

Server-Side - Flask

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

Professor: PhD Andrei Carniel (Prof. Andrei)

Contato: andrei.carniel@gmail.com



1. Primeiros sites em flask

Importante!

- <https://flask.palletsprojects.com/en/2.3.x/>
- Digite “flask documentation” no Google.

Primeiro Site

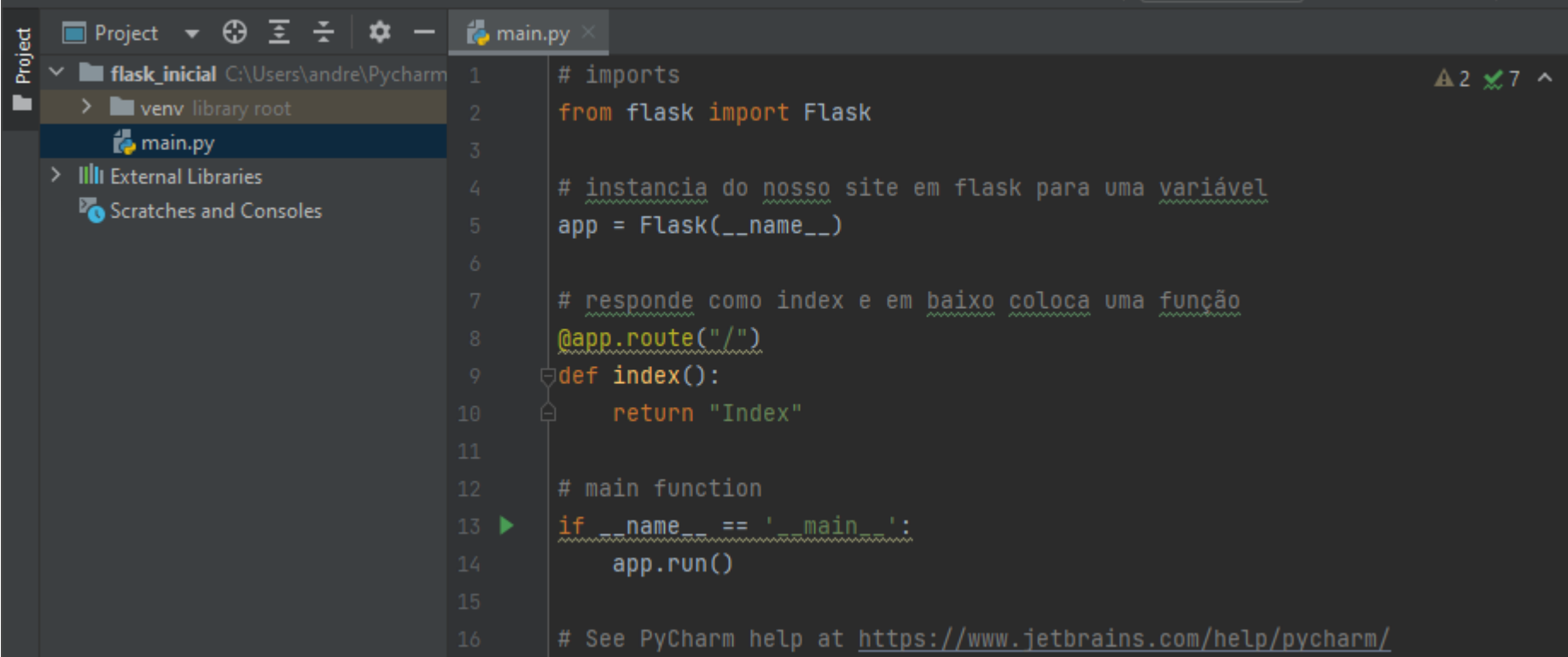
1. Abra PyCharm.
2. Criar um novo projeto.
3. Instalar o flask: `pip install Flask`
4. Importar: `from flask import Flask`
5. 3 - Salvar instância da variável: `app = Flask(__name__)`
6. 4 - rodar na main function: `app.run()`
7. 5 - criar roto do index: e criar função:

```
@app.route("/")
```

```
def index()
```

```
    return "<h1>Index</h1>"
```

Resultado



The screenshot shows the PyCharm IDE interface. On the left, the 'Project' tool window displays the file structure: a project named 'flask_inicial' located at 'C:\Users\andre\Pycharm', containing a 'venv' directory (labeled 'library root') and a 'main.py' file. The 'main.py' file is selected and open in the editor. The editor shows the following Python code:

```
1 # imports
2 from flask import Flask
3
4 # instancia do nosso site em flask para uma variável
5 app = Flask(__name__)
6
7 # responde como index e em baixo coloca uma função
8 @app.route("/")
9 def index():
10     return "Index"
11
12 # main function
13 if __name__ == '__main__':
14     app.run()
15
16 # See PyCharm help at https://www.jetbrains.com/help/pycharm/
```

Line numbers 1 through 16 are visible on the left side of the code editor. The code includes comments in Portuguese explaining each step: importing Flask, creating the app instance, defining the index route, and running the application.

Continuando...

1. Criar uma rota teste:

#criando Rota teste

```
@app.route("/teste")
def teste():
    return "Teste"
```

2. add debug e port

3. add rule:

```
def rule():
    return "<h1>Rule </h1>"
```

```
app.add_url_rule("/rule", "rule", rule) #(endereço, nome, função)
```

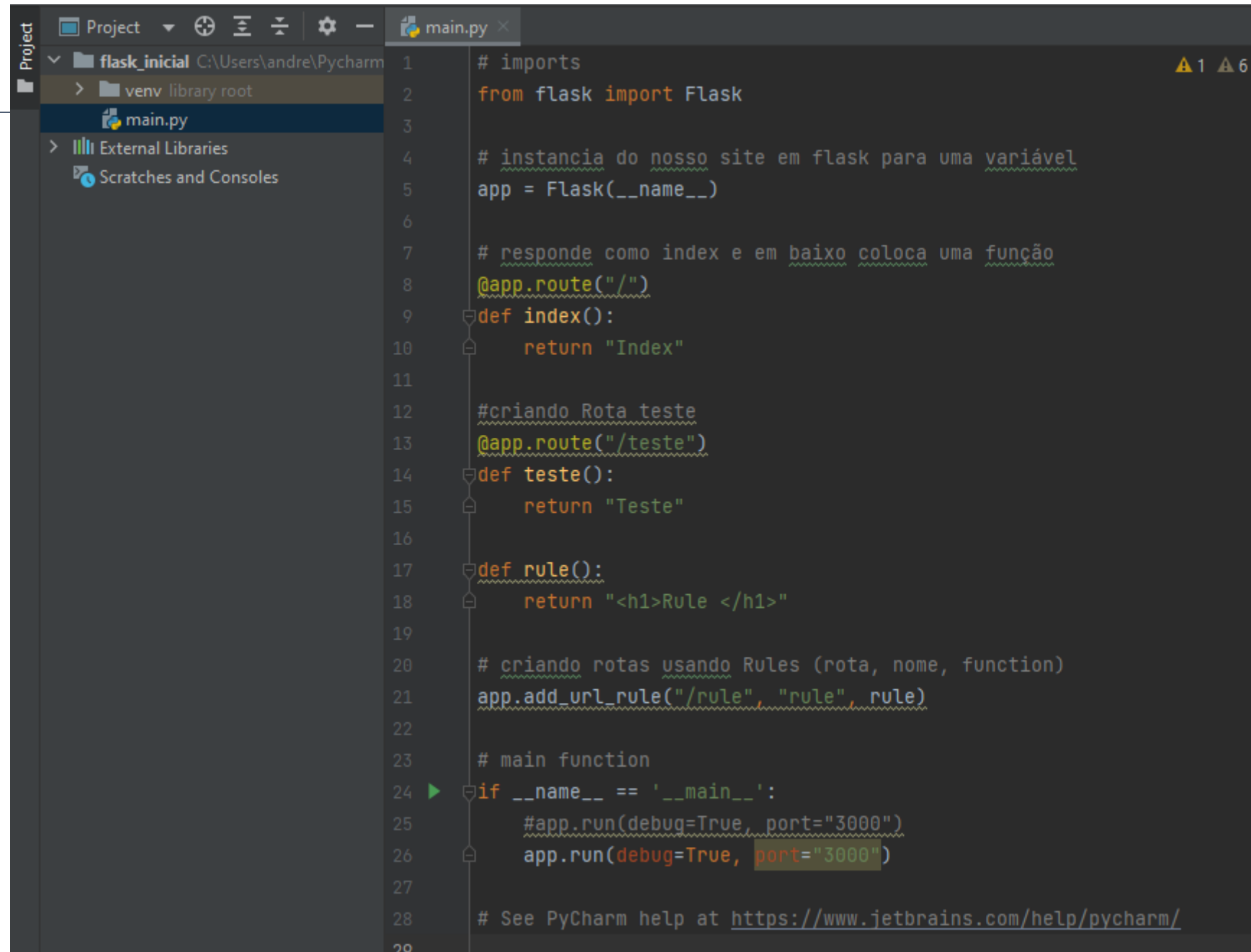
4. Altere a porta:

```
app.run(port="3000")
```

5. Coloque o debug como true e teste

```
app.run(debug=True, port="3000")
```

Resultado



The image shows a PyCharm IDE window with a project named 'flask_inicial'. The file explorer on the left shows the project structure: 'flask_inicial' (C:\Users\andre\Pycharm) containing a 'venv' folder (library root) and a 'main.py' file. The 'main.py' file is open in the editor, showing the following Python code:

```
1  # imports
2  from flask import Flask
3
4  # instancia do nosso site em flask para uma variável
5  app = Flask(__name__)
6
7  # responde como index e em baixo coloca uma função
8  @app.route("/")
9  def index():
10     return "Index"
11
12  #criando Rota teste
13  @app.route("/teste")
14  def teste():
15     return "Teste"
16
17  def rule():
18     return "<h1>Rule </h1>"
19
20  # criando rotas usando Rules (rota, nome, function)
21  app.add_url_rule("/rule", "rule", rule)
22
23  # main function
24  if __name__ == '__main__':
25     #app.run(debug=True, port="3000")
26     app.run(debug=True, port="3000")
27
28  # See PyCharm help at https://www.jetbrains.com/help/pycharm/
29
```

URL Dinâmica

- Podemos usar URLs dinâmicas para carregar diferentes conteúdos.

```
# criando URL dinâmica
```

```
@app.route("/hello")
```

```
@app.route("/hello/<nome>")
```

```
def hello(nome = ""):
```

```
    return "<h1>Hello {}</h1>".format(nome)
```

- Lembre-se
 - “/hello” responde apenas por “/hello”. Mas pode-se adicionar “/hello/” para responder por “/hello” e “/hello/”.
 - Precisamos garantir nesse exemplo que a variável nome sempre estará preenchida.

URL Dinâmica

- Podemos usar URLs dinâmicas para carregar diferentes conteúdos e receber variáveis.

```
# URL dinamica com int
```

```
@app.route("/user/")
```

```
@app.route("/user/<int:userID>")
```

```
def user(userID = 0):
```

```
    if userID > 0:
```

```
        return "<h1>User ID: {}</h1>".format(userID)
```

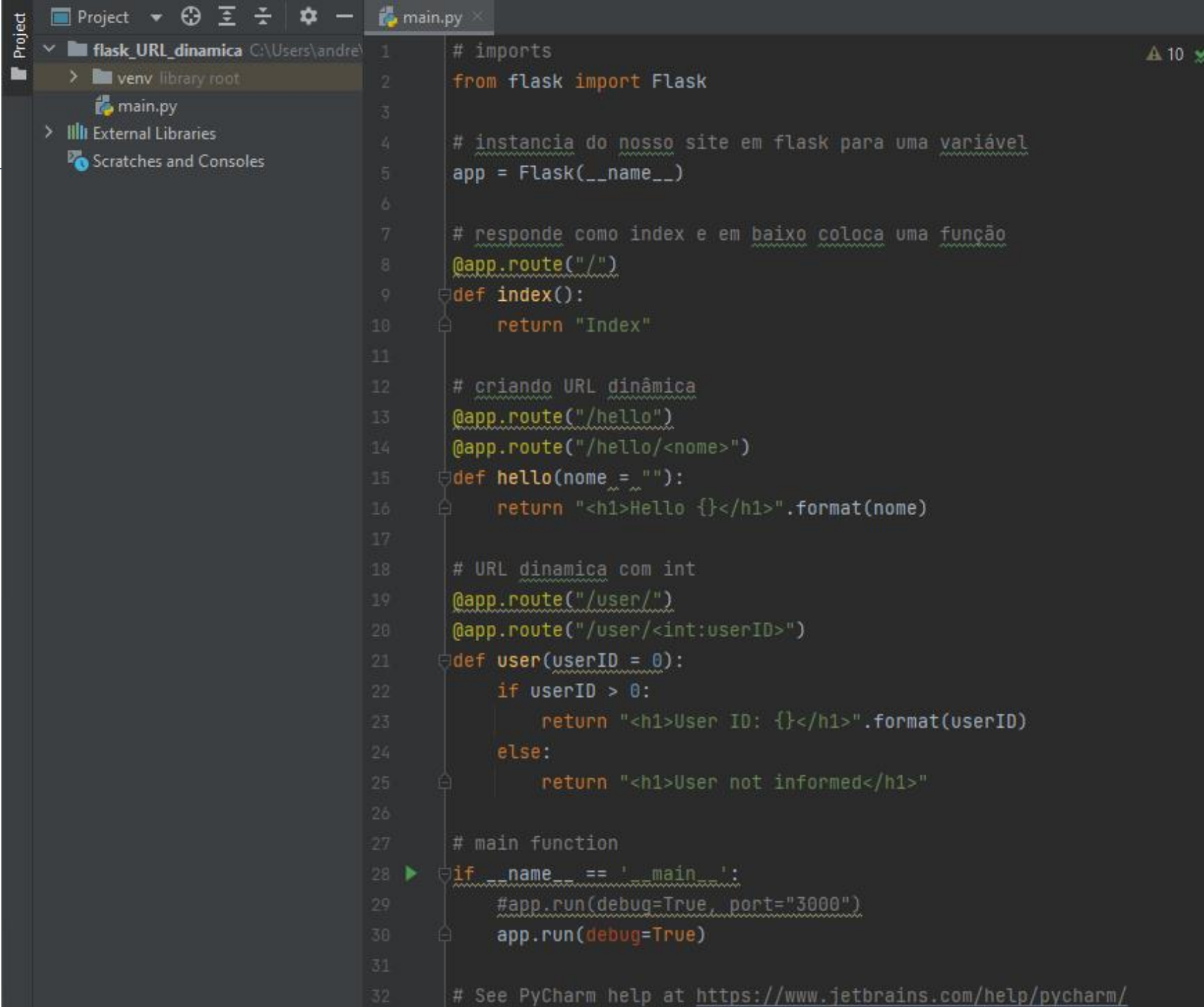
```
    else:
```

```
        return "<h1>User not informed</h1>"
```

- Lembre-se

- Precisamos garantir nesse exemplo que a variável userID sempre estará preenchida para depois realizar a verificação dela.

Resultado...



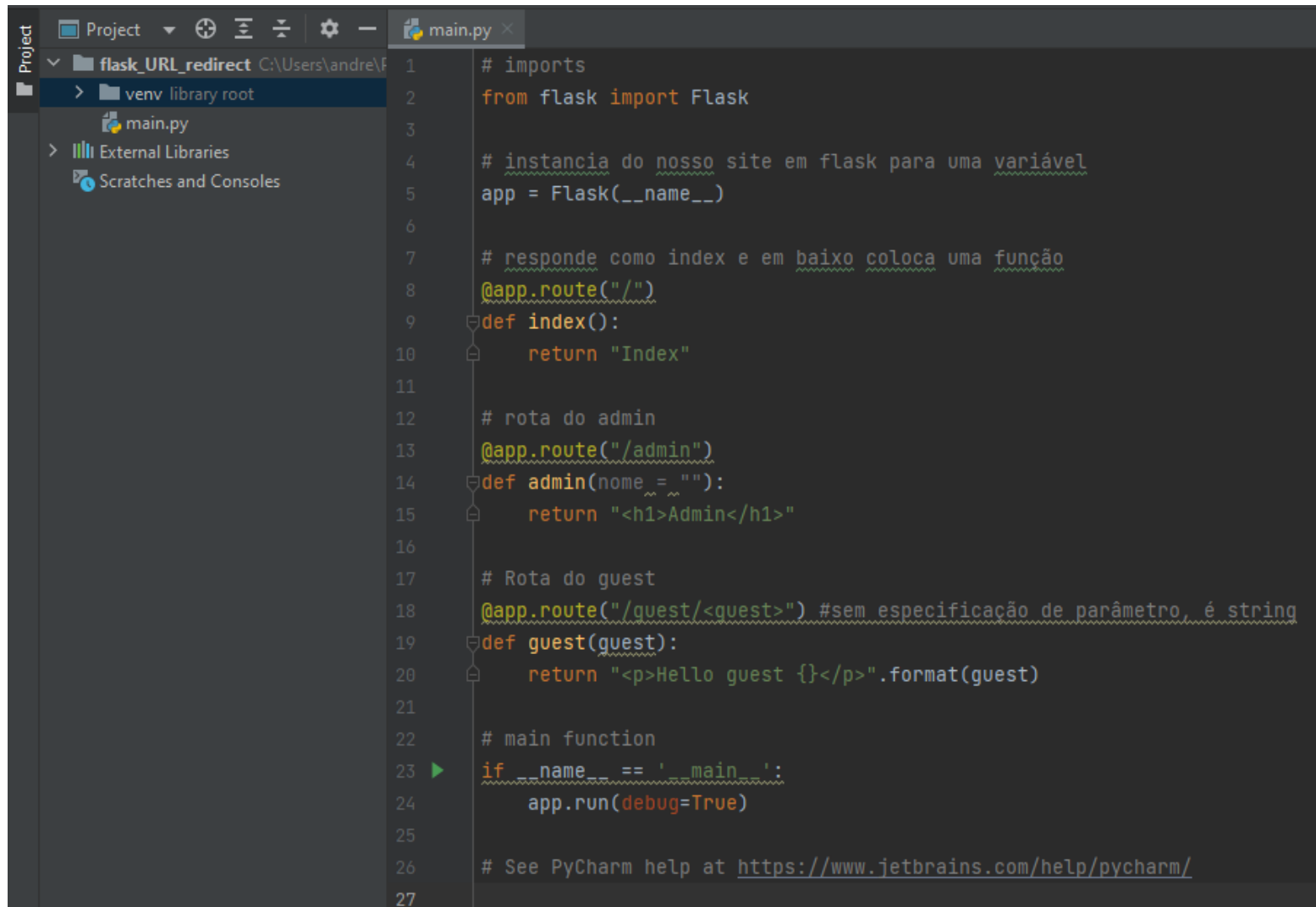
```
1 # imports
2 from flask import Flask
3
4 # instancia do nosso site em flask para uma variável
5 app = Flask(__name__)
6
7 # responde como index e em baixo coloca uma função
8 @app.route("/")
9 def index():
10     return "Index"
11
12 # criando URL dinâmica
13 @app.route("/hello")
14 @app.route("/hello/<nome>")
15 def hello(nome = ""):
16     return "<h1>Hello {}.format(nome)
17
18 # URL dinamica com int
19 @app.route("/user/")
20 @app.route("/user/<int:userID>")
21 def user(userID = 0):
22     if userID > 0:
23         return "<h1>User ID: {}.format(userID)
24     else:
25         return "<h1>User not informed</h1>"
26
27 # main function
28 if __name__ == '__main__':
29     #app.run(debug=True, port="3000")
30     app.run(debug=True)
31
32 # See PyCharm help at https://www.jetbrains.com/help/pycharm/
```

Redirecionamentos

- Podemos redirecionar para uma página diferente.

- Para esse exemplo, crie a seguinte estrutura:

- Lembre-se:
 - Sem não especificar o tipo de parâmetro, o Flask assume que é String.



The screenshot shows a PyCharm IDE with a project named 'flask_URL_redirect'. The project structure in the left sidebar includes a 'venv' folder (library root) and a 'main.py' file. The 'main.py' file contains the following Python code:

```
1 # imports
2 from flask import Flask
3
4 # instancia do nosso site em flask para uma variável
5 app = Flask(__name__)
6
7 # responde como index e em baixo coloca uma função
8 @app.route("/")
9 def index():
10     return "Index"
11
12 # rota do admin
13 @app.route("/admin")
14 def admin(nome = ""):
15     return "<h1>Admin</h1>"
16
17 # Rota do guest
18 @app.route("/quest/<quest>") #sem especificação de parâmetro, é string
19 def quest(quest):
20     return "<p>Hello guest {}</p>".format(quest)
21
22 # main function
23 if __name__ == '__main__':
24     app.run(debug=True)
25
26 # See PyCharm help at https://www.jetbrains.com/help/pycharm/
27
```

Redirecionando...

- Redirecionamentos normal.
- Redirecionamento com parâmetro.
- Redirecionamento para URL.
- Como melhorar o código?

```
Project
  flask_URL_redirect C:\Users\andre\F
    venv library root
    main.py
  External Libraries
  Scratches and Consoles

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__)
6
7 # responde como index e em baixo coloca uma função
8 @app.route("/")
9 def index():
10     return "Index"
11
12 # rota do admin
13 @app.route("/admin")
14 def admin(nome = ""):
15     return "<h1>Admin</h1>"
16
17 # Rota do guest
18 @app.route("/quest/<quest>") #sem especificação de parâmetro, é string
19 def quest(quest):
20     aux = "<p>Hello guest {}</p>".format(quest)
21     aux += "</br>"
22     aux += "<p>Hello guest <b>%s</b></p>" % quest
23     return aux
24
25 @app.route("/user/<name>")
26 def user(name):
27     if name == "admin":
28         return redirect(url_for('admin'))
29     else:
30         return redirect(url_for('quest', guest=name))
31
32 @app.route("/google")
33 def google():
34     return redirect("http://google.com")
35
36 # main function
37 if __name__ == '__main__':
38     app.run(debug=True)
```

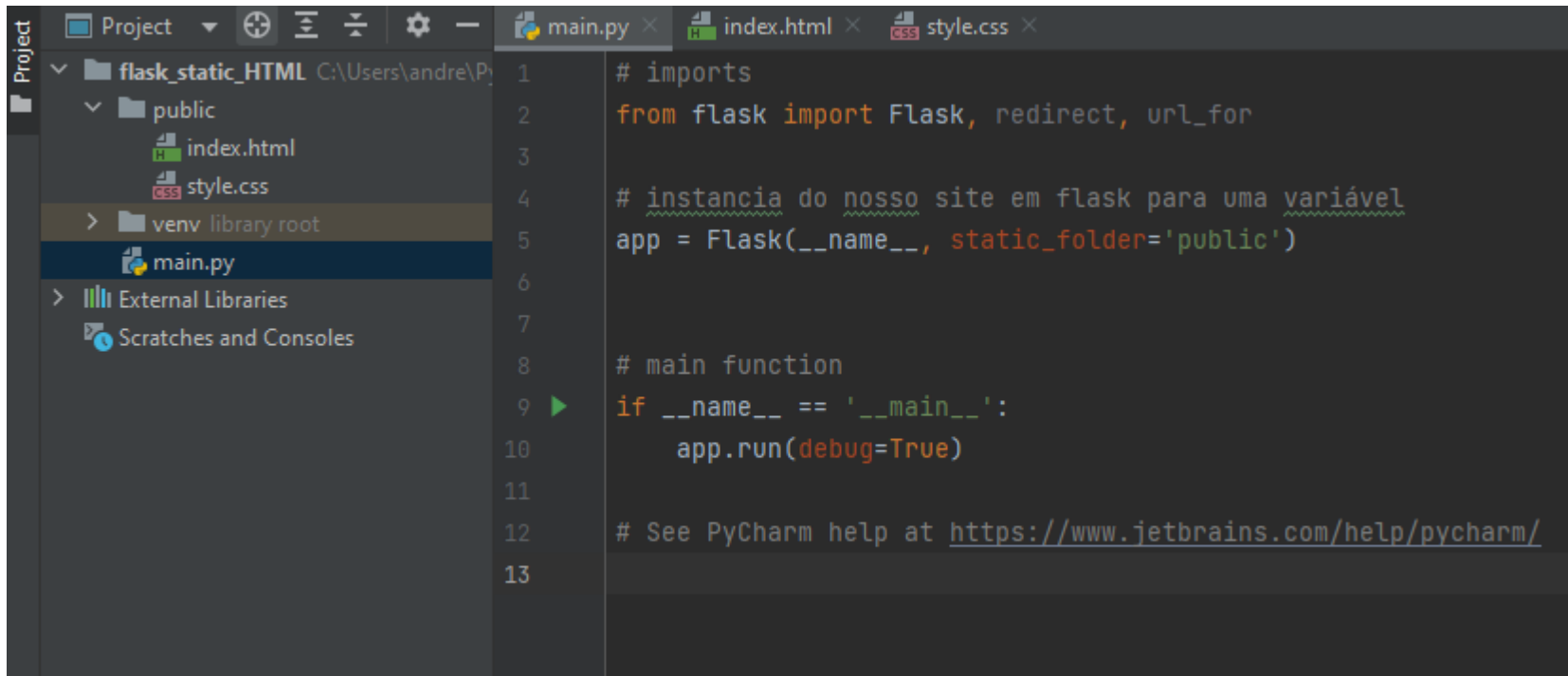
Momento para conversar sobre Python

- O que acharam da identificação?
- Finalização de linha?
- Erros comuns?
- Diferenças?
- Passagem de parâmetros?
- Objetivo na programação?

- Qual a sua opinião?

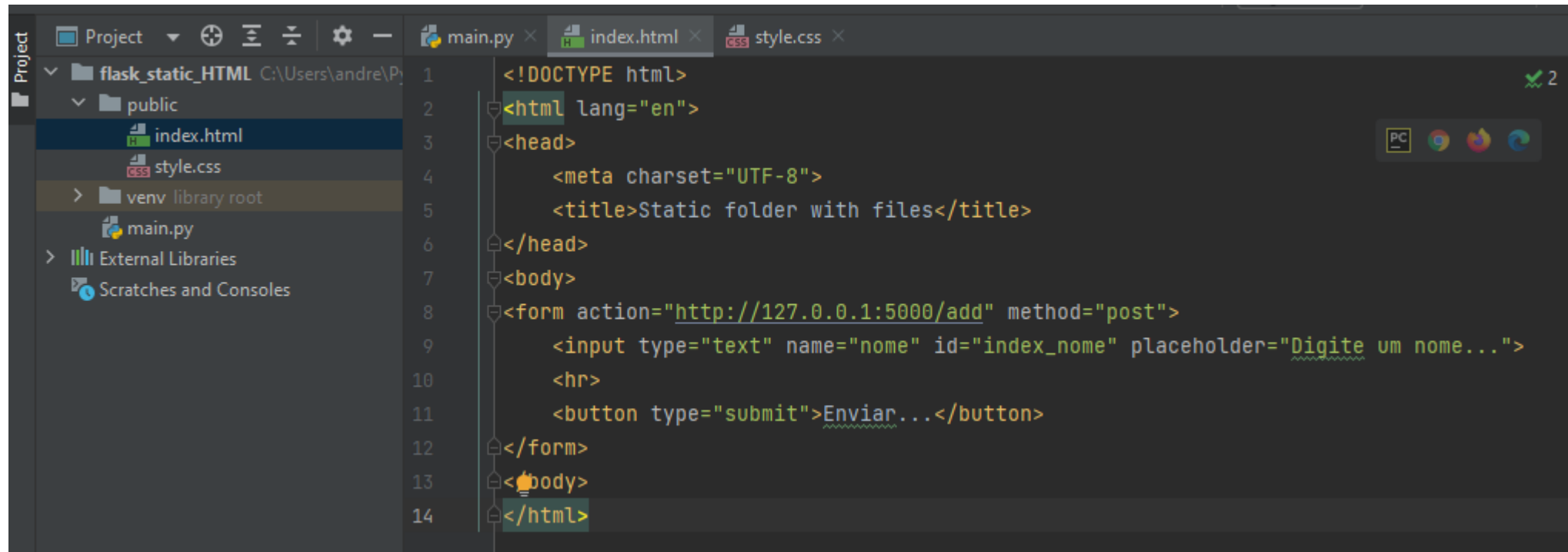
Exemplos de arquivos estáticos

- Primeiro vamos configurar uma pasta local ou “static_folder” para a variável app.
- Crie uma pasta (nome “public”) e crie um arquivo (nome “index.html”).



```
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 # main function
9 if __name__ == '__main__':
10     app.run(debug=True)
11
12 # See PyCharm help at https://www.jetbrains.com/help/pycharm/
13
```

Conteúdo do arquivo HTML



The screenshot shows a code editor with a project named 'flask_static_HTML'. The file explorer on the left shows the project structure: 'public' folder containing 'index.html' and 'style.css', and a 'venv' folder. The main editor displays the content of 'index.html' with line numbers 1 through 14. The code is as follows:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Static folder with files</title>
6 </head>
7 <body>
8   <form action="http://127.0.0.1:5000/add" method="post">
9     <input type="text" name="nome" id="index_nome" placeholder="Digite um nome...">
10    <hr>
11    <button type="submit">Enviar...</button>
12  </form>
13 </body>
14 </html>
```

- Execute o programa ou pressione F5 se estiver em debug.

Acesse...

- <http://127.0.0.1:5000/public/index.html>

Modificando...

- Adicione um arquivo com o nome de “style.css”, com o seguinte conteúdo.

```
h1 {  
    text-align: center;  
    color: purple;  
}
```

- Adicione as seguintes linhas no “index.html”.
 - Form:
 - `<h1>Exemplo de arquivos estáticos</h1>`
 - Head (precisa informar que está usando um arquivo css):
 - `<link rel="stylesheet" href="style.css"/>`

Resultado esperado

