Etec
Registro

python™

Flask

Programação Web III

Prof. Diego Max

Etec
Registro

python™

Aula 03: *Templates e Arquivos Estáticos*

Arquivo base (template) HTML

Flask

Aprenderemos agora como utilizar um página HTML base que servirá como modelo para todas as outras páginas de nossa aplicação. O primeiro passo é criar esse arquivo servirá de modelo.

Para isso, criaremos um arquivo com o nome “base.html” dentro da nossa pasta de views.

Nesse arquivo iremos montar a estrutura que queremos que se repita nas demais páginas.

Arquivo base (template) HTML

Flask

Parte do código do arquivo “base.html” ficará assim (coloque a estrutura completa do HTML):

```
...
    <title>The Games</title>
</head>
<body>
    <header>
        <h1 style="text-align: center;">The Games</h1>
        <hr>
    </header>
    <!-- Aqui irá o conteúdo das outras páginas -->
    <footer>
        <hr>
        <p style="text-align: center;"> Todos os direitos reservados ®</p>
    </footer>
</body>
</html>
```

Arquivo base (template) HTML

Flask

Note que na parte que deixamos comentado será inserido o conteúdo das demais páginas. Agora só precisamos informar isso com as tags do Jinja na página HTML. Para isso, criamos um bloco de conteúdo e atribuímos a esse bloco um nome. No nosso caso, chamaremos esse bloco de “content”, logo o código ficará assim:

```
    <header>
        <h1 style="text-align: center;">The Games</h1>
        <hr>
    </header>
    {% block content %}
    <!-- Aqui irá o conteúdo das outras páginas -->
    {% endblock content %}
    <footer>
        <hr>
        <p style="text-align: center;"> Todos os direitos reservados ®</p>
    </footer>
```

Arquivo base (template) HTML

A saída do arquivo “base.html” será esse:

Tendo nosso arquivo base pronto, agora iremos importá-lo nas demais páginas, ou seja, nas páginas: “index.html”, “games.html” e “cadgames.html”.

Para importar o template em outra página utilizaremos o código {% extends 'base.html' %}, no início do arquivo. Depois devemos nomear o bloco de conteúdo que queremos inserir na página base com o código {% block content %}, sendo “content” o nome do bloco. No final, nosso arquivo “index.html” ficará assim:

```
{% extends 'base.html' %}
{% block content %}
    <h2>Esta é a Homepage.</h2>
{% endblock content %}
```

Arquivo base (template) HTML

E o resultado será esse:

Faremos isso agora, para as páginas restantes. Primeiro na "**games.html**":

```
{% extends 'base.html' %}
{% block content %}
    <h2>Esta é a página de Games.</h2>
    <hr>
    <p><strong>Game mais jogado no momento:</strong></p>
    {% for k, v in game.items() %}
        <details>
            <summary>{{k}}</summary>
            <p>{{v}}</p>
        </details>
    {% endfor %}
```

Arquivo base (template) HTML

E o resultado será esse:

Arquivo base (template) HTML

Flask

Também na página “**cadgames.html**”:

```
{% extends 'base.html' %}
{% block content %}
    <h2>Cadastro de Games</h2>
    <form action="{{url_for('cadgames')}}", method="POST">
        <label for="titulo">Título:</label>
        <input type="text" name="titulo" required>
        <label for="ano">Ano:</label>
        <input type="text" name="ano" required>
        <label for="categoria">Categoria:</label>
        <input type="text" name="categoria" required>
        <input type="submit" value="Cadastrar">
    </form>
    <br>
    <hr>
    <h4>Games cadastrados:</h4>
    {% for game in gamelist %}
    Título: <strong>{{game.Título}}</strong> <br>
    Ano: <strong>{{game.Ano}}</strong> <br>
    Categoria: <strong>{{game.Categoria}}</strong> <br><br>
    {% endfor %}
{% endblock content %}
```

Arquivo base (template) HTML

Resultado:

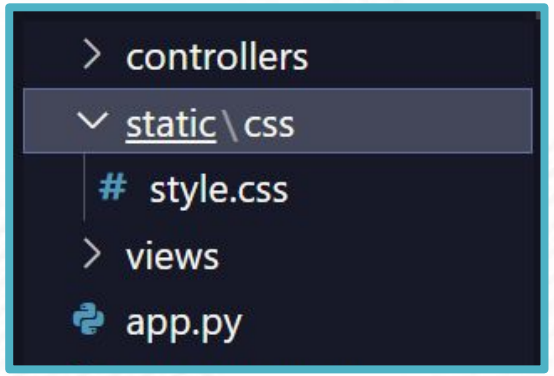
Agora as páginas de nossa aplicação fazem o uso de arquivo base HTML como template. No próximo tópico aprenderemos a manipular arquivos estáticos como folhas de estilo, scripts e imagens.

Configurando arquivos estáticos no Flask

Os arquivos estáticos de uma aplicação são arquivos usados para melhorar a aparência das páginas e, consequentemente, a experiência do usuário.

Esses arquivos podem ser folhas de estilo, scripts e imagens. É uma boa prática que todos os arquivos estáticos de uma aplicação Flask sejam armazenados em um diretório “static” na raiz do projeto e, dentro deste diretório, uma nova pasta contendo os tipos de arquivos estáticos (CSS, JS e IMG).

Em nosso projeto, inicialmente adicionaremos um arquivo CSS. Para isso, o primeiro passo é criar uma pasta como o nome “static” na raiz do projeto. Dentro da pasta “static” criaremos uma pasta com o nome “css” e por fim dentro da pasta “css” criaremos o arquivo “style.css”. A estrutura ficará assim:



Configurando arquivos estáticos no Flask

No arquivo “style.css” faremos apenas duas configurações para testar as modificações, conforme código a seguir.

```
* {
    color: blue;
    margin: 5px;
}
```

Agora devemos importar nosso CSS dentro do arquivo “base.html” que está servindo como template para as demais páginas, assim a folha de estilo será importado também nas demais páginas. Para isso, dentro das tags <head> do arquivo “base.html” inclua a seguinte linha:

```
<link rel="stylesheet" href="{{url_for('static', filename='css/style.css')}}">
```

Veja que utilizamos o {{url_for()}} do Jinja para indicar o diretório onde está nosso arquivo CSS, informando o nome da pasta e o nome do arquivo.

Configurando arquivos estáticos no Flask

Feito isso, as modificações já devem ser aplicadas na página, conforme o resultado a seguir:

Para importar arquivos de scripts ou imagens o processo se repete, lembrando que arquivos Javascript por padrão ficam armazenados em uma pasta com o nome “js” e imagens em uma pasta com o nome “img”. Essas pastas por sua vez, devem estar dentro do diretório “static”.

Encerramos aqui a Aula 02, onde vimos sobre Controllers, Requisições HTTP, Templates e Static Files, na próxima aula veremos como integrar nossa aplicação com uma API.

Etec
Registro

python™

Flask

Programação Web III

Prof. Diego Max