**CEETEPS**

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**

**ETEC ILZA NASCIMENTO PINTUS**

**Enzo Daniel Guedes da Silva**

**Gustavo Soares Pimentel**

**Victor Diniz Bento**

**Victor Gabriel Ferreira de Souza**

**CADS: SISTEMA DE CADASTRO DE PATRIMÔNIO**

**São José dos Campos, São Paulo.**

**2025**

**CEETEPS**

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**

**ETEC ILZA NASCIMENTO PINTUS**

**Enzo Daniel Guedes da Silva**

**Gustavo Soares Pimentel**

**Victor Diniz Bento**

**Victor Gabriel Ferreira de Souza**

**CADS**

Trabalho de conclusão de Curso apresentado ao Curso Teórico em desenvolvimento de sistemas na Etec Ilza Nascimento Pintus, orientado pelo Profº Jean Lourenço da Costa, como requisito parcial para obtenção do título técnico em Desenvolvimento de Sistemas.

**São Paulo**

**2025**

**CEETEPS**

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**

**ETEC ILZA NASCIMENTO PINTUS**

**Enzo Daniel Guedes da Silva**

**Gustavo Soares Pimentel**

**Victor Diniz Bento**

**Victor Gabriel Ferreira de Souza**

**CADS: Sistema de Gerenciamento de Patrimônio**

Aprovado em \_\_/\_\_/\_\_ para obtenção do Curso Técnico em Desenvolvimento de Sistemas.

BANCA EXAMINADORA:

Professor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assinatura: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Professor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assinatura: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Professor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assinatura: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Professor: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assinatura: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**AGRADECIMENTOS**

Pela conclusão desse trabalho, agradecemos a todos que nos apoiaram e auxiliaram nesse caminho longo e duradouro. Somos gratos a todos que nos apoiaram e caminharam juntos convosco para a finalização do projeto, e com enorme felicidade podemos dizer que o projeto está finalizado, e assim, podemos continuar nosso caminho, obtendo experiência e conhecimento profissional para continuarmos melhorando e aprimorando nossa técnica na área profissional.

“E são tantas marcas  
Que já fazem parte  
Do que eu sou agora  
Mas ainda sei me virar”

Paralamas do Sucesso, banda brasileira.

**RESUMO**

O projeto foi criado com a finalidade de auxiliar na gestão eficiente de patrimônios da Etec, de modo que ela aja de maneira mais eficiente e menos errante. Atualmente, a instituição passa por dificuldades com relação a gestão patrimonial, onde os registros são feitos à mão, dificultando a sua eficiência e possibilitando erros humanos, que ocorrem repentinamente, devido à excessiva quantidade de itens patrimoniais presentes na instituição, e por falta de atenção, ocorrem erros feitos pelo sistema. Nesse processo, ocorrem erros de digitação referente ao patrimônio, repetição de códigos, e até mesmo a exaustão causada devido ser uma gestão feita à mão. Com o CADS, ele possibilita o registro desses patrimônios através de um site web responsivo, onde ele poderá alterar, inserir, e gerar relatórios para o seu sistema de cadastramento, corrigindo assim, os possíveis erros humanos. As tecnologias aplicadas foram o *Python* com *framework* Flask, o banco de dados responsável foi o *MySQL*. Para o armazenamento de informações e codificações foi utilizado *GitHub*.

Palavras-chave: Patrimônios, sistema de gestão patrimonial escolar, cadastros de patrimônios.

**ABSTRACT**

The project was created to assist in the efficient management of Etec's assets, so that it can act more efficiently and less erratically. Currently, the institution is experiencing difficulties in asset management, where records are made by hand, hindering its efficiency and allowing human errors, which occur suddenly due to the excessive amount of assets present in the institution, and due to lack of attention, errors made by the system occur. In this process, typing errors related to assets, repetition of codes, and even exhaustion caused by manual management occur. With CADS, it is possible to register these assets through a responsive website, where it can change, insert, and generate reports for its registration system, thus correcting possible human errors. The technologies applied were Python with the Flask framework, and the database responsible was MySQL. GitHub was used to store information and codes.

Keywords: Assets, system, school management, records, changes.

**LISTA DE FIGURAS**

Gráfico 1 – Investimentos em Sistema de Gestão Patrimonial

Figura 1 – Tecnologias utilizadas no projeto CADS

Quadro 1 – Comparativo das principais características

Imagem 1 – Interface do sistema *GLPI*

Imagem 2 – Interface do sistema *CMDBuild*

Imagem 3 – Interface do software *Asset Panda*

Quadro 2 – Levantamento de requisitos do projeto CADS

Quadro 3 – Requisitos funcionais do CADS

Quadro 4 – Requisitos não funcionais do sistema CADS

Tabela 1 – Backlog do produto

Gráfico 2 - Projeção de investimentos em gestão patrimonial

Gráfico 3 – Impactos da Ausência de Controle Patrimonial Adequado

Figura 2 – Diagrama de caso de uso do projeto CADS

Imagem 4 – Modelo lógico do banco de dados

Quadro 5 - Ilustração da tabela patrimônio.

Quadro 6 - Ilustração da tabela users

Figura 3 – Tela de login

Figura 4 – Tela principal

Figura 5 – Tela de filtro

Figura 6 – Relatórios do patrimônio

Figura 7 – Tela de alteração

Figura 8 – Diagrama de classe do projeto CADS

Quadro 7 - Comparativo CADS e demais soluções existentes

**SIGLAS E ABREVIATURAS**

CSS - *Cascading Style Sheets*

HTML - *Hypertext Markup Language*

SQL - *Structure Query Language*

MVC - *Model-View-Controller*

ERP - *Enterprise Resource Planning*

GRASP - *General Responsibility Assignment Software Patterns*

DB – *Database*

*PHP - Hypertext Preprocessor*

*CRUD - Create, read, update and delete.*

**SUMÁRIO**

[1 INTRODUÇÃO 12](#_Toc250891261)

[1.2 Hipóteses 15](#_Toc1659051274)

[1.3 Objetivo Geral 16](#_Toc583822321)

[1.3.1 Objetivos Específicos 16](#_Toc2068898107)

[1.4 Referencial Teórico 16](#_Toc355257753)

[2. CONTEXTUALIZAÇÃO TECNOLÓGICA 17](#_Toc442779287)

[2.1 Tecnologias utilizadas 18](#_Toc1342618776)

[2.1.1 Linguagem de programação 19](#_Toc824285604)

[2.1.2 Linguagens de marcação 20](#_Toc2128636477)

[2.1.3 MySQL e Banco de Dados 21](#_Toc695124576)

[2.1.4 Python e Framework Flask 22](#_Toc907531928)

[2.2 Soluções existentes 23](#_Toc1578425070)

[2.2.1 GLPI 24](#_Toc728436021)

[2.2.2 CMDBuild 25](#_Toc1742351426)

[2.2.3 Asset Panda 25](#_Toc513529846)

[2.3 Levantamento de requisitos 26](#_Toc2071164293)

[2.3.1 Definição dos Stakeholders 27](#_Toc392113410)

[2.3.2 Metodologia Ágil 28](#_Toc1915365224)

[2.3.4 Requisitos funcionais 29](#_Toc97938273)

[2.3.5 Requisitos não funcionais 31](#_Toc1202735197)

[2.3.6 Backlog do produto 33](#_Toc615926314)

[2.4 Gestão patrimonial 35](#_Toc1407994313)

[2.5 UML 37](#_Toc1274189941)

[3. DESENVOLVIMENTO 39](#_Toc806667626)

[3.1 Modelo de dados 40](#_Toc633963257)

[Fonte: Autores (2025) 42](#_Toc1129053582)

[3.2 Estrutura do site 42](#_Toc681373077)

[3.2.1 Arquitetura do sistema 47](#_Toc1560994485)

[3.3 Módulo 1 48](#_Toc1115137850)

[3.4 Deploy 49](#_Toc529862906)

[4. RESULTADOS E DISCUSSÃO 50](#_Toc1527925980)

[4.1 Resultados obtidos 51](#_Toc463070021)

[4.2 Comparativo com soluções existentes 51](#_Toc2026493724)

[4.3 Avaliação do cliente 52](#_Toc1499443231)

[4.4 Análise dos resultados 52](#_Toc568773599)

[5. TRABALHOS FUTUROS 53](#_Toc2074917405)

[REFERÊNCIAS 54](#_Toc1234526010)

[APÊNDICE A (Se tiver) 57](#_Toc1393245203)

[ANEXOS 57](#_Toc1478653061)

# 1 INTRODUÇÃO

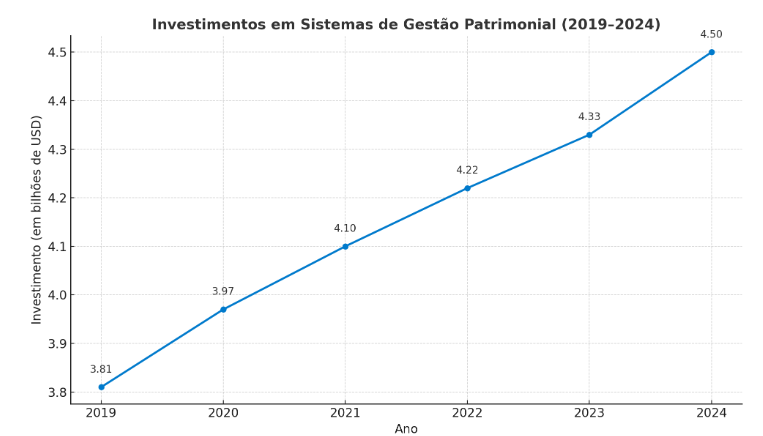
A gestão patrimonial em instituições de ensino representa um desafio constante, sobretudo quando realizada de forma manual. A ausência de um sistema automatizado compromete a eficiência, gera inconsistências nos registros, perdas financeiras e dificulta o controle dos bens públicos. Segundo Cury (2005), uma administração patrimonial eficaz é essencial para a transparência e otimização dos recursos públicos. No contexto da ETEC, essa realidade se faz ainda mais evidente. Como destaca Dantas (2020), a digitalização dos processos burocráticos não é mais uma alternativa, mas uma necessidade. O uso de planilhas e documentos físicos para controle de patrimônio está sujeito a falhas humanas e limita a agilidade na tomada de decisões. Além disso, erros como duplicidade de cadastros e extravio de materiais tornam-se recorrentes.

A evolução tecnológica tem impactado positivamente diversos setores da sociedade, inclusive o educacional. O Portal FGV (2022) aponta que o uso de ferramentas digitais aprimora significativamente a administração e oferece novos modelos de gestão. Investimentos em tecnologias da informação, capacitação de profissionais e integração dos setores são fatores que vêm impulsionando a modernização da gestão patrimonial em instituições públicas e privadas (Bernardes, 2008; Santos, 2016).

De acordo com a IDC (2024), os investimentos globais em transformação digital saltaram de US$ 1,18 trilhão em 2019 para mais de US$ 2,5 trilhões em 2024, evidenciando a tendência de substituição de métodos manuais por sistemas informatizados. A McKinsey (2023) reforça que mais de 90% das empresas adotaram estratégias de digitalização nesse período, visando maior confiabilidade nos dados e redução de custos. No Brasil, segundo o Sebrae, cerca de 76% das pequenas e médias empresas ainda não utilizam sistemas de gestão, mas 90% delas afirmam planejar esse investimento (Contábeis, 2024). Isso demonstra uma mudança gradual, mas crescente, no modo como as organizações compreendem a importância da transformação digital.

O Gráfico 1, a seguir ilustra os investimentos globais em tecnologias e transformação digital nos últimos anos, reforçando a urgência da automação nos processos administrativos.

Gráfico 1 – Investimentos em Sistema de Gestão Patrimonial

 Fonte: Statista Research Department, adaptado (2024)

Como afirma Maximiano (2006), o desempenho organizacional está diretamente ligado ao uso correto dos recursos disponíveis, sendo responsabilidade do gestor público empregar esses recursos de forma racional e eficiente. Viana (2006) complementa que o controle eficaz da cadeia de suprimentos é essencial para evitar desperdícios e manter a funcionalidade das instituições. Neste cenário, os sistemas web surgem como soluções viáveis e estratégicas. Laudon e Laudon (2019) afirmam que os sistemas de informação não apenas reduzem erros, mas também oferecem ferramentas de análise e controle cada vez mais sofisticadas. Entre as tecnologias disponíveis, o *Python* e o *microframework* Flask destacam-se por sua leveza, flexibilidade e simplicidade, sendo ideais para aplicações de pequeno e médio porte (Grinberg, 2018).

Diante disso, o presente trabalho tem como objetivo o desenvolvimento de um sistema *web* de cadastro e gerenciamento de patrimônios da ETEC de São José dos Campos. A proposta visa modernizar os processos internos da escola, garantindo maior controle, segurança, transparência e eficiência na administração dos bens patrimoniais.

**1.1 Justificativa**

O **CADS (Sistema de Cadastro de Patrimônio)** foi desenvolvido para suprir uma necessidade concreta da ETEC Profª Ilza Nascimento Pintus: a modernização do controle patrimonial. Atualmente, a escola enfrenta dificuldades relacionadas ao uso de processos manuais, que resultam em falhas recorrentes, como a duplicidade de cadastros, inconsistências nos registros e dificuldade na geração de relatórios precisos. Nesse contexto, a proposta do sistema visa promover uma gestão mais eficiente, segura e automatizada. De acordo com Sommerville (2019), o desenvolvimento de software deve sempre estar fundamentado em necessidades reais, com requisitos bem definidos e metodologias estruturadas. Um sistema bem planejado não apenas resolve demandas imediatas, mas também cria bases sólidas para futuras expansões e aprimoramentos. Nesse sentido, o CADS busca não apenas substituir métodos manuais, mas oferecer uma solução tecnológica acessível, escalável e de fácil uso para os colaboradores da instituição.

Segundo Cury (2005), uma gestão patrimonial eficiente é essencial para garantir o uso racional dos recursos públicos e facilitar o planejamento e controle administrativo. A digitalização do cadastro de bens escolares propicia maior confiabilidade nas informações, além de agilizar o acesso, a consulta e a realização de auditorias internas ou externas. A escolha pelo banco de dados *MySQL* também se justifica tecnicamente. Como observa Dufour (2018), aplicações que demandam alto desempenho e confiabilidade nas operações *CRUD* (*Create, Read, Update, Delete*) se beneficiam de bancos relacionais como o *MySQL*, que oferecem integridade de dados, suporte a transações e ampla documentação.

Diante disso, levanta-se o seguinte questionamento norteador: **de que forma um sistema digital de cadastramento pode contribuir para a organização, controle e consulta de patrimônios, promovendo uma gestão mais eficiente e confiável dentro de instituições públicas de ensino?**

## **1.2 Hipóteses**

* A digitalização do controle patrimonial reduzirá erros e inconsistências no cadastro dos bens, tornando o processo de registro mais ágil e confiável.
* A implementação do sistema facilitará a consulta, movimentação e auditoria dos patrimônios.
* A segurança das informações será aprimorada, evitando extravios e acessos não autorizados.

## **1.3 Objetivo Geral**

Desenvolver um sistema de gerenciamento de patrimônios da escola Etec Ilza Nascimento Pintus de São José dos Campos, proporcionando eficiência e segurança no controle dos bens.

### **1.3.1 Objetivos Específicos**

Para alcançar o objetivo principal, é necessário realizar os seguintes objetivos específicos:

* Desenvolver um sistema web para gerenciar patrimônios.
* Desenvolver um aplicativo (app) para cadastro de patrimônios.
* Implementar um sistema de banco de dados.
* Implementar níveis de controle de acesso para usuários.
* Assegurar um sistema seguro e de fácil usabilidade.
* Emitir um relatório dos ativos da escola.

Como sintetiza Pressman (2016), a qualidade de um sistema é medida não apenas por sua funcionalidade, mas por sua capacidade de resolver problemas reais de forma eficiente e sustentável.

## **1.4 Referencial Teórico**

O referencial teórico será baseado em autores como Cury (2005), que discute a importância da gestão patrimonial no setor público, além de Pressman (2016), Laudon e Laudon (2019), e Gil (2019), que abordam a implementação de sistemas de informação, engenharia de *software* e metodologias de pesquisa.

Portanto, espera-se que esse projeto contribua para a organização e modernização da gestão escolar, aprimorando a eficiência administrativa, segurança da informação e a transparência na utilização dos recursos patrimoniais.

# 2. CONTEXTUALIZAÇÃO TECNOLÓGICA

Este capítulo apresenta os fundamentos teóricos e práticos utilizados para o desenvolvimento do sistema, além de descrever a metodologia adotada, as tecnologias empregadas, o andamento do projeto e a modelagem utilizando a UML. Os tópicos estão organizados em: tecnologias utilizadas, conceitos, metodologia utilizada, gestão patrimonial e diagramas.

## 2.1 Tecnologias utilizadas

***Visual Studio Code***: É um editor de código-fonte gratuito e multiplataforma, muito utilizado no desenvolvimento de *software*, que suporta extensões, integração com o *Git* (Sistema de controle de versões) e realce de sintaxe para várias linguagens. Além disso, é um editor leve e de fácil acesso, poderoso e muito popular, que inclui suporte embutido para *JavaScript*, *TypeScript* e *Node.js*, e possui um vasto ecossistema de extensões. (MICROSOFT, 2024).

***Python:*** É uma linguagem de programação de alto nível, popular pela sua simplicidade e legibilidade, ela é utilizada em diversas áreas, como desenvolvimento *web*, automação de tarefas e serviços *web*, ciência de dados e inteligência artificial. *Python* é uma linguagem poderosa e fácil de aprender, que oferece estruturas de alto nível para desenvolvimento rápido e eficaz. (DEITEL; DEITEL, 2020).

***Flask***(*Microframework Python*): É um *framework web* leve e flexível para *Python*, recomendado para aplicações simples e médias. Permite a criação de APIs e sistemas web com poucas linhas de códigos. Em resumo, *Flask* é um *microframework* minimalista, que permite desenvolver aplicações web com rapidez e simplicidade. (GRINBERG, 2018).

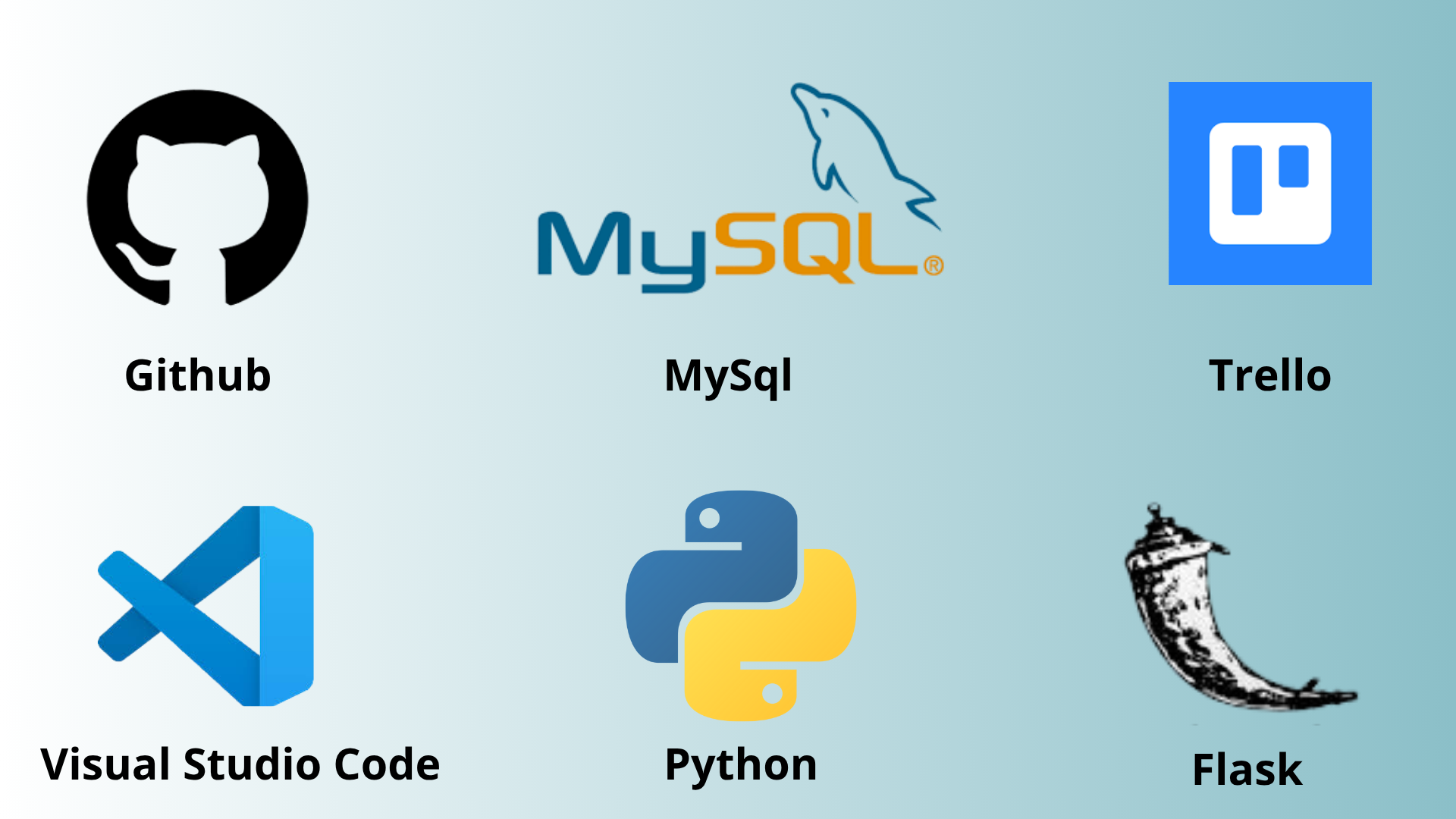
***Github:*** É uma plataforma de hospedagem de código-fonte baseada em *Git*, voltada à colaboração e versionamento, ela permite controle de versões, trabalho em equipe e integração contínua. Em síntese, *GitHub* oferece uma interface web para o sistema *Git*, promovendo colaboração e controle de versões de maneira eficiente. (CHACÓN; STRAUB, 2014).

***Trello***: Popularmente conhecido por ser uma ferramenta de gerenciamento de projetos baseada na metodologia *Kanban*. Utiliza quadros, listas e cartões para organizar e acompanhar tarefas de forma visual. Sinteticamente, Trello é uma aplicação de gerenciamento de projetos que usa cartões e listas para representar tarefas e fluxos de trabalho. (ATLASSIAN, 2024).

***MySQL***: Sistema de gerenciamento de banco de dados relacional (SGBD), baseado em SQL. É seguro, rápido e amplamente utilizado em aplicações web. MySQL é um sistema de gerenciamento de banco de dados que oferece alta performance e escalabilidade, sendo ideal para aplicações que exigem robustez. (ELMASRI; NAVATHE, 2011).

Segue a figura a seguir, referente as tecnologias que foram utilizadas para a criação do projeto CADS

Figura 1 – Tecnologias utilizadas no projeto CADS.



### 2.1.1 Linguagem de programação

Linguagens de programação são ferramentas essenciais para o desenvolvimento de softwares, pois permitem a comunicação entre seres humanos e máquinas. Elas fornecem um conjunto de regras sintáticas e semânticas que possibilitam escrever algoritmos, ou seja, instruções que os computadores podem entender e executar. (GUNTER, 1992)

Segundo Deitel e Deitel (2020), “uma linguagem de programação serve como meio de comunicação entre o programador humano e o computador, permitindo criar soluções digitais para problemas do mundo real”. As linguagens são classificadas, de forma geral, em dois grandes grupos: linguagens de baixo nível e alto nível. Linguagens de baixo nível, como Assembly, estão mais próximas da linguagem de máquina e exigem um maior conhecimento da arquitetura do *hardware*. Elas oferecem maior controle sobre os recursos do sistema, mas tendem a ser mais complexas e difíceis de manter. Já as linguagens de alto nível, como Python, Java e C#, são projetadas para serem mais compreensíveis ao ser humano, com sintaxe mais próxima da linguagem natural.

Essas linguagens facilitam o desenvolvimento de sistemas complexos e são as mais utilizadas atualmente no desenvolvimento de aplicações *web*, *mobile* e corporativas. Além disso, linguagens de programação podem ser categorizadas por paradigmas, como o imperativo, orientado a objetos, funcional e lógico. Cada paradigma oferece uma maneira distinta de organizar e estruturar o código, influenciando diretamente na forma como os problemas são resolvidos. *Python*, por exemplo, suporta múltiplos paradigmas, tornando-se uma linguagem versátil para diversos tipos de aplicações (Lutz, 2013).

As linguagens de programação permitem traduzir algoritmos em instruções compreensíveis ao computador, viabilizando o desenvolvimento de sistemas complexos (BEZERRA, 2011).

Para a criação do projeto CADS, foram utilizados o *Python* e o *framework Flask* para o funcionamento do site, segue a seguir o link para instalação dessas tecnologias:

*Python:* <https://www.python.org/downloads/>

*Framework Flask:* <https://flask.palletsprojects.com/en/stable/installation/>

### 2.1.2 Linguagens de marcação

Diferente das linguagens de programação, as linguagens de marcação não servem para processar dados, mas sim para estruturar e apresentar informações. Elas são utilizadas para organizar o conteúdo em documentos digitais, especialmente páginas da web. A mais conhecida é a HTML (*HyperText Markup Language*), padrão para criação de sites e aplicações web. Segundo Castro e Hyslop (2013), “HTML não é uma linguagem de programação, mas uma linguagem de marcação usada para estruturar o conteúdo da web”. Conforme Laudon e Laudon (2019), as linguagens de marcação representam um dos pilares para a interface entre usuários e sistemas, garantindo acessibilidade, padronização e integração com outras tecnologias da web.

Linguagens de marcação utilizam "tags" (etiquetas) para delimitar blocos de informação, definindo títulos, parágrafos, tabelas, links e imagens. Junto com o CSS (*Cascading Style Sheets*), responsável pela formatação visual, o *HTML* permite o desenvolvimento de interfaces amigáveis ao usuário. Além do *HTML*, outras linguagens de marcação como *XML* (*Extensible Markup Language*) são utilizadas para representar dados estruturados de forma legível tanto por humanos quanto por máquinas, muito comum em troca de informações entre sistemas.

Para o projeto CADS, foi utilizado o *Visual Studio Code* para a criação do site *web*, segue a seguir o link para a instalação da tecnologia:

*Visual Studio Code*: <https://code.visualstudio.com/download>

## 2.1.3 MySQL e Banco de Dados

O *MySQL* é um dos sistemas de gerenciamento de banco de dados mais populares do mundo, sendo um **banco relacional** que utiliza a linguagem *SQL* (*Structured Query Language*) para manipulação de dados. De acordo com Elmasri e Navathe (2011), “bancos de dados relacionais armazenam os dados em tabelas com colunas e linhas, permitindo relacionamentos entre diferentes conjuntos de dados por meio de chaves primárias e estrangeiras". O MySQL é amplamente utilizado em aplicações web, sistemas corporativos e plataformas educacionais devido à sua confiabilidade, desempenho e código aberto. Ele é especialmente eficiente para operações CRUD (*Create*, *Read*, *Update*, *Delete*), muito comuns em sistemas de cadastro, como o proposto neste TCC.

Além do MySQL, existem outros bancos de dados relacionais como *PostgreSQL*, *Oracle Database*, *Microsoft SQL Server* e *SQLite*. Todos seguem o modelo relacional proposto por Codd (1970), baseado na teoria dos conjuntos. No entanto, há também os bancos de dadosnão relacionais (*NoSQL*), como *MongoDB*, Cassandra e *Firebase*, que foram desenvolvidos para atender novas demandas, como o armazenamento de grandes volumes de dados não estruturados e a escalabilidade horizontal. Bancos relacionais são ideais para dados estruturados e integridade referencial, sendo muito utilizados em sistemas administrativos, financeiros e educacionais. Já os bancos não relacionais são indicados para aplicações que lidam com dados dinâmicos, de alto volume e sem uma estrutura rígida, como redes sociais, IoT e sistemas baseados em nuvem. Segundo Sadalage e Fowler (2013), “os bancos *NoSQL* oferecem flexibilidade e desempenho para tipos de dados que não se encaixam bem no modelo tabular tradicional”.

Segue a seguir o link para instalação da tecnologia utilizada para a criação do banco de dados:

MySQL: <https://www.mysql.com/downloads/>

## 2.1.4 Python e Framework Flask

O *Python* é uma linguagem de programação de alto nível, interpretada e de propósito geral, conhecida por sua sintaxe clara e legibilidade, o que facilita o aprendizado e acelera o desenvolvimento de sistemas. Seu *design* prioriza a produtividade do programador e a simplicidade do código. Segundo Deitel e Deitel (2020), *Python* é uma linguagem poderosa e fácil de aprender, que oferece estruturas de alto nível para desenvolvimento rápido e eficaz. Além de sua simplicidade, *Python* é extremamente versátil, sendo utilizado nas mais diversas áreas da tecnologia: desenvolvimento *web*, ciência de dados, automação, inteligência artificial, engenharia de *software* e muito mais. Como reforça Lutz (2013), “*Python* combina clareza de código com uma ampla gama de bibliotecas, tornando-a ideal para desenvolvedores de todos os níveis”. No projeto CADS, *Python* foi escolhido como linguagem base por sua integração facilitada com bancos de dados e *frameworks web*, além da vasta comunidade e documentação robusta. Sua flexibilidade permite o uso de diferentes paradigmas de programação — como o imperativo, orientado a objetos e funcional — o que contribui para uma estrutura de sistema modular, reutilizável e de fácil manutenção.

Para a camada *web*, foi utilizado o *Flask*, um *microframework* desenvolvido em *Python* que fornece as ferramentas necessárias para a construção de aplicações *web* com rapidez e leveza. *Flask* é considerado um "micro" *framework* não por ser limitado, mas por não impor camadas ou bibliotecas específicas, permitindo ao desenvolvedor escolher livremente os componentes que deseja integrar à aplicação. Como explica Grinberg (2018), “*Flask* é ideal para projetos pequenos e médios, proporcionando controle total da aplicação com uma curva de aprendizado amigável”. Entre os benefícios do *Flask*, destacam-se a simplicidade na criação de rotas, a integração facilitada com bancos de dados como o *MySQL* e a estrutura enxuta baseada no padrão *WSGI* (*Web Server Gateway Interface*). Ele é amplamente utilizado em startups, instituições educacionais e até em protótipos de grandes empresas por sua agilidade no desenvolvimento e testes.

No caso do sistema CADS, o uso do *Flask* foi estratégico: além de ser leve e eficiente para projetos de médio porte, ele possibilita a criação de interfaces web dinâmicas e responsivas com baixo consumo de recursos. Sua estrutura modular também permite que o sistema cresça gradualmente, facilitando futuras implementações, como autenticação de usuários, dashboards e relatórios. Com essa arquitetura, *Python* e *Flask* proporcionam uma solução tecnológica acessível, escalável e sustentável, atendendo às necessidades da ETEC no gerenciamento digital de seus bens patrimoniais.

Segue a seguir o link para instalação das tecnologias *Python* e o *framework Flask:*

*Python:* <https://www.python.org/downloads/>

*Framework Flask:* <https://flask.palletsprojects.com/en/stable/installation/>

## 2.2 Soluções existentes

Nesse tópico, abordaremos sobre outros tipos de solução existentes para o nosso problema, ou seja, realizaremos um comparativo com outros softwares que possuem a mesma funcionalidade que a nossa.

Quadro 1 – Comparativo das principais características

|  |  |  |  |
| --- | --- | --- | --- |
| Características | GLPI | CMDBuild | Asset Panda |
| Custo | Grátis | Grátis (*open source*) | Pago |
| Licença | GPL v3 | AGPL | Proprietária |
| Público-alvo | Público e empresas. | Uso público, alta customização. | Empresas com necessidade mobile |
| Funcionalidades | Inventário, *help desk,* manutenção | Custom apps, facilities, ITSM | Rastreamento, auditoria mobile |
| Plataforma | *Web (PHP)* | *Web (Java)* | *Web/Cloud + Mobile* |
| Escabilidade | Pequeno a médio porte | Médio a grande porte, flexível | Médio porte, altamente customizável |

## 2.2.1 *GLPI*

O *GLPI* (*Gestionnaire Libre de Parc Informatique*) é um *software* gratuito e de código aberto, licenciado sob a *GPL v3*, muito utilizado para gestão de inventário e suporte técnico. Desenvolvido em *PHP*, é baseado em plataforma web, permitindo acesso remoto via navegadores. Entre suas principais funções estão o cadastro e controle de ativos, a gestão de chamados de suporte (*help desk*), monitoramento de manutenção preventiva e corretiva, além da geração de relatórios gerenciais detalhados e precisos.

Uma das principais vantagens do *GLPI* é sua flexibilidade e personalização, permitindo a integração com outros sistemas e a instalação de diversos plugins (um programa de computador usado para adicionar funções a outros programas maiores, provendo alguma funcionalidade especial ou muito específica.) para ampliar recursos. Por ser *open source*(*softwares* cujo código-fonte é disponibilizado publicamente para que qualquer pessoa possa visualizá-lo, modificá-lo e distribuí-lo), não possui custos de licenciamento, o que reduz despesas para organizações de pequeno e médio porte. Além disso, conta com uma comunidade ativa que mantém atualizações e suporte colaborativo.

*GLPI* também apresenta desvantagens, como a necessidade de conhecimentos técnicos para instalação, configuração e manutenção do servidor, o que pode exigir uma equipe de TI dedicada. Além disso, sua interface, embora funcional, pode parecer menos intuitiva para usuários iniciantes quando comparada a soluções comerciais mais modernas. A imagem 1 a seguir, ilustrará a interface do sistema em questão.

Imagem 1 – Interface do sistema *GLPI*

Fonte: GLPI, (2025)

Link para página: <https://glpi-project.org>

Link para código-fonte: <https://github.com/glpi-project/glpi>

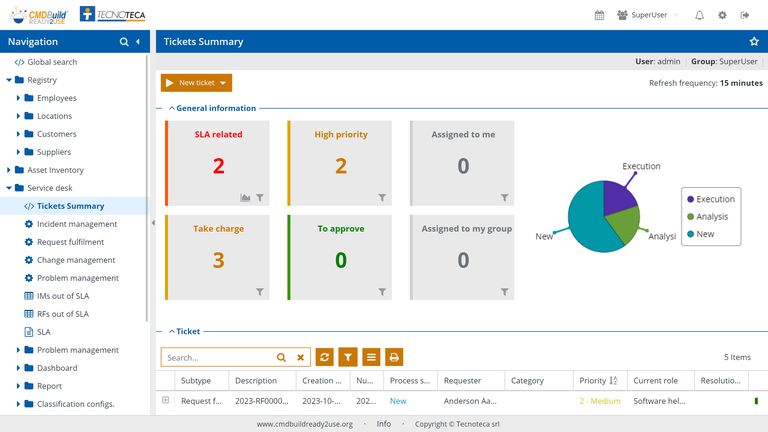
## 2.2.2 *CMDBuild*

O *CMDBuild* é uma plataforma *open source* licenciada sob a *AGPL*, desenvolvida em Java, voltada à gestão de ativos físicos e de infraestrutura. Uma de suas versões mais conhecidas, o *openMAINT*, é direcionada para o gerenciamento de *facilities*, manutenção de prédios, móveis e inventário de bens móveis e imóveis. Por ser uma plataforma de desenvolvimento de baixo código, permite a criação de aplicações personalizadas adaptadas às necessidades específicas de cada organização.

Entre suas vantagens, destacam-se a alta escalabilidade e a possibilidade de configurar fluxos de trabalho (*workflows*) complexos, tornando-o adequado para médias e grandes empresas. Ele oferece módulos para gestão de manutenção, controle de documentos, rastreamento de bens e integração com sistemas de georreferenciamento (*GIS*).

Por outro lado, sua complexidade pode ser considerada uma desvantagem, especialmente para empresas que buscam soluções mais simples e rápidas de implementar. A curva de aprendizado é relativamente alta e, assim como o *GLPI*, requer infraestrutura de servidor e equipe técnica para configuração e suporte contínuo. A imagem 2 a seguir, ilustrará a interface do sistema *CMDBuild*.

Imagem 2 – Interface do sistema *CMDBuild*

Fonte: CMDBuild, (2025)

Link para o site CMDBuild: <https://www.cmdbuild.org>

Link para o site openMAINT:<https://www.openmaint.org>

Link para Código-fonte: <https://github.com/tecnoteca/cmdbuild>

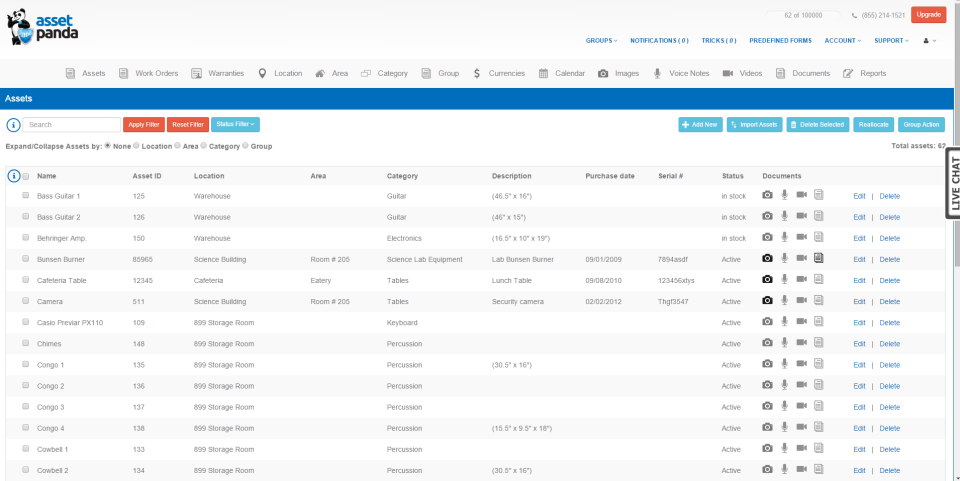
## 2.2.3 Asset Panda

O Asset Panda é uma solução comercial voltada à gestão de ativos com forte ênfase na mobilidade e acessibilidade em nuvem. Diferente das soluções anteriores, não é gratuito, mas oferece um período de teste. É acessado por navegadores e aplicativos móveis, permitindo o rastreamento de bens por meio de códigos de barras, QR *codes* e etiquetas *RFID*. Suas funcionalidades incluem controle de inventário, auditoria de ativos, manutenção preventiva, gerenciamento de documentos e integração com outros sistemas corporativos.

Sua maior vantagem é a facilidade de uso: a interface é moderna, intuitiva e permite operação sem conhecimentos técnicos avançados. Além disso, por ser hospedado em nuvem, não exige infraestrutura própria, o que agiliza a implantação e reduz a necessidade de manutenção interna. O suporte ao cliente é especializado e contínuo, atendendo demandas rapidamente.

Como desvantagens, está o custo de licenciamento, que pode ser elevado dependendo do número de usuários e funcionalidades contratadas. Além disso, por ser um serviço hospedado em servidores externos, sua utilização depende totalmente de conexão à internet e da política de segurança da empresa provedora. A imagem a seguir ilustrará a interface do software.

Imagem 3 – Interface do software *Asset Panda*

Fonte: PCMag, 2025

Site oficial: <https://www.assetpanda.com>

### 2.3 Levantamento de requisitos

Durante o desenvolvimento do CADS, realizamos diversas reuniões com a equipe administrativa da ETEC Ilza Nascimento Pintus, incluindo a colaboradora Renata, para identificar as necessidades prioritárias e secundárias do sistema. Essas discussões nos permitiram classificar os requisitos em obrigatórios (essenciais para o funcionamento básico do sistema) e desejáveis (que agregam valor, mas não são críticos para a operação inicial).

Abaixo, apresentaremos a Tabela 2, que organiza esses requisitos, incluindo tanto funcionais (o que o sistema faz) quanto não funcionais (como o sistema deve funcionar).

Quadro 2 – Levantamento de requisitos do projeto CADS

|  |  |  |  |
| --- | --- | --- | --- |
| Categoria | Requisito | Desejável | Obrigatório |
| Cadastro de bens | O sistema deve permitir o cadastro de bens com informações básicas (nome, número de patrimônio, localização, status). |  | X |
| Autenticação | O sistema deve exigir login e senha para acesso. |  | X |
| Inventário | O sistema deve permitir a realização de inventários via leitura de QR *Code*. |  | X |
| Relatórios | O sistema deve gerar relatórios básicos de bens cadastrados e movimentações. |  | X |
| Sincronização | O aplicativo mobile deve funcionar offline e sincronizar dados quando houver conexão. |  | X |
| Desempenho | O sistema deve responder em menos de 3 segundos para consultas básicas. | X |  |
| Acessibilidade | A interface web deve ser compatível com leitores de tela. | X |  |

## 2.3.1 Definição dos Stakeholders

No projeto CADS – Controle e Administração Digital de Sistemas Patrimoniais, os stakeholders são todas as pessoas ou grupos que vão ajudar a implementar ou se beneficiar do sistema. São eles:

* **Instituição e gestão escolar:**
  + A ETEC Profª Ilza Nascimento Pintus, que terá mais controle e segurança sobre os bens da escola.
  + A gestão, que precisa de relatórios claros para tomar decisões e manter a organização do patrimônio.

* **Equipe de desenvolvimento:**
  + Os integrantes do grupo de TCC — Enzo Daniel Guedes da Silva, Gustavo Soares Pimentel, Victor Diniz Bento e Victor Gabriel Ferreira de Souza — responsáveis por criar, testar e entregar o sistema funcionando.
* **Pessoas de apoio:**
  + Professor orientador (Jean Lourenço da Costa), que dá suporte técnico e acompanha o andamento do projeto.

**Quem vai usar o CADS?**

O sistema será usado principalmente por quem cuida da parte administrativa e de manutenção da ETEC. Eles vão ter acesso para cadastrar, consultar e atualizar os patrimônios. Gestores e coordenadores vão poder gerar relatórios e ter uma visão geral do que a escola possui.

## **2.3.2 Metodologia Ágil**

A engenharia de *software* é a área responsável por todos os processos relacionados ao desenvolvimento de sistemas, desde a análise e especificação até a manutenção contínua. Segundo Sommerville (2019, p. 12), “a engenharia de *software* é uma disciplina que lida com todos os aspectos da produção de *software* — desde os estágios iniciais de especificação do sistema até a sua manutenção após estar em uso”.

Com o tempo, surgiram as metodologias ágeis como resposta à rigidez dos métodos tradicionais, como o modelo cascata (*Waterfall*), que segue etapas sequenciais e pouco flexíveis. As abordagens ágeis priorizam a colaboração, adaptação contínua e entregas incrementais de valor. Elas possibilitam maior flexibilidade e respostas rápidas às mudanças ao longo do desenvolvimento (BECK et al., 2001).

Dentre essas metodologias, destaca-se o *Scrum*, um *framework* leve e iterativo, baseado em ciclos curtos chamados sprints. De acordo com Sutherland (2020), “*Scrum* é um *framework* para desenvolver e sustentar produtos complexos, baseado em transparência, inspeção e adaptação contínua”. Cada *sprint* (de duas a quatro semanas) tem por objetivo entregar um incremento funcional do produto. O processo envolve três papéis principais: *Product Owner*, *Scrum Master* e a equipe de desenvolvimento (SCHWABER; SUTHERLAND, 2020).

Ao final de cada *sprint*, realiza-se uma revisão para apresentar os resultados e discutir melhorias. Esse ciclo contínuo promove entregas frequentes e alinhadas às necessidades do cliente. Apesar de suas vantagens, o *Scrum* exige alta dedicação e organização da equipe, o que pode ser um desafio em contextos menos estruturados.

A base do desenvolvimento ágil está no Manifesto Ágil (2001), que estabelece quatro valores principais:

* Indivíduos e interações mais que processos e ferramentas;
* *Software* em funcionamento mais que documentação abrangente;
* Colaboração com o cliente mais que negociação de contratos;
* Responder a mudanças mais que seguir um plano.

Além disso, o manifesto apresenta 12 princípios, como entregas frequentes, aceitação de mudanças tardias, cooperação entre equipe e cliente, simplicidade e melhoria contínua. Segundo *Atlassian* (2024), o ágil não se limita à área de *software*, influenciando também áreas como *marketing*, educação e gestão de produtos.

Em comparação com o modelo *Waterfall*, onde o escopo é fixo e o tempo ajustável, no ágil o tempo é fixo e o escopo é adaptável, permitindo maior flexibilidade e foco na entrega de valor contínuo.

### 2.3.4 Requisitos funcionais

Nesse tópico será abordado sobre os requisitos funcionais obtidos através da reunião com o nosso cliente.

Requisitos funcionais são descrições claras das funções que um sistema deve executar para atender às necessidades de seus usuários e alcançar os objetivos para os quais foi desenvolvido. Eles definem “o que o sistema deve fazer” e representam as funcionalidades, serviços e comportamentos esperados durante sua operação. No caso de um sistema de gestão patrimonial, esses requisitos abrangem atividades como o cadastro de bens, registro de movimentações, controle de manutenção, inventário, geração de relatórios e envio de notificações. São itens diretamente relacionados às interações do usuário com o sistema, e não aos aspectos técnicos de desempenho ou segurança. Um requisito funcional bem definido garante que desenvolvedores e usuários tenham um entendimento comum sobre as funcionalidades que estarão presentes no produto final, evitando ambiguidades e garantindo que o sistema atenda exatamente às demandas estabelecidas no início do projeto. (SOMMERVILLE, 2019.)

O sistema **CADS** (Controle e Administração de Dados Patrimoniais) é uma solução sob demanda desenvolvida para atender às necessidades de gestão patrimonial da **ETEC Ilza Nascimento Pintus**, a pedido da colaboradora Renata, integrante da equipe administrativa da instituição. Seu objetivo é centralizar e otimizar o gerenciamento de bens, permitindo maior controle, rastreabilidade e transparência sobre os ativos da escola.

A plataforma será disponibilizada em versão web e aplicativo mobile (Android e iOS), com integração total entre os dois ambientes, garantindo que todas as informações sejam sincronizadas em tempo real. O sistema contemplará funcionalidades como o cadastro e edição de bens, registro de movimentações, execução de inventários por meio de leitura de códigos QR ou de barras, controle de manutenções, cálculo de depreciação e emissão de relatórios e *dashboards*.

O aplicativo será voltado principalmente para operações em campo, como conferência de bens e registro de movimentações, permitindo o funcionamento em modo offline e sincronização posterior. Já a versão web será focada na gestão estratégica, geração de relatórios, configuração de usuários e análise de dados.

A tabela a seguir ilustrará os requisitos funcionais obtidos, descrição e sua extensão.

Quadro 3 – Requisitos funcionais do CADS

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requisito funcional | Descrição | Plataforma |
| RF001 | Cadastro de bens | Permitir cadastrar bens com informações de nome, categoria, número de patrimônio, entre outros. | Web + App |
| RF002 | Edição de itens | Permitir alterar itens cadastrados, como o nome, imagem, sua condição e entre outros. | Web + App |
| RF003 | Upload de fotos | Permitir anexar imagens e arquivos relacionados ao bem. | Web + App |
| RF004 | Consulta de bens | Pesquisar bens por código, nome, categoria ou localização. | Web + App |
| RF005 | Registro de movimentações | Registrar transferências de bens. | Web + App |
| RF006 | Controle de usuários | Criar perfis com permissões diferentes (Administrador, gestor, colaborador). | Web |
| RF007 | Relatórios | Emitir relatórios em PDF/Excel, categoria, data, status. | Web |

### 2.3.5 Requisitos não funcionais

Os Requisitos Não Funcionais (RNFs) estabelecem os critérios de qualidade e desempenho que o sistema deve atender, definindo "como o sistema deve funcionar" em vez de "o que o sistema deve fazer" (SOMMERVILLE, 2019). Eles abrangem aspectos como segurança, usabilidade, desempenho e confiabilidade, garantindo que a solução atenda às expectativas técnicas e operacionais da instituição. Diferentemente dos requisitos funcionais, que tratam das funcionalidades específicas, os RNFs focam na experiência do usuário e na infraestrutura necessária para suportar o sistema de forma eficiente e segura (PRESSMAN, 2016).

O CADS foi desenvolvido como uma solução sob demanda para a ETEC Ilza Nascimento Pintus, atendendo a uma solicitação da equipe administrativa, representada pela colaboradora Renata. Além das funcionalidades essenciais, o sistema foi projetado para oferecer um ambiente robusto, intuitivo e seguro, garantindo que a gestão patrimonial seja realizada com eficiência e confiabilidade.

A plataforma, disponível em versão web e aplicativo mobile (Android e iOS), foi desenvolvida com uma arquitetura que prioriza a integração contínua entre os ambientes, assegurando que todas as informações sejam sincronizadas em tempo real. Para isso, foram estabelecidos requisitos não funcionais. A tabela 3 a seguir, representará os requisitos não funcionais (BORDAS SUPERIOR E INFERIOR PARA TíTULO, INFERIOR PARA DEMAIS ITENS, RETIRAR BORDAS LATERAIS DA TABELA.) (BORDA

Quadro 4 – Requisitos não funcionais do sistema CADS

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Requisito não funcional | Descrição | Plataforma |
| RNF 001 | Desempenho | O sistema deve responder às requisições em menos de 2 segundos em condições normais de uso | Web |
| RNF 002 | Segurança | Todos os acessos devem ser autenticados por login e senha, com suporte a dois fatores (2FA) para funções administrativas | Web + App |
| RNF 003 | Usabilidade | A interface deve seguir as diretrizes de acessibilidade WCAG 2.1, garantindo inclusão para todos os usuários. | Web + App |
| RNF 004 | Disponibilidade | O sistema deve manter uma disponibilidade de 99% durante o horário comercial (7h às 22h). | Web |
| RNF 005 | Portabilidade | A versão web deve ser compatível com os principais navegadores (Chrome, Firefox, Edge, Safari). | Web + App |
| RNF 006 | Resiliência | O aplicativo mobile deve operar em modo offline, armazenando dados localmente e sincronizando quando a conexão for restabelecida. | Web + App |

### 2.3.6 Backlog do produto

Nesse tópico será introduzido o backlog do nosso produto, composto pelos critérios de avaliação, histórias de usuário, seu grau de importância (sendo eles a baixa, média e alta.) e entre outros quesitos. A Tabela a seguir ilustrará o backlog do produto.

Tabela 1 – Backlog do produto (Alterar os demais)

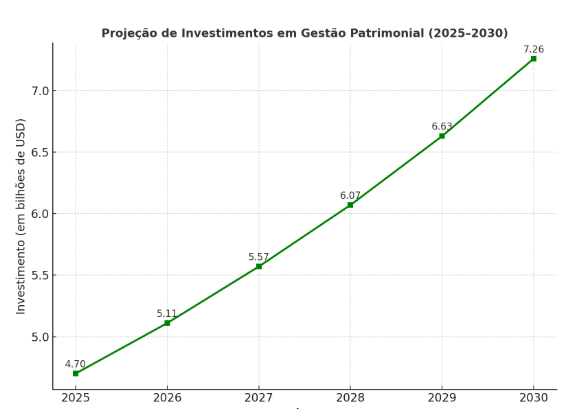
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Rank** | **Prioridade** | **User Story** | **Estimativa** | **Sprint** |
| 1 | Alta | Como administrador, quero cadastrar bens patrimoniais (nome, código, local, quantidade, status) para que a escola tenha controle atualizado. | 5 | 1 |
| 2 | Alta | Como usuário autorizado, quero acessar o sistema via login e senha para garantir segurança no acesso. | 4 | 2 |
| 3 | Alta | Como gestor, quero emitir relatórios em PDF/Excel para visualizar os bens cadastrados de forma organizada. | 8 | 3 |
| 4 | Alta | Como colaborador, quero pesquisar bens cadastrados por filtros (nome, código, local, status) para encontrar rapidamente o item desejado. | 6 | 2 |
| 5 | Alta | Como administrador, quero editar informações de patrimônios já cadastrados para corrigir erros ou atualizar dados. | 7 | 2 |
| 6 | Média | Como gestor, quero controlar diferentes níveis de acesso (administrador, colaborador, gestor) para que cada perfil tenha permissões específicas. | -- | 4 |
| 7 | Média | Como usuário mobile, quero utilizar o sistema em aplicativo com funcionamento offline para cadastrar bens mesmo sem internet. | 5 | 5 |
| 8 | Média | Como gestor, quero registrar movimentações de bens (transferência, manutenção, descarte) para rastrear o histórico de cada patrimônio. | 5 | 2 |
| 9 | Baixa | Como usuário, quero anexar imagens aos bens cadastrados para melhor identificação. | 3 | 2 |

## **2.4 Gestão patrimonial**

A gestão patrimonial eficiente é essencial para garantir a sustentabilidade dos recursos públicos e para o planejamento e controle administrativo das instituições" (CURY, 2005, p. 45).

De acordo com Sommerville (2019), o desenvolvimento de software requer um ciclo contínuo de testes e aprimoramentos, com o objetivo de alcançar um produto funcional e confiável. A automação de processos, promovida por sistemas computacionais, tem como principal vantagem a substituição de tarefas manuais, reduzindo erros, aumentando a produtividade e possibilitando a inovação institucional. Laudon e Laudon (2019) destacam que os sistemas de informação contribuem para a eficiência operacional e estratégica das organizações. Ilustrada no Gráfico 2.

Gráfico 2 - Projeção de investimentos em gestão patrimonial



Fonte: International Data Corporation, adaptado (2024)

Segundo o Portal FGV (2022), a gestão patrimonial é um processo essencial da administração, que envolve o controle e a conservação de bens, assegurando seu uso adequado e a otimização de recursos.

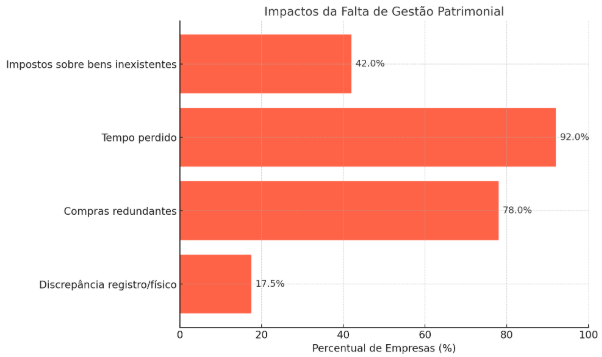
Em ambientes escolares, a gestão patrimonial envolve o controle de bens móveis e imóveis. A ausência de um sistema adequado pode acarretar falhas nos registros, perdas de informações e dificuldades administrativas. De acordo com o Portal Contábeis (2024), a falta de controle patrimonial pode expor a instituição a autuações fiscais e penalidades legais.

O *Manual de Patrimônio* (2024) enfatiza a importância da avaliação contínua dos ativos, permitindo a identificação de desgastes, depreciações e necessidades de manutenção. Um sistema de gestão eficaz garante que os usuários compreendam os bens sob sua responsabilidade e tomem decisões com base em dados confiáveis.

Além disso, conforme o SAP (2023), a gestão patrimonial pode ser aplicada em diferentes contextos, como gestão empresarial (bens fixos e intangíveis), pessoal, pública e de investimentos. O sistema aborda etapas como o cadastro, controle, avaliação e manutenção de bens, sejam eles móveis, imóveis ou direitos (como patentes e marcas).

A gestão possui diversos tipos, todos esses referenciados à patrimônios, como a gestão empresarial, que em resumo, visa a administração de ativos fixos como imóveis, equipamentos e estoques, além de bens intangíveis. Gestão patrimonial pessoal é aquela gerenciada por um indivíduo ou uma família, envolvendo bens e recursos financeiros. A gestão patrimonial de terceiros envolve pessoas (profissionais que atuam na área), onde atuam como administradores de patrimônios em troca de remuneração. Temos também a gestão patrimonial pública, geralmente gerenciada pelo governo, autarquias, instituições públicas e fundações. E por último a gestão patrimonial de investimentos, onde é envolvido a análise do mercado financeiro, envolvendo riscos e muitos benefícios caso o lucro. (SAP, 2023). Ilustrado no Gráfico 3.

Gráfico 3 – Impactos da Ausência de Controle Patrimonial Adequado



Fonte: AfixCode, adaptado (2025)

### **2.5 UML**

A UML (*Unified Modeling Language*) é uma linguagem gráfica amplamente utilizada para representar, de forma visual, os elementos que compõem um sistema de *software*. Seu objetivo principal é tornar mais claro o entendimento sobre o funcionamento interno e externo de um sistema, especialmente quando se trabalha com programação orientada a objetos. Ela surgiu da necessidade de padronizar diferentes formas de modelagem que existiam entre os desenvolvedores nas décadas de 1980 e 1990.

Segundo Bezerra (2011), a *UML* utiliza símbolos gráficos que ajudam a expressar com clareza os conceitos da orientação a objetos durante o processo de projeto e desenvolvimento de *software*. Através dela, é possível criar diagramas que representam desde a estrutura estática do sistema, como os diagramas de classe, até aspectos mais dinâmicos, como as interações entre os componentes (diagrama de sequência) ou os fluxos de atividades (diagrama de atividades).

O uso da UML também está alinhado com metodologias modernas de desenvolvimento, principalmente aquelas baseadas em entregas incrementais e melhoria contínua. Larman (2007) destaca que a *UML* funciona muito bem em conjunto com práticas de análise e design orientado a objetos, especialmente quando são aplicados casos de uso e padrões de responsabilidade, como os princípios *GRASP* (*General Responsibility Assignment Software Patterns*), que ajudam a distribuir melhor as funções entre os componentes do sistema.

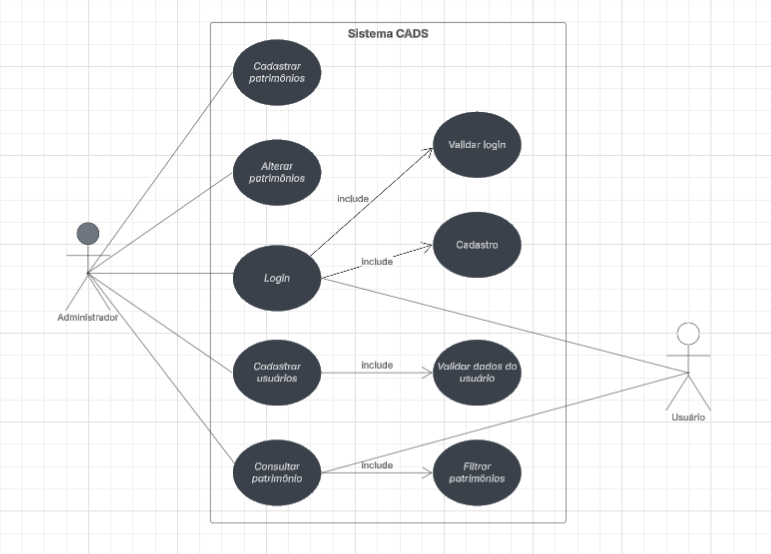
No desenvolvimento do sistema CADS, a *UML* foi essencial para organizar as ideias e representar de maneira clara como o sistema deveria funcionar. Os diagramas facilitaram a comunicação entre os integrantes da equipe, diminuindo ambiguidades e contribuindo para que as funcionalidades do projeto fossem planejadas e construídas com maior segurança e alinhamento aos objetivos da aplicação.

**2.6 Diagrama de caso de uso**

Conforme Bezerra (2011), na Linguagem de Modelagem Unificada (*UML*), o diagrama de caso de uso refere-se a uma representação gráfica e abstrata das funcionalidades centrais de um sistema sob a perspectiva do usuário. Ele retrata a interação entre elementos externos –denominados atores- e os casos de uso, que representam os serviços oferecidos pelo sistema. Sua principal finalidade é prover uma visão geral, de alto nível, destacando e distinguindo quais usuários ou sistemas externos interagem com determinadas funcionalidades. Portanto, um *DCU* pode ser entendido como uma contextualização do sistema, pois delimita suas funcionalidades e evidencia os tipos de interação com o ambiente externo.

Para criar um, use um conjunto de símbolos e conectores especializados. Um bom diagrama de caso de uso ajuda sua equipe a representar e discutir cenários onde o sistema interage com pessoas, organizações ou sistemas externos, metas que o sistema ou aplicativo ajuda essas entidades a atingir. Nesse tópico, abordaremos como a aplicação será controlada, o diagrama serve para que possamos entender melhor como o sistema será gerenciado, quem irá gerenciar, e quais são seus níveis de acesso e entre outros, de acordo com a Figura 2 que ilustra o diagrama de caso de uso referente ao sistema de gestão patrimonial.

Figura 2 – Diagrama de caso de uso do projeto CADS (Arrumar)

Fonte: Autores (2025)

# 3. DESENVOLVIMENTO

Nesse tópico será abordado sobre o desenvolvimento do projeto, tais como as telas, o banco de dados, e entre outros.

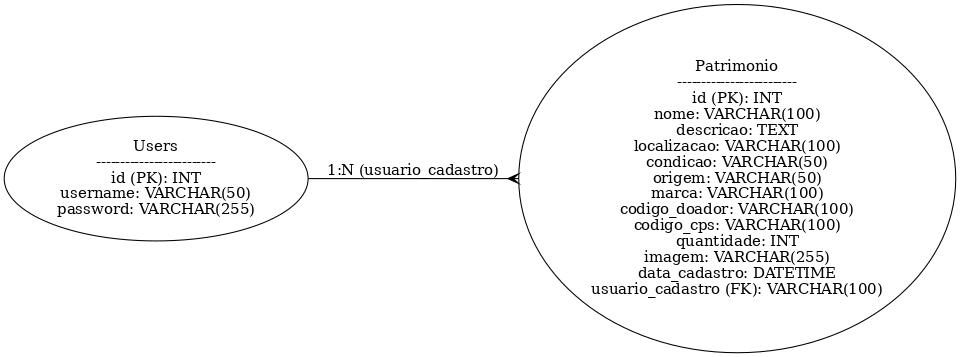
## 3.1 Modelo de dados

Nesse tópico iremos apresentar sobre o modelo de dados referente ao projeto, tais como as tabelas inseridas, suas relações e o dicionário de dados.

O modelo lógico do banco de dados do projeto CADS foi estruturado para atender às necessidades de gestão e controle patrimonial da instituição. Ele é composto por duas tabelas principais: patrimonio e *users*, que se relacionam de forma a garantir a rastreabilidade e a segurança das informações. A tabela patrimonio armazena os dados referentes aos bens da instituição, incluindo informações como nome, descrição, localização, condição, origem, marca, códigos de identificação, quantidade, imagem e dados de cadastro. O campo usuario\_cadastro estabelece uma relação lógica com a tabela *users*, indicando qual usuário foi responsável pelo registro. Já a tabela *users* concentra os dados de autenticação e controle de acesso dos usuários do sistema, com campos como *username*, senha criptografada, tentativas de acesso e status de bloqueio.

Dessa forma, o modelo lógico garante não apenas a organização dos patrimônios cadastrados, mas também a segurança e o monitoramento do acesso ao sistema, a imagem 1 a seguir ilustrará o modelo lógico do banco de dados, e logo em seguida, será explicado cada parte da estrutura desse modelo.

Imagem 4 – Modelo lógico do banco de dados

(Autores, 2025)

Logo em seguida, a quadro 5 e 6 irão ilustrar o dicionário de dados utilizados, explicando seus tipos, suas funções e seus campos.

Quadro 5 - Ilustração da tabela patrimônio.

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descrição** |
| **id** | Inteiro (PK) | Identificador único (chave primária). |
| **nome** | Texto | Nome do bem patrimonial (ex.: "Notebook Dell", "Mesa de Reunião"). |
| **descricao** | Texto | Descrição detalhada do item, incluindo características relevantes. |
| **localizacao** | Texto | Local físico onde o patrimônio está alocado (ex.: "Sala 203", "Almoxarifado"). |
| **condicao** | Texto | Estado de conservação (ex.: "Novo", "Usado - Bom Estado", "Necessita Reparo"). |
| **origem** | Texto | Forma de aquisição (ex.: "Compra", "Doação", "Licitação"). |
| **marca** | Texto | Marca ou fabricante, quando aplicável (ex.: "Dell", "Tramontina"). |
| **codigo\_doador** | Texto (Opcional) | Código identificador do doador, caso o item tenha sido recebido por doação. |
| **codigo\_cps** | Texto | Código único de controle patrimonial da instituição (etiqueta de identificação). |
| **quantidade** | Inteiro (Default = 1) | Número de itens idênticos cadastrados no mesmo registro. |
| **imagem** | Texto (Opcional) | Caminho do arquivo ou referência para uma imagem ilustrativa do bem. |
| **data\_cadastro** | Data/Hora (Default = CURRENT\_TIMESTAMP) | Registro automático da data e hora de inclusão no sistema. |
| **usuario\_cadastro** | Inteiro (FK) | Chave estrangeira referenciando o usuário que realizou o cadastro (tabela users). |

Fonte: Autores (2025)

Quadro 6 - Ilustração da tabela *users*

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descrição** |
| **id** | Inteiro (PK) | Identificador único do usuário (chave primária). |
| **username** | Texto (Único) | Nome de usuário utilizado para login no sistema. |
| **password** | Texto | Senha criptografada para autenticação. |
| **username\_acess** | Inteiro | Status de acesso: 0 (ativo) ou 1 (bloqueado). |

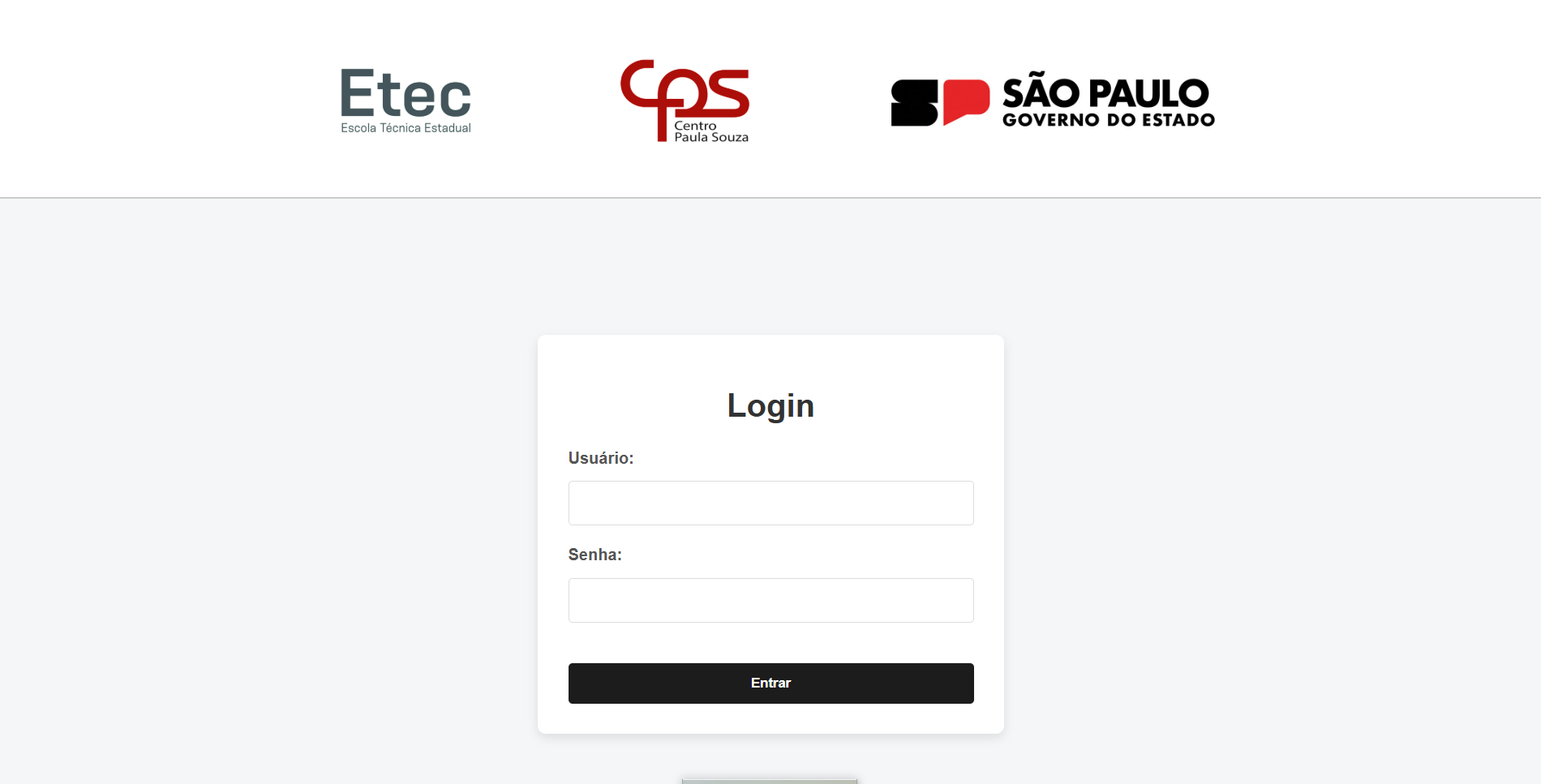
## Fonte: Autores (2025)

## 3.2 Estrutura do site

A simplicidade da interface está alinhada aos princípios de usabilidade, segundo os quais o sistema deve ser intuitivo, eficiente e reduzir erros do usuário (NIELSEN, 2010).

Inicialmente, a primeira tela exibida é a página de login, onde serão colocados o usuário e a sua senha. Ela contém logos da instituição, além de que possui um sistema de segurança onde, caso o usuário escreva a senha incorreta pela terceira vez consecutiva, a conta é bloqueada diretamente no banco de dados. A figura 3 a seguir ilustrará a tela de login.

Figura 3 - Tela de login

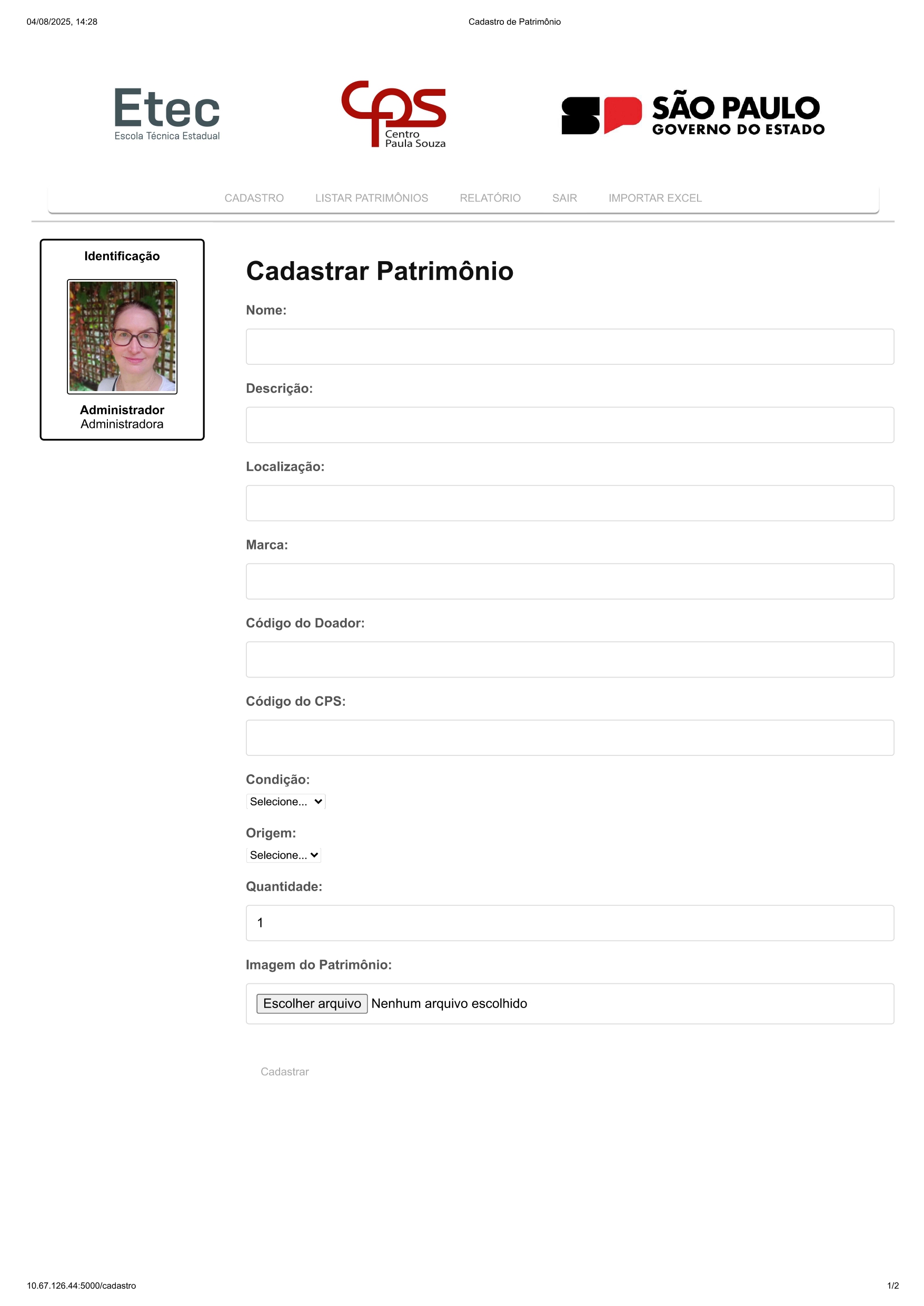


Fonte: Autores (2025)

Após acessar o sistema pela tela de login, o usuário é redirecionado para a terceira tela, que é a página principal. Responsável por realizar o cadastramento de patrimônios.

Visualmente, a tela principal apresenta uma interface simples e intuitiva. De forma geral, são exibidas a foto de perfil do usuário juntamente do seu nível de acesso, uma barra de navegação que redireciona para as diferentes funcionalidades do sistema e as logos das instituições. Em relação a operacionalidade, para efetuar o cadastro de patrimônios, é necessário que o usuário informe o nome, código (que se diferencia entre código de bens cedidos pela CPS e bens doados por outras instituições) do patrimônio, quantidade, marca, local localizado, condição, origem e, opcionalmente, uma descrição, além da possibilidade de anexar uma foto do objeto. A figura 4 ilustra a interface da tela principal.

Figura 4 – Tela principal

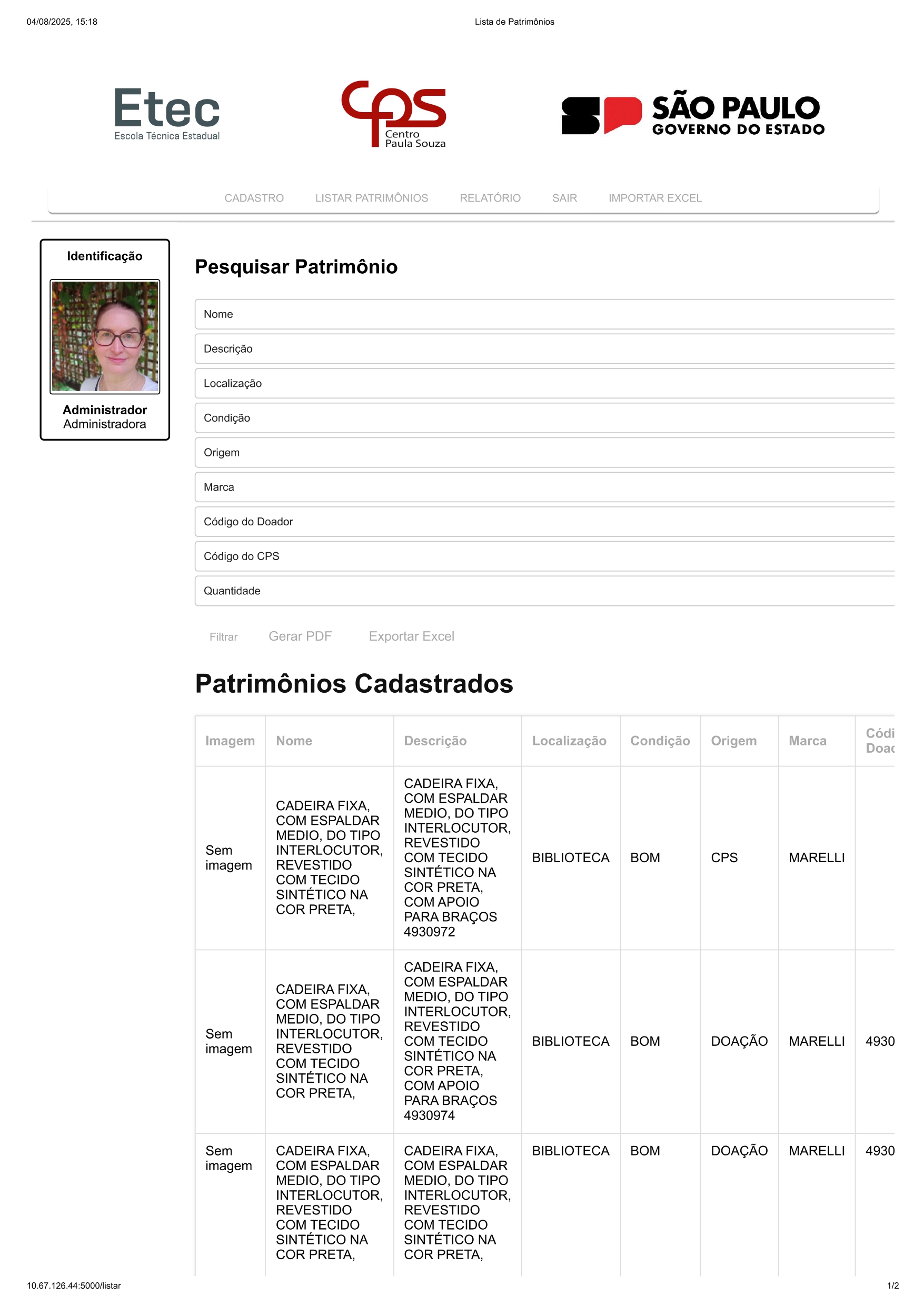


Fonte: Autores (2025)

A tela principal, é composta basicamente por todo o funcionamento do site. Além da funcionalidade de cadastro, o sistema possibilita a consulta de bens previamente registrados, dispondo também de recursos de filtragem da lista de forma eficiente e organizada.

Na página de filtragem, o usuário pode selecionar os critérios que atendem ao motivo de sua pesquisa, informando nome, descrição, localização, condição, origem, marca, código do patrimônio ou quantidade. A interface do sistema nesta página é projetada para tornar o processo de consulta simples e rápido. A figura 5 ilustra a interface da página de consulta.

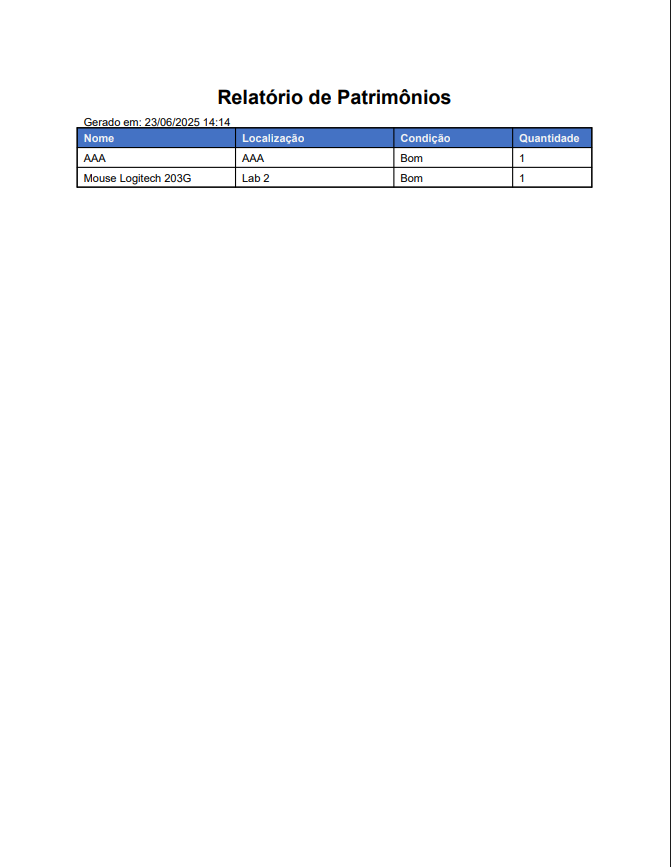
Figura 5 – Listar patrimônios



Fonte: Autores (2025)

Outras funcionalidades da página de filtragem incluem a emissão de um relatório em formato PDF (*Portable Document Format*) e a exportação dos dados para arquivos no formato Excel. Os registros previamente cadastrados são organizados e formatados para garantir acessibilidade e facilitar a documentação dos bens. A figura 7 ilustra o resultado da emissão de um relatório.

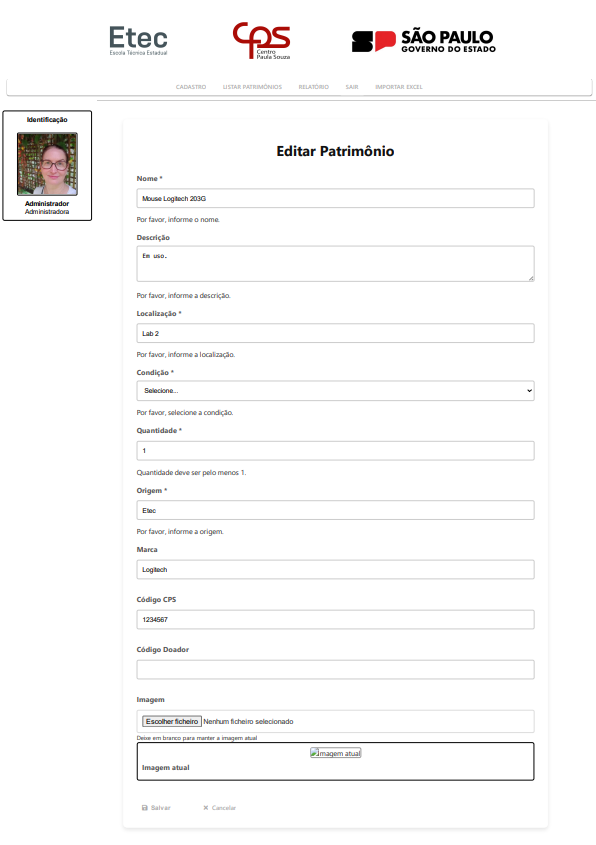
Figura 6 – Relatório dos patrimônios (PDF)



Fonte: Autores (2025)

Nessa tela possuímos o relatório referente aos patrimônios cadastrados, onde são exibidos nome, localização, condição e quantidade. Entretanto, em caso de erro ou necessidade de atualizações das condições dos bens, o sistema oferece a funcionalidade de alteração, permitindo que o usuário modifique as informações ou corrija possíveis equívocos. A figura a seguir ilustrará a tela de alteração.

Figura 7- Alteração de patrimônio



Nessa tela é feito a alteração de um patrimônio, em caso de erro de código ou quaisquer outros erros gramaticais, essa tela é responsável por fazer alteração de patrimônios já existentes, para corrigir possíveis erros.

### 3.2.1 Arquitetura do sistema

O sistema CADS foi estruturado sob uma arquitetura cliente-servidor, composta por uma aplicação web responsiva desenvolvida em *Python* com o microframework *Flask*, banco de dados *MySQL*, e camadas de interface desenvolvidas em HTML, CSS e *JavaScript*. O controle de versão foi realizado no *GitHub*, e a organização do desenvolvimento foi apoiada pela metodologia Scrum, com auxílio da ferramenta Trello.

A Figura X apresenta o diagrama de arquitetura geral do sistema: (Inserir)

* *Frontend*: construído em HTML, CSS e *JavaScript*, oferecendo uma interface responsiva e acessível, alinhada às boas práticas de desenvolvimento web (CASTRO; HYSLOP, 2013).
* *Backend*: desenvolvido em *Python* (*Flask*), responsável pelo processamento das requisições, aplicação das regras de negócio e comunicação com o banco de dados (GRINBERG, 2018).
* Banco de dados: gerenciado pelo *MySQL*, um *SGBD* relacional robusto, amplamente utilizado em sistemas de médio porte devido à confiabilidade e escalabilidade (ELMASRI; NAVATHE, 2011).
* Controle de versão: realizado por meio do *GitHub*, garantindo rastreabilidade do código, colaboração entre a equipe e integração contínua (CHACÓN; STRAUB, 2014).
* Padrão de projeto: adota princípios do MVC (*Model-View-Controller*), separando as camadas de dados, interface e lógica de negócio, facilitando a manutenção e evolução do sistema (SOMMERVILLE, 2019).

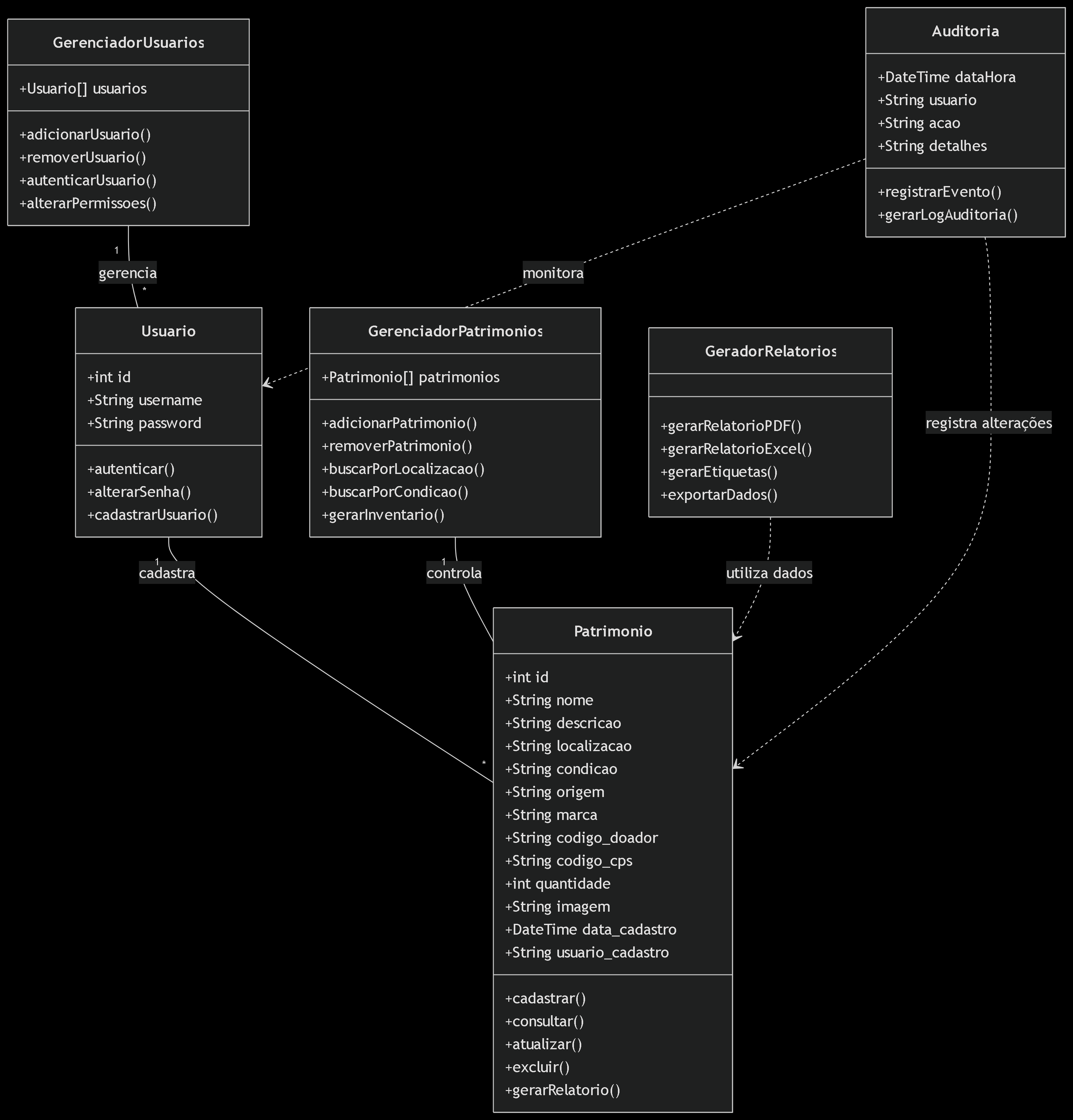
## 3.3 Módulo 1

O primeiro módulo contemplou as funções básicas do sistema: cadastro, consulta e edição de patrimônios, com autenticação de usuários. A modularização reduz a complexidade e facilita expansões futuras (SOMMERVILLE, 2019).

O primeiro módulo registrado no sistema CADS foi voltado ao cadastro e gerenciamento de patrimônios. Ele contemplou as funcionalidades básicas de login, autenticação de usuários e cadastro de bens, estabelecendo a base do CRUD (Create, Read, Update, Delete). Essa etapa foi essencial para garantir a segurança no acesso, permitindo diferentes níveis de permissão, como administrador e colaborador.  
 Segundo Sommerville (2019), a modularização é fundamental para reduzir a complexidade do sistema e facilitar sua evolução. Nesse sentido, o módulo 1 foi estruturado de forma a assegurar escalabilidade, permitindo que novos módulos sejam adicionados futuramente sem comprometer a arquitetura existente. O módulo incorpora um sistema hierárquico de usuários com diferentes níveis de acesso, incluindo administradores com permissões totais e colaboradores com acesso restrito a determinadas funcionalidades. Esta abordagem segue o princípio de menor privilégio, fundamental para a segurança de sistemas de informação, pois "o controle de acesso deve garantir que cada usuário tenha acesso apenas aos recursos estritamente necessários para desempenhar suas funções" (PFLEEGER & PFLEEGER, 2015).

A implementação da autenticação utilizou técnicas modernas de hash para armazenamento seguro de senhas, incorporando a recomendação de que "credenciais de usuários devem ser protegidas usando funções de hash criptográficas apropriadas, preferencialmente com salt único para cada credencial" (OWASP, 2021). Esta camada de segurança inicial é crucial para proteger os dados patrimoniais contra acessos não autorizados. A figura a seguir ilustrará o diagrama de classes referente ao projeto CADS.

Figura 8 – Diagrama de classe do projeto CADS



Fonte: Autores (2025)

# 3.4 Deploy

O processo de deploy consistiu na disponibilização do sistema em um ambiente de produção, tornando-o acessível aos usuários finais. O sistema foi hospedado em servidor local, utilizando o *Flask* para execução e o *MySQL* como banco de dados. O *GitHub* foi empregado como repositório para versionamento e colaboração da equipe.  
 Conforme Chacón e Straub (2014), o uso do *GitHub* em projetos colaborativos garante rastreabilidade e integração contínua, reduzindo falhas no versionamento. O processo de deploy incluiu as seguintes etapas: configuração do ambiente virtual Python, importação do banco de dados, upload dos arquivos para o servidor e testes de funcionalidade.  
 Embora o sistema esteja em ambiente local, sua estrutura permite futura implantação em nuvem, como no *Heroku* ou *Render*, garantindo maior escalabilidade e acessibilidade.

O deploy consiste na etapa final do desenvolvimento, onde o sistema é disponibilizado em um ambiente de produção, permitindo seu uso efetivo pelos gestores patrimoniais. Abaixo são descritos os principais aspectos dessa fase:

Hospedagem: O sistema CADS está hospedado em dois locais, o *GitHub* e na máquina da área administrativa da instituição, as ferramentas utilizadas para o *deploy* foi o *XAMPP* e o ambiente virtual do *Python*.

Para o deploy, será listado as seguintes máquinas que foram utilizadas para a codificação do site, realização de testes e funcionalidades:

* Notebook Enzo: Intel i5, 8GB RAM, SSD 240GB;
* Notebook Victor Bento: Intel i7, 16GB RAM, SSD 512GB;
* Notebook Victor Gabriel: Ryzen 5, 8GB RAM, HD 1TB;
* Máquina da escola: Intel i3, 4GB RAM, HD 500GB.

# 4. RESULTADOS E DISCUSSÃO

Nesse tópico será apresentado os resultados obtidos através do projeto e discussões acerca do mesmo.

## 4.1 Resultados obtidos

A implementação do CADS possibilitou:

* Cadastro seguro de bens patrimoniais.
* Sistema de login com bloqueio após tentativas inválidas.
* Relatórios exportáveis em PDF e Excel.
* Consulta de patrimônios com filtros eficientes.
* Banco de dados relacional integrado ao sistema.

Esses resultados demonstram que as hipóteses levantadas foram confirmadas: a digitalização reduziu inconsistências, agilizou consultas e aumentou a segurança dos dados, em consonância com estudos que apontam a relevância da transformação digital para eficiência administrativa (Portal FGV, 2022; Contábeis, 2024).

## 4.2 Comparativo com soluções existentes

O sistema CADS foi comparado com *GLPI*, *CMDBuild* e *Asset Panda*. Diferente das soluções de mercado, o CADS é gratuito, leve e adaptado especificamente para a realidade da escola pública. Sua principal vantagem é a simplicidade e acessibilidade, fatores que favorecem sua adoção imediata.  
 No entanto, apresenta limitações, como a ausência de recursos avançados de auditoria e integração com sistemas de georreferenciamento, presentes no *CMDBuild* (2025), e de mobilidade avançada, como no Asset Panda (2025). Assim, o CADS mostra-se uma solução viável para instituições de médio porte, embora ainda precise de aprimoramentos. A tabela a seguir demonstrará o comparativo com as soluções existentes.

Quadro 7 – Comparativo CADS e demais soluções existentes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Critério** | **CADS** | **GLPI (Open Source)** | **CMDBuild (Open Source)** | **Asset Panda (Pago)** |
| **Custo** | Gratuito (desenvolvido sob demanda) | Gratuito (GPL v3) | Gratuito (AGPL) | Pago (licença por usuário) |
| **Público-alvo** | Instituições da CPS | Empresas e instituições | Empresas médias e grandes | Empresas privadas |
| **Funcionalidades** | Cadastro, consulta, relatórios básicos | Inventário, help desk, manutenção | Facilities, fluxos de trabalho, GIS | Auditoria, mobilidade, QR Code, RFID |
| **Plataforma** | Web local (Flask + MySQL) | Web (PHP) | Web (Java) | Web + Mobile (Cloud) |
| **Escalabilidade** | Médio porte escolar | Pequeno a médio porte | Médio a grande porte | Médio porte |
| **Facilidade de uso** | Simples e intuitivo | Média (requer conhecimento técnico) | Complexo, curva de aprendizado alta | Alta, interface moderna e intuitiva |
| **Diferenciais** | Adaptado à realidade da ETEC | Comunidade ativa, plugins | Alta customização e workflows | Mobilidade e suporte em nuvem |

## 4.3 Avaliação do cliente

A colaboradora Renata, da equipe administrativa da ETEC, avaliou positivamente o sistema, destacando a facilidade de uso e a agilidade na emissão de relatórios. Ela apontou que o processo de cadastro, antes feito manualmente, tornou-se mais confiável e menos suscetível a erros humanos. Entretanto, sugeriu melhorias futuras, como a inclusão de inventário por QR Code e o desenvolvimento da versão mobile.  
 Esse feedback confirma a necessidade de evolução contínua, característica de projetos ágeis (Sutherland, 2020).

(Inserir avaliação da Renata, seja foto ou mensagem.)

## 4.4 Análise dos resultados

Os resultados comprovam que o sistema atendeu aos objetivos gerais e específicos. Houve redução significativa de falhas nos registros, maior controle dos bens e transparência na gestão.  
 De acordo com Statista Research Department (2024), o aumento nos investimentos em digitalização é reflexo da busca por eficiência organizacional, tendência que também se aplica às instituições educacionais. Além disso, a SAP (2023) reforça que a gestão patrimonial, quando apoiada por tecnologia, garante maior confiabilidade dos ativos e eficiência nos processos internos.

# 5. TRABALHOS FUTUROS

O desenvolvimento do projeto progrediu evidentemente bem a cada etapa, entretanto, ainda que atenda as necessidades do cliente, funcionalidades adicionais que auxiliaram o usuário em questão de praticidade, foram tratadas com menor prioridade devido a limitações de tempo para a entrega do sistema. A partir deste ponto, alguns dos serviços que a equipe gostaria de ter aplicado incluem:

* **Desenvolvimento de um Assistente IA**
* Suporte do sistema.
* Contribui para necessidades do usuário em relação a busca por ativos.
* **aplicação mobile:**
* Agiliza processo de consulta de patrimônio.
* Facilita o cadastro de patrimônios diretamente de um dispositivo móvel.

# REFERÊNCIAS

ATLASSIAN. Trello. Disponível em: https://trello.com. Acesso em: 20 set. 2025.

BERNARDES, Roberto. Gestão Pública e Modernização Administrativa. São Paulo: Atlas, 2008.

BEZERRA, Eduardo. Princípios de Análise e Projeto de Sistemas com UML. Rio de Janeiro: Elsevier, 2011.

CASTRO, Elizabeth; HYSLOP, Bruce. HTML5 e CSS3: guia visual. 8. ed. Rio de Janeiro: Alta Books, 2013.

CHACÓN, Scott; STRAUB, Ben. Pro Git. Berkeley: Apress, 2014.

CONTÁBEIS. Pesquisa Sebrae: uso de sistemas de gestão em pequenas empresas. São Paulo, 2024. Disponível em: https://www.contabeis.com.br. Acesso em: 20 set. 2025.

CURY, Antonio Carlos. Gestão Patrimonial no Setor Público. São Paulo: Atlas, 2005.

DANTAS, José. A necessidade de digitalização nas empresas brasileiras. São Paulo: FGV, 2020.

DEITEL, Paul; DEITEL, Harvey. Python para Programação. Porto Alegre: Bookman, 2020.

DUFOUR, Benoit. Banco de Dados: fundamentos e aplicações. São Paulo: Pearson, 2018.

ELMASRI, Ramez; NAVATHE, Shamkant. Sistemas de Banco de Dados. 6. ed. São Paulo: Pearson Addison Wesley, 2011.

GIL, Antonio Carlos. Métodos e Técnicas de Pesquisa Social. 7. ed. São Paulo: Atlas, 2019.

GRINBERG, Miguel. Flask Web Development. 2. ed. Sebastopol: O’Reilly Media, 2018.

GUNTER, Carl. Semantics of Programming Languages. Cambridge: MIT Press, 1992.

IDC. Worldwide Digital Transformation Spending Guide. Framingham: IDC, 2024.

LAUDON, Kenneth C.; LAUDON, Jane P. Sistemas de Informação Gerenciais. 15. ed. São Paulo: Pearson, 2019.

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

MAXIMIANO, Antonio César Amaru. Teoria Geral da Administração. 2. ed. São Paulo: Atlas, 2006.

McKINSEY & Company. Digital adoption and business transformation report. New York, 2023. Disponível em: https://www.mckinsey.com. Acesso em: 20 set. 2025.

MICROSOFT. Visual Studio Code. Disponível em: https://code.visualstudio.com. Acesso em: 20 set. 2025.

PRESSMAN, Roger S. Engenharia de Software: uma abordagem profissional. 8. ed. Porto Alegre: AMGH, 2016.

SADALAGE, Pramod J.; FOWLER, Martin. NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence. Boston: Addison-Wesley, 2013.

SANTOS, Edmilson. Tecnologias da Informação na Gestão Pública. Brasília: Ipea, 2016.

SEBRAE. Transformação digital em micro e pequenas empresas. Brasília: Sebrae, 2024. Disponível em: https://www.sebrae.com.br. Acesso em: 20 set. 2025.

STATISTA RESEARCH DEPARTMENT. Global digital transformation investments. Hamburg: Statista, 2024. Disponível em: https://www.statista.com. Acesso em: 20 set. 2025.

GLPI Project. GLPI – IT Asset Management software. Disponível em: https://glpi-project.org. Acesso em: 20 set. 2025.

CMDBUILD. Open-source web environment for the configuration of a CMDB. Disponível em: https://www.cmdbuild.org. Acesso em: 20 set. 2025.

ASSET PANDA. Asset Panda Official Website. Disponível em: https://www.assetpanda.com. Acesso em: 20 set. 2025.

PCMAG. Asset Panda review. Nova Iorque, 2025. Disponível em: https://www.pcmag.com. Acesso em: 20 set. 2025.

PORTAL FGV. Transformação digital na gestão pública. Fundação Getúlio Vargas, 2022. Disponível em: [https://portal.fgv.br](https://portal.fgv.br/?utm_source=chatgpt.com). Acesso em: 08 jun. 2025.

SAP. Gestão patrimonial: conceitos e aplicações. São Paulo: SAP Brasil, 2023. Disponível em: [https://www.sap.com/brazil](https://www.sap.com/brazil?utm_source=chatgpt.com). Acesso em: 08 jun. 2025.

SOMMERVILLE, Ian. Engenharia de software. 10. ed. São Paulo: Pearson, 2019.

TIOBE SOFTWARE. TIOBE Index for October 2024. Disponível em: <https://www.tiobe.com/tiobe-index/>. Acesso em: 08 jun. 2025.

VIANA, João José. Administração de materiais: um enfoque prático. São Paulo: Atlas, 2006.

# APÊNDICE A (Se tiver)

# ANEXOS