# Lista 8

*Gerado com IA*

**Exercício: Implementando Cadastro e Listagem de Usuários com Banco de Dados SQLite (Versão Simplificada)**

**Objetivo:** Modificar a aplicação para salvar os dados dos usuários em um banco de dados SQLite, mantendo as funcionalidades de cadastro, login e listagem de usuários apenas para usuários logados.

---

## Passo 1: Configurando o Banco de Dados

1. **Prepare o banco de dados:**
   - No início do arquivo `app.py`, adicione o seguinte código para conectar diretamente ao SQLite:

```
import sqlite3

DATABASE = 'usuarios.db'

def conectar_banco():
    return sqlite3.connect(DATABASE)

def criar_tabela():
    conn = conectar_banco()
    cursor = conn.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS usuarios (
            id INTEGER PRIMARY KEY AUTOINCREMENT,
            username TEXT UNIQUE NOT NULL,
            password TEXT NOT NULL,
            nome TEXT NOT NULL
        )
    ''')
    conn.commit()
    conn.close()

# Inicializa o banco de dados ao iniciar a aplicação
criar_tabela()
```

2. **Objetivo deste passo:** Configurar a conexão e garantir que a tabela de usuários seja criada.

---

## Passo 2: Atualizando o Cadastro de Usuários

1. **Atualize a rota `/cadastro`:**
   - Modifique a lógica para salvar usuários diretamente no banco:

```
@app.route('/cadastro', methods=['GET', 'POST'])
def cadastro():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        nome = request.form['nome']

        try:
            conn = conectar_banco()
            cursor = conn.cursor()
            cursor.execute('''
                INSERT INTO usuarios (username, password, nome)
                VALUES (?, ?, ?)
            ''', (username, password, nome))
            conn.commit()
            conn.close()
            return redirect(url_for('login'))
        except sqlite3.IntegrityError:
            return render_template('cadastro.html', erro='Usuário já
cadastrado.')

    return render_template('cadastro.html')
```

2. **Objetivo deste passo:** Garantir que os dados do usuário sejam persistidos no banco.

---

## Passo 3: Atualizando o Login

1. **Modifique a rota `/login`:**
    - Atualize a lógica para validar usuários diretamente no banco:

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']

        conn = conectar_banco()
        cursor = conn.cursor()
        cursor.execute('''
            SELECT * FROM usuarios WHERE username = ? AND password = ?
        ''', (username, password))
        usuario = cursor.fetchone()
        conn.close()

        if usuario:
            session['username'] = username
            resposta = make_response(redirect(url_for('dashboard')))
            resposta.set_cookie('username', username, max_age=60*60*24)
```

```
            return resposta

        return render_template('login.html', erro='Usuário ou senha
inválidos.')

    if 'username' in session:
        return redirect(url_for('dashboard'))

    return render_template('login.html')
```

2. **Objetivo deste passo:** Validar credenciais diretamente do banco de dados.

---

## Passo 4: Atualizando a Listagem de Usuários

1. **Crie ou atualize a rota `/usuarios`:**
   - Atualize a lógica para buscar a lista de usuários no banco:

```
@app.route('/usuarios')
def listar_usuarios():
    if 'username' not in session:
        return redirect(url_for('login'))

    conn = conectar_banco()
    cursor = conn.cursor()
    cursor.execute('SELECT nome, username FROM usuarios')
    usuarios = cursor.fetchall()
    conn.close()

    return render_template('usuarios.html', usuarios=usuarios)
```

2. **Objetivo deste passo:** Obter a lista de usuários cadastrados diretamente do banco.

---

## Passo 5: Testando a Aplicação

1. **Inicie a aplicação:**
   - Execute o servidor com:

```
python app.py
```

2. **Siga os passos:**
   - Acesse `http://127.0.0.1:5000/cadastro` para registrar novos usuários.
   - Faça login com um dos usuários cadastrados.

- Acesse `http://127.0.0.1:5000/usuarios` e verifique que a lista exibe todos os usuários registrados no banco.

3. **Verifique a proteção das rotas:**
   - Tente acessar `/usuarios` sem fazer login e confirme que você é redirecionado para a página de login.

---

**Conclusão do Exercício**

Neste exercício, você aprendeu a:

1. Substituir armazenamento em listas pela persistência em banco de dados SQLite.
2. Garantir a proteção de rotas com base no estado de autenticação do usuário.
3. Trabalhar diretamente com conexões SQLite para manipular dados.

**Desafio Extra:**

1. Adicione um recurso para verificar a força da senha no cadastro.
2. Use hashing para salvar as senhas com segurança (utilize `werkzeug.security`).