



**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**

**Nomes**

Cauã Brandão da Costa

Gabriel Alves Moreira

Josué Abraão Carvalho dos Santos

Luiz Gustavo da Silva Borges

Maciel Costa da Silva

Maycon Douglas Inácio Silva

**R.A**

R8327B0

H67HJI4

H67HJJ2

H4062J7

R280985

H719CD3

**SÃO JOSÉ DOS CAMPOS – SP**

**NOVEMBRO/2025**

| <b>Nomes</b>                     | <b>R.A</b> |
|----------------------------------|------------|
| Cauã Brandão da Costa            | R8327B0    |
| Gabriel Alves Moreira            | H67HJI4    |
| Josué Abraão Carvalho dos Santos | H67HJJ2    |
| Luiz Gustavo da Silva Borges     | H4062J7    |
| Maciel Costa da Silva            | R280985    |
| Maycon Douglas Inácio Silva      | H719CD3    |

**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.

## RESUMO

O projeto aborda o desenvolvimento de um Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial, concebido para aprimorar a gestão e a interação entre alunos, professores e administradores. A proposta integra dois módulos principais: um desenvolvido em linguagem C, responsável pelo cadastro e armazenamento básico de usuários em arquivos CSV, e outro em Python, encarregado da interface gráfica (Tkinter) e da visualização de desempenho (matplotlib). O projeto foi conduzido com base nas práticas de Engenharia de Software Ágil, utilizando as metodologias Scrum e Kanban para o levantamento de requisitos, modelagem UML, implementação, integração e testes. Foram definidos requisitos funcionais, como autenticação por níveis de acesso, lançamento e visualização de notas, e consulta ao corpo docente, bem como requisitos não funcionais, relacionados ao armazenamento de dados em arquivo CSV, limitações de segurança e arquitetura modular. Os testes locais evidenciaram a viabilidade da integração entre módulos e a utilidade do feedback motivacional automatizado para o aluno. Conclui-se que a solução desenvolvida serve como prova de conceito para ambientes acadêmicos de pequeno porte, sendo recomendadas futuras melhorias, como a migração para banco de dados relacional, o fortalecimento da segurança das credenciais e a ampliação do componente de Inteligência Artificial com técnicas de aprendizado de máquina.

**Palavras-chave:** Sistema Acadêmico; Inteligência Artificial; Engenharia de Software Ágil; Linguagem C; Python

## SUMÁRIO

|   |           |
|---|-----------|
| <b>1 INTRODUÇÃO .....</b>   | <b>6</b>  |
| 1.1 Objetivo Geral .....  | 6         |
| 1.2 Objetivos Específicos .....   | 6         |
| 1.3 Contextualização do Caso.....   | 7         |
| <b>2 CARACTERIZAÇÃO DA EMPRESA .....</b>  | <b>8</b>  |
| <b>3 DESENVOLVIMENTO DO PROJETO.....</b>  | <b>8</b>  |
| 3.1 Caracterização do Ambiente de Estudo .....  | 8         |
| 3.2 Programação Estruturada em C.....   | 9         |
| 3.2.1 Virtudes da Linguagem C.....  | 10        |
| 3.2.2 Limitações da Linguagem C.....  | 10        |
| 3.3 Engenharia de Software Ágil.....  | 11        |
| 3.4 Estrutura de Dados em Python .....  | 16        |
| 3.4.1 Caracterização da Linguagem Python como Parte do Processo de<br>Desenvolvimento de Sistemas ..... | 16        |
| 3.4.2 Descrição do Processo de Desenvolvimento e Utilização dos Recursos de<br>Python .....             | 17        |
| 3.4.3 Estruturas de Programação e Elementos de Python Utilizados .....                                  | 18        |
| 3.5 Análise e Projeto de Sistemas .....   | 19        |
| 3.6 Redes de Computadores e Sistemas Distribuídos.....  | 20        |
| 3.7 Educação Ambiental .....  | 22        |
| 3.8 Inteligência Artificial .....   | 23        |
| 3.9 Pesquisa, Tecnologia e Inovação.....  | 25        |
| <b>4 DESENVOLVIMENTO .....</b>  | <b>26</b> |
| 4.10 Implementação e Resultados do Projeto.....   | 28        |
| 4.10.1 Artefatos de Modelagem do Sistema .....  | 28        |
| 4.10.2 Implementação do Módulo em C (Back-End).....   | 33        |
| 4.10.3 Interface Gráfica em Python (Front-End) .....  | 34        |
| 4.10.4 Funcionalidades de Análise e Gráficos .....  | 35        |
| 4.10.5 Gestão Ágil do Projeto (Kanban).....   | 36        |
| 4.10.6 Infraestrutura de Rede e Layout (Planta Baixa) .....   | 37        |
| <b>5 CONCLUSÃO .....</b>  | <b>39</b> |
| <b>REFERÊNCIAS .....</b>  | <b>40</b> |

**SUMÁRIO DE FIGURAS**

**Figura 1 ..... 29**  
**Figura 2..... 30**  
**Figura 3..... 31**  
**Figura 4..... 32**  
**Figura 5..... 33**  
**Figura 6..... 34**  
**Figura 7 ..... 35**  
**Figura 8..... 36**  
**Figura 9..... 37**

## 1 INTRODUÇÃO

O presente Projeto Integrado Multidisciplinar (PIM II) apresenta o desenvolvimento do Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial, uma solução tecnológica voltada para modernizar a gestão da "Instituição Modelo de Tecnologia". O projeto surge da necessidade de solucionar problemas como a descentralização de informações acadêmicas e a dependência de processos manuais, propondo uma ferramenta que integre alunos, professores e administradores.

A solução adota uma arquitetura híbrida que combina a eficiência da Linguagem C no gerenciamento administrativo de dados com a interatividade da Linguagem Python na interface gráfica. O sistema inova ao trazer um módulo de Inteligência Artificial baseado em regras heurísticas, capaz de analisar o desempenho do aluno e fornecer *feedback* motivacional automático, auxiliando no engajamento estudantil.

Todo o desenvolvimento foi fundamentado nos princípios da Engenharia de Software Ágil, utilizando as metodologias Scrum e Kanban para o planejamento e controle das tarefas. O trabalho documenta o ciclo completo do software, desde a modelagem de requisitos (UML) até a implementação técnica e infraestrutura de redes, demonstrando a aplicação prática dos conhecimentos adquiridos no curso de Análise e Desenvolvimento de Sistemas.

### 1.1 Objetivo Geral

O objetivo geral deste projeto é desenvolver um sistema acadêmico colaborativo voltado à gestão eficiente de informações estudantis, promovendo a comunicação entre alunos, professores e administradores de forma integrada e organizada, utilizando práticas modernas de desenvolvimento de software.

### 1.2 Objetivos Específicos

Para alcançar o objetivo geral serão desenvolvidos os seguintes objetivos específicos.

- Levantar e analisar os requisitos funcionais e não funcionais necessários ao sistema.
- Estruturar os módulos de software de forma integrada, utilizando linguagens de programação adequadas ao contexto acadêmico.
- Implementar uma interface gráfica intuitiva e acessível para diferentes tipos de usuários.
- Aplicar metodologias ágeis no planejamento e acompanhamento das etapas do projeto.
- Garantir a segurança, a consistência e a persistência dos dados acadêmicos armazenados.
- Promover a colaboração e a troca de informações entre os usuários do sistema.
- Documentar todas as etapas de desenvolvimento, assegurando clareza e padronização técnica.

### 1.3 Contextualização do Caso

A necessidade de modernização dos processos acadêmicos tem impulsionado o desenvolvimento de soluções tecnológicas que centralizam informações e facilitam a interação entre os diferentes setores de uma instituição de ensino. Nesse cenário, este projeto apresenta a proposta de criação de um Sistema Acadêmico Colaborativo, projetado para simplificar o gerenciamento de dados e otimizar o fluxo de informações entre alunos, professores e administração. O sistema foi concebido para permitir o cadastro e o controle de usuários, o lançamento e acompanhamento de notas, além da visualização de relatórios acadêmicos de forma prática e segura. Sua estrutura modular possibilita a integração entre diferentes linguagens de programação, demonstrando a aplicação de conceitos de engenharia de software, programação estruturada e análise de sistemas. Assim, o projeto contribui para o aprimoramento da gestão educacional e serve como exemplo de aplicação dos conhecimentos adquiridos ao longo do curso de Análise e Desenvolvimento de Sistemas, unindo teoria e prática em um contexto real de desenvolvimento tecnológico.

## **2 CARACTERIZAÇÃO DA EMPRESA**

O projeto foi desenvolvido tendo como cenário uma Instituição de Ensino Superior (IES) de pequeno a médio porte, doravante denominada "Instituição Modelo de Tecnologia". A instituição atua no segmento educacional oferecendo cursos de graduação e pós-graduação na área de Tecnologia da Informação e Gestão.

Atualmente, a organização enfrenta desafios relacionados à descentralização de suas informações acadêmicas. O processo vigente de lançamento de notas e comunicação entre secretaria, corpo docente e discentes ainda depende, em partes, de planilhas manuais e sistemas legados que não conversam entre si. Essa fragmentação gera retrabalho, inconsistência de dados e dificuldades no acompanhamento do desempenho estudantil.

A estrutura organizacional da empresa é composta pela Diretoria Acadêmica, Secretaria Geral, Coordenação de Cursos e o Corpo Docente. O sistema proposto visa atender transversalmente a esses departamentos, focando na otimização do fluxo de dados do curso de Análise e Desenvolvimento de Sistemas. A adoção do "Sistema Acadêmico Colaborativo com Apoio de IA" busca modernizar essa infraestrutura, reduzindo o uso de papel e agilizando a tomada de decisão pedagógica através da centralização dos dados em arquivos digitais estruturados.

## **3 DESENVOLVIMENTO DO PROJETO**

### **3.1 Caracterização do Ambiente de Estudo**

O ambiente de estudo para o desenvolvimento do Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial foi estruturado de forma a simular o funcionamento real de uma instituição de ensino superior, envolvendo múltiplos perfis de usuários, como administradores, professores e alunos. O foco principal foi proporcionar uma plataforma integrada que facilitasse a comunicação e o gerenciamento de informações acadêmicas, contemplando funções essenciais como cadastro de usuários, lançamento de notas, visualização de desempenho e geração de relatórios.

O projeto foi desenvolvido em ambiente multiplataforma, utilizando computadores pessoais com sistema operacional Windows 10/11, linguagens de



programação C e Python, além de bibliotecas específicas para interface gráfica e visualização de dados. Os testes funcionais foram realizados localmente, em rede simulada (LAN), de modo a representar o fluxo de comunicação entre cliente e servidor em um contexto acadêmico.

O sistema foi planejado com base em metodologias ágeis, utilizando Scrum e Kanban como suporte para o acompanhamento das tarefas e definição de sprints. A plataforma Trello foi adotada como ferramenta de controle visual das etapas, garantindo uma visão geral do progresso do desenvolvimento e facilitando a colaboração entre os membros da equipe.

A estrutura lógica foi concebida para permitir integração entre diferentes linguagens e camadas do sistema, reforçando conceitos das disciplinas de Programação Estruturada, Engenharia de Software e Análise de Sistemas. Cada integrante da equipe ficou responsável por uma parte específica do desenvolvimento, como programação, testes, modelagem e documentação.

### 3.2 Programação Estruturada em C

De acordo com Damas (2007), a linguagem C surgiu como uma linguagem de propósito geral, combinando a eficiência e o controle das linguagens de baixo nível com os princípios da programação estruturada, o que permite desenvolver programas mais claros, modulares e de fácil manutenção. Essa estruturação é alcançada, por meio do uso de funções, blocos lógicos e estruturas de controle como *if*, *for*, *while* e *switch*, que proporcionam uma lógica sequencial e hierárquica ao desenvolvimento de software.

Segundo Damas (2007), “a simplicidade da linguagem C e sua proximidade com o hardware permitem um controle eficiente dos recursos do computador, mantendo uma estrutura de código organizada elegível”. A linguagem C consolidou-se como uma das mais influentes da história, servindo de base para diversas outras linguagens posteriores, como C++, C#, Java e Python. Sua versatilidade e eficiência a tornam amplamente utilizada em aplicações que exigem alto desempenho e controle direto sobre o hardware, como sistemas integrados, sistemas operacionais e compiladores.

### 3.2.1 Virtudes da Linguagem C

A linguagem C apresenta um conjunto de virtudes que justificam sua permanência como uma das linguagens mais utilizadas e ensinadas no campo da programação. Entre essas virtudes, destacam-se:

- Eficiência e desempenho: C é capaz de gerar código executável altamente otimizado, o que a torna adequada para o desenvolvimento de softwares que exigem velocidade e uso eficiente de recursos computacionais.
- Portabilidade: os programas escritos em C podem ser compilados em diferentes plataformas com poucas modificações, característica essencial para o desenvolvimento multiplataforma.
- Controle de hardware: a linguagem oferece acesso direto à memória e aos dispositivos, permitindo a manipulação de ponteiros e endereços, fator indispensável para aplicações de baixo nível e sistemas embarcados.
- Flexibilidade e poder expressivo: C permite ao programador empregar diferentes abordagens de solução e integrar-se com outras linguagens e bibliotecas.
- Base conceitual para outras linguagens: por sua proximidade com a arquitetura do computador e sua estrutura lógica, o aprendizado de C fornece uma base sólida para o entendimento de paradigmas e linguagens mais avançadas.

A aplicação prática desses conceitos pode ser observada no desenvolvimento do módulo de *backend* administrativo, cuja interface de linha de comando e funcionamento são apresentados na Figura 5, localizada no tópico 3.10.2.

### 3.2.2 Limitações da Linguagem C

Apesar de suas inúmeras virtudes, a linguagem C também apresenta limitações reconhecidas pela literatura técnica. Essas limitações decorrem, principalmente, de seu foco em eficiência e controle, o que implica menor abstração e segurança em relação a linguagens modernas. Entre as principais restrições, destacam-se:

- Ausência de orientação a objetos: C não possui mecanismos nativos de encapsulamento, herança ou polimorfismo, o que limita a modularidade e a reutilização de código em projetos de grande porte.

- Gerenciamento manual de memória: a alocação e liberação de memória são de responsabilidade do programador, o que aumenta a possibilidade de erros, como vazamentos de memória (*memory leaks*) e falhas de segmentação (*segmentation faults*).
- Baixo nível de segurança: a linguagem não realiza verificações rigorosas de tipo e limites durante a compilação, o que pode resultar em falhas de execução e vulnerabilidades de segurança.
- Curva de aprendizado acentuada: conceitos como ponteiros, endereçamento e manipulação de memória podem representar uma barreira inicial para programadores iniciantes.

### 3.3 Engenharia de Software Ágil

A Engenharia de Software é um campo essencial da Computação, responsável por organizar e padronizar o desenvolvimento de sistemas, garantindo qualidade, confiabilidade e eficiência. No contexto acadêmico, sua aplicação é fundamental para estruturar soluções tecnológicas que atendam às demandas de alunos, professores e administradores.

Este capítulo tem como objetivo caracterizar a Engenharia de Software Ágil, apresentar os requisitos funcionais e não funcionais do Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial, e descrever a adoção das metodologias ágeis Scrum e Kanban como base para o gerenciamento e o controle do projeto.

### **Engenharia de Software no Processo de Desenvolvimento**

A Engenharia de Software envolve processos, métodos e ferramentas que abrangem todas as etapas do ciclo de vida de um sistema, desde a análise de requisitos até a manutenção. Ela promove documentação estruturada, comunicação eficiente e qualidade no produto final (SOMMERVILLE, 2011).

## LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos, neste contexto, foi realizado por meio da engenharia reversa dos módulos de softwares desenvolvidos (em C e Python). Esta abordagem, frequentemente necessária ao lidar com sistemas existentes onde a documentação original pode estar ausente, foca na análise do código-fonte para inferir as funcionalidades e características técnicas implementadas (PRESSMAN e MAXIM, 2016).

### Requisitos Funcionais (RF)

Os requisitos funcionais descrevem o que o sistema faz.

#### **RF01:** Autenticar Usuários por Nível de Acesso

- O sistema deve autenticar três níveis de acesso (Admin, Professor, Aluno) via login, carregando dinamicamente as credenciais dos arquivos CSV (credenciais\_\*.csv) a cada tentativa.

#### **RF02:** Cadastrar Aluno

- O sistema deve permitir o auto cadastro de Alunos, com geração automática de login (baseado no primeiro nome) e e-mail padrão.

#### **RF03:** Gerenciar Professores

- O sistema deve permitir ao Admin cadastrar novos Professores, associando-os a uma disciplina. Durante esse processo, o Admin pode substituir um professor existente e renomear a disciplina "Extra".

#### **RF04:** Excluir Usuário

- O sistema deve permitir ao Admin excluir Alunos (identificados por RA) ou Professores (identificados por Login), removendo-os de todos os arquivos de dados e credenciais.

**RF05: Carregar Dados Acadêmicos**

- O sistema deve carregar os dados dos alunos (alunos.csv). Admins podem carregar manualmente; Professores e Alunos carregam o arquivo padrão automaticamente ao logar.

**RF06: Lançar Notas**

- O sistema deve permitir o lançamento de notas (com validação de 0.0 a 10.0) e persistir a alteração. A permissão é restrita: Admins podem lançar em qualquer disciplina, Professores apenas na sua.

**RF07: Visualizar Tabela de Notas**

- O sistema deve exibir os dados dos alunos em uma tabela, aplicando regras de visualização e autorização:
- Alunos veem apenas seus próprios dados.
- Professores veem notas apenas da sua disciplina.
- A coluna "Extra" só é visível se estiver em uso.
- Botões da interface (como "Lançar Notas" ou "I.A.") são ocultos dinamicamente com base no nível de acesso.

**RF08: Visualizar Desempenho**

- O sistema deve permitir a visualização de detalhes de notas e gerar um gráfico de barras (matplotlib) para análise visual do desempenho de um aluno (restringindo a visão do Professor à sua disciplina).

**RF09: Analisar Desempenho**

- O sistema deve fornecer uma funcionalidade "I.A." (restrita ao Aluno) que analisa o desempenho (notas altas/baixas) e exibe uma mensagem motivacional.

**RF10: Consultar Corpo Docente**

- O sistema deve prover uma funcionalidade para visualizar a lista de professores e suas disciplinas (incluindo disciplinas "Vagas").

## Requisitos Não Funcionais (RNF)

Os requisitos não funcionais descrevem como o sistema opera, suas qualidades e restrições.

### RNF01: Plataforma e Interface

- O sistema opera em dois módulos (GUI/Python e CLI/C). A GUI utiliza uma arquitetura de janela única (que transita do login para a interface principal) e janelas modais (grab\_set) que bloqueiam a interação. A GUI requer as bibliotecas Pillow e matplotlib.

### RNF02: Persistência e Estrutura de Dados

- Todos os dados da aplicação (usuários, alunos, mapeamentos) são persistidos em arquivos de formato CSV, localizados em um subdiretório dados\_confidenciais.

### RNF03: Gerenciamento de Dados e Desempenho

- O sistema adota um modelo "Read-All, Write-All": todo o conteúdo dos arquivos CSV é carregado na memória RAM (arrays/listas) para operações, e qualquer modificação (cadastro, exclusão, nota) resulta na reescrita completa do arquivo (modo "w"), garantindo a limpeza de linhas em branco, mas com baixa eficiência de escrita.

### RNF04: Robustez

- O sistema garante a presença de cabeçalhos nos arquivos CSV (lendo, verificando e reescrevendo se ausente) e trata erros de codificação de caracteres (tentando utf-8 e latin-1).

### RNF05: Segurança e Restrições

- O sistema possui baixo nível de segurança (senhas em texto plano, senha de admin fixa). Não é portátil, pois depende de caminhos de arquivo absolutos (C:\Users\mayco\...) em ambos os módulos C e Python. A escalabilidade do módulo C é limitada por arrays de tamanho fixo (MAX\_ALUNOS 500).

## Metodologia Ágil: Scrum e Kanban

A Engenharia de Software Ágil caracteriza-se pelo uso de métodos iterativos e incrementais, priorizando a entrega contínua de valor e a adaptação rápida a mudanças (Pressman & Maxim, 2016). Entre as metodologias mais difundidas está o Scrum, que organiza o trabalho em ciclos curtos chamados sprints, define papéis claros e promove inspeção e adaptação constantes (Schwaber & Sutherland, 2020).

### Scrum em Detalhe

#### Papéis:

- **Product Owner:** Responsável por maximizar o valor do produto e priorizar o Product Backlog.
- **Scrum Master:** Atua como facilitador, removendo impedimentos e garantindo a aderência às práticas ágeis.
- **Time de Desenvolvimento:** Equipe multifuncional que entrega incrementos de software funcionais ao fim de cada Sprint.

#### Artefatos:

- **Product Backlog:** Lista ordenada e dinâmica de requisitos e funcionalidades.
- **Sprint Backlog:** Conjunto de itens selecionados para a Sprint atual, com plano de execução.
- **Incremento:** Soma de todos os itens concluídos, resultando em uma versão utilizável do sistema.

#### Eventos:

- **Sprint Planning:** Definição do objetivo e seleção das tarefas da Sprint.
- **Daily Scrum:** Reunião diária para alinhamento de progresso.
- **Sprint Review:** Apresentação do incremento e coleta de feedback.
- **Sprint Retrospective:** Reflexão sobre o processo e definição de melhorias.

O Scrum oferece um processo estruturado e colaborativo, ideal para ambientes complexos e em constante evolução, como o desenvolvimento de sistemas acadêmicos com IA.

## Kanban como Ferramenta de Apoio

Além do Scrum, a equipe adota o Kanban como ferramenta de apoio para o gerenciamento visual das tarefas. Inspirado no sistema Toyota de produção, o Kanban promove fluxo contínuo e eficiência operacional, permitindo visualizar o andamento do trabalho de forma clara (SOMMERVILLE, 2011).

Para a implementação prática desta abordagem visual, a equipe utilizou a plataforma Trello.com, que permite criar quadros digitais e gerenciar o fluxo de trabalho de forma colaborativa. O quadro Kanban no Trello permite acompanhar o andamento das atividades em tempo real, identificar gargalos e tornar a comunicação mais clara para toda a equipe. A visualização prática deste quadro e a organização das tarefas da equipe encontram-se ilustradas na Figura 8, apresentada no tópico 3.10.5.

### 3.4 Estrutura de Dados em Python

De acordo com Lutz (2013), Python é uma linguagem de programação de alto nível, interpretada e de tipagem dinâmica, amplamente empregada no desenvolvimento de aplicações modernas. Sua sintaxe simples e legível favorece a manutenção, a clareza e a produtividade no desenvolvimento de software. Este capítulo apresenta a fundamentação teórica da disciplina de Estrutura de Dados em Python, enfatizando os princípios da linguagem e seu papel no contexto do desenvolvimento de sistemas.

#### 3.4.1 Caracterização da Linguagem Python como Parte do Processo de Desenvolvimento de Sistemas

Segundo Lutz (2013), a linguagem Python combina a expressividade e a facilidade das linguagens de alto nível com recursos poderosos de abstração e modularidade, o que a torna ideal para o desenvolvimento rápido de aplicações (*Rapid Application Development* – RAD). No contexto do Sistema Acadêmico Colaborativo, Python foi caracterizada e adotada como parte essencial do processo de desenvolvimento, em virtude de suas propriedades de ser uma linguagem interpretada, multiplataforma e de tipagem dinâmica.



Além disso, Python é amplamente reconhecida por sua sintaxe clara e legível, o que contribui significativamente para a manutenibilidade do código e permite a aplicação eficiente de práticas de Engenharia de Software, como reutilização de componentes, modularização e testes automatizados.

### 3.4.2 Descrição do Processo de Desenvolvimento e Utilização dos Recursos de Python

O processo de desenvolvimento do sistema envolveu a utilização de recursos nativos e de bibliotecas externas da linguagem Python para a construção dos principais componentes do software. De acordo com Van Rossum e Drake Jr. (2009), a extensa biblioteca padrão e a capacidade de integração com outros módulos constituem uma das maiores forças da linguagem.

Os componentes e recursos utilizados no sistema e sua importância são descritos a seguir:

- **Módulo tkinter:** kit de ferramentas padrão para criação de Interfaces Gráficas do Usuário (GUI). Sua utilização foi crucial para atender ao requisito de criar soluções de interface eficientes, implementando janelas de login, tabelas de dados (*Treeview*) e botões interativos.
- **Módulos csv e os:** utilizados para o gerenciamento da persistência de dados (leitura e escrita de arquivos .csv), garantindo que as informações de alunos e credenciais fossem armazenadas de forma estruturada. Esses módulos também serviram de base para estratégias de interação de dados com outras plataformas.
- **Biblioteca matplotlib:** componente de visualização de dados fundamental para o sistema, permitindo a geração de gráficos de barras de desempenho acadêmico, agregando clareza e valor informativo aos relatórios.

A descrição desses componentes é essencial para o entendimento do software desenvolvido, pois eles representam as tecnologias que dão suporte à aplicação e demonstram a flexibilidade do ecossistema Python no desenvolvimento de sistemas interativos. Os resultados visuais da implementação dessa interface gráfica, incluindo

a tela de login e os dashboards de desempenho, estão demonstrados nas Figuras 6 e 7, localizadas nos tópicos 3.10.3 e 3.10.4, respectivamente.

### 3.4.3 Estruturas de Programação e Elementos de Python Utilizados

Os elementos de programação da linguagem Python consistem nas estruturas e na lógica empregadas para manipular dados e controlar o fluxo do sistema, seguindo os princípios da programação estruturada. Segundo Lutz (2013), as estruturas de dados de Python são projetadas para serem simples, porém poderosas, permitindo o gerenciamento eficiente de informações em diferentes contextos.

#### Estruturas de Dados:

- **Listas (*list*):** utilizadas como estrutura primária para armazenar coleções ordenadas de dados, como a lista de alunos (*dados\_alunos*) e as notas em vetores sequenciais.
- **Dicionários (*dict*):** estruturas do tipo chave-valor empregadas para mapear informações de forma eficiente, como o gerenciamento de credenciais (*CREDENCIAIS*) e o mapeamento de professores para suas respectivas disciplinas (*PROFESSOR\_TO\_COLNAME*).

#### Estruturas de Controle de Fluxo:

- **Decisão (*if/elif/else*):** utilizadas para o controle de acesso (login), validação de entrada de dados (notas) e habilitação de funções de acordo com o tipo de usuário (Administrador, Professor ou Aluno).
- **Repetição (*for*):** aplicadas em processos como a iteração sobre a lista de alunos para exibição na interface e no cálculo da média geral, garantindo que todas as notas sejam processadas de forma sistemática.

Essas estruturas reforçam a flexibilidade, clareza e poder expressivo da linguagem Python, consolidando-a como uma das principais ferramentas para o ensino e a aplicação prática de Estrutura de Dados.

### 3.5 Análise e Projeto de Sistemas

A análise de sistemas é a etapa responsável por identificar as necessidades e requisitos que o software deve atender, enquanto o projeto de sistemas traduz essas necessidades em uma solução técnica estruturada. Segundo Sommerville (2011), a modelagem de sistemas é o processo de desenvolver modelos abstratos de um sistema, sendo uma forma de documentar sua estrutura e funcionamento antes da implementação.

Para este projeto, a análise e a modelagem foram fundamentais para integrar os módulos em C e Python, garantindo que ambos operassem sobre a mesma base de dados. Foram utilizadas duas abordagens principais de modelagem: a UML (Unified Modeling Language) e o DFD (Diagrama de Fluxo de Dados).

#### **Processo de Desenvolvimento do Diagrama de Fluxo de Dados (DFD)**

O Diagrama de Fluxo de Dados (DFD) é uma técnica de modelagem que foca no movimento das informações através do sistema. Diferente dos fluxogramas que mostram a lógica de controle, o DFD mostra a origem, o destino, o armazenamento e o processamento dos dados.

Para o Sistema Acadêmico, o DFD foi desenvolvido identificando primeiramente as entidades externas (neste caso, os atores: Professor, Aluno e Administrador). Em seguida, mapearam-se os fluxos de dados (entradas como login e notas, e saídas como relatórios e feedback). Por fim, identificou-se o processo central (o sistema colaborativo) e os depósitos de dados (os arquivos CSV). O diagrama resultante, apresentado no capítulo de Desenvolvimento, ilustra como as permissões e os dados transitam entre os módulos.

#### **Processo de Modelagem UML**

A UML foi utilizada para representar a estrutura orientada a objetos (ou simulada através de estruturas) e o comportamento do sistema. Foram elaborados três tipos de diagramas:

- **Diagrama de Caso de Uso:** Focado nos requisitos funcionais, identificando os atores e suas interações com o sistema.

- **Diagrama de Classes:** Focado na estrutura estática, definindo as entidades (Aluno, Professor, Disciplina), seus atributos e relacionamentos.
- **Diagrama de Sequência:** Focado na dinâmica temporal, ilustrando a ordem das mensagens trocadas entre a interface, o autenticador e a base de dados durante processos críticos como o login.

A consolidação visual destes artefatos (DFD e UML) encontra-se detalhada no capítulo 3.10 Desenvolvimento do Projeto.

### 3.6 Redes de Computadores e Sistemas Distribuídos

Segundo Kurose e Ross (2017), uma rede de computadores consiste em dispositivos conectados entre si que compartilham recursos físicos e lógicos. No contexto do sistema acadêmico, será utilizado o modelo cliente-servidor, no qual os computadores dos usuários (máquinas clientes) poderão acessar serviços disponibilizados por um servidor central, como o banco de dados e o painel administrativo.

O sistema acadêmico proposto depende de uma infraestrutura de rede confiável e bem planejada, que permita a comunicação entre servidores, estações de trabalho e dispositivos móveis utilizados por alunos, professores e administradores. As redes de computadores são essenciais para o compartilhamento de dados, controle de acessos, autenticação e integração de sistemas distribuídos, como o banco de dados e os módulos de login e cadastro.

Os principais componentes da rede incluem:

- Switches e roteadores: responsáveis por encaminhar pacotes e interligar os diferentes setores da instituição.
- Servidores: dedicados à hospedagem do sistema acadêmico, banco de dados e serviços como DNS e DHCP.
- Estações de trabalho: computadores dos usuários (secretaria, professores e alunos) conectados via rede local (LAN).
- Pontos de acesso wireless: para conexão de dispositivos móveis e notebooks por Wi-Fi, garantindo mobilidade no campus.

## Tipos de Rede e Layout

A rede utilizada será uma LAN (*Local Area Network*) estruturada em topologia estrela, na qual todos os equipamentos são conectados a concentradores (*switches*) que distribuem o sinal. Essa configuração proporciona maior controle, facilidade de manutenção e melhor desempenho da rede, isolando falhas em conexões individuais.

A comunicação externa com servidores de *backup* ou serviços em nuvem poderá ocorrer via WAN (*Wide Area Network*), com uso de conexões seguras e criptografadas (VPN). O diagrama detalhado da topologia de rede planejada, bem como a distribuição lógica dos equipamentos na planta baixa da instituição, pode ser consultado na Figura 9, disponível no tópico 3.10.6.

## Configuração e Serviços de Rede

A rede será configurada para permitir a comunicação entre os setores administrativos e acadêmicos.

- Serviço DHCP: para atribuição automática de endereços IP aos dispositivos.
- Serviço DNS: para resolução de nomes e acesso aos recursos internos.
- Endereçamento IP: baseado em IPv4, com divisão em sub-redes para separar setores (ex.: secretaria, professores, alunos).
- Segurança: utilização de firewall e controle de acesso baseado em credenciais.

O sistema acadêmico proposto se enquadra no conceito de sistema distribuído, pois diferentes módulos (cadastro, login, notas, comunicação interna) funcionam de forma independente, mas coordenada, através da rede. Conforme Tanenbaum e Steen (2007), sistemas distribuídos permitem que vários computadores operem de maneira integrada, apresentando-se aos usuários como um sistema único e coerente.

No caso do sistema acadêmico, o servidor principal gerencia os dados centralmente, enquanto os usuários acessam as informações remotamente, garantindo escalabilidade, tolerância a falhas e segurança.

### 3.7 Educação Ambiental

A Educação Ambiental (EA) é um componente curricular fundamental, que visa à conscientização sobre a complexa relação entre as sociedades e o meio ambiente. No contexto do desenvolvimento de sistemas, esta disciplina orienta os futuros profissionais de Análise e Desenvolvimento de Sistemas (ADS) a incorporarem práticas de sustentabilidade desde a concepção de um projeto.

A abordagem da EA Crítica, proposta por Loureiro (2011), defende que a transformação ambiental é inseparável da transformação social, exigindo dos indivíduos uma postura ética e reflexiva sobre suas práticas, incluindo aquelas realizadas no ambiente de trabalho e no ciclo de vida do software.

No setor de Tecnologia da Informação (TI), a aplicação dos princípios da EA é conhecida como TI Verde ou Green IT. Trata-se de uma estratégia que busca reduzir o impacto ambiental gerado pela produção, uso e descarte de equipamentos e sistemas eletrônicos.

Conforme Loureiro (2011), a educação ambiental deve permear todas as áreas do conhecimento, e a TI não é exceção. A contribuição do Sistema Acadêmico Colaborativo para a sustentabilidade se materializa na eliminação do uso do papel.

"A adoção de tecnologias digitais para o gerenciamento de documentos e informações representa uma estratégia poderosa de redução de consumo de recursos naturais, como papel e energia, minimizando a pegada ecológica das instituições" (CAMPINA et al., 2024, p. 115).

Ao digitalizar processos como emissão de históricos e relatórios, o sistema proposto atua diretamente na redução da demanda por papel e impressoras. Portanto, o desenvolvimento do projeto está alinhado com a Educação Ambiental, demonstrando a importância de o profissional de ADS projetar sistemas com consciência ecológica.

### 3.8 Inteligência Artificial

A Inteligência Artificial (IA) no contexto educacional oferece mecanismos para personalizar a jornada do aluno. No Sistema Acadêmico desenvolvido, a IA é aplicada não como um sistema complexo de recomendação de novos conteúdos, mas como um mecanismo de análise de desempenho e feedback motivacional, focado diretamente na experiência do usuário "Aluno".

Esta funcionalidade está diretamente ligada ao Requisito Funcional RF09 (Analisar Desempenho (I.A.)) e ao Requisito Não Funcional RNF01 (Plataforma e Interface). O objetivo não é sugerir materiais de estudo externos, mas sim analisar os dados de desempenho existentes (as notas) e fornecer ao aluno um feedback proativo e personalizado sobre onde focar seus estudos.

#### **Análise de Implementação (Baseada no Código)**

Diferente de sistemas que utilizam aprendizado supervisionado ou IA Generativa, a implementação atual utiliza uma abordagem pragmática baseada em regras (heurística) para prover valor imediato. Esta abordagem está alinhada com conceitos de Sistemas Especialistas, onde a inteligência é encapsulada em regras de negócio claras para classificar o desempenho e sugerir ações (Russell & Norvig, 2021).

O mecanismo, implementado na função `analisar_por_ia()`, opera da seguinte forma:

1. **Restrição de Acesso:** A funcionalidade é exclusiva para o nível de acesso "Aluno". O sistema bloqueia ativamente o acesso de "Professores" e "Administradores" através da função `habilitar_botoes`.
2. **Análise de Dados:** O sistema coleta as notas do aluno logado (armazenadas na variável `dados_alunos`).
3. **Filtragem de Dados:** O sistema ignora notas 0.00 (assumindo que não foram lançadas) e também ignora a disciplina "Extra" se nenhum professor estiver associado a ela (verificando `PROFESSORES_POR_DISCIPLINA`).

4. **Identificação:** O sistema identifica a disciplina com a nota mais alta e a disciplina com a nota mais baixa (entre as notas válidas e visíveis).
5. **Análise de Contexto:** A média geral (`media_geral`) do aluno é calculada e usada como um classificador de desempenho (ex: abaixo de 5.0, entre 5.0 e 7.0, ou acima de 7.0).
6. **Geração de Feedback:** Com base nessa classificação, o sistema seleciona aleatoriamente (`random.choice`) uma mensagem de uma das três listas pré-definidas (motivacionais, parabens, neutras).
7. **Personalização:** A mensagem é personalizada usando f-strings do Python para inserir dinamicamente o nome da disciplina mais forte ou mais fraca do aluno. Por exemplo: `f"Não desista! Sua disciplina mais fraca é {disc_display_fraca} ({menor_nota:.2f})..."`
8. **Interface Modal (RNF01):** A análise é apresentada em uma nova janela (`Toplevel`) que é configurada como modal (`j.grab_set()`) e centralizada, bloqueando a janela principal até que o usuário feche o pop-up da I.A.

A visualização desta interface modal, onde o aluno recebe o feedback motivacional gerado por essa lógica, está ilustrada na **Figura 7** (Dashboard e Módulo de Análise), disponível no tópico **3.10.4**.

### Histórias de Usuário (Conforme Implementado)

A funcionalidade é centrada exclusivamente no aluno. Como aluno, quero receber um feedback motivacional e sugestões de foco (como a "Ação Rápida") baseadas no meu desempenho (notas altas e baixas), para entender onde devo concentrar meus esforços.

### Conclusão da Análise

A "Inteligência Artificial" deste projeto se materializa como um sistema de feedback inteligente e proativo. O sistema não utiliza IA Generativa nem aprendizado de máquina para recomendar conteúdos externos.

Em vez disso, ele cumpre o Requisito Funcional RF09: Fornece uma análise de dados real (identificação de pontos fortes e fracos) e transforma essa análise em



um conselho acionável e motivacional para o aluno. Esta abordagem foca na motivação do aluno, transformando dados brutos (notas) em um conselho personalizado, imediato e de fácil compreensão, alinhando-se aos princípios de sistemas baseados em conhecimento que auxiliam na tomada de decisão (Russell & Norvig, 2021).

### 3.9 Pesquisa, Tecnologia e Inovação

A pesquisa científica é o ponto de partida para o avanço tecnológico e para a criação de inovações que transformam a sociedade. Segundo Damiani (2025), a ciência constitui um empreendimento coletivo que busca compreender e aprimorar a realidade por meio de métodos sistemáticos, baseando-se na observação, na experimentação e na análise crítica dos resultados obtidos. Assim, a pesquisa é essencial para o desenvolvimento de soluções que ampliem o conhecimento e sirvam de base para novas tecnologias.

A tecnologia, por sua vez, representa a aplicação prática desse conhecimento científico, transformando ideias em produtos, processos e sistemas que melhoram a eficiência e o bem-estar humano. De acordo com Vasconcellos (2021), a inovação pode ser entendida como uma ideia que passou por alguma ação, resultando em benefícios concretos para as partes envolvidas. Ela é, portanto, o elo entre a pesquisa e a prática de mercado — um ciclo contínuo de geração de conhecimento, aplicação e aperfeiçoamento.

No contexto deste projeto, o desenvolvimento de um Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial (IA) exemplifica esse processo de integração entre pesquisa, tecnologia e inovação. A concepção do sistema parte de estudos e metodologias da área de computação, especialmente no campo da IA e do aprendizado de máquina, aplicando princípios científicos para resolver desafios práticos de comunicação e colaboração acadêmica. Por meio de técnicas de processamento de dados e automação, o sistema busca oferecer suporte inteligente a estudantes e docentes, otimizando tarefas como organização de atividades, compartilhamento de informações e interação em grupo.

A aplicação da IA representa uma inovação tecnológica relevante, pois permite que o sistema aprenda com as interações dos usuários e melhore continuamente seu desempenho. Conforme Cordaro (2024), a demanda é o principal fator motivador da inovação, neste caso, a necessidade de plataformas acadêmicas mais dinâmicas e acessíveis impulsionou a criação de uma solução que utiliza recursos de pesquisa científica para gerar impacto positivo no ambiente educacional.

Dessa forma, o projeto reflete o ciclo **pesquisa** → **inovação** → **prática de mercado**, apresentado por Damiani (2025): a pesquisa fornece o embasamento teórico e técnico, a tecnologia permite a implementação prática das ideias e a inovação resulta na criação de um produto aplicável, com potencial para contribuir com a transformação digital da educação.

#### **4 DESENVOLVIMENTO**

O desenvolvimento do sistema foi dividido em etapas, com o objetivo de garantir organização, modularidade e rastreabilidade do progresso. A seguir, são descritas as principais fases e os resultados alcançados.

##### **Planejamento e levantamento de requisitos:**

Nesta etapa, foram definidos os requisitos funcionais e não funcionais com base nas necessidades identificadas para o sistema acadêmico. O grupo determinou as funções essenciais, como autenticação de usuários, cadastro de professores e alunos, lançamento de notas e visualização de relatórios. Também foram consideradas as restrições técnicas e limitações da infraestrutura.

##### **Modelagem do sistema:**

A modelagem foi realizada utilizando diagramas UML, abrangendo caso de uso, classes e sequência. Essa etapa teve o propósito de representar graficamente a estrutura e o comportamento do sistema antes da implementação, facilitando a comunicação entre os desenvolvedores e garantindo alinhamento com os objetivos do projeto.

**Implementação do módulo em C:**

O módulo desenvolvido em linguagem C tem como função principal o cadastro e o gerenciamento básico de usuários. Ele utiliza uma interface de linha de comando (CLI) e arquivos no formato CSV para armazenamento persistente das informações. Esse módulo garante que os dados de alunos e professores sejam gravados e organizados adequadamente, servindo como base para o módulo gráfico em Python.

**Implementação do módulo em Python:**

O módulo Python foi responsável pela criação da interface gráfica (GUI), desenvolvida com a biblioteca Tkinter. Ele realiza a autenticação de usuários com diferentes níveis de acesso, apresenta tabelas de notas (Treeview) e permite o lançamento e consulta de dados em tempo real. Foram utilizadas também as bibliotecas csv, os, matplotlib e Pillow para persistência de dados, tratamento de arquivos e geração de gráficos de desempenho.

**Integração entre os módulos:**

A integração entre os módulos C e Python ocorre por meio da leitura e escrita dos mesmos arquivos CSV. Essa estratégia garante compatibilidade e sincronização dos dados entre as partes do sistema, funcionando como uma base comum e centralizada de informações acadêmicas.

**Implementação do módulo de Inteligência Artificial:**

O recurso de IA foi implementado com base em regras heurísticas simples, aplicadas à análise de notas. A funcionalidade coleta os dados de desempenho dos alunos, identifica os pontos fortes e fracos e gera uma mensagem motivacional personalizada. Essa abordagem, embora não utilize aprendizado de máquina, representa um primeiro passo na aplicação de inteligência computacional voltada à educação.

**Testes e validação:**

Após a implementação, o sistema passou por uma fase de testes unitários e integrados. Foram verificadas funcionalidades de login, cadastro, lançamento de

notas, persistência de dados e exibição correta de gráficos. Durante os testes, foram observados aspectos de usabilidade, desempenho e estabilidade da aplicação, garantindo que as funções essenciais estivessem operando conforme os requisitos definidos.

### **Documentação e entrega:**

Por fim, foi elaborada a documentação técnica e acadêmica do projeto, incluindo descrição dos requisitos, modelagem, fundamentação teórica e resultados obtidos. A documentação visa não apenas atender às exigências do PIM, mas também servir como guia para futuras melhorias e manutenções do sistema.

Em síntese, o desenvolvimento do Sistema Acadêmico Colaborativo com Apoio de IA demonstrou a integração prática dos conhecimentos adquiridos nas disciplinas do curso, resultando em uma aplicação funcional e coerente com as demandas do ambiente educacional moderno.

## **4.10 IMPLEMENTAÇÃO E RESULTADOS DO PROJETO**

O desenvolvimento do Sistema Acadêmico Colaborativo com Apoio de IA seguiu uma abordagem prática e iterativa. Esta seção apresenta os artefatos de modelagem gerados na fase de análise, os resultados da implementação técnica em C e Python, a gestão do projeto e a infraestrutura de rede planejada.

### **4.10.1 Artefatos de Modelagem do Sistema**

Antes da codificação, a estrutura lógica do sistema foi mapeada através de diagramas para garantir a integridade dos dados e a clareza dos requisitos.

#### **A. Diagrama de Fluxo de Dados (DFD)**

O DFD abaixo ilustra o fluxo de informações entre os usuários e o sistema, destacando como os dados de autenticação, notas e gestão transitam para o processo central.

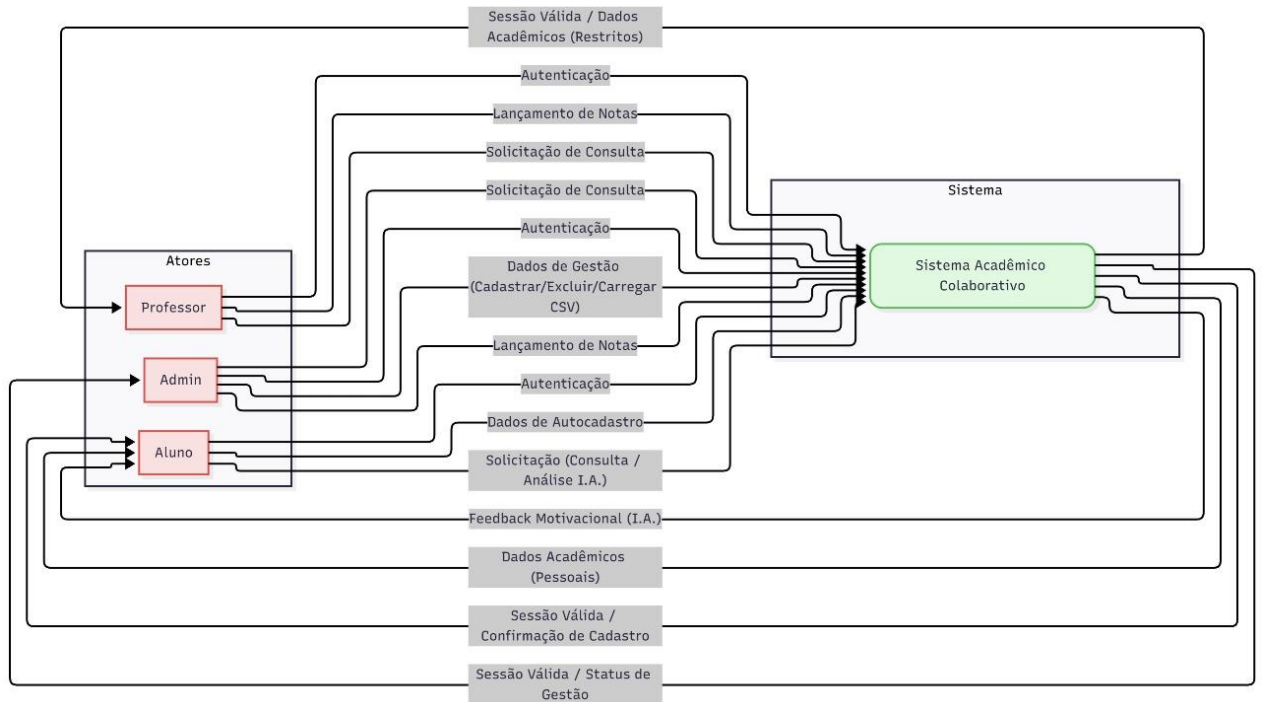


Figura 1 – Diagrama de contexto do Sistema Acadêmico

Fonte: Criado pelos autores, 2025.

Conforme observado na Figura 1, o sistema atua como um núcleo centralizador que recebe solicitações de três atores distintos, processa as permissões e retorna dados específicos (restritos para professores, pessoais para alunos e gerenciais para administradores).

## B. Diagramas UML

Para detalhar a estrutura e o comportamento, foram gerados os seguintes diagramas UML:

**Diagrama de Caso de Uso:** Apresentado na **Figura 2**, detalha as funcionalidades acessíveis a cada perfil, delimitando as fronteiras de atuação de Alunos, Professores e Administradores."



Figura 2 – Diagrama de Caso de Uso

Fonte: Criado pelos autores, 2025.

**Diagrama de Classes:** A estrutura estática do sistema, incluindo suas entidades, atributos e o relacionamento com a base de dados CSV, está representada no **Diagrama de Classes** da **Figura 3**.

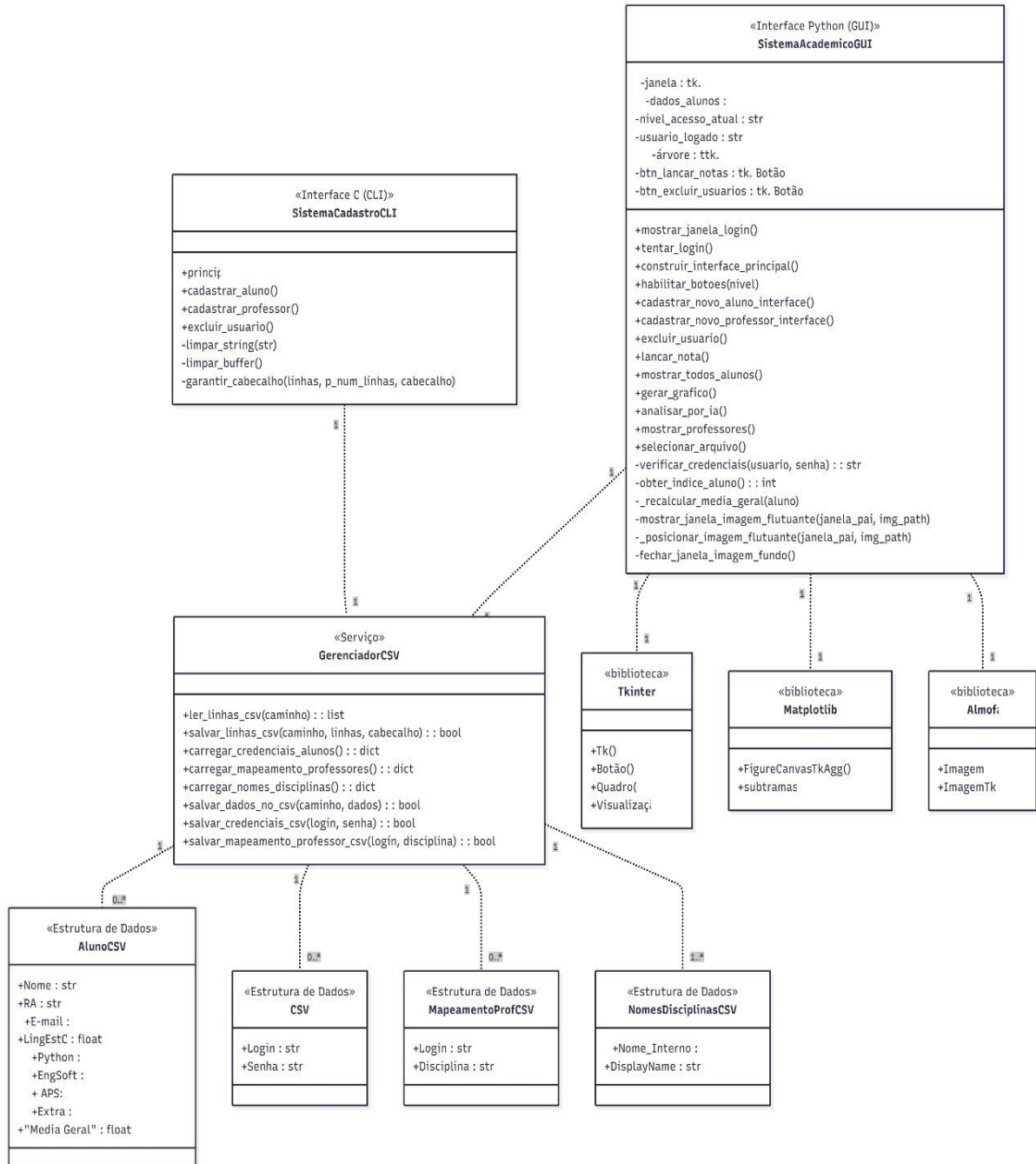


Figura 3 – Diagrama de Classes

Fonte: Criado pelos autores, 2025.

**Diagrama de Sequência:** O fluxo temporal de troca de mensagens durante o processo de autenticação é ilustrado no **Diagrama de Sequência** da **Figura 4**, evidenciando a interação entre interface e lógica de negócios.

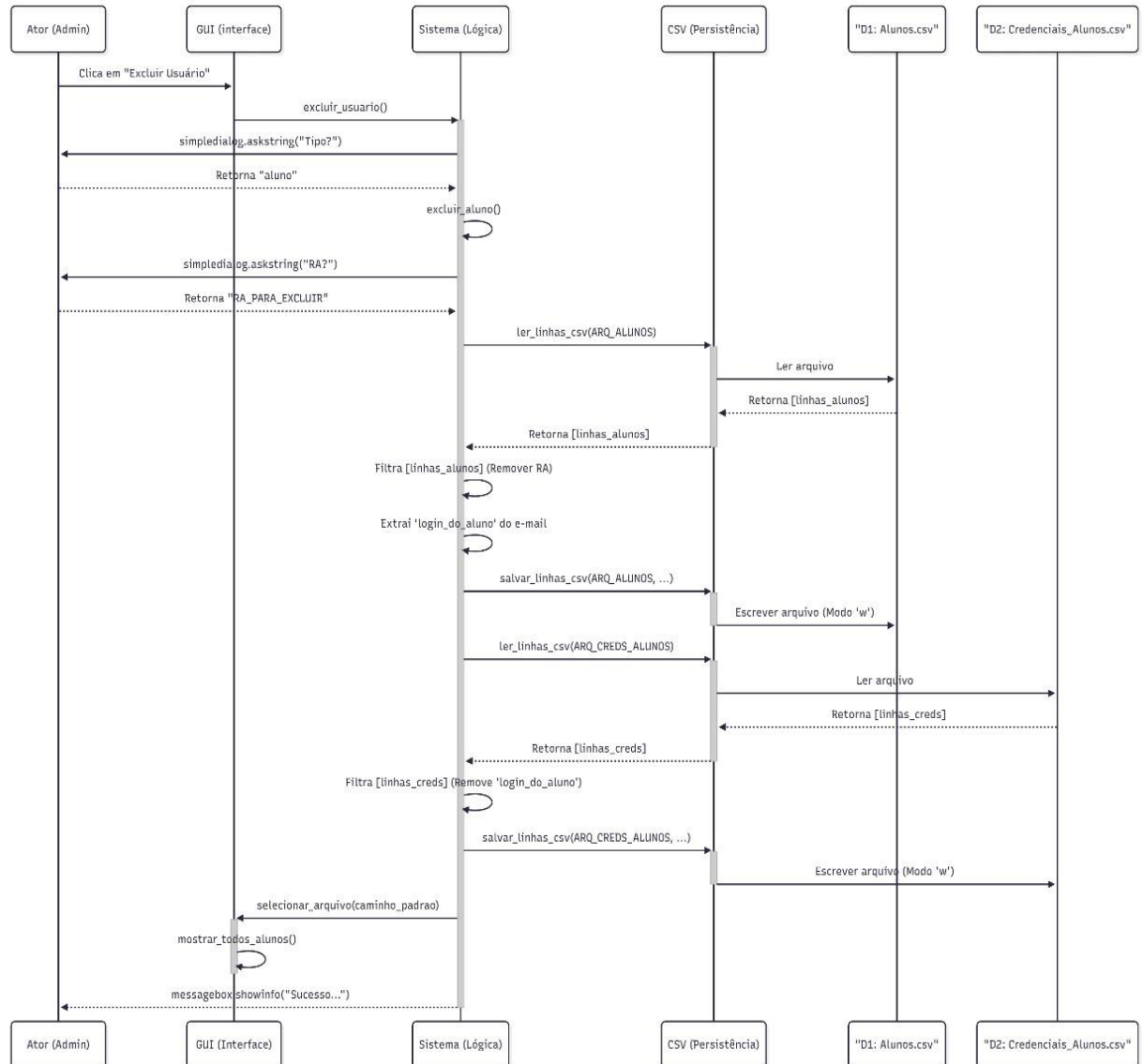


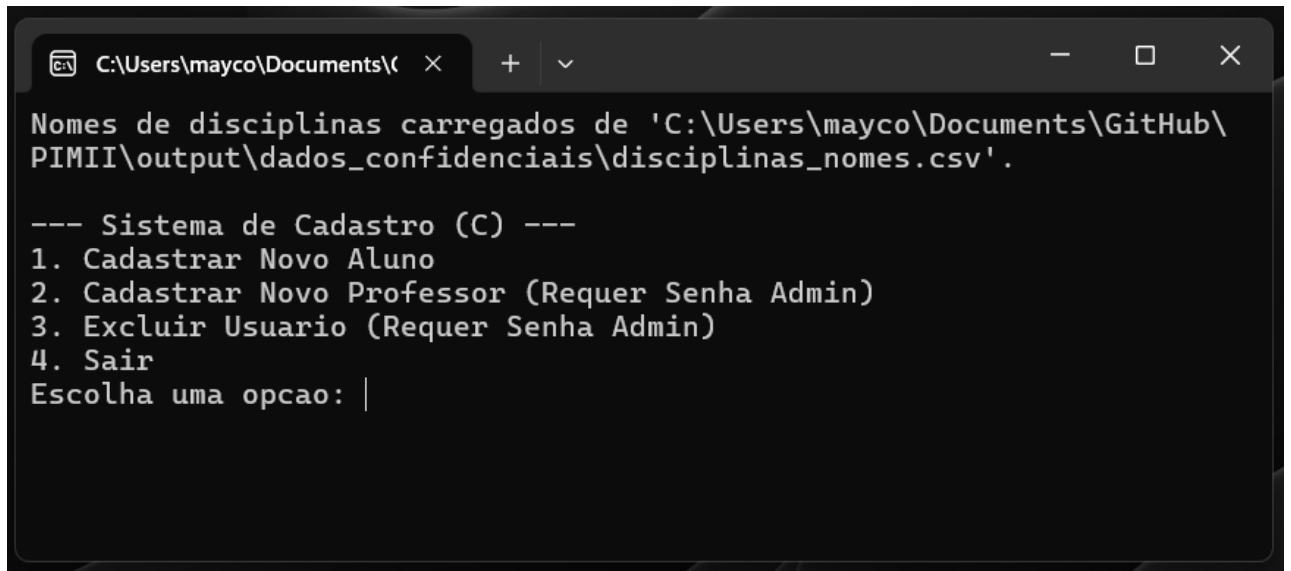
Figura 4 – Diagrama de Sequência

Fonte: Criado pelos autores, 2025.



#### 4.10.2 Implementação do Módulo em C (Back-End)

O módulo em C foi desenvolvido para operar via linha de comando, garantindo rapidez no cadastro massivo de usuários e a integridade inicial dos arquivos CSV, conforme demonstrado na interface da **Figura 5**.



```
C:\Users\mayco\Documents\...
Nomes de disciplinas carregados de 'C:\Users\mayco\Documents\GitHub\
PIMII\output\dados_confidenciais\disciplinas_nomes.csv'.

--- Sistema de Cadastro (C) ---
1. Cadastrar Novo Aluno
2. Cadastrar Novo Professor (Requer Senha Admin)
3. Excluir Usuario (Requer Senha Admin)
4. Sair
Escolha uma opcao: |
```

Figura 5 – Interface do módulo administrativo em C

Fonte: Criado pelos autores, 2025.

#### 4.10.3 Interface Gráfica em Python (Front-End)

A interface gráfica, desenvolvida com Tkinter, constitui o *front-end* da aplicação. A **Figura 6** demonstra a tela de login, porta de entrada para todos os níveis de acesso.

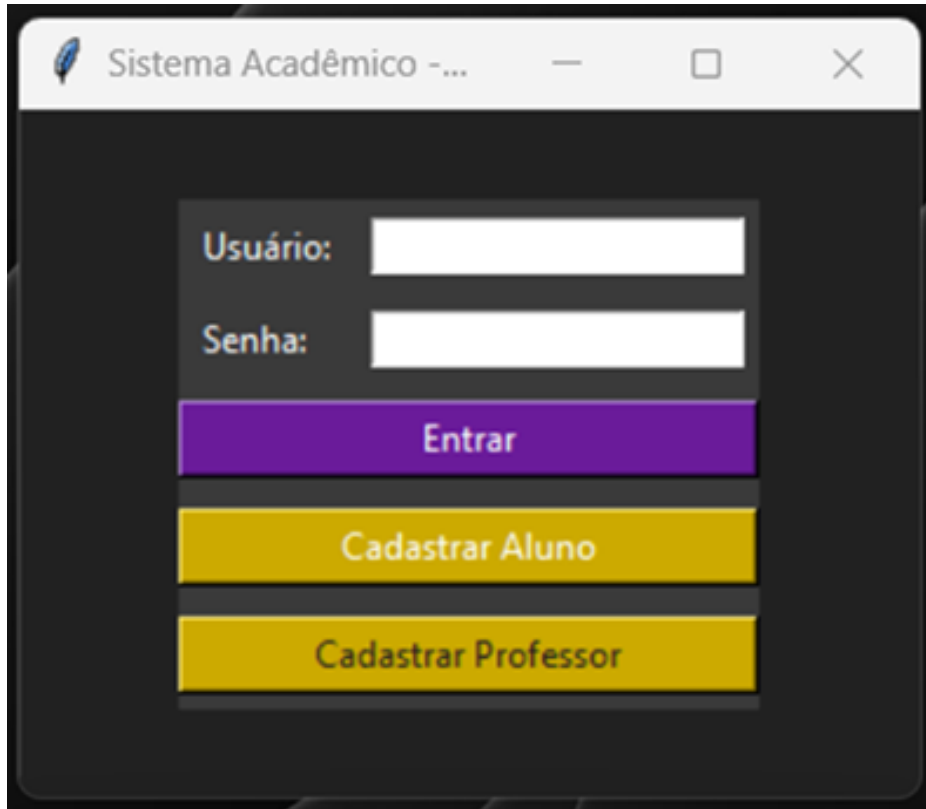


Figura 6 – Tela de Login da aplicação principal

Fonte: Criado pelos autores, 2025.

#### 4.10.4 Funcionalidades de Análise e Gráficos

Após a autenticação, o sistema apresenta dashboards personalizados. O destaque é a geração de gráficos de desempenho e o feedback motivacional via IA, exclusivo para o perfil de Aluno. A **Figura 7** apresenta a visualização integrada deste *dashboard*, exibindo a tabela de notas, o menu de ações e a janela modal de *feedback* da Inteligência Artificial.

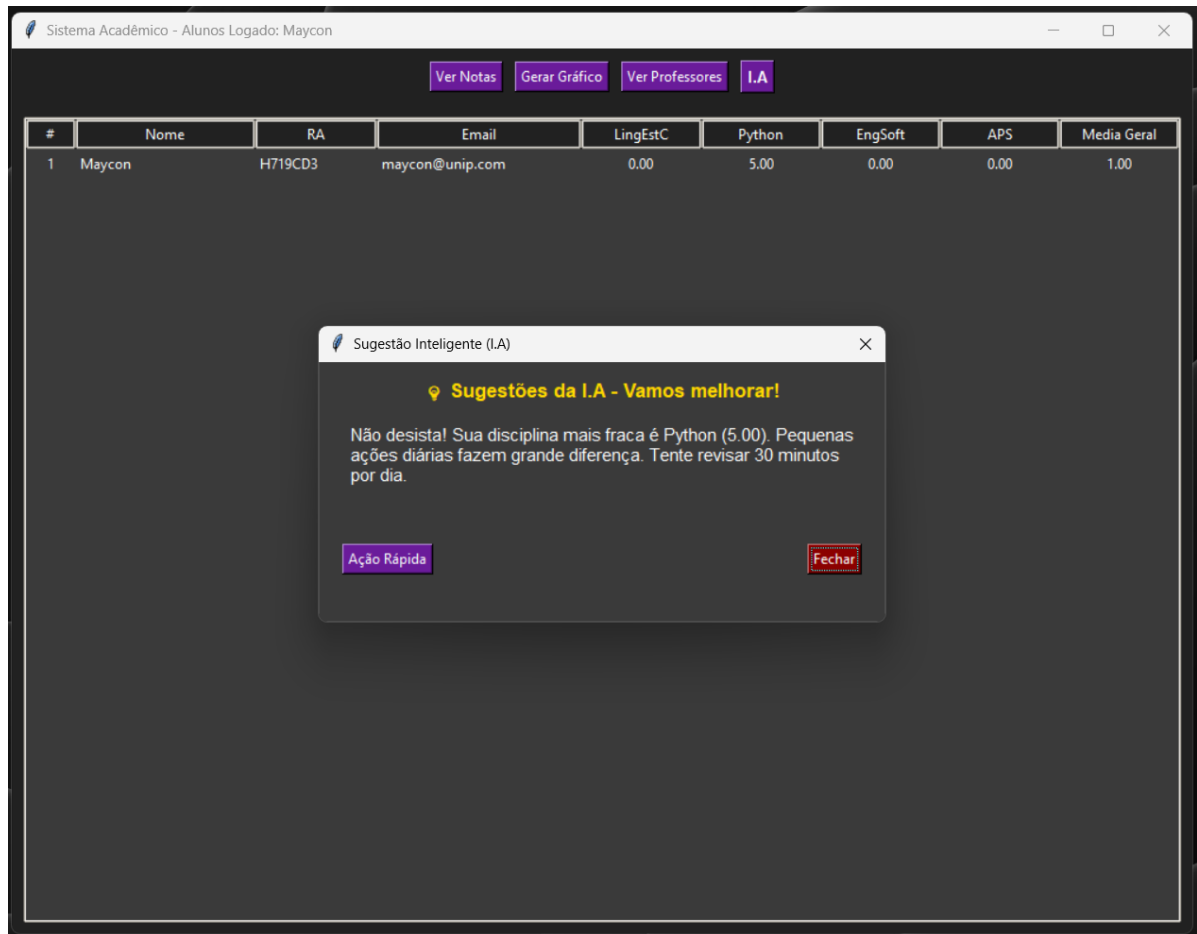


Figura 7 – Visualização de desempenho e módulo de análise

Fonte: Criado pelos autores, 2025.

#### 4.10.5 Gestão Ágil do Projeto (Kanban)

Para garantir a organização das tarefas e o cumprimento dos prazos estabelecidos, a equipe utilizou a metodologia Kanban aplicada através da plataforma Trello. Essa ferramenta permitiu a visualização clara do fluxo de trabalho, dividindo as atividades em colunas como "A Fazer" (Backlog), "Em Andamento" (Doing) e "Concluído" (Done).

A gestão visual facilitou a distribuição de responsabilidades entre os integrantes do grupo, permitindo identificar rapidamente gargalos no desenvolvimento e priorizar as funcionalidades críticas, conforme ilustrado no quadro Kanban da **Figura 8**.

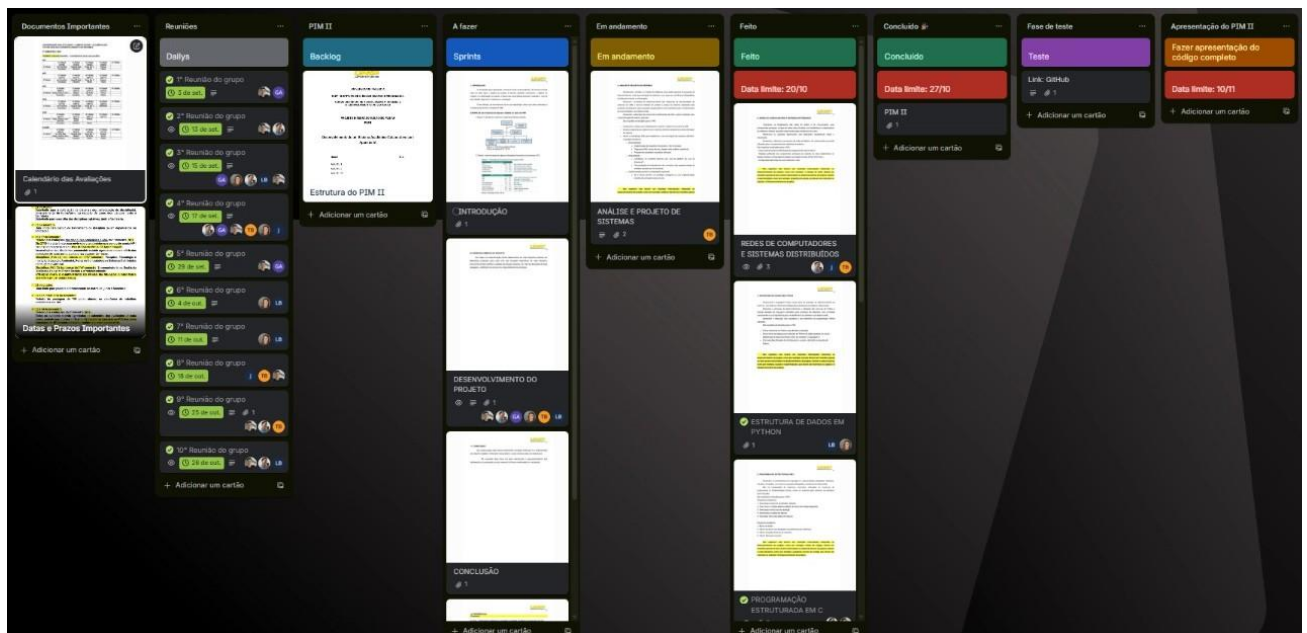


Figura 8 – Quadro Kanban utilizado para gestão do projeto

Fonte: Criado pelos autores, 2025.

#### 4.10.6 Infraestrutura de Rede e Layout (Planta Baixa)

Para suportar a execução do Sistema Acadêmico Colaborativo em ambiente distribuído, foi planejada uma infraestrutura de rede local (LAN) que interliga os setores críticos da instituição, conforme ilustrado na planta baixa lógica da **Figura 9**.

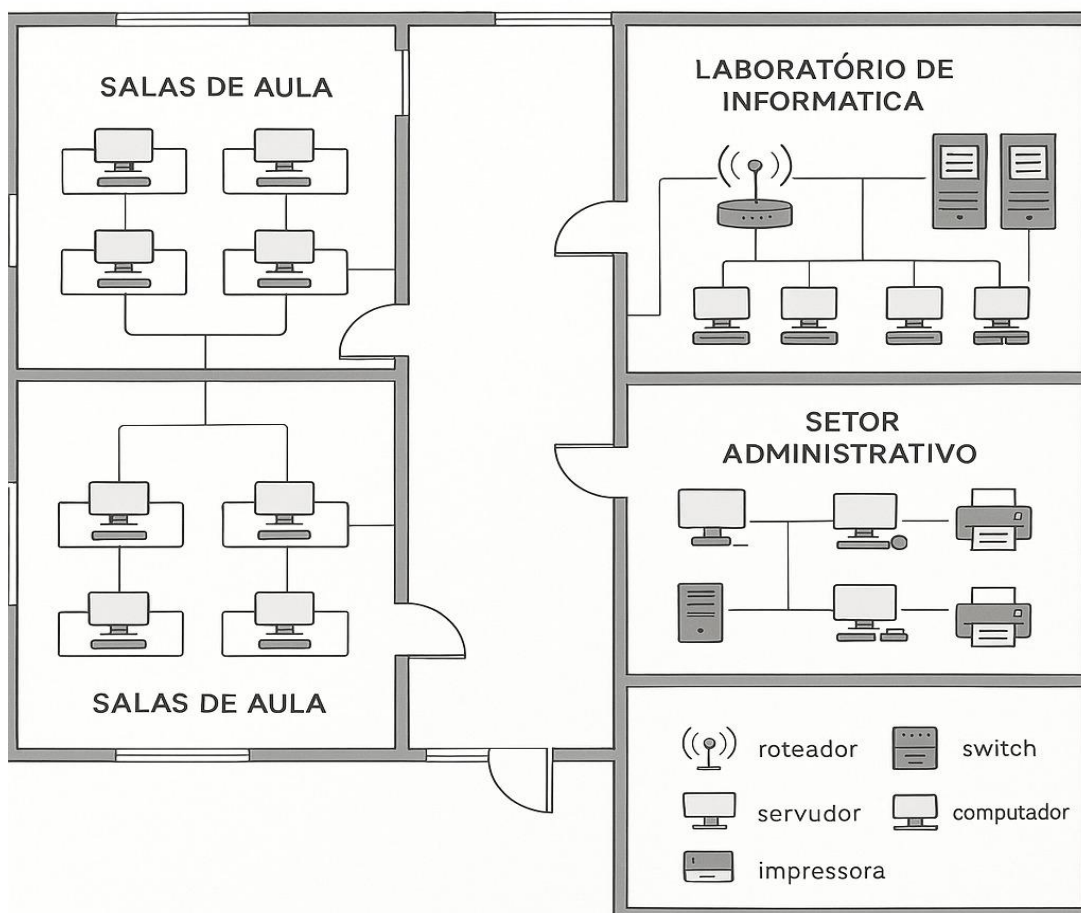


Figura 9 – Diagrama de Distribuição dos Equipamentos na Rede

Fonte: Criado pelos autores, 2025.

#### A. Layout de Distribuição

A distribuição física dos equipamentos segue uma topologia centralizada, segmentada por departamentos para garantir segurança e organização do tráfego:

- **Setor Administrativo:** Área crítica onde residem as impressoras de rede e computadores administrativos. É neste setor que se localizam os servidores principais responsáveis pelo armazenamento dos arquivos CSV (dados\_confidenciais), garantindo a segurança física dos dados.

- **Laboratório de Informática:** Equipado com servidores de apoio e um roteador wireless para prover conectividade móvel. As estações de trabalho são cabeadas para permitir que os alunos acessem o módulo Python (consulta e I.A.) com estabilidade.
- **Salas de Aula:** Ambientes com estações de trabalho interligadas à rede central, permitindo que professores realizem chamadas, lancem notas e consultem dados durante as aulas.

## **B. Relação Justificada dos Componentes**

1. **Cabeamento Estruturado:** Conecta as estações fixas das Salas de Aula e do Setor Administrativo, essencial para alta velocidade na leitura/escrita dos arquivos no servidor.
2. **Roteador Wireless:** Localizado no Laboratório, expande a rede para dispositivos móveis (notebooks e smartphones).
3. **Servidores:** Centralizam o armazenamento e o processamento das requisições. O diagrama destaca a redundância de servidores no laboratório e no administrativo.
4. **Impressoras:** Compartilhadas via rede no setor administrativo, otimizando recursos e centralizando a emissão de documentos oficiais.

## 5 CONCLUSÃO

O desenvolvimento do **Sistema Acadêmico Colaborativo com Apoio de Inteligência Artificial** consolidou a aplicação prática e integrada dos conhecimentos teóricos adquiridos no curso de Análise e Desenvolvimento de Sistemas. A execução deste projeto permitiu vivenciar integralmente o ciclo de vida de desenvolvimento de software, abrangendo desde o levantamento de requisitos e a modelagem do sistema até a sua implementação, validação e documentação técnica.

No decorrer do processo, a adoção de metodologias ágeis e a estratégia de modularização mostraram-se fundamentais para o sucesso da integração entre as linguagens **C** e **Python**, resultando em uma solução funcional e escalável. A escolha arquitetural por arquivos CSV como base de dados local validou-se como uma alternativa eficaz para garantir a persistência e a integridade das informações transitadas entre os módulos.

Destaca-se a implementação do módulo de Inteligência Artificial que, embora fundamentado em heurísticas, agregou valor significativo ao sistema ao prover *feedback* personalizado. Essa funcionalidade demonstrou como a tecnologia pode atuar estrategicamente no fomento ao engajamento estudantil e na melhoria da experiência de aprendizagem.

Conclui-se que o sistema atingiu os objetivos propostos, estabelecendo-se como uma ferramenta viável para auxiliar na organização de dados acadêmicos em instituições de ensino. Como perspectiva de trabalhos futuros, recomenda-se a evolução da camada de persistência para um banco de dados relacional, o fortalecimento dos protocolos de segurança de credenciais e a sofisticação do módulo de IA através de técnicas de aprendizado de máquina (*Machine Learning*).

## REFERÊNCIAS

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.

CAMPINA, N. N.; NASCIMENTO, F. M.; NASCIMENTO, D. R.; MIRANDA, M. C. Educação Ambiental. São Paulo: Editora Sol, 2024.

CORDARO, F. Pesquisa, Inovação e Prática de Mercado. São Paulo: Editora Sol, 2024.

DAMAS, Luís. Linguagem C. 10. ed. Rio de Janeiro: LTC, 2007.

DAMIANI, L. R. Pesquisa, Tecnologia e Inovação. São Paulo: Editora Sol, 2025.

KUROSE, James F.; ROSS, Keith W. Redes de Computadores e a Internet: uma abordagem top-down. 7. ed. São Paulo: Pearson, 2017.

LESSA, Vanessa Santos. Inteligência Artificial. São Paulo: Editora Sol, 2025.

LOUREIRO, C. F. B. Educação ambiental crítica: contribuições e desafios. 3. ed. São Paulo: Cortez, 2011.

LUTZ, Mark. Learning Python. 5th ed. Sebastopol: O'Reilly Media, 2013.

PRESSMAN, Roger S.; MAXIM, Bruce R. Engenharia de Software: Uma Abordagem Profissional. 8. ed. Porto Alegre: AMGH, 2016.

RUSSELL, Stuart; NORVIG, Peter. Inteligência Artificial: Uma Abordagem Moderna. 4. ed. Rio de Janeiro: LTC, 2021.

SCHWABER, Ken; SUTHERLAND, Jeff. The Scrum Guide. [S. l.]: Scrum Guides, 2020. Disponível em: <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf>. Acesso em: 29 out. 2025.

SOMMERVILLE, Ian. Engenharia de Software. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

TANENBAUM, Andrew S.; STEEN, Maarten Van. Sistemas Distribuídos: princípios e paradigmas. 2. ed. São Paulo: Pearson Prentice Hall, 2007.

VAN ROSSUM, Guido; DRAKE JR., Fred L. Python 3 Reference Manual. Scotts Valley, CA: CreateSpace, 2009.

VASCONCELLOS, E. Gestão da Inovação Tecnológica. São Paulo: Atlas, 2021.



## FICHA DE CONTROLE DO PIM

Grupo Nº: 02 Ano: 2025 Período: Diurno Orientadores: Coordeiro e Flávio

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

Alunos:

| RA      | Nome                             | E-mail                       | Curso | Visto do aluno |
|---------|----------------------------------|------------------------------|-------|----------------|
| R8327B0 | Cauã Brandão da Costa            |                              | ADS   |                |
| H67HJI4 | Gabriel Alves Moreira            |                              | ADS   |                |
| H67HJJ2 | Josué Abraão Carvalho dos Santos |                              | ADS   |                |
| H4062J7 | Luiz Gustavo da Silva Borges     |                              | ADS   |                |
| R280985 | Maciel Costa da Silva            | maciel.silva18@aluno.unip.br | ADS   |                |
| H719CD3 | Maycon Douglas Inácio Silva      | maycon.silva87@aluno.unip.br | ADS   |                |

Registros:

| Data do encontro | Observações  |
|------------------|--|
| 03/09            | 1ª Reunião: Conversamos com o Profº Cordeiro sobre a estrutura da atividade da NP1.                                    |
| 13/09            | 2ª Reunião: Trabalhamos na NP1 de Engenharia de Software.  |
| 15/09            | 3ª Reunião: Finalizamos a NP1 e organizamos as tarefas.  |
| 17/09            | 4ª Reunião: Reunião sobre AVA e PIM. Maciel iniciou o código, finalizamos o AVA e configuramos o processo ágil Kanban. |
| 29/09            | 5ª Reunião: Atualizando os tópicos: Engenharia de Software Ágil e Educação Ambiental.                                  |
| 04/10            | 6ª Reunião: Revisão do código em Python.   |
| 11/10            | 7ª Reunião: Atualização do código em Python.   |
| 18/10            | 8ª Reunião: Atualização e Revisão do código em C.  |
| 25/10            | 9ª Reunião: Rever o que falta no projeto e trabalhar na solução; Elaborar e discutir Relatório de Progresso PIM II.    |
| 26/10            | 10ª Reunião: Revisão do código; Revisão das matérias do PIM II; Planejamento das próximas sprints.                     |