : char
- criarUsuario(Usuário : USUÁRIO) : void
- editarUsuario() : void
+ excluiUsuario() : void

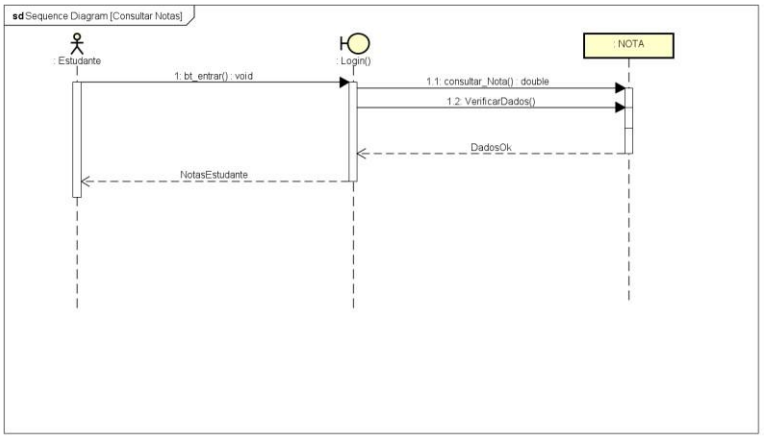


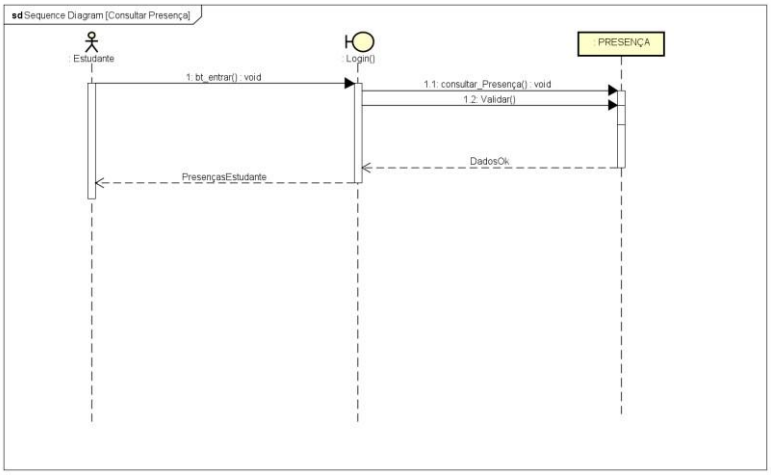
Fonte: As autoras (2025)

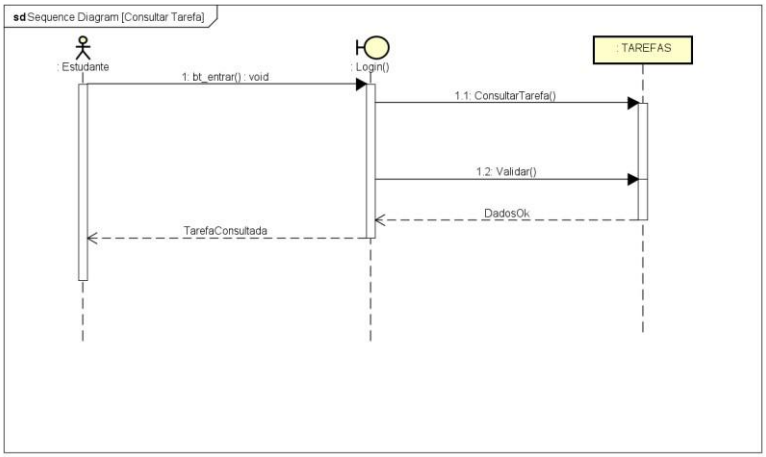
O Diagrama de Sequência mostrou a ordem cronológica das interações entre os objetos, representando a troca de mensagens e a lógica de funcionamento interno do sistema. A união desses diagramas permitiu a padronização da comunicação entre os integrantes do grupo e assegurou o alinhamento das decisões técnicas.

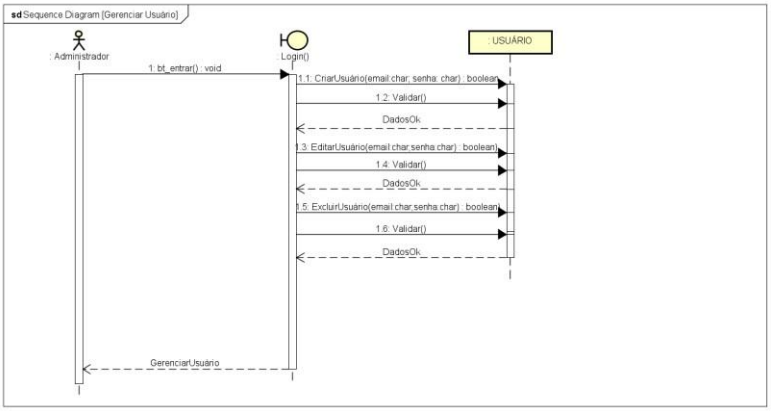
Figura 11 a 18: Diagrama de Sequência

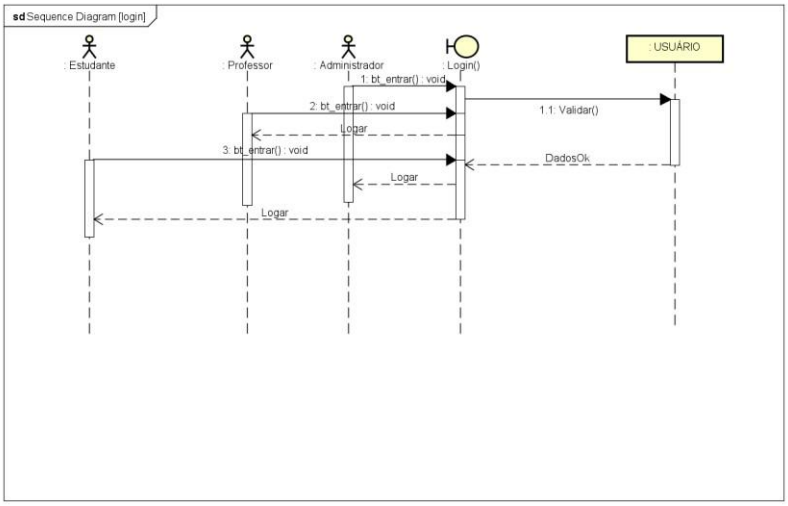


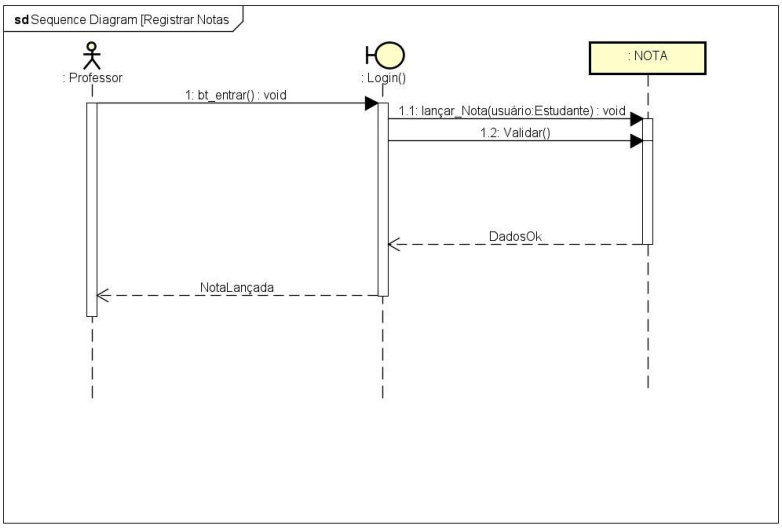

sd Sequence Diagram [Consultar Notas]
: Estudante
: Login()
: NOTA
1: bt_entrar() : void
1.1: consultar_Nota() : double
1.2: VerificarDados()
DadosOk
NotasEstudante

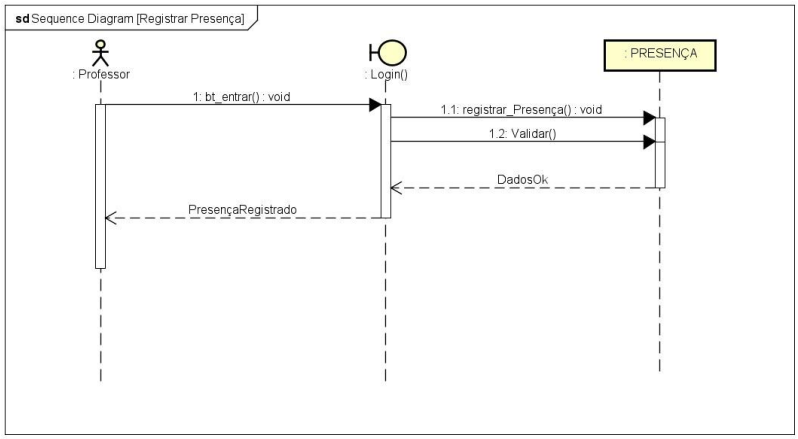

sd Sequence Diagram [Consultar Presença]
: Estudante
: Login()
: PRESENÇA
1: bt_entrar() : void
1.1: consultar_Presença() : void
1.2: Validar()
DadosOk
PresençasEstudante


sd Sequence Diagram [Consultar Tarefa]
: Estudante
: Login()
: TAREFAS
1: bt_entrar() : void
1.1: ConsultarTarefa()
1.2: Validar()
DadosOk
TarefaConsultada


sd Sequence Diagram [Gerenciar Usuário]
: Administrador
Login()
: USUÁRIO
1: bt_entrar() : void
1.1: CriarUsuário(email:char; senha: char) : boolean
1.2: Validar()
DadosOk
1.3: EditarUsuário(email:char;senha:char) : boolean
1.4: Validar()
DadosOk
1.5: ExcluirUsuário(email:char;senha:char) : boolean
1.6: Validar()
DadosOk
GerenciarUsuário


sd Sequence Diagram [login]
: Estudante
: Professor
: Administrador
: Login()
: USUÁRIO
1: bt_entrar() : void
2: bt_entrar() : void
1.1: Validar()
Logar
3: bt_entrar() : void
DadosOk
Logar
Logar


sd Sequence Diagram [Registrar Notas
: Professor
: Login()
: NOTA
1: bt_entrar() : void
1.1: lançar_Nota(usuário:Estudante) : void
1.2: Validar()
DadosOk
NotaLançada

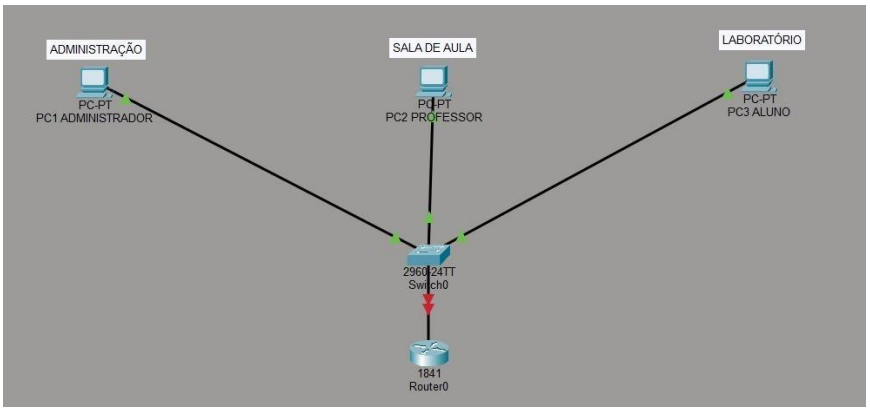


Fonte: As autoras (2025)

Em Python, foi desenvolvida a interface gráfica com a biblioteca Tkinter, criando janelas e menus interativos para cada perfil de usuário. Os arquivos usuario_dados.csv e dados_academicos.csv armazenam as informações de forma persistente, comunicando-se com os módulos em C e demonstrando integração entre linguagens.

A disciplina de Redes de Computadores e Sistemas Distribuídos foi aplicada na configuração da infraestrutura lógica, validando o sistema em topologia clienteservidor, realizando também testes de conectividade, desempenho e segurança, permitindo acessos simultâneos e estabilidade. Essa arquitetura distribuída possibilita futuras expansões e integrações, reforçando a base técnica do projeto.

Figura 19: Diagrama de Casos de Uso



Fonte: As autoras (2025)

A planta baixa apresentada representa a topologia física da rede local (LAN) utilizada pelo sistema acadêmico. Sua estrutura utiliza cabeamento Ethernet e dispositivos como switch e roteador para garantir conectividade estável entre os setores da Administração, Sala de Aula e Laboratório. Essa distribuição permite o acesso simultâneo ao sistema por três perfis distintos (Administrador, Professor e Aluno), assegurando desempenho, organização e facilidade de manutenção da infraestrutura de rede.

A Educação Ambiental foi incorporada ao projeto por meio da substituição de relatórios impressos por registros digitais e pela realização de campanhas de conscientização e reciclagem, reforçando a responsabilidade ambiental do grupo.

A Inteligência Artificial foi utilizada como ferramenta de apoio, auxiliando na geração e otimização de códigos, documentação técnica e boas práticas de desenvolvimento. Já em Pesquisa, Tecnologia e Inovação, aplicaram-se estudos sobre computação em nuvem e modularização orientada a serviços (SOA), além da criação de um plano de atualização contínua para o sistema.

11. Conclusão

Em suma do projeto apresentado, possui como meta implementar de maneira eficiente e sustentável o desenvolvimento de um sistema educacional para automação dos processos em uma instituição educacional para auxiliar professores e alunos em acessos rápidos e coesos, além de troca de informações.

Logo foi identificado ao longo do processo, as necessidades enfrentadas pelos usuários. Procuramos entender seus problemas diários, o funcionamento de cada etapa para assim implementar um sistema dinâmico para utilização com a utilização da Python para interface. Foi produzido na linguagem de programação C pela sua estruturação e agilidade, o armazenamento das informações em arquivos CSV sem utilização de banco de dados tornando o programa mais leve e acessível. No caso de obstáculos, foram utilizadas diversas ferramentas de consulta, como a IA (Inteligência Artificial), por exemplo, para ajustes básicos até mesmo em conceitos de estruturas do software.

Assim documentado todo o processo em diagramas UML, se mostrou eficiente no entendimento do processo de troca de dados na estrutura do software. Sendo possível realizar alterações pontuais e assertivas, diminuindo falhas na programação e erros futuros. Com a UML foi possível aumentar o conhecimento acerca do sistema e funcionalidades, guiando nosso “dev time” para o escopo do projeto com objetivo de entregar um software de qualidade. Em busca de melhor comunicação utilizamos o scrum como metodologia, foi possível identificar e mapear entregas e prazos de forma mais rápida. Isso nos ajudou a buscar correções em reuniões para mapear soluções e documentá-las. O sistema moderniza o ambiente escolar com a automatização de entregas, promovendo o desperdício zero de matéria prima aliadas à sustentabilidade.

Para trabalhos futuros, recomenda-se ampliar as funcionalidades do sistema e integrá-lo a outras plataformas digitais voltadas à educação. Para trabalhos futuros, recomenda-se ampliar as funcionalidades do sistema e integrá-lo a outras plataformas digitais voltadas à educação.

12. Referências Bibliográficas

ALURA. *Engenharia de Software: um guia sobre a área, carreira, mercado e formação*. [S. l.]: Alura, [s. d.]. Disponível em: https://www.alura.com.br/artigos/engenharia-de-software. Acesso em: 28 out. 2025.

BOOCH, Grady. *UML: guia do usuário* (tradução). [S. l.]: PDF Coffee, [s. d.]. Disponível em: https://pdfcoffee.com/uml-guia-do-usuario-traducao-grady-boochpdf-4-pdf-free.html. Acesso em: 28 out. 2025.

BRASIL. Lei nº 9.795, de 27 de abril de 1999. Dispõe sobre a Educação Ambiental, institui a Política Nacional de Educação Ambiental e dá outras providências. *Diário Oficial da União*, Brasília, 1999.

CHESBROUGH, H. *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Boston: Harvard Business School Press, 2003. Acesso em: 29 out. 2025.

CLICKUP. *Como escrever documentação de engenharia de software*. [S. l.]: ClickUp, [s. d.]. Disponível em: https://clickup.com/pt-BR/blog/238692/documentacao-de-engenharia-de-software. Acesso em: 28 out. 2025.

COULOURIS, G.; DOLLIMORE, J.; KINDBERG, T.; BLAIR, G. *Sistemas distribuídos: conceitos e projeto*. 5. ed. Porto Alegre: Bookman, 2013.

DAMAS, L. *Linguagem C*. 10. ed. Lisboa: FCA Editora, 2020.

DEITEL, P.; DEITEL, H. *C: how to program*. 8. ed. Pearson, 2017.

DEVMEDIA. *Artigo Engenharia de Software 10 – Documento de requisitos*. [S. l.]: DevMedia, [s. d.]. Disponível em: https://www.devmedia.com.br/artigo-engenharia-de-software-10-documento-de-requisitos/11909. Acesso em: 28 out. 2025.

DIAS, G. F. *Educação ambiental: princípios e práticas*. 12. ed. São Paulo: Gaia, 2015.

DIO. *Os processos de desenvolvimento de software: engenharia de software*. [S. l.]: Digital Innovation One (DIO), [s. d.]. Disponível em: https://www.dio.me/articles/os-processos-de-desenvolvimento-de-software-engenharia-de-software. Acesso em: 28 out. 2025.

EDUCAPES. *Engenharia de Software*. [Material didático]. Capes, [s. d.]. Disponível em: https://educapes.capes.gov.br/bitstream/capes/177122/2/Material%20Didatico-Engenharia%20de%20Software.pdf. Acesso em: 28 out. 2025.

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. *Lógica de programação: a construção de algoritmos e estruturas de dados*. 4. ed. São Paulo: Pearson, 2015.
GODOY, A. B. *Programação em C: princípios, conceitos e aplicações*. São Paulo: Novatec, 2021.

GUIA DEV. *Documentação técnica*. [S. l.]: Guia Dev, [s. d.]. Disponível em: https://guia.dev/pt/pillars/software-architecture/technical-documentation.html. Acesso em: 28 out. 2025.

IFFAR. *PDS: Processo de Desenvolvimento de Software*. Instituto Federal Farroupilha (IFFar), [s. d.]. Disponível em: https://www.iffarroupilha.edu.br/component/k2/attachments/download/39458/9baba407379e6651a7587cfa980e8b03. Acesso em: 28 out. 2025.
JACOBI, P. R. Educação ambiental, cidadania e sustentabilidade. *Cadernos de Pesquisa*, n. 118, p. 189–205, 2003.

KERNIGHAN, B W.; RITCHIE, D. M. *The C programming language*. 2. ed. Prentice Hall, 1988.

KUROSE, J. F.; ROSS, K. W. *Redes de computadores e a Internet: uma abordagem top-down*. 8. ed. São Paulo: Pearson, 2021.

MAITINO NETO, R. *Engenharia de Software*. Londrina: Editora e Distribuidora Educacional S.A., 2016.

MARR, B. *Big Data in practice*. 2. ed. Wiley, 2018. Acesso em: 29 out. 2025.

MIZRAHI, V. V. *Estruturas de dados e algoritmos em C*. 3. ed. São Paulo: LTC, 2018.

PRESSMAN, R. S. *Engenharia de software: uma abordagem profissional*. 8. ed. McGraw-Hill, 2016.

PRESSMAN, R. S. *Engenharia de software: uma abordagem profissional*. 9. ed. São Paulo: McGraw-Hill, 2020. Acesso em: 29 out. 2025.

RUSSELL, S.; NORVIG, P. *Artificial intelligence: a modern approach*. 4. ed. Harlow: Pearson, 2021. Acesso em: 29 out. 2025.

RUSSELL, S.; NORVIG, P. *Inteligência artificial: uma abordagem moderna*. 4. ed. São Paulo: Pearson, 2021.

SAMAECN. *Engenharia de Software*. [S. l.]: Samaecn, [s. d.]. Disponível em: https://samaecn.com.br/dinamico/noticias/2/pdf/engenharia_do_software.pdf. Acesso em: 28 out. 2025.

SANTOS, C. M.; PIRES, J. H. Tecnologia da informação e sustentabilidade. *Revista Brasileira de Gestão Ambiental*, v. 15, n. 2, p. 22–34, 2021.

SCHUMPETER, J. A. *Capitalismo, socialismo e democracia*. 3. ed. Rio de Janeiro: Fundo de Cultura, 1934.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Sistemas de banco de dados*. 7. ed. Pearson, 2020.

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

SOUZA, J. P. *Programação estruturada em C*. São Paulo: Érica, 2019.

STALLINGS, W. *Data and computer communications*. 10. ed. Pearson, 2014.

SYDLE ONE. *Documentação de software*. [S. l.]: Sydle One, [s. d.]. Disponível em: https://www.sydle.com/br/blog/documentacao-de-software-67607a278f7ac06b8fb6bbcc. Acesso em: 28 out. 2025.

TANENBAUM, A. S. *Structured computer organization*. 6. ed. Pearson, 2013.

TANENBAUM, A. S.; WETHERALL, D. J. *Redes de computadores*. 5. ed. São Paulo: Pearson, 2011.

THOMÉ, A. C. *Desenvolvimento de software por engenheiros: diretrizes e metodologias*. 1998. Dissertação (Mestrado) – INPE, São José dos Campos, 1998. Disponível em: http://marte3.sid.inpe.br/… Acesso em: 28 out. 2025.

TIDD, J.; BESSANT, J. *Managing innovation*. 6. ed. Wiley, 2018.

UNESCO. *Education and digital transformation: global guidelines*. Paris: UNESCO, 2020. Acesso em: 29 out. 2025.
UNICESUMAR. *O que é Engenharia de Software?* [S. l.]: Unicesumar, [s. d.]. Disponível em: https://www.unicesumar.edu.br/blog/o-que-e-engenharia-de-software/. Acesso em: 28 out. 2025.

ZUP INNOVATION. *Documentação de software: por que é tão importante*. [S. l.]: Zup Innovation, [s. d.]. Disponível em: https://zup.com.br/blog/documentacao-de-software/. Acesso em: 28 out. 2025.

FICHA DE CONTROLE DO PIM

Grupo Nº II Ano 2025 Período: 2° Orientador:_____

Tema: Desenvolvimento de um Sistema Acadêmico Colaborativo com Apoio de IA

Alunos:

RA	Nome	E-mail	Curso	Visto do aluno
H665442	Camille Vitória dos Reis Silva	camille.silva12@aluno.unip.br	CST em ADS	
R8674A8	Fernanda Leticia do Prado Andrade	fernanda.andr@aluno.unip.br	CST em ADS	
F362FI3	Keyla de Souza Fróes	keyla.froes@aluno.unip.br	CST em ADS	
H610010	Nicole Aparecida Gomes	nicole.gomes8@aluno.unip.br	CST em ADS	
H6611J0	Nicole Grassini Gonçalves	nicole.goncalves10@aluno.unip.br	CST em ADS	
R263JE1	Sabrina Helen dos Santos Reis Ferreira	sabrina.reis15@unip.com.br	CST em ADS	

Registros:

Data do encontro	Observações
18/08/2025	Reunião para decisão da ferramenta Jira como apoio na organização do projeto.
12/09/2025	Reunião de alinhamento para distribuição das disciplinas para cada membro.
15/09/2025	Reunião com Eduardo Sakaue para levantamento de requisitos do Sistema Educacional.
08/10/2025	Reunião com professor Egydio para feedback do diagrama de casos de uso e explicação do diagrama de classes e sequência.
10/10/2025	Reunião PO e Scrum Master para ajuste de quadro kanban e Jira com nossos requisitos não funcionais e funcionais.

16/10/2025	Reunião para alteração dos atores do sistema: adicionado o usuário Administrador; organizado o código em Python e iniciado em C.
07/11/2025	Reunião entre PO e Scrum Master para a formatação e o ajuste do desenvolvimento da documentação do projeto.
21/11/2025	Reunião pré-apresentação do projeto, com ajustes finais à documentação e leitura em grupo para modificações de acordo com o código.
28/11/2025	Reunião final de revisão de código e tira-dúvidas geral.