### Conteúdo

- render\_template
- url\_for
- request
- formulários HTML
- métodos HTTP POST e GET

### **Usando Templates**

### Importando o render\_template

from flask import Flask, render\_template

• Para usar templates HTML no Flask, precisamos importar a função render\_template do módulo flask.

### <u>Usando Template no Flask</u>

```
@app.route('/formulario')
def form():
    return render_template('formulario.html')
```

- Definimos uma rota '/formulario' utilizando o decorador @app.route().
- A função form() retorna o conteúdo renderizado do arquivo formulario.html usando render\_template.
- O Flask procura por templates na pasta templates por padrão, então certifique-se de ter um arquivo formulario.html nesta pasta.

# Configurar rota para receber Formulário

### Alterando a rota 'formulario'

- Atualização da Rota /formulario:
  - Adicionamos a lista de métodos permitidos ['POST', 'GET']
     como um parâmetro na definição da rota utilizando o decorador
     @app.route()
  - Isso permite que a rota /formulario aceite requisições tanto
     POST quanto GET.
  - Agora a função associada à rota /formulario pode processar dados enviados por formulários HTML usando POST e também retornar o formulário HTML usando GET.

### Alterando a rota 'formulario'

Código-fonte após alteração:

```
# trecho de código app.py
@app.route('/formulario', methods=['POST', 'GET'])
def form():
    if request.method == 'GET':
        return render_template('formulario.html')
    else:
        name = request.form['name']
        return name
```

## Formulário HTML e url\_for

### url\_for

- url\_for('form') : Gera dinamicamente o URL para a rota associada à função form() no Flask.
- Para utilizarmos a função url\_for devemos importá-la a partir do módulo flask:

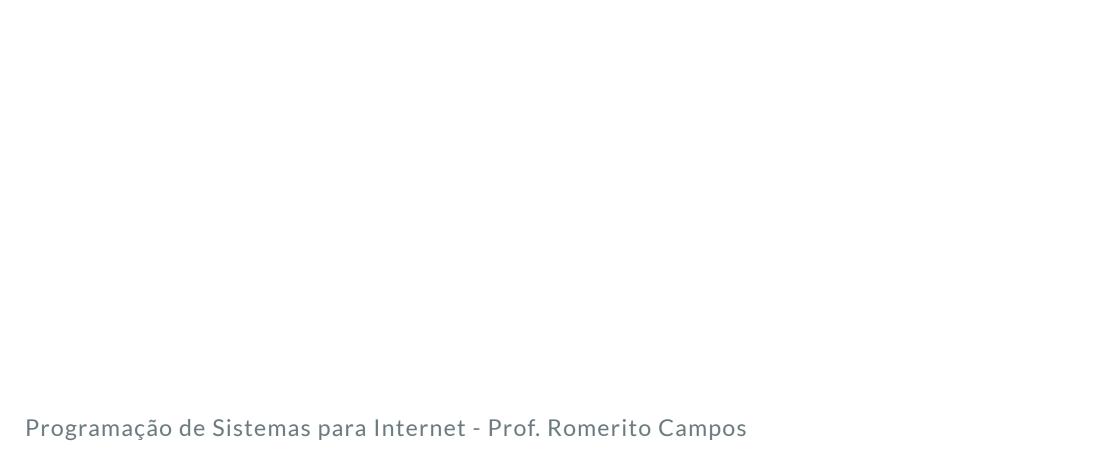
from flask import url\_for

### Explicação do Código

Trecho de código de formulario.html onde usamos url\_for:

### Funcionamento do url\_for

- url\_for('form'): No contexto do template HTML do Flask, url\_for é uma função que gera URLs para funções view. Neste caso específico, 'form' refere-se à função form() definida no arquivo app.py que manipula a rota associada.
- method="POST": Define que os dados do formulário serão enviados via método POST, o que é adequado para enviar dados sensíveis ou que precisam ser processados no servidor.



# Objeto request

### Explicação do Código

```
@app.route('/formulario', methods=['POST', 'GET'])
def form():
    if request.method == 'GET':
        return render_template('formulario.html')
    else:
        name = request.form['name']
        return name
```

### Funcionamento do request

• request : No Flask, request é um objeto global que representa a requisição HTTP recebida do cliente. Ele contém informações sobre a requisição atual, como parâmetros de URL, dados do formulário, cabeçalhos e métodos HTTP.

### Papel do request no Código

- 1. Rota /formulario e Métodos Permitidos:
  - A rota /formulario é configurada para aceitar tanto requisições
     GET quanto POST através do parâmetro
     methods=['POST', 'GET'].

### Papel do request no Código

#### 2. Condicionais Baseados no Método da Requisição:

- o if request.method == 'GET': : Verifica se a requisição é do tipo
  GET . Nesse caso, o código retorna o template HTML
  formulario.html usando render\_template
- else: Se a requisição não for GET (ou seja, é POST neste caso), o código continua a executar. Aqui, request.form['name'] é usado para acessar os dados enviados com o nome name. Isso é possível porque o formulário HTML enviado via método POST envia os dados como um objeto form dentro do request.

### Papel do request no Código

#### 3. Retorno de Dados:

return name: Retorna o valor inserido no campo de formulário name de volta como resposta HTTP. No entanto, em uma aplicação real, geralmente você processaria os dados recebidos antes de retorná-los.

### Exemplo de Uso

No exemplo acima, request é crucial para determinar o método da requisição (GET ou POST) e para acessar os dados enviados pelo usuário através do formulário HTML.

### **Considerações**

- Certifique-se de importar request no início do seu arquivo app.py usando from flask import request.
- Use condicionais apropriados ( if request.method == 'GET': e else: ) para manipular diferentes tipos de requisições HTTP dentro das suas funções view no Flask.

Esta explicação mostra como o request é utilizado para manipular dados de formulário e gerenciar diferentes tipos de requisições HTTP em uma aplicação Flask.

