## **Introdução**

Neste projeto, modelei um sistema de gerenciamento para uma loja de vestuário fictícia. Essa loja oferece uma variedade de roupas e acessórios, como camisetas, calças, e tênis. O sistema será utilizado por três tipos principais de usuários: gerentes, funcionários e clientes. Os gerentes terão permissões amplas para gerenciar tanto usuários quanto produtos, enquanto os funcionários terão permissões mais restritas, focadas no gerenciamento de produtos. Os clientes terão acesso ao sistema para visualizar produtos e realizar compras.

## **Implementação**

### **Usuários**

#### **Estrutura de Dados**

Para armazenar as informações dos usuários, utilizei um dicionário Python chaves:

* id: um identificador único para cada usuário.
* nome: o nome do usuário.
* senha: a senha do usuário.
* nivel\_permissao: o nível de permissão do usuário (por exemplo, gerente, funcionário, cliente).

#### **Estrutura do Arquivo de Registro**

Os dados dos usuários são armazenados em um arquivo CSV (usuarios.csv) com as colunas:

* id: identificador único do usuário.
* nome: nome do usuário.
* senha: senha do usuário.
* nivel\_permissao: nível de permissão do usuário.

#### **Funcionalidades**

1. **Criação de Usuário (Create)**

Função: criar\_usuario(nome, senha, nivel\_permissao)

Descrição: Cria um novo usuário com as informações fornecidas e o adiciona ao arquivo de registro.

1. **Leitura de Usuário (Read)**

Função: listar\_usuarios()

Descrição: Lista todos os usuários registrados no sistema.

Função: buscar\_usuario\_por\_id(user\_id)

Descrição: Busca um usuário específico pelo seu ID.

1. **Atualização de Usuário (Update)**

Função: atualizar\_usuario(user\_id, nome, senha, nivel\_permissao)

Descrição: Atualiza as informações de um usuário existente com base no seu ID.

1. **Remoção de Usuário (Delete)**

Função: remover\_usuario(user\_id)

Descrição: Remove um usuário do sistema com base no seu ID.

### **Produtos**

#### **Estrutura de Dados**

Para armazenar as informações dos produtos, utilizei um dicionário Python com as seguintes chaves:

* nome: o nome do produto.
* preco: o preço do produto.
* quantidade: a quantidade disponível do produto.

#### **Estrutura do Arquivo de Registro**

Os dados dos produtos são armazenados em um arquivo CSV (produtos.csv) com as colunas:

* nome: nome do produto.
* preco: preço do produto.
* quantidade: quantidade disponível do produto.

#### **Funcionalidades**

1. **Criação de Produto (Create)**

Função: criar\_produto(nome, preco, quantidade)

Descrição: Cria um novo produto com as informações fornecidas e o adiciona ao arquivo de registro.

1. **Leitura de Produto (Read)**

Função: listar\_produtos()

Descrição: Lista todos os produtos registrados no sistema.

Função: buscar\_produto\_por\_nome(nome)

Descrição: Busca um produto específico pelo seu nome.

1. **Atualização de Produto (Update)**

Função: atualizar\_produto(nome, novo\_nome, novo\_preco, nova\_quantidade)

Descrição: Atualiza as informações de um produto existente com base no seu nome.

1. **Remoção de Produto (Delete)**

Função: remover\_produto(nome)

Descrição: Remove um produto do sistema com base no seu nome.

## **Conclusão**

A maior dificuldade foi implementar as funções para receber parâmetros enviados pelo usuário via terminal, por isso acabei preferindo utilizar usuários e produtos já determinados. Infelizmente não consegui criar de forma funcional essa interação entre o script e o terminal. Porém, acho que essa implementação seria mais tranquila com o script já pronto e funcionando caso, em um mundo fictício, ele fosse realmente usado em alguma loja.

Tive algumas dificuldades também na escrita do código, porém achei vários guias na internet e contei com a ajuda de uma pessoa já experiente nesse ramo. Com certeza sem essas ajudas teria sido impossível a conclusão do código, e dessa forma creio que consegui aprender e evoluir muito após término dele.