



**PUC**  
**CAMPINAS**  
PONTIFÍCIA UNIVERSIDADE CATÓLICA

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS**

**JULIA DE SOUZA LEANDRO - RA: 25009148**

**MARIA EDUARDA PRUDENTE CASTRO - RA: 25005944**

**RAFAEL DE SOUZA BATISTA - RA: 25005945**

**VITOR SOARES POLIZEL - RA: 24008087**

**RELATÓRIO DE PROJETO:**

**Sistema de Gerenciamento de Estoque de um Armazém**

**CAMPINAS**

**2025**

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS  
ESCOLA POLITÉCNICA,  
Ciências de Dados e Inteligência Artificial**

**RELATÓRIO DE PROJETO:  
Sistema de Gerenciamento de Estoque de um Armazém**

Relatório de projeto de sistema, apresentado no componente curricular Programação e Estrutura de Dados, do curso de Ciências de Dados e Inteligência Artificial, da Escola Politécnica da Pontifícia Universidade Católica de Campinas.

Orientador: Lúcia Filomena de Almeida Guimarães

**CAMPINAS**

**2025**  
**SUMÁRIO**

1.	INTRODUÇÃO.....	1
2.	REQUISITOS .....	2
3.	DESENVOLVIMENTO DO PROJETO.....	3
4.	CONCLUSÃO.....	5
5.	REFERÊNCIAS .....	6

## **1. INTRODUÇÃO**

O projeto consiste no desenvolvimento de um Sistema de Gerenciamento de Estoque de um Armazém, utilizando a linguagem C. O objetivo principal é aplicar os conceitos fundamentais da programação estruturada, como vetores, matrizes, funções, condicionais e laços de repetição, para criar um programa capaz de registrar, atualizar e consultar o estoque de diferentes produtos distribuídos entre diversas filiais ou setores.

Além disso, o sistema visa facilitar a visualização do total de itens disponíveis, a análise de médias e a identificação de produtos e filiais com maior ou menor quantidade em estoque, oferecendo ao usuário um menu interativo e intuitivo.

## 2. REQUISITOS

Os requisitos avaliados definidos pela professora Lúcia Guimarães são:

### 1. CADASTRO INICIAL

- Solicitar ao usuário o número de produtos e o número de filiais (ou setores);
- Criar e preencher uma matriz bidimensional representando o estoque;
- Máximo de 15 produtos e 10 filiais.

### 2. ATUALIZAÇÃO DO ESTOQUE

- Permitir o aumento ou redução do estoque de um produto em uma filial específica;
- Impedir valores negativos no estoque;

### 3. CONSULTAS

#### 3.1. O sistema deve oferecer as seguintes consultas:

- Mostrar o estoque total de cada produto (somando todas as filiais);
- Mostrar o estoque total de cada filial (somando todos os produtos);
- Mostrar o produto com maior quantidade de estoque e menor quantidade de estoque;
- Mostrar a filial com mais estoque e a filial com menos estoque;
- A média de estoque em uma determinada filial, onde a filial é fornecida pelo usuário;
- A média de estoque de um determinado produto, onde o produto é fornecido pelo usuário;

### **3. DESENVOLVIMENTO DO PROJETO**

#### **3.1. Funções Utilizadas**

O projeto foi desenvolvido em linguagem C utilizando a função principal `main()` e funções auxiliares para organizar o código e facilitar a manutenção do sistema. A função `main()` foi responsável pelo menu principal, permitindo ao usuário escolher entre cadastrar, editar, ler informações do estoque ou encerrar o programa.

Já a função `matriz()` teve como objetivo o cadastro e preenchimento dos dados do estoque, utilizando uma matriz bidimensional para representar os produtos e setores. Essa função também foi responsável por validar as entradas do usuário, garantindo que os valores inseridos estivessem dentro dos limites definidos (10 setores e 15 produtos). Além disso, foram implementadas estruturas de repetição e condicionais para controlar o fluxo do programa e evitar erros de entrada.

Durante a implementação da função de cadastro, foi adicionada uma rotina para descartar caracteres residuais da entrada do teclado, utilizando o comando `while (getchar() != '\n');`. Essa verificação impede que o programa interprete entradas anteriores, como o “Enter” digitado pelo usuário, evitando erros nas próximas leituras feitas pelo `scanf()`. Assim, o sistema garante que cada dado inserido seja lido de forma correta e independente, contribuindo para uma execução mais estável e precisa.

### 3.2. Dificuldades enfrentadas e Soluções

Integrantes	Dificuldades e Soluções
Rafael	<p>Creio que minhas principais dificuldades foram em relação ao trabalho em equipe e integração de códigos no mesmo arquivo do diretório.</p> <p>A dificuldade se deu por conta de que, como o projeto foi dividido entre 3 programadores e sem o auxílio de uma ferramenta de versionamento, estabelecer uma conexão entre três arquivos com tipagens e definições de variáveis diferentes, se mostrou ser um desafio. Por vezes me via corrigindo sintaxe de variáveis iguais, contudo com nomes diferentes para cada aluno, ou ainda, percebendo que variáveis que tinham sido declaradas singularmente nas funções específicas mas não haviam sido declaradas ou referenciadas corretamente na main.</p> <p>Contudo, apesar das dificuldades de sintaxe e a necessidade de correções constantes. Muitas das dificuldades foram sanadas com a estrutura que adotamos para o projeto, a saber: criação de funções void que são responsáveis pelas funcionalidades do CRUD e são referenciadas na main e ‘chamadas’ por um Switch. Proporcionando um parsing melhor e maior, tornando mais fácil a mobilidade e correção no código.</p>
Maria Eduarda	<p>Durante o desenvolvimento da minha parte do trabalho, enfrentei algumas dificuldades. Uma delas foi compreender a lógica de programação necessária para implementar a funcionalidade de atualização de dados e garantir a segurança dos dados da matriz.</p> <p>Além disso, precisei garantir que a matriz utilizada não excedesse o tamanho pré-definido, para garantir que o cliente estivesse utilizando um produto de estoque já existente, e também evitar que valores negativos fossem inseridos. Para isso, implementei verificações que garantiam que os dados respeitassem esses limites.</p>
Vitor	<p>As dificuldades encontradas durante o desenvolvimento do tópico de consultas estiveram relacionadas principalmente à dependência entre as partes do código, o que exigia que os módulos de outros integrantes estivessem prontos para que o meu pudesse ser testado e integrado corretamente. Além disso, houve incertezas quanto ao tempo disponível para finalização, o que exigiu uma boa organização do grupo.</p> <p>Também surgiram problemas de sintaxe e lógica, que precisaram ser revisados e estudados com mais atenção para garantir o correto funcionamento do código.</p> <p>Com boa comunicação entre os integrantes e prazos bem definidos, foi possível superar essas dificuldades, realizar os ajustes necessários e concluir a implementação de forma satisfatória.</p>
Julia	<p>Durante a elaboração do relatório, não foram encontradas dificuldades significativas. O principal desafio esteve na organização das informações de forma clara e objetiva, garantindo que o relatório atende-se aos requisitos exigidos pela professora, como também, que estivesse de acordo com o padrão ABNT. Além disso, foi necessário revisar a formatação do documento para manter a padronização entre as seções e as referências bibliográficas.</p>

Figura 1 – Tabela com as dificuldades e soluções encontradas pelos integrantes durante o desenvolvimento do projeto

#### **4. CONCLUSÃO**

O desenvolvimento deste projeto permitiu aplicar de forma prática os conceitos de vetores, matrizes, estruturas de controle e modularização de código em C. A implementação das funções e do menu interativo demonstrou como é possível estruturar programas mais organizados e reutilizáveis. O trabalho contribuiu para consolidar o raciocínio lógico e a compreensão da importância da validação de dados e da interação com o usuário em sistemas simples de gerenciamento.



## **5. REFERÊNCIAS**

BRASIL. PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS.  
Material didático da disciplina Programação e Estrutura de Dados – Curso de Ciência de  
Dados e Inteligência Artificial. Campinas: PUC-Campinas, 2025. Material interno.