

analise e desenvolvimento de sistemas

GUSTAVO ALMEIDA VASCONCELOS

PROJETO INTEGRADO INOVAÇÃO - ADS:

2º Semestre 2024.2

Sobral

2024

gustavo almeida vasconcelos

PROJETO INTEGRADO INOVAÇÃO - ADS:

2º Semestre 2024.2

Projeto Integrado Síntese Inovação apresentado como requisito parcial para a obtenção de média semestral.

Sobral

2024

SUMÁRIO

[1 INTRODUÇÃO 3](#_Toc181823691)

[2 DESENVOLVIMENTO 4](#_Toc181823692)

[**2.1** **PASSO 1** 4](#_Toc181823693)

[**2.1.1** **RELATÓRIO SPRINT 1** 4](#_Toc181823694)

[**2.1.2** **RELATÓRIO SPRINT 2** 7](#_Toc181823695)

[**2.1.3** **RELATÓRIO SPRINT 3** 9](#_Toc181823696)

[2.2 PASSO 2: TABELA VERDADE 10](#_Toc181823697)

[2.3 PASSO 3: CÓDIGO 11](#_Toc181823698)

[2.3.1 DEFINIÇÃO DE ESTRUTURAS DE DADOS 12](#_Toc181823699)

[2.3.1.1 PRODUTO 13](#_Toc181823700)

[2.3.1.2 MOVIMENTAÇÕES 13](#_Toc181823701)

[2.3.2 ALGORITMOS DE CADASTRO E CONSULTA 14](#_Toc181823702)

[2.3.3 ALGORITMOS DE MOVIMENTAÇÃO 16](#_Toc181823703)

[2.3.4 RELATÓRIOS E CONSULTAS 17](#_Toc181823704)

[2.4 PASSO 4: DIAGRAMA DE CASOS DE USO 18](#_Toc181823705)

[3 CONCLUSÃO 19](#_Toc181823706)

[REFERÊNCIAS 20](#_Toc181823707)

# INTRODUÇÃO

Este relatório apresenta o desenvolvimento de um sistema de gerenciamento de estoque para uma empresa de comércio eletrônico em fase de expansão, a mesma tem enfrentado alguns passos em gerenciar seus estoques de maneira eficiente, falta e excesso de produtos, dificuldades em localizar itens nos depósitos, etc. O objetivo do projeto é desenvolver uma solução que automatize e otimize esses processos, melhorando a precisão e a eficiência na gestão de estoque, pondo a teste o conhecimento dos estudantes, além de sua criatividade, conhecimento da área e aumentando sua experiência em atividades relacionadas.

Para atender às necessidades propostas, o sistema deverá incluir funcionalidades essenciais como:

* cadastro de produtos;
* atualização de estoque;
* rastreamento da localização dos itens;
* geração de relatórios.

Além disso, será seguido um processo de desenvolvimento ágil utilizando a metodologia **Scrum**, permitindo uma entrega iterativa e incremental das funcionalidades do sistema. Esse processo será gerido por meio de um quadro Scrum no Trello, onde as tarefas serão organizadas, priorizadas e monitoradas ao longo das sprints.

As atividades do projeto geral estão divididas em 4 partes, de acordo com o conteúdo estudado ao longo do semestre 2024.2 pela turma de Análise e Desenvolvimento de Sistemas, sendo eles a elaboração de um quadro Scrum para planejamento do projeto, criação de uma tabela verdade (Lógica e Matemática Computacional), desenvolvimento do código referente ao desenvolvimento do software (Linguagem de Programação) e por fim a confecção do diagrama de casos de uso do sistema (Análise e Modelagem de Sistemas).

# DESENVOLVIMENTO

## **PASSO 1**

A atividade se baseia em torno do problema central de uma **gestão ineficiente** de estoque enfrentada pela empresa de comércio eletrônico, resultando em falta e excesso de produtos e dificuldades para localizar itens nos depósitos. Para solucionar esses passos, a empresa necessita de um **sistema de gerenciamento de estoque** que otimize suas operações, aprimorando a precisão e a eficiência na gestão dos produtos.

O primeiro passo proposto a ser concluído pela atividade é o do **PLANEJAMENTO** por meio de um quadro ***SCRUM*** utilizando a ferramenta online **Trello**. Assim, organizando o quadro, obtém-se o modelo padrão de quadros *SCRUM* prontos para serem definidos as durações das *Sprints*.

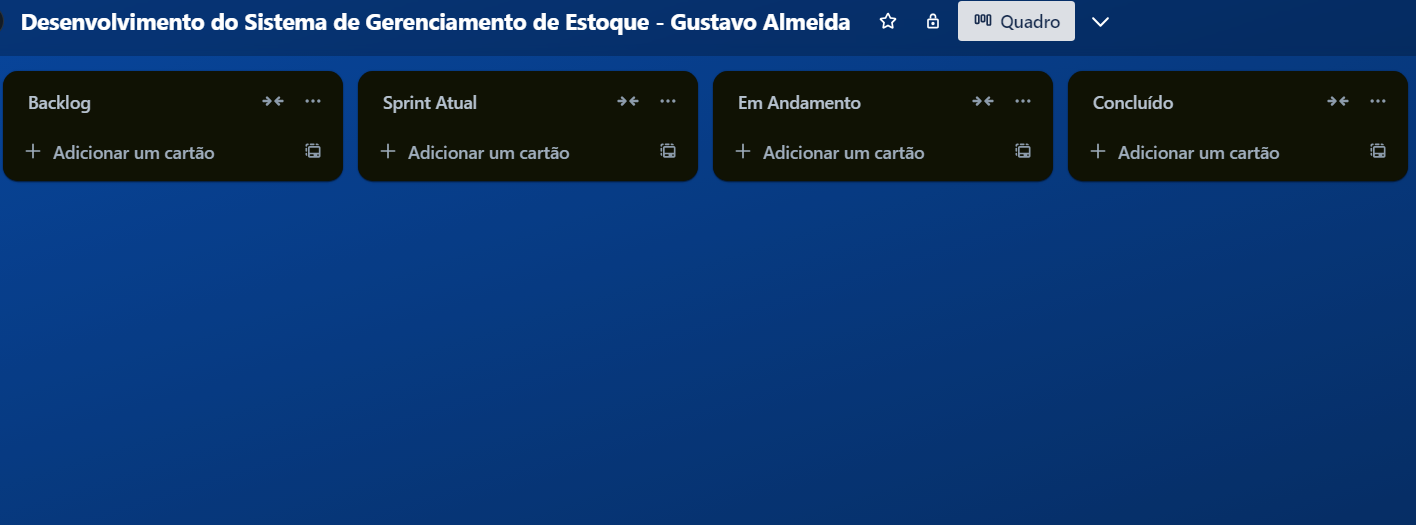


Figura 1 - Quadro Scrum padrão

Sendo decidido a duração das *Sprints* para 15 dias, foi dado início ao planejamento do projeto, o quadro pode ser encontrado no link: https://trello.com/b/jCQSklwP/desenvolvimento-do-sistema-de-gerenciamento-de-estoque-gustavo-almeida.

### **RELATÓRIO SPRINT 1**

As atividades iniciais decididas para o primeiro Sprint foram:

* Criação da Interface inicial;
* Levantamento de todo estoque atual;

 Implementar interface para cadastro de produtos.

* Desenvolver a lógica para salvar produtos no banco de dados.
* Planejamento do banco de dados (colunas);
* Mapeamento do estoque.

A primeira *Sprint* ocorreu sem nenhum problema não previsto, o planejamento do banco de dados foi o primeiro a ser completo, assim, possibilitando a conclusão rápida da interface inicial logo após.

Então, os passos de implementar interface para cadastro de produtos, desenvolver a lógica para salvar produtos no banco de dados e a página inicial foram concluídos em conjunto.

O banco de dados decidido para o sistema de armazenar dados dos produtos foi o **mySQL**, e todas funções posteriormente citadas serão voltadas para sua linguagem de programação.

Foi decidido, para o banco de dados, armazenar as informações de:

* Nome do produto; (Informação principal para identificação do produto)
* Código do produto; (Normalmente duplicado do código de barras, para diferenciar produtos com nomes semelhantes)
* Preço; (Valor de venda do produto)
* Estoque atual; (Quantidade em estoque no momento atual)
* Categoria; (Em que classificação o item se encontra, ex’s: periféricos, computadores, variedades...)
* Subcategoria; (Ramificação de categoria, busca exemplos mais específicos para categorizar os itens)
* Marca;
* Localização no estoque;
* Estoque mínimo e estoque máximo. (valor máximo e mínimo aceito no estoque)

A última informação não é muito utilizada em sistemas de estoque, porém, foi vista como necessária para solucionar os problemas de **falta de produtos** e **excesso de estoque**, de forma que quando próximas de serem alcançadas, seja notificado do sistema para o usuário esta possibilidade.

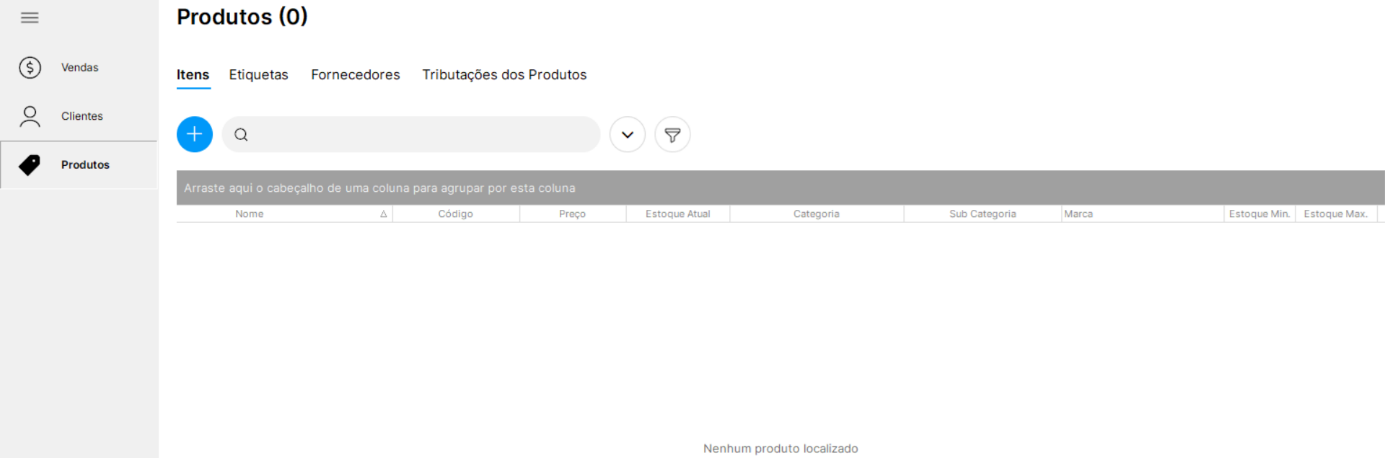


Figura 2 - Design ideal para o sistema de produtos

Sobre o mapeamento do estoque, esta foi uma tarefa significante pois será necessária no sistema para informar visualmente a localização do produto, a informando numa imagem de como está organizado o estoque e marcando sua posição exata.

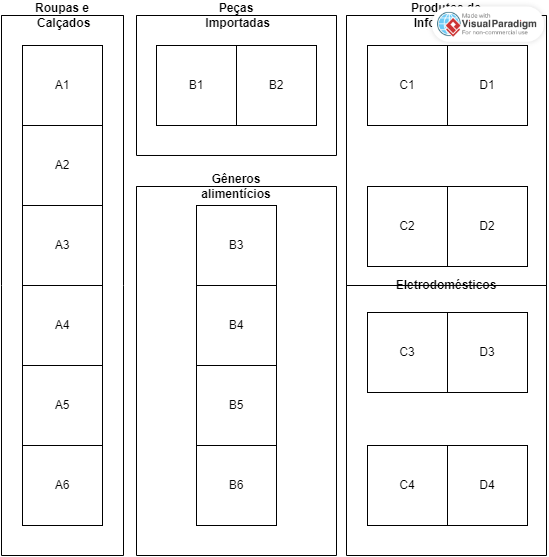


Figura 3 - Estoque da loja com exemplo de indicação da localização do produto.

Quanto ao levantamento de produtos, sua conclusão não foi possível durante apenas uma *string*, foi levantado apenas 45% de todo o estoque pois a tarefa foi dificultada com a atividade sendo feita ainda com o estoque e loja em funcionamento.



Figura 4 - Sprint 1 em seu último dia de prazo.

### **RELATÓRIO SPRINT 2**

As atividades iniciais decididas para o segundo Sprint foram:

* Inserção de todo o estoque para o sistema
* Lógica da Interface de login
* Design da interface de login
* Reunião de Entregáveis (Andamento do projeto com o solicitante.)

Foi notado que para a primeira atividade seria necessário concluir o levantamento de produtos, assim, foi atribuído a atividade à mais técnicos visando sua conclusão com máxima urgência.

Enquanto o sistema já estava recebendo os itens, as duas tarefas progrediam simultaneamente e foram concluídas sem problemas.

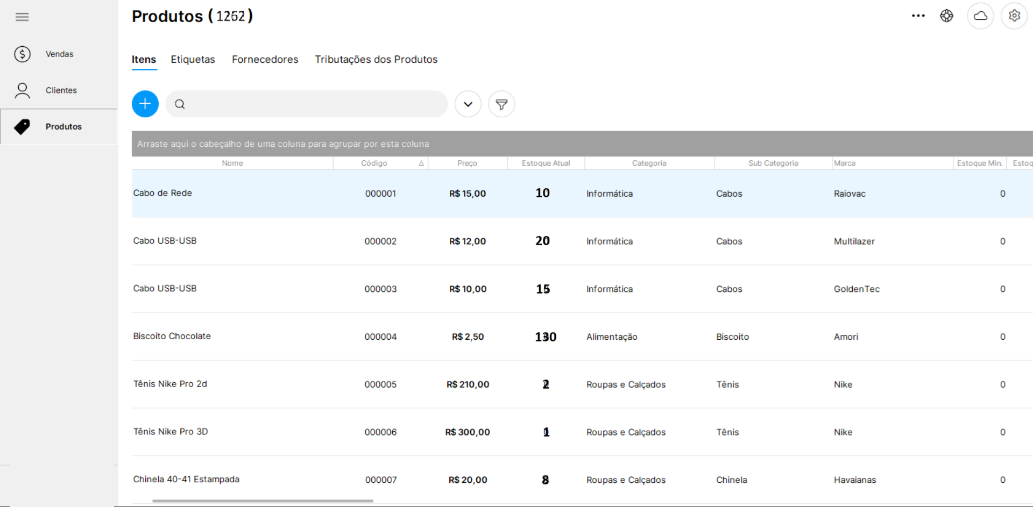


Figura 5 - Evidência de Produtos Cadastrados

A interface de login também não teve problemas durante sua criação, tanto a lógica quanto o design foram concluídos.

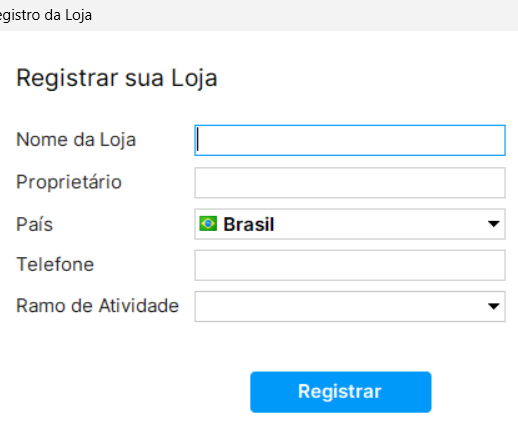


Figura 6 - Evidência da interface de login

A reunião de entregáveis foi remarcada para a próxima segunda (28/10/24) pois empregador não pode comparecer.



Figura 7 - Sprint 2 em seu último dia de prazo

### **RELATÓRIO SPRINT 3**

A Sprint 3 foi dedicada à correção de bugs e à implementação de novas funcionalidades solicitadas para aprimorar o sistema de gerenciamento de estoque da empresa de comércio eletrônico.

As atividades iniciais decididas para o terceiro Sprint foram:

* Reunião com o empregador (remarcada para o dia 28/10/24);
* Funcionalidade de edição e exclusão dos produtos;
* Lógica de verificação para estoque mín. e máx. e notificação no sistema;
* Lógica do relatório de movimentação de produtos;
* Criar função para adicionar e subtrair itens na venda dos produtos ou compra dos mesmos.

A reunião foi realizada com o empregador para alinhar as expectativas e discutir o andamento das novas funcionalidades e correções de bugs, permitiu validação dos requisitos e feedback sobre as melhorias.

Foram adicionadas funcionalidades que permitem a edição e exclusão de produtos no sistema. Agora, o usuário pode atualizar informações dos produtos cadastrados, como quantidade, preço e localização, bem como excluir registros obsoletos.

Também, foi implementada uma lógica que verifica automaticamente o estoque de cada produto. Notificações são geradas quando a quantidade em estoque atinge níveis mínimos ou excede o máximo permitido.

Continuamente concluiu-se a lógica que permite a geração de relatórios detalhados sobre a movimentação de produtos, incluindo entradas e saídas. O relatório mostra transações de vendas, reposições e ajustes manuais no estoque.

Por último foi criada uma função que permite o ajuste do estoque conforme a venda ou compra de produtos. Ao realizar uma venda, a quantidade em estoque é automaticamente subtraída; no caso de compras, é adicionada.

“  
BEGIN TRANSACTION;

UPDATE produtos

SET quantidade\_estoque = quantidade\_estoque – quantidade\_vendida;

WHERE id\_produto = produto AND quantidade\_estoque >= quantidade\_vendida;

COMMIT;”

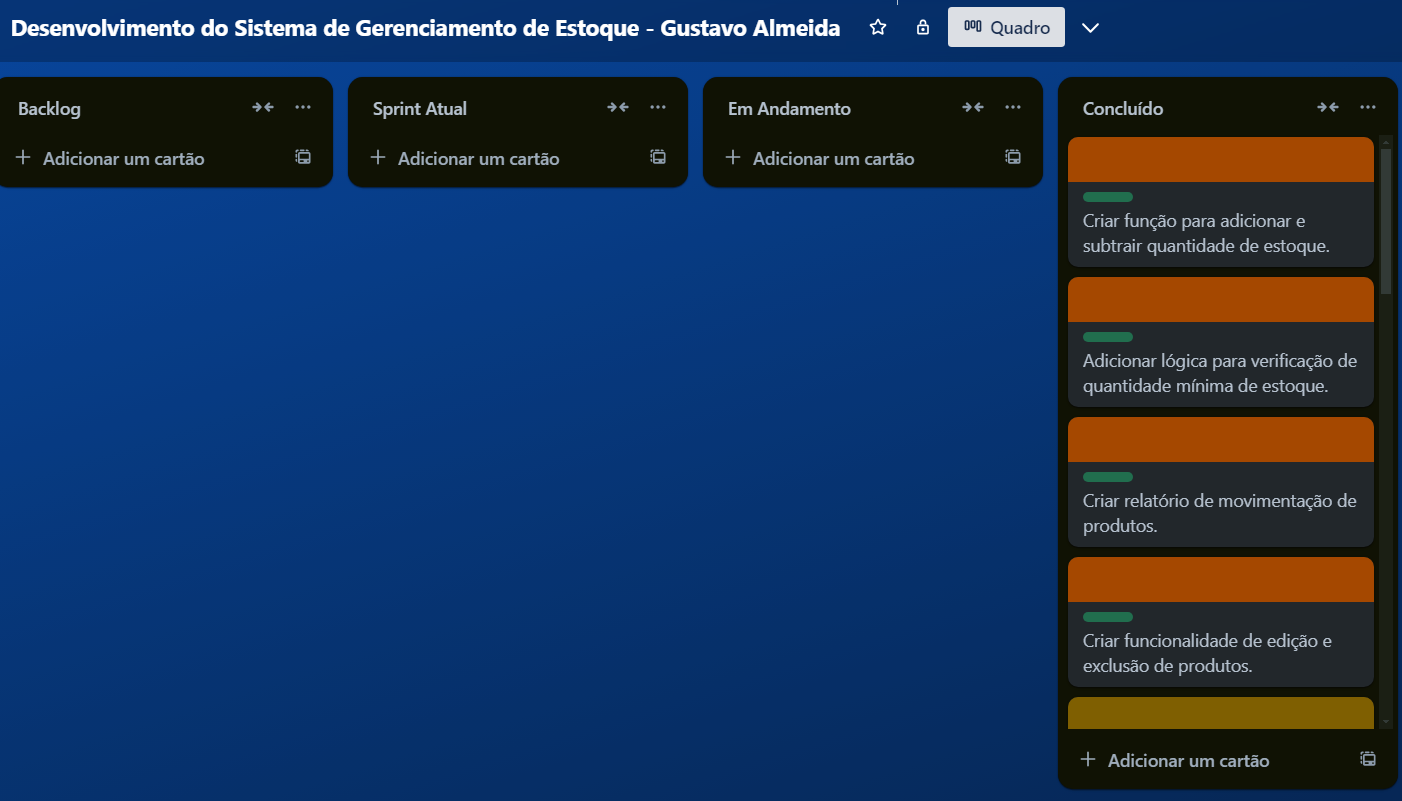


Figura 8 - Sprint 3 no seu último dia de prazo.

## PASSO 2: TABELA VERDADE

O software utilizado para a criação da tabela verdade foi o Microsoft Excel. Foram estabelecidas cinco colunas, sendo uma para cada variável booleana e uma adicional para o resultado. Em relação às linhas, foram alocadas 16 linhas para os dados, considerando que temos quatro variáveis, o que resulta em 24=162^4 = 1624=16 combinações possíveis.

Após preenchimento dos valores utilizando o método de preenchimento da tabela verdade, onde as colunas (ou variáveis) alternam de a cada 2^(*n-m)* repetições, onde “*n*” é o total de variáveis e “*m*” o número da coluna, até a última coluna, que varia a cada linha (Esse padrão garante que todas as combinações possíveis das variáveis booleanas sejam incluídas na tabela verdade), foi preciso definir o método de cálculo de da solução completa.

Transcrevendo a expressão lógica “P /\ E /\ L /\ R” para Excel obtem-se algo parecido com “=E(P;E;L;R)” onde “=E()” é a função de verificação lógica AND e “P;E;L;R” são os argumentos lógicos.

Tabela 1 - Tabela Verdade P;E;L;R

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **P (Produtos)** | **E (Estoque)** | **L (Localização)** | **R (Relatórios)** | **T (Solução Completa)** |
| 0 | 0 | 0 | 0 | FALSO |
| 0 | 0 | 0 | 1 | FALSO |
| 0 | 0 | 1 | 0 | FALSO |
| 0 | 0 | 1 | 1 | FALSO |
| 0 | 1 | 0 | 0 | FALSO |
| 0 | 1 | 0 | 1 | FALSO |
| 0 | 1 | 1 | 0 | FALSO |
| 0 | 1 | 1 | 1 | FALSO |
| 1 | 0 | 0 | 0 | FALSO |
| 1 | 0 | 0 | 1 | FALSO |
| 1 | 0 | 1 | 0 | FALSO |
| 1 | 0 | 1 | 1 | FALSO |
| 1 | 1 | 0 | 0 | FALSO |
| 1 | 1 | 0 | 1 | FALSO |
| 1 | 1 | 1 | 0 | FALSO |
| 1 | 1 | 1 | 1 | VERDADEIRO |

Assim, trocando 1 por True e 0 por False, verifica-se que a única "solução completa" que resulta em verdadeiro ocorre quando todos os argumentos são verdadeiros. Essa condição é atendida pelo sistema planejado nas sprints do primeiro passo, o qual cumpre todos os requisitos estabelecidos.

## PASSO 3: CÓDIGO

No terceiro passo é solicitado consolidar algumas definições de alta importância para a estrutura de um código. Antes de apresentar como foram desenvolvidos os requerimentos da atividade, é importante informar que foi trabalhado com o visual gráfico das ações, para isso foi utilizada a biblioteca integrada de python tkinter.

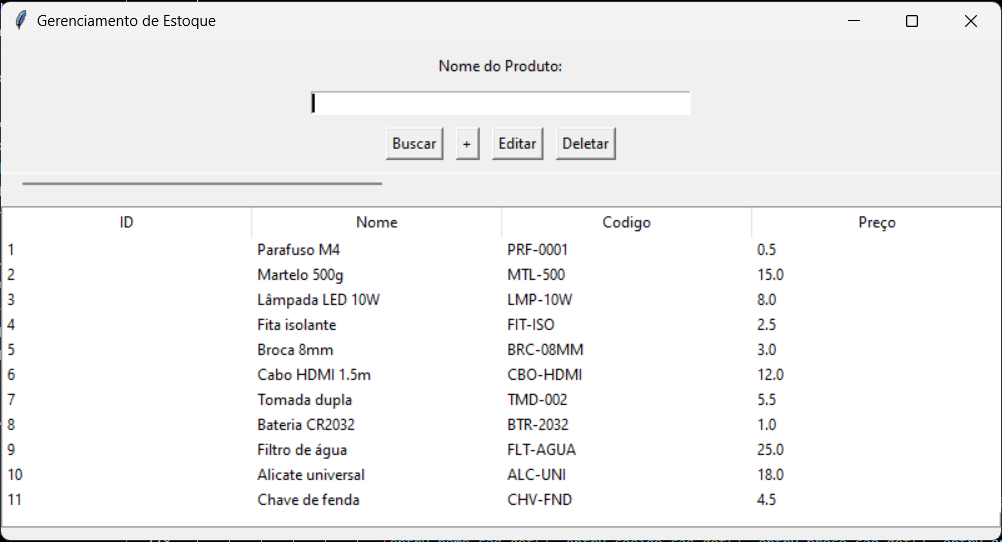
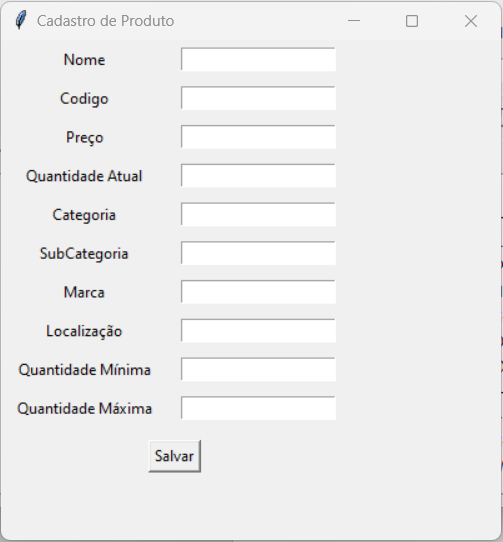
 

Figura 9 e 10 – Tela inicial do software / Tela de cadastro de produtos.

### DEFINIÇÃO DE ESTRUTURAS DE DADOS

*“Estruturas bem definidas para produtos, categorias e movimentações.”*

Antes de definir as estruturas, foi analisado todo o objetivo do projeto e decidido como essas estruturas ficariam organizadas e em que linguagem de programação seriam feitas.

Foi analisado que, em um projeto de gerenciamento de estoque, um banco de dados desempenharia um papel crucial na organização, armazenamento e manipulação eficiente das informações relacionadas aos produtos, entradas, saídas, quantidades disponíveis e outras variáveis essenciais para a administração do estoque.

A linguagem de programação selecionada para a implementação do sistema de gerenciamento de estoque foi o Python, em virtude de sua sintaxe clara e da facilidade na estruturação de soluções, o que reduz a complexidade do desenvolvimento. Além disso, o Python oferece suporte à biblioteca sqlite3, que facilita o manejo e manipulação de bancos de dados armazenados em arquivos com a extensão .db. Esta biblioteca proporciona uma interface eficiente e integrada para trabalhar com bancos de dados SQLite, permitindo realizar operações como criação, leitura, atualização e exclusão de dados de maneira prática e com baixo custo computacional, sem a necessidade de um servidor de banco de dados dedicado.

Dessa forma, a utilização do sqlite3 no contexto do Python oferece uma solução leve e eficiente para o gerenciamento de dados, sendo especialmente adequada para projetos de pequeno e médio porte, como o controle de estoque.

Além disso, os softwares VS Code e DB Browser for SQLite foram empregados para o desenvolvimento e visualização das atividades relacionadas ao banco de dados. O VS Code foi utilizado como ambiente de desenvolvimento integrado (IDE) para escrever e executar o código Python, enquanto o DB Browser for SQLite facilitou o manejo e a visualização do banco de dados SQLite, proporcionando uma interface gráfica intuitiva para a consulta e manipulação dos dados de estoque de forma eficiente.

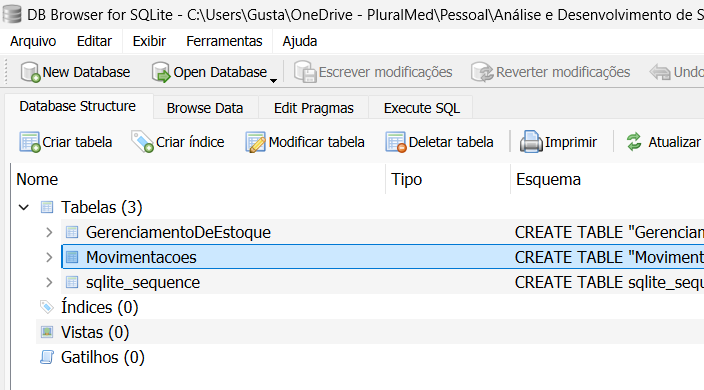
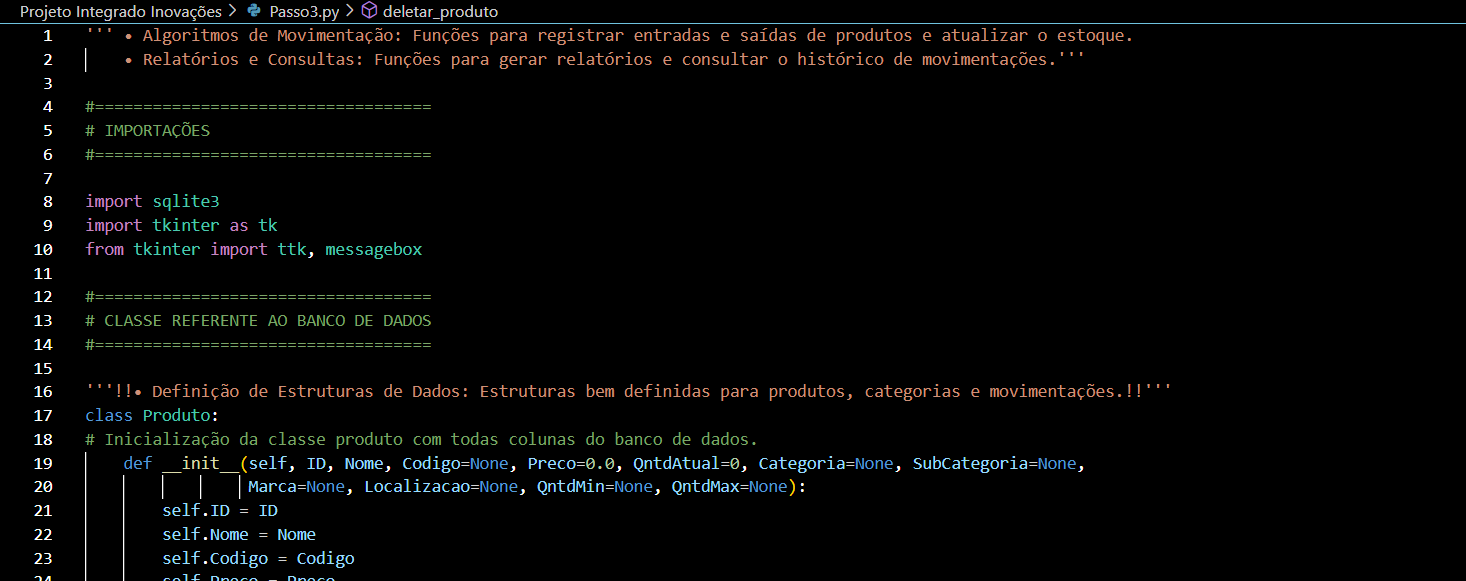


Figura 10 e 11 - Visualização do código no VS CODE / Visualização do Banco de Dados no DB Browser.

#### PRODUTO

No Relatório Sprint 1, foram previamente definidos os dados essenciais para caracterizar um produto, o que facilitou significativamente a elaboração do banco de dados. Com essas definições claras, a estruturação do banco de dados tornou-se mais objetiva, permitindo que as decisões relacionadas a aspectos técnicos fossem concentradas em apenas decisões das informações dos dados como: PK (Primary Key), NN (Not Null), U (Unique), AI (Auto Increment) e qual seu tipo (Integer, Numeric, Text...).

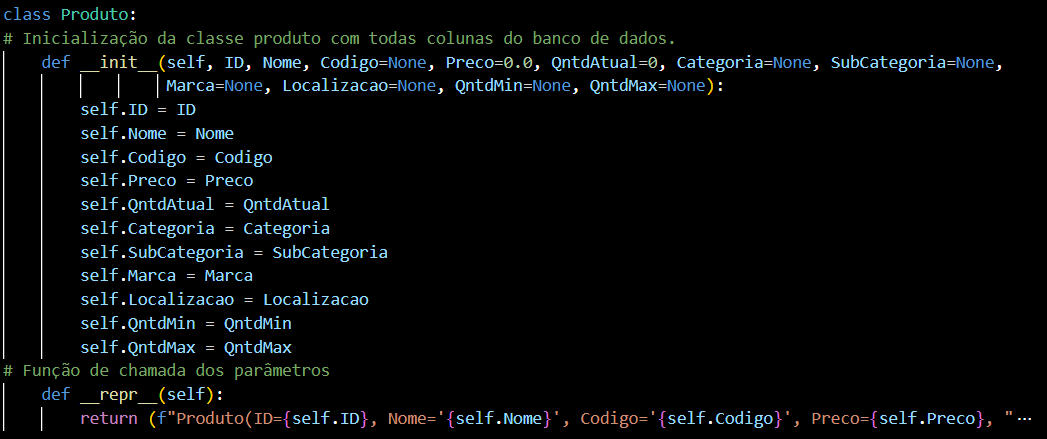


Figura 12 e 13 – Estrutura do produto em python (classe) / Estrutura dos dados do produto (SQL)

#### MOVIMENTAÇÕES

A estrutura de dados das movimentações foi definida para registrar todas as transações de entrada e saída de produtos, permitindo o controle de estoque e o histórico de alterações para cada item. Cada movimentação armazena as informações essenciais para caracterizar a transação e associá-la a um produto específico no banco de dados, garantindo integridade e rastreabilidade das ações.

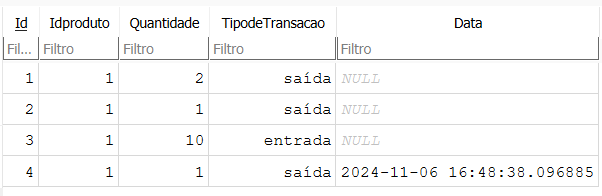
Suas colunas são:

Figura 14 – Estrutura de dados da tabela ‘Movimentações’

* ID: Identificador único para cada movimentação.
* IdProduto: Referência ao produto relacionado à movimentação.
* Quantidade: Quantidade de unidades movimentadas na transação.
* TipoDeTransacao: Especifica o tipo de transação, podendo ser "entrada" para acréscimos ao estoque ou "saída" para retiradas.
* Data: Registra a data e a hora da movimentação, facilitando o rastreamento cronológico.

### ALGORITMOS DE CADASTRO E CONSULTA

Para a implementação dos algoritmos de cadastro e consulta, foram utilizadas funções em Python que interagem diretamente com o banco de dados. Segue abaixo descrições mais detalhadas referentes a essas funções.

* Cadastro

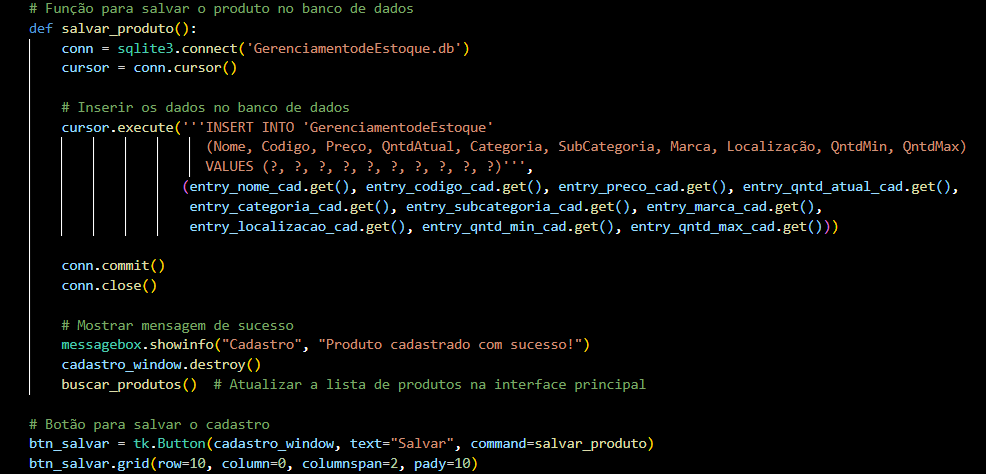


Figura 14 – Função de cadastro de produtos

A função de cadastro de produtos é responsável por permitir a inserção de novos registros de produtos no banco de dados. O processo pode ser descrito como segue:

* + **Criação da Conexão com o Banco de Dados:** Inicialmente, a função estabelece uma conexão com o banco de dados utilizando a variável conn. Essa conexão permite que o Python interaja com o banco de dados SQLite para realizar operações como inserção, leitura e atualização dos dados.
  + **Uso do Cursor para Interagir com o Banco de Dados:** A variável cursor é obtida a partir da conexão, ela que permite a execução de comandos SQL no banco de dados. O cursor funciona como um intermediário entre o Python e o banco de dados, permitindo que as instruções SQL sejam enviadas e os resultados retornados.
  + **Leitura dos Dados da Interface Gráfica:** A interface gráfica do sistema utiliza campos Entry para capturar as informações inseridas pelo usuário (como nome do produto, descrição, preço, etc.). A função coleta esses dados e os prepara para inserção no banco de dados.
  + **Inserção dos Dados no Banco de Dados:** Utilizando o cursor, a função executa um comando SQL de inserção no banco de dados, utilizando os dados capturados nos campos Entry da interface gráfica.
  + **Confirmação de Sucesso e Feedback para o Usuário:** Após a execução do comando de inserção, a função fornece um feedback ao usuário sobre o sucesso da operação.
  + **Atualização da Lista (TreeView):** A lista visual (normalmente implementada com o componente TreeView da biblioteca tkinter) que exibe os dados do banco de dados é atualizada
* Consulta

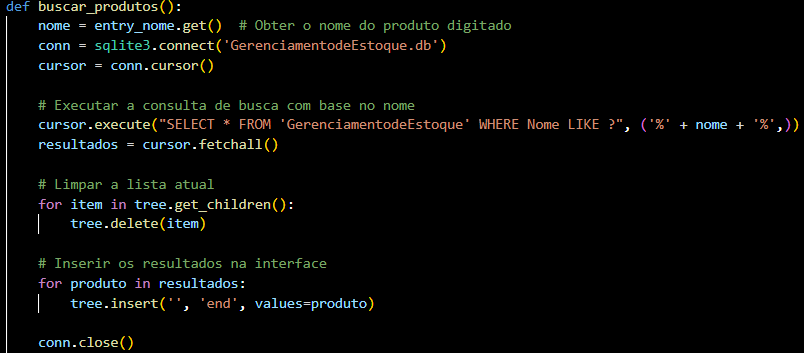


Figura 15 – Função de consulta.

A função de consulta utiliza o valor em Entry\_nome para filtrar a busca dos produtos, ela é implementada no botão “buscar” da tela inicial. O passo a passo do projeto é definido na ordem de:

* + **Criação da Conexão com o Banco de Dados**
  + **Uso do Cursor para Interagir com o Banco de Dados**
  + **Leitura dos Dados da Interface Gráfica**
  + **Execução da Consulta:** é utilizada a instrução *“"SELECT \* FROM 'GerenciamentodeEstoque' WHERE Nome LIKE ?", ('%' + nome + '%',)”* para filtrar os resultados da lista conforme o digitado na Entry.
  + **Atualização da Lista:** Deleta os itens fora do filtro da *Treeview* e a atualiza.

### ALGORITMOS DE MOVIMENTAÇÃO

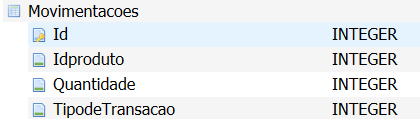


Figura 16 – Estrutura de Organização dos dados de Movimentação.

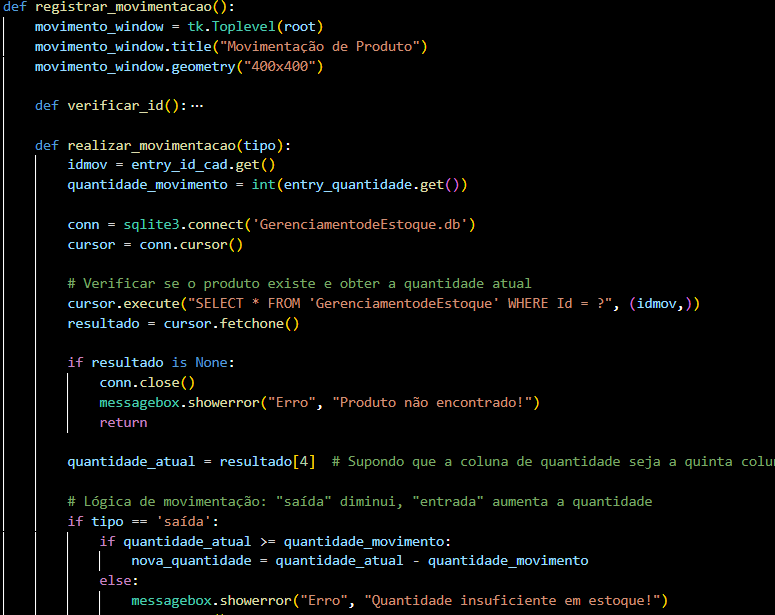
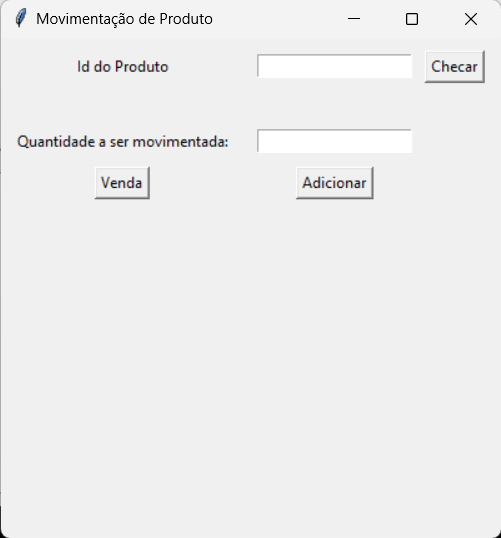
Para implementar os algoritmos de movimentação dentro do código, viu-se a necessidade de outra janela exclusiva para essas movimentações. 

Figura 17 e 18– Janela de movimentação de produtos e código de função dos botões de movimentação..

* Função de verificar id (do produto):

Verifica se o produto com o ID inserido (entry\_id\_cad) existe na tabela. Caso o produto seja encontrado, o nome do produto é exibido em label\_produto. Caso contrário, uma mensagem "Produto não encontrado" é exibida.

* Função de realizar movimentação:

Realiza a movimentação de estoque, seja para **venda** (saída) ou **adição** (entrada), de acordo com o parâmetro tipo, busca o produto na tabela e obtém a quantidade atual. Se o tipo for "saída", verifica se há estoque suficiente. Caso sim, subtrai a quantidade; e não, exibe uma mensagem de erro. Se o tipo for "entrada", adiciona a quantidade ao estoque. Atualiza a quantidade no banco de dados e registra a movimentação na tabela ‘movimentacoes’, incluindo o ID do produto, a quantidade movimentada, o tipo de transação e a data/hora atual (datetime.now()).

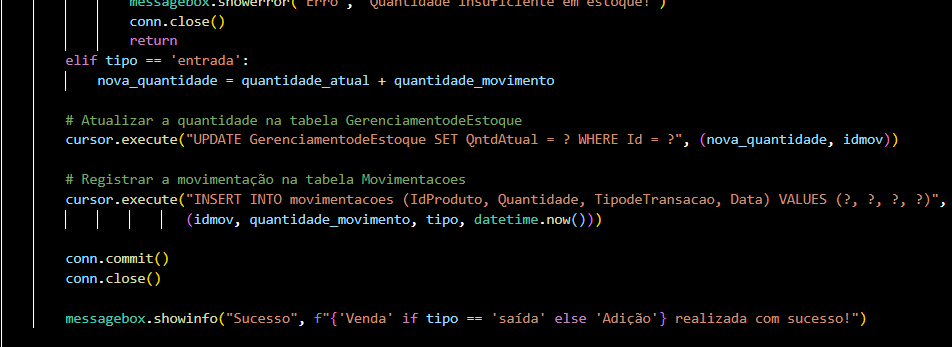


Figura 19 – Continuação da função dos botões de movimentação

### RELATÓRIOS E CONSULTAS

Como já foi evidenciado na figura 9, o sistema já disponibiliza de um relatório visual dos produtos cadastrados, para fins de relatórios escritos e copiáveis foi implementado uma nova função para gerar esses relatórios.

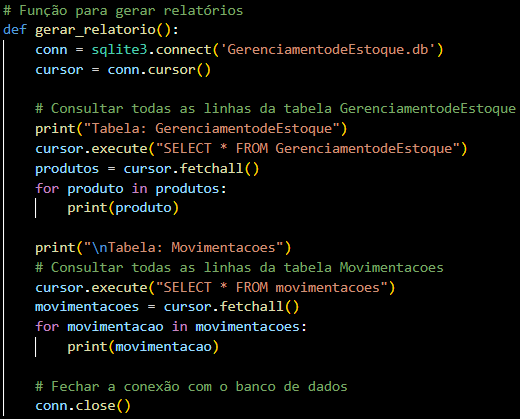


Figura 20 – Função gerar relatório.

* **Criação da Conexão com o Banco de Dados**
* **Uso do Cursor para Interagir com o Banco de Dados**
* **Leitura dos Dados da Interface Gráfica**
* **Impressão dos resultados obtidos:** as variáveis produtos e movimentações recebem todos os itens das tabelas no banco de dados, depois, são utilizadas para impressão no terminal com a função print().

## PASSO 4: DIAGRAMA DE CASOS DE USO

O primeiro passo para a criação do diagrama de casos de uso solicitado para conclusão do 4º passo foi a definição dos atores e as funcionalidades do sistema, os atores foram dados na própria atividade como Usuário, Estoquista e Gerente de Setor.

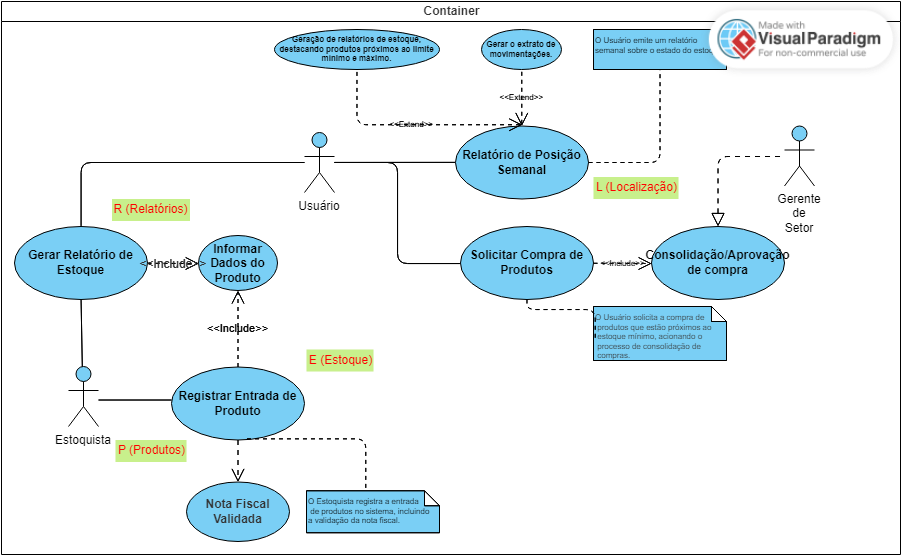


Figura 21 – Diagrama de casos de uso.

Os casos de uso / funcionalidades foram definidos com base nas necessidades operacionais do sistema de gerenciamento de estoque:

* **Gerar Relatório de Estoque**: Permite que o usuário gere relatórios de estoque, destacando produtos que estão próximos dos limites mínimos e máximos.
* **Informar Dados do Produto**: Relaciona-se à obtenção de informações detalhadas sobre produtos específicos no sistema.
* **Registrar Entrada de Produto**: Responsável por registrar a entrada de produtos no estoque. Inclui a validação da nota fiscal para garantir a conformidade dos itens.
* **Solicitar Compra de Produtos**: O usuário pode solicitar a compra de produtos que estão próximos ao limite mínimo no estoque.
* **Consolidação/Aprovação de Compra**: Processo que envolve a consolidação e a aprovação da compra dos produtos solicitados por parte do gerente.
* **Relatório de Posição Semanal:** Emite um relatório semanal com o estado atualizado do estoque.

# CONCLUSÃO

O desenvolvimento do sistema de gerenciamento de estoque para uma empresa de comércio eletrônico em expansão proporcionou uma boa oportunidade de aprendizado e consolidação de conhecimentos nas áreas trabalhadas no curso de Análise e Desenvolvimento de Sistemas. Ao longo do projeto, foi possível aplicar a metodologias ágil Scrum para gerenciar o ciclo de desenvolvimento, comunicação e o monitoramento das atividades. Esse processo não só melhorou o entendimento sobre práticas de gerenciamento de projetos, mas também demonstrou a importância da organização e do planejamento para entregas eficientes e iterativas.

A criação de funcionalidades essenciais, como o cadastro de produtos, atualização de estoque, rastreamento de localização e geração de relatórios, ampliou as habilidades técnicas do programador, que aplicou conhecimentos em lógica de programação, banco de dados e modelagem de sistemas. A definição das estruturas de dados para produtos e movimentações, assim como o uso do banco de dados SQLite, reforçou a compreensão de conceitos de armazenamento e manipulação de dados. As funcionalidades foram desenvolvidas em Python, demonstrando o uso eficiente da linguagem e sua integração com a interface gráfica Tkinter e a biblioteca sqlite3.

As atividades práticas também incluíram a construção de algoritmos de consulta e movimentação de produtos, consolidando o entendimento da lógica por trás de operações comuns em sistemas de estoque.

Por fim, o diagrama de casos de uso possibilitou uma visão ampla das interações entre os atores e o sistema, auxiliando na visualização e validação dos requisitos funcionais.

Esse projeto representou um avanço significativo em termos de aprendizado e aplicação prática de conceitos fundamentais no desenvolvimento de sistemas, contribuindo para o aprimoramento das competências técnicas e metodológicas.

###### REFERÊNCIAS

VISUAL PARADIGM. Sobre nós. Disponível em: https://online.visual-paradigm.com/pt/about-us/. Acesso em: 22/10/2024.

LUCIDCHART, Diagrama de caso de uso UML: O que é, como fazer e exemplos. Disponível em: [https://www.lucidchart.com/pages/pt/diagrama-de-caso-de-uso-uml. Acesso em 22/10/2024](https://www.lucidchart.com/pages/pt/diagrama-de-caso-de-uso-uml.%20Acesso%20em%2022/10/2024).

FREECODECAMP. Como criar e manipular bancos de dados SQL com Python. Disponível em: <https://www.freecodecamp.org/portuguese/news/como-criar-e-manipular-bancos-de-dados-sql-com-python/>. Acesso em: 06/11/2024.

DEVMEDIA. Primeiros Passos no SQL. Disponível em: <https://www.devmedia.com.br/primeiros-passos-no-mysql/28438>. Acesso em: 06/11/2024.