Aluno: SIMONE MAYARA DUARTE SOARES
Curso: ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ATIVIDADE 03

Atenção: Antes de realizar a atividade, leia as orientações no AVA.

- **Definição de restrições e acessos**

Neste documento eu descrevo o desenvolvimento e a implantação web "Mr. Temperos III", um sistema de e-commerce para compra e venda de produtos, conforme os requisitos e critérios solicitados.

Para garantir a segurança e a integridade dos dados, implementei um sistema de autenticação utilizando a biblioteca Flask-Login. As seguintes páginas e funcionalidades são acessíveis somente após o usuário realizar o login:

Página Principal (/index): Embora a página principal seja visível, as ações de gestão só estão disponíveis para utilizadores autenticados.

Criação de Anúncios **(/anúncio/novo)**: Apenas usuários logados podem criar anúncios de produtos.

Edição de Anúncios **(/anúncio/editar/<id>):** Usuário só pode editar os anúncios que ele próprio criou.

Exclusão de Anúncios **(/anúncio/deletar/<id>):** Usuário só pode excluir os seus próprios anúncios.

Realizar Perguntas **(/anúncio/<id>/perguntar)**: Apenas usuários logados podem fazer perguntas nos anúncios de outros.

Responder a Perguntas **(/pergunta/<id>/responder**): Apenas o proprietário do anúncio pode responder às perguntas feitas no anúncio.

Comprar um Anúncio **(/anúncio/<id>/comprar):** A funcionalidade de compra está restrita a usuários autenticados.

Adicionar/Remover Favoritos **(/favoritar/<id> e desfavoritar/<id>):** Apenas usuários logados podem gerir a sua lista de favoritos.

Páginas de Relatórios Pessoais: Todas as páginas que exibem dados específicos do utilizador exigem login, segue as páginas:

/meus_anuncios

/minhas_compras

/minhas_vendas

/meus_favoritos

Justificativa: Conforme as boas práticas de segurança apresentadas em aulas, o controle de acesso é fundamental em qualquer aplicação que manipule dados de usuários. O uso do decorador @login_required do Flask-Login em cada rota, garante que nenhuma ação de modificação de dados possa ser executada por um usuário anônimo, protegendo a aplicação contra acessos não autorizados.

- **Atualização da Aparência com Bootstrap**

Na interface da aplicação eu utilizei o framework Bootstrap, uma aparência organizada e responsiva.

Menus: Foi implementada uma barra de navegação responsiva que se adapta a diferentes tamanhos de ecrã.

Formulários: eles foram estilizados com as classes do Bootstrap, proporcionando uma experiência bonita e fácil.

Tabelas e Listas: As listas de produtos, compras e vendas são apresentadas em tabelas e cartões, tornando a visualização dos dados clara e organizada.

Justificativa: conforme foi discutido em aula, o uso do Bootstrap acelera o desenvolvimento e garante a consistência visual resolvendo problemas de responsividade, de forma padronizada e eficiente.

- **Publicação do Código-Fonte no GitHub**

O código-fonte final e funcional da aplicação foi publicado num repositório público no GitHub, conforme foi pedido na atividade III.

Link do Repositório: https://github.com/SimoneSoaress/mr_tempero_III

- **Implantação no PythonAnywhere**

A aplicação foi implantada na plataforma PythonAnywhere. O processo seguiu os seguintes passos:

Criação da Conta: Foi criada uma conta gratuita do tipo "Beginner" no PythonAnywhere.

Criação da Web App: No painel, foi criada uma nova "Web App" utilizando o framework Flask e Python 3.10.

Clone do Repositório: Através de um terminal Bash no PythonAnywhere, a pasta padrão mysite foi apagada e o projeto foi clonado do GitHub para o meu projeto no lugar desse, com o comando git clone.

Criação do Ambiente Virtual: Foi criado um ambiente virtual (venv) e todas as dependências do ficheiro requirements.txt foram instaladas com pip install -r requirements.txt.

Configuração do WSGI: O ficheiro de configuração WSGI foi editado para apontar para a instância da aplicação Flask.

Configuração do Virtualenv: Na aba "Web", o caminho para o ambiente virtual (/home/SimoneMayaraSoares/mysite/venv) foi configurado.

Criação da Base de Dados: No terminal Bash, o comando flask db upgrade foi executado para criar a base de dados app.db e todas as suas tabelas no servidor.

Criação do Primeiro Usuário: Através do flask shell, foi criado um usuário administrador para permitir o primeiro acesso à aplicação.

Reload da Aplicação: A aplicação foi recarregada na aba "Web" para aplicar todas as configurações.

- **Link de Acesso ao Sistema Implantado**

A aplicação está totalmente funcional e pode ser acedida através do seguinte link público:

Link de Acesso: http://SimoneMayaraSoares.pythonanywhere.com