

# Server-Side – Métodos HTTP

---

Católica de Santa Catarina – Centro Universitário  
Jaraguá do Sul - SC, Brasil

Professor: Ph.D. Andrei Carniel (Prof. Andrei)

Contato: [andrei.carniel@gmail.com](mailto:andrei.carniel@gmail.com)

[linktr.ee/andrei.carniel](https://linktr.ee/andrei.carniel)



# 1. Primeiros sites em flask – Continuando...

---

# Métodos HTTP

---

- GET

- Solicita a representação de um recurso para um servidor. Retornam apenas dados. Pode usar paginação (fragmenta a resposta em páginas e lê página a página).

- POST

- Submeter uma entidade a um recurso específico causando uma mudança de estado no recurso (Envia informações a um servidor, geralmente de formulários).

- PUT

- Substitui as atuais representações do recurso de destino pela carga de dados da requisição (edita arquivos já existentes).

- DELETE

- Remove um recurso específico.

- PATCH

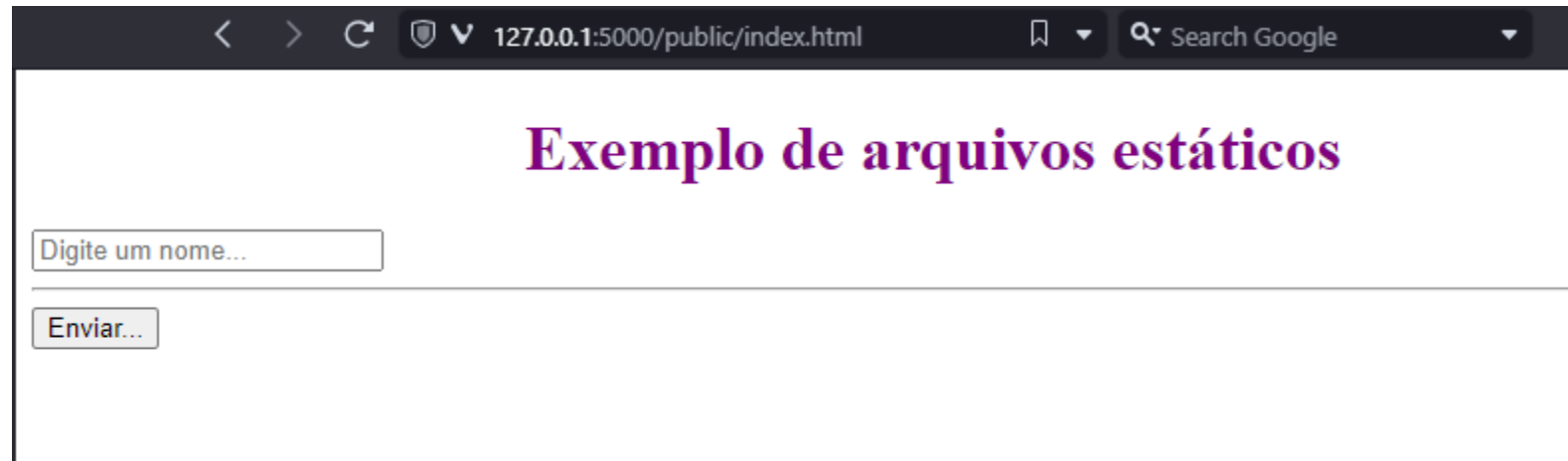
- Utilizado para aplicar modificações parciais.

- HEAD

- Solicita uma resposta de forma idêntica ao GET, porém sem conter o corpo da resposta (retorna apenas o cabeçalho da resposta).

# Aula passada

---



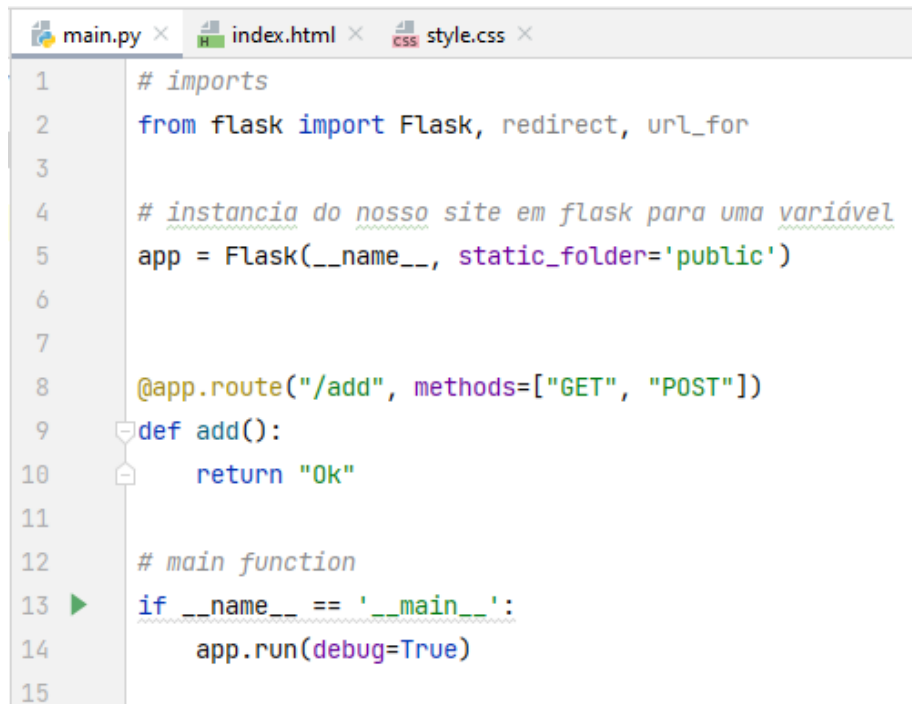
# Vamos modificar o main.py



```
1  # imports
2  from flask import Flask, redirect, url_for
3
4  # instancia do nosso site em flask para uma variável
5  app = Flask(__name__, static_folder='public')
6
7
8  @app.route("/add", methods=["POST"])
9  def add():
10     return "Ok"
11
12  # main function
13  if __name__ == '__main__':
14     app.run(debug=True)
15
```

# Teste a tela...

- No Firefox, clique com o botão direito e inspecionar.
  - Teste o método com POST.
  - Teste o método com GET.
- Modifique e teste novamente:



```
main.py x index.html x style.css x
1  # imports
2  from flask import Flask, redirect, url_for
3
4  # instancia do nosso site em flask para uma variável
5  app = Flask(__name__, static_folder='public')
6
7
8  @app.route("/add", methods=["GET", "POST"])
9  def add():
10     return "Ok"
11
12  # main function
13  if __name__ == '__main__':
14     app.run(debug=True)
15
```

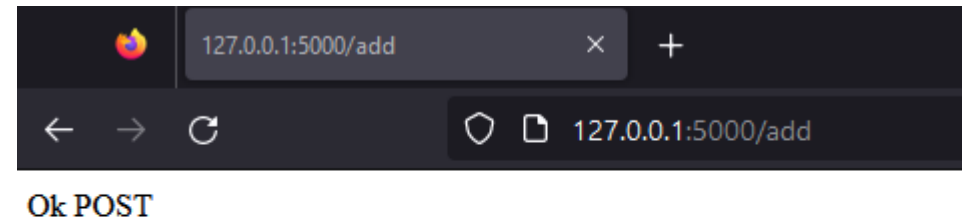
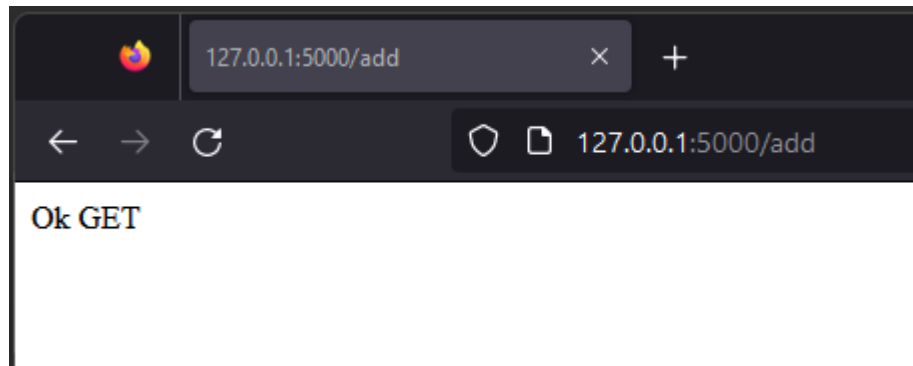
# Podemos usar GET e POST simultaneamente

- Para isso precisamos delimitar os métodos.
- Importar o *request*.
- Comparar o *request*.

```
main.py x index.html x style.css x
1  # imports
2  from flask import Flask, request
3
4  # instancia do nosso site em flask para uma variável
5  app = Flask(__name__, static_folder='public')
6
7  @app.route("/add", methods=["GET", "POST"])
8  def add():
9      if (request.method == "POST"):
10         return "Ok POST"
11     else:
12         return "Ok GET"
13
14  # main function
15  if __name__ == '__main__':
16     app.run(debug=True)
17
```

# Resultados esperados...

---



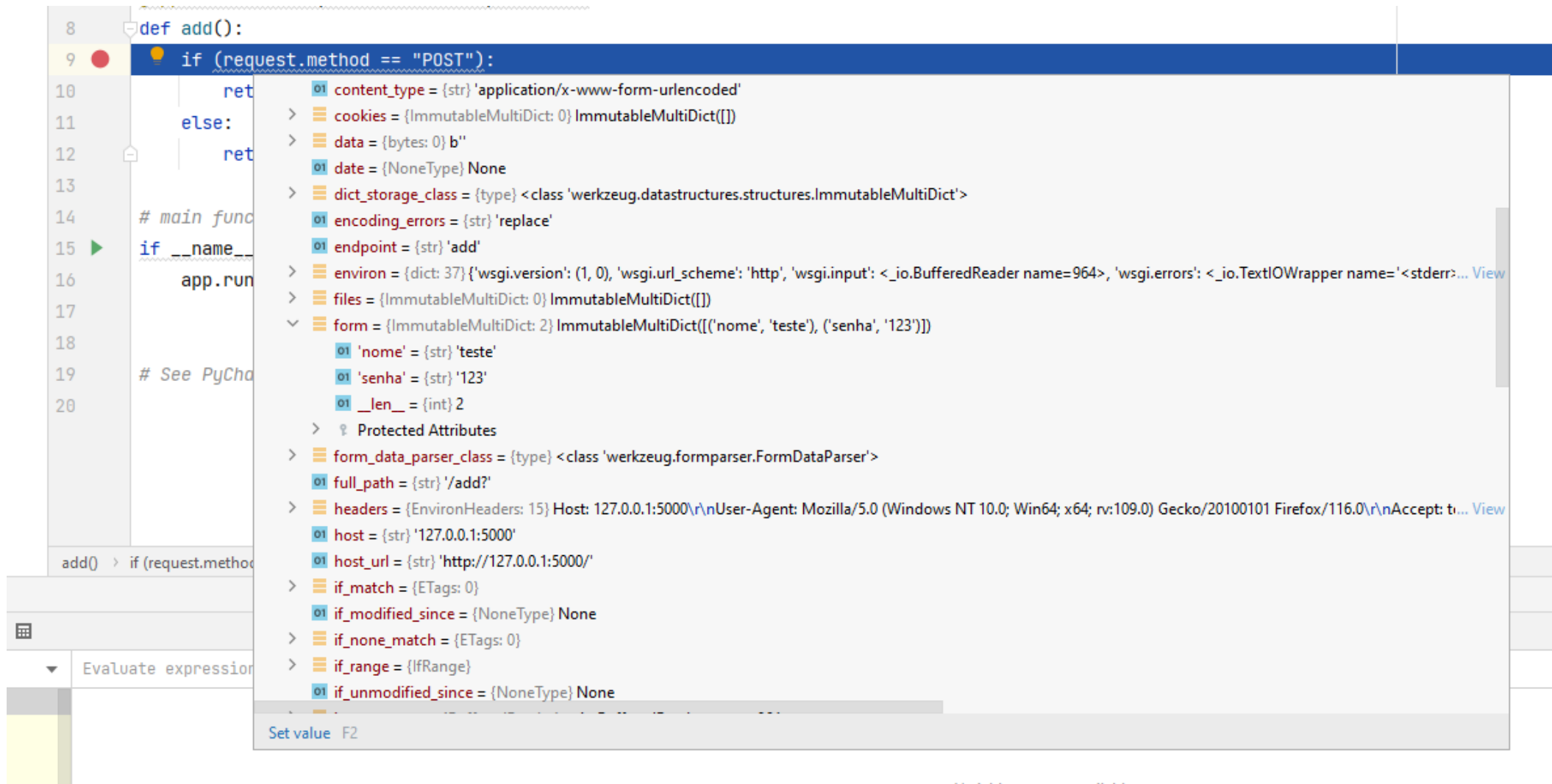


# Fazendo um login...

- Altere o arquivo index.html

```
main.py x index.html x style.css x
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Static folder with files</title>
6   <link rel="stylesheet" href="style.css"/>
7 </head>
8 <body>
9 <form action="http://127.0.0.1:5000/add" method="post">
10   <h1>Exemplo de arquivos estáticos</h1>
11   <input type="text" name="nome" id="index_nome" placeholder="Digite seu nome...">
12   <input type="password" name="senha" id="index_senha" placeholder="Digite sua senha...">
13   <hr>
14   <button type="submit">Enviar...</button>
15 </form>
16 </body>
17 </html>
```

# Debugando...



```
8 def add():
9     if (request.method == "POST"):
10         ret
11     else:
12         ret
13
14 # main func
15 if __name__ == '__main__':
16     app.run()
17
18 # See PyCharm documentation for more details
19
20
```

Variable inspection window:

- content\_type = {str} 'application/x-www-form-urlencoded'
- cookies = {ImmutableMultiDict: 0} ImmutableMultiDict({})
- data = {bytes: 0} b''
- date = {NoneType} None
- dict\_storage\_class = {type} <class 'werkzeug.datastructures.structures.ImmutableMultiDict'>
- encoding\_errors = {str} 'replace'
- endpoint = {str} 'add'
- environ = {dict: 37} {'wsgi.version': (1, 0), 'wsgi.url\_scheme': 'http', 'wsgi.input': <\_io.BufferedReader name=964>, 'wsgi.errors': <\_io.TextIOWrapper name='<stderr>...'>, 'wsgi.path': '/add?', 'REQUEST\_METHOD': 'POST', 'SCRIPT\_NAME': '', 'SERVER\_NAME': '127.0.0.1', 'SERVER\_PORT': '5000', 'HTTP\_HOST': '127.0.0.1:5000', 'HTTP\_USER\_AGENT': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/116.0', 'HTTP\_ACCEPT': 'text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8', 'HTTP\_REFERER': 'http://127.0.0.1:5000/', 'HTTP\_COOKIE': '', 'PATH\_INFO': '/add?', 'QUERY\_STRING': ''}
- files = {ImmutableMultiDict: 0} ImmutableMultiDict({})
- form = {ImmutableMultiDict: 2} ImmutableMultiDict([('nome', 'teste'), ('senha', '123')])
  - 'nome' = {str} 'teste'
  - 'senha' = {str} '123'
  - \_\_len\_\_ = {int} 2
- Protected Attributes
- form\_data\_parser\_class = {type} <class 'werkzeug.formparser.FormDataParser'>
- full\_path = {str} '/add?'
- headers = {EnvironHeaders: 15} Host: 127.0.0.1:5000\r\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/116.0\r\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8\r\nReferer: http://127.0.0.1:5000/\r\nCookie: ''
- host = {str} '127.0.0.1:5000'
- host\_url = {str} 'http://127.0.0.1:5000/'
- if\_match = {ETags: 0}
- if\_modified\_since = {NoneType} None
- if\_none\_match = {ETags: 0}
- if\_range = {IfRange}
- if\_unmodified\_since = {NoneType} None

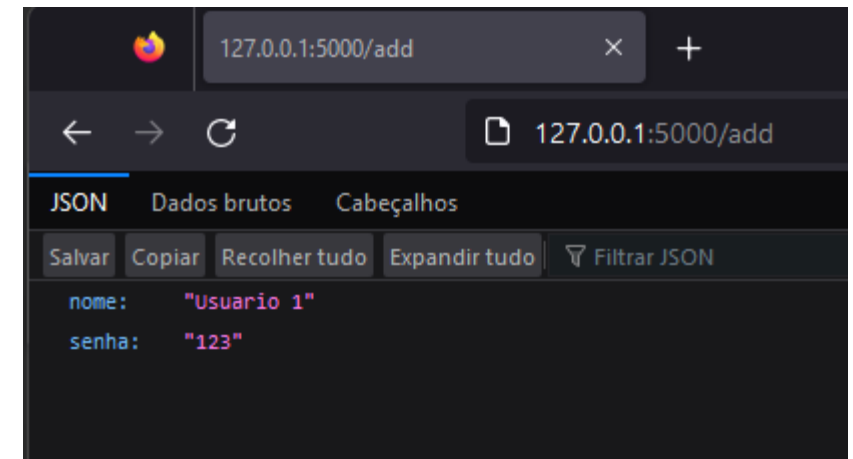
Set value F2

# Adicione o seguinte código...

```
main.py x index.html x style.css x
1  # imports
2  from flask import Flask, request
3
4  # instancia do nosso site em flask para uma variável
5  app = Flask(__name__, static_folder='public')
6
7  @app.route("/add", methods=["GET", "POST"])
8  def add():
9      if (request.method == "POST"):
10         return "Ok POST result = %s." % request.form['nome']
11     else:
12         return "Ok GET"
13
14  # main function
15  if __name__ == '__main__':
16     app.run(debug=True)
17
```

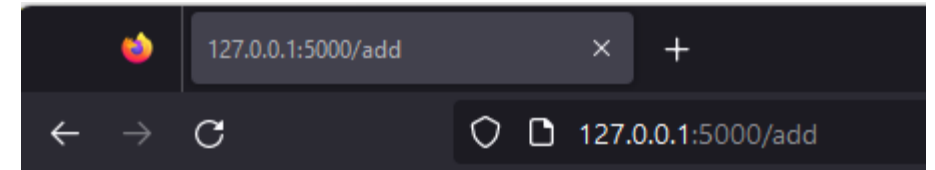
# Podemos recuperar os dados

```
main.py x index.html x style.css x
1  # imports
2  from flask import Flask, request
3
4  # instancia do nosso site em flask para uma variável
5  app = Flask(__name__, static_folder='public')
6
7  @app.route("/add", methods=["GET", "POST"])
8  def add():
9      if (request.method == "POST"):
10         data = request.form
11         print(data['nome'])
12         print(data['senha'])
13         return data
14     else:
15         return "Ok GET"
16
17 # main function
18 if __name__ == '__main__':
19     app.run(debug=True)
```



# Podemos fazer com JSON

```
main.py x index.html x style.css x
1  # imports
2  from flask import Flask, request
3  from json import dumps
4
5  # instancia do nosso site em flask para uma variável
6  app = Flask(__name__, static_folder='public')
7
8  @app.route("/add", methods=["GET", "POST"])
9  def add():
10     if (request.method == "POST"):
11         js = dumps(request.form)
12         return js
13     else:
14         return "Ok GET"
15
16 # main function
17 if __name__ == '__main__':
18     app.run(debug=True)
19
```



{"nome": "Usuario 1", "senha": "123"}