**Banco de Dados**

Instruções para a criação do banco em SQL

-- Tabelas auxiliares (Modelo, Marca, Cor, Tamanho, Estampa)

CREATE TABLE Modelo (

    idModelo INT PRIMARY KEY IDENTITY(1,1),

    nome\_Modelo NVARCHAR(100) NOT NULL

)

CREATE TABLE Marca (

    idMarca INT PRIMARY KEY IDENTITY(1,1),

    nome\_Marca NVARCHAR(100) NOT NULL

)

CREATE TABLE Cor (

    idCor INT PRIMARY KEY IDENTITY(1,1),

    nome\_Cor NVARCHAR(50) NOT NULL

)

CREATE TABLE Tamanho (

    idTamanho INT PRIMARY KEY IDENTITY(1,1),

    tamanho NVARCHAR(10) NOT NULL

)

CREATE TABLE Estampa (

    idEstampa INT PRIMARY KEY IDENTITY(1,1),

    tipo\_Estampa NVARCHAR(100) NOT NULL

)

-- Tabelas Principas (Produto, Estoque, Venda, ItemVenda e Usuarios)

CREATE TABLE Produto (

    etiqueta NVARCHAR(50) PRIMARY KEY NOT NULL,

    idModelo INT FOREIGN KEY REFERENCES Modelo(idModelo),

    idMarca INT FOREIGN KEY REFERENCES Marca(idMarca),

    idCor INT FOREIGN KEY REFERENCES Cor(idCor),

    idTamanho INT FOREIGN KEY REFERENCES Tamanho(idTamanho),

    idEstampa INT FOREIGN KEY REFERENCES Estampa(idEstampa),

    preco\_unitario DECIMAL(10, 2) NOT NULL

)

CREATE TABLE Estoque (

    etiqueta NVARCHAR(50) FOREIGN KEY REFERENCES Produto(etiqueta),

    quantidade INT NOT NULL

)

CREATE TABLE Venda (

    idVenda INT PRIMARY KEY IDENTITY(1,1),

    valor\_total DECIMAL(10, 2) NOT NULL,

    data\_venda DATETIME DEFAULT GETDATE()

)

CREATE TABLE ItemVenda (

    idItemVenda INT PRIMARY KEY IDENTITY(1,1),

    idVenda INT FOREIGN KEY REFERENCES Venda(idVenda),

    etiqueta NVARCHAR(50) FOREIGN KEY REFERENCES Produto(etiqueta),

    quantidade INT NOT NULL,

    preco\_unitario DECIMAL(10, 2) NOT NULL

)

CREATE TABLE Usuarios (

    id INT PRIMARY KEY IDENTITY(1,1),

    username NVARCHAR(50) NOT NULL,

    senha NVARCHAR(255) NOT NULL

)

CREATE VIEW CONSULTAR\_VENDAS AS (

SELECT

    V.idVenda,

    V.data\_venda,

    P.etiqueta,

    M.nome\_Modelo,

    Marca.nome\_Marca,

    C.nome\_Cor,

    T.tamanho,

    E.tipo\_Estampa,

    IV.quantidade,

    IV.preco\_unitario,

    IV.quantidade \* IV.preco\_unitario AS valor\_total\_item

FROM ItemVenda IV

JOIN Produto P

ON IV.etiqueta = P.etiqueta

JOIN Modelo M

ON P.idModelo = M.idModelo

JOIN Marca

ON P.idMarca = Marca.idMarca

JOIN Cor C

ON P.idCor = C.idCor

JOIN Tamanho T

ON P.idTamanho = T.idTamanho

JOIN Estampa E

ON P.idEstampa = E.idEstampa

JOIN Venda V

ON IV.idVenda = V.idVenda

)

CREATE VIEW VENDAS\_SEMPLES AS (

SELECT

    V.idVenda,

    V.data\_venda,

    SUM(IV.quantidade) AS total\_produtos\_vendidos,

    SUM(IV.quantidade \* IV.preco\_unitario) AS valor\_total\_venda

FROM Venda V

JOIN ItemVenda IV

ON V.idVenda = IV.idVenda

GROUP BY V.idVenda, V.data\_venda

)

SELECT \* FROM CONSULTAR\_VENDAS

SELECT \* FROM VENDAS\_SEMPLES

SELECT \* FROM Modelo

SELECT \* FROM Marca

SELECT \* FROM Cor

SELECT \* FROM Tamanho

SELECT \* FROM Estampa

SELECT \* FROM Produto

SELECT \* FROM Estoque

SELECT \* FROM Venda

SELECT \* FROM ItemVenda

SELECT \* FROM Usuarios

Instruções do código em Python através do Pyodbc

**import pyodbc** #Biblioteca usada para conectar-se a bancos de dados via ODBC (neste caso, SQL Server).

**Função de Conexão ao Banco de Dados**

def connect\_db():

conn = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};'

'SERVER=localhost;'

'DATABASE=BANCOPI;'

'Trusted\_Connection=yes;'

'Encrypt=yes;'

'TrustServerCertificate=yes;')

return conn

**connect\_db():** Função que conecta ao banco de dados BANCOPI no servidor local usando o driver ODBC. Ela retorna uma conexão ativa para ser usada nas operações de banco de dados.

**INTERFACE WEB**

1. **Flask**

Instruções no código em Python

**from flask import Flask**, render\_template, request, redirect, url\_for, flash

Importa as funções e classes necessárias do Flask:

1. Flask: Para criar a aplicação web.
2. render\_template: Renderiza arquivos HTML.
3. request: Acessa dados de uma requisição HTTP (por exemplo, dados de formulários).
4. redirect e url\_for: Redirecionam o usuário para outras rotas dentro da aplicação.
5. flash: Exibe mensagens temporárias (notificações de sucesso ou erro).

import **webbrowser**: Permite abrir URLs no navegador web

import **threading**: Importa a biblioteca de threads para executar ações em paralelo (aqui usada para abrir o navegador ao iniciar o servidor Flask).

Inicializando a Aplicação Flask

app = Flask(\_\_name\_\_)

app.secret\_key = "chave\_secreta\_para\_flash\_messages"

1. **app = Flask(\_\_name\_\_):** Cria uma instância da aplicação Flask, onde \_\_name\_\_ é o nome do módulo atual.
2. **app.secret\_key = "chave\_secreta\_para\_flash\_messages":** Define uma chave secreta usada pelo Flask para proteger as mensagens flash e outras funcionalidades que exigem segurança (como sessões).

Variável de Controle para o Navegador

browser\_opened = False

**browser\_opened = False:** Variável usada para garantir que o navegador só seja aberto uma vez.

Rota Principal

@app.route('/')

def index():

return render\_template('index.html')

1. **@app.route('/'):** Define a rota principal do site (quando acessamos /).
2. **def index():** Função associada à rota. Retorna o template index.html, que será renderizado no navegador.

Rota de Cadastro de Produtos

@app.route('/cadastrar-produto', methods=['GET', 'POST'])

def cadastrar\_produto():

if request.method == 'POST':

1. **@app.route('/cadastrar-produto', methods=['GET', 'POST']):** Define a rota para cadastrar produtos. Essa rota aceita os métodos HTTP GET (para exibir o formulário) e POST (para enviar dados).
2. **if request.method == 'POST'::** Verifica se o método da requisição é POST, ou seja, se o formulário foi enviado.

Processamento do Formulário

etiqueta = request.form['etiqueta']

modelo = request.form['modelo']

marca = request.form['marca']

cor = request.form['cor']

tamanho = request.form['tamanho']

estampa = request.form['estampa']

preco\_unitario = request.form['preco\_unitario']

request.form: Extrai os dados enviados no formulário via POST, como etiqueta, modelo, marca, etc.

1. **HTMl/CSS/JavaScript**