Exercício Prático: Construindo um Blog Dinâmico com Flask e Bootstrap

Baseado nos conceitos da **Aula 10**

Objetivo

Este exercício prático permitirá que você aplique os conceitos de **back-end**, **Flask**, **rotas**, **views**, **templates com Jinja2** e a **integração com o Bootstrap** usando o CDN para criar um blog simples, mas funcional. O projeto incluirá navegação, páginas dinâmicas e tratamento de erros, cobrindo todos os tópicos da aula.

Passo 1: Configuração do Projeto

1. Crie um novo diretório para o seu projeto e, dentro dele, configure um **ambiente virtual** Python.
2. Ative o ambiente virtual e instale o Flask com o comando `pip install flask`.
3. Crie o arquivo principal da sua aplicação (`app.py`) e um diretório chamado `templates` para os seus arquivos HTML.

Passo 2: Construindo o Template Base (`base.html`)

Crie um arquivo `base.html` dentro do diretório `templates`. Este será o template principal, que as outras páginas herdarão.

1. Adicione a estrutura HTML básica e o link para o **Bootstrap 5 CSS** via CDN no `<head>`.
2. Defina um bloco para o título da página (`{ % block title % }{ % endblock % }`).
3. Adicione a barra de navegação (`<nav>`) do Bootstrap no `<body>`. A navegação deve ter links para as páginas "Home"e "Sobre"(que você criará no próximo passo).
4. Defina o bloco de conteúdo principal (`{ % block body % }{ % endblock % }`).
5. Inclua o **Bootstrap 5 JS** via CDN antes da tag de fechamento `</body>`.

Passo 3: Implementando Rotas e Views

No arquivo `app.py`, importe a classe `Flask` e a função `render_template`. Crie a instância da aplicação `app = Flask(__name__)`.

1. Crie a rota principal (`/`) para a página inicial, que renderizará o template `index.html`.
2. Crie a rota `/about` para a página "Sobre", que renderizará o template `about.html`.
3. Crie a rota `/user/<name>` para o perfil do usuário. A função de view deve aceitar um argumento `name` e renderizar o template `user.html`, passando o nome do usuário para o template.

Passo 4: Criando Páginas Específicas

1. Crie o arquivo `index.html` que **herda** o `base.html`. No bloco de conteúdo, adicione um cabeçalho e um parágrafo que descrevem o blog.
2. Crie o arquivo `about.html`, que também herda o `base.html`. No bloco de conteúdo, adicione informações sobre o projeto ou sobre você.
3. Crie o arquivo `user.html`, herdando o `base.html`. No bloco de conteúdo, use um cabeçalho `<h1>` para saudar o usuário, por exemplo, `Olá, {{ name }}!`.

Passo 5: Tratamento de Erros

1. Crie um template `404.html` para lidar com páginas não encontradas.
2. No arquivo `app.py`, use o decorador `@app.errorhandler(404)` para criar uma função que renderize o `404.html` quando um erro 404 ocorrer.

Passo 6: Execução

1. Defina a variável de ambiente `FLASK_APP` para o nome do seu arquivo Python (ex: `export FLASK_APP=app.py`).
2. Execute o servidor com `flask run` ou `python app.py`.
3. Teste as rotas no seu navegador. Tente acessar a página principal, a página "Sobre"e a página de usuário com diferentes nomes. Por fim, tente uma URL que não existe para testar a página de erro 404.