HTML Templates

In this flask tutorial I will show you how to render and create HTML templates. I will also discuss how to dynamically create HTML with the use of python code in your html files.

Redirecting Continued

Starting from where we left off in the last tutorial. I wanted to show how to redirect to a function that takes an argument (like our user function). To do this we simply need to define the parameter name and a value in the url_for function, like below.

tutorial1.py

```
from flask import Flask, redirect, url_for

app = Flask(__name__)

@app.route("/")
def home():
    return "Hello! this is the main page <h1>HELLO</h1>"


@app.route("/<name>")
def user(name):
    return f"Hello {name}!"

@app.route("/admin")
def admin():
    return redirect(url_for("user", name="Admin!"))
    # Now we when we go to /admin we will redirect to user
with the argument "Admin!"


if __name__ == "__main__":
    app.run()
```

Rendering HTML

Now as beautiful as our website is we probably want to render proper HTML files. To do this we need to follow a few steps.

Step 1: Create new python file called *tutorial2.py*

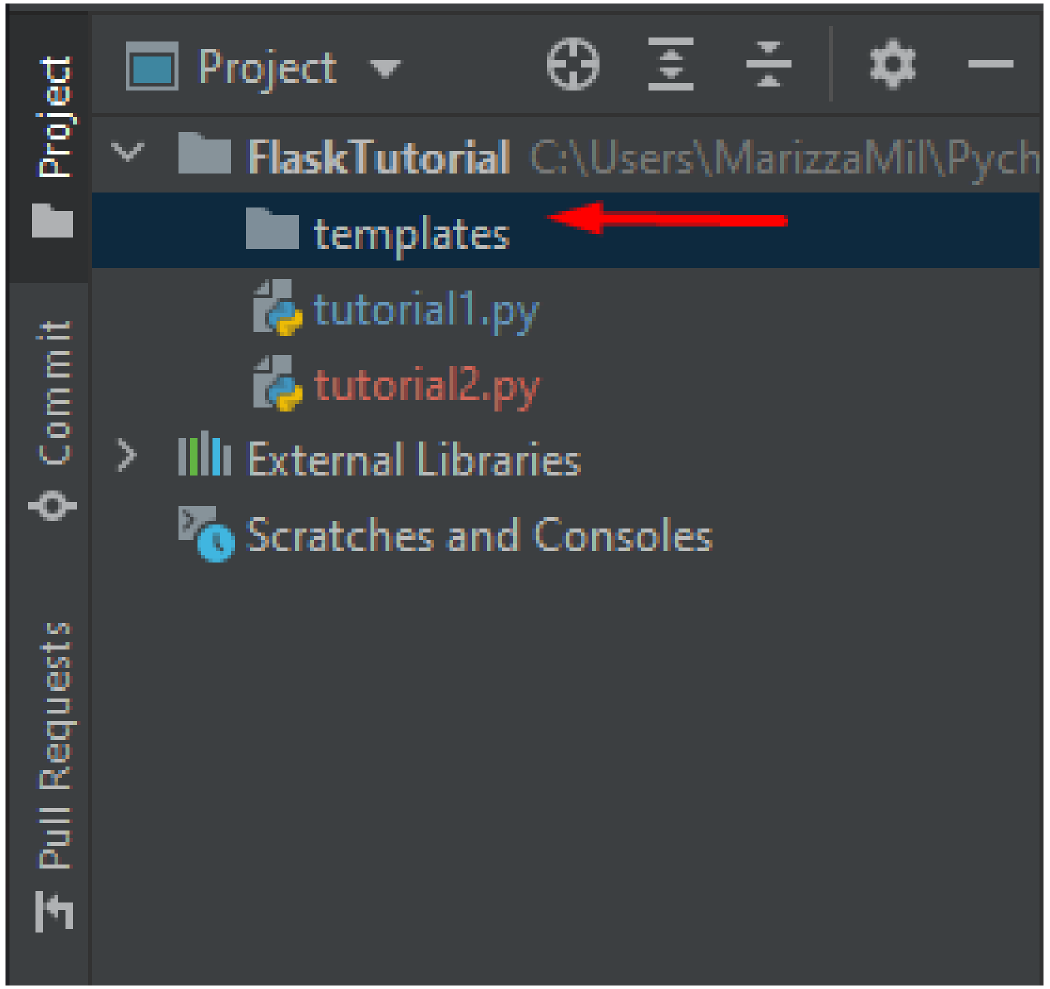
Import the render_template function from flask

tutorial2.py

```
from flask import Flask, render_template


app = Flask(__name__)



@app.route("/")
def home():
    return "Hello! this is the main page <h1>HELLO</h1>"



if __name__ == "__main__":
    app.run()
```

Rendering HTML

Step 2: Create a new folder called *templates* inside the SAME directory as our python script.



Rendering HTML

Step 3: Create an html file, I've named mine *index.html*. Make sure to **put it in the templates folder!**

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <title>Home page</title>
</head>
<body>
   <h1>Home Page!</h1>
</body>
</html>
```

Rendering HTML

Step 4: Render the template from a function in python.

The render_template function will look in the "templates" folder for a file called "index.html" and render it to the screen. Now try running the script and visiting "/". You should see that html rendered.

tutorial2.py

```
from flask import Flask, render_template


app = Flask(__name__)


@app.route("/")
def home():
    return render_template("index.html")


if __name__ == "__main__":
    app.run()
```

Dynamic HTML

Flask uses a templating engine called jinja. This allows you to write python code inside your html files. It also allows you to pass information from your back-end (the python script) to your HTML files.
In your HTML file you can use the following syntax to evaluate python statements.
{{Variable/Statement}} Placing a variable or statement inside of {{}} will tell flask to evaluate the statement inside the brackets and render the text equivalent to it.

Dynamic HTML

Let's look at an example.

Here we are defining that we will have a variable passed to this HTML file called content. So from our back-end, when we render the template we need to pass it a value for content.

index.html

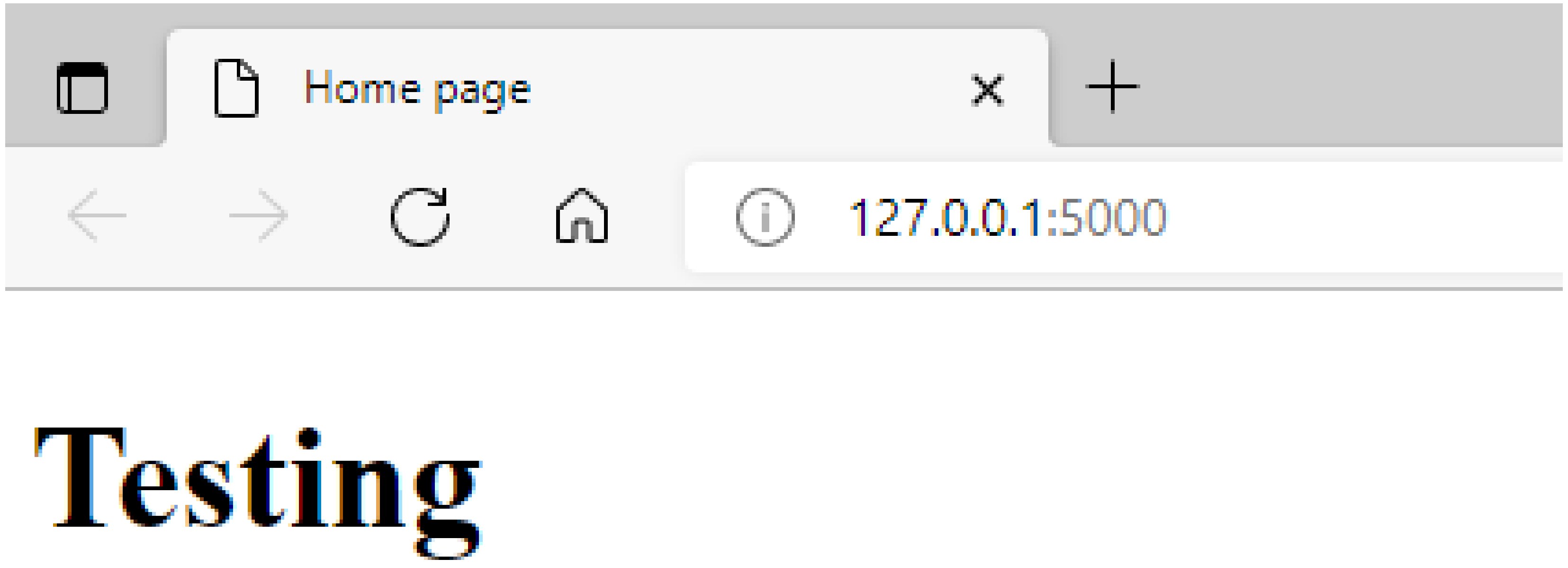
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Home page</title>
</head>

<body>
    <h1>{{content}}</h1>
</body>

</html>
```

tutorial2.py

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route("/")
def home():
    return render_template("index.html", content="Testing")

if __name__ == "__main__":
    app.run()
```

Dynamic HTML

When we run the script and navigate to the home page we get the following.



Testing

Templates Continued

There are a few other things you can write in your HTML relating to python code. The most popular is to use for loops and if statements.
You can place python expressions inside **{% %}**.

Templates Continued

Here's a quick example of the syntax for if statements

index.html

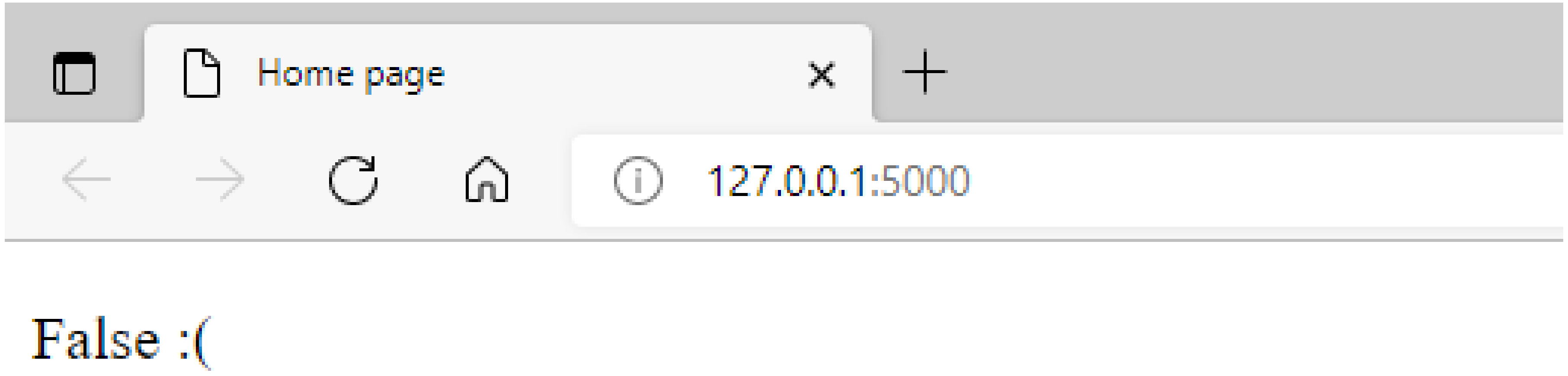
```html
<!doctype html>
<html>
<head>
    <title>Home page</title>
    </head>
    <body>
        {% if content == "true" %}
            <p>True!</p>
        {% else %}
            <p>False :(</p>
        {% endif %}
    </body>
</html>
```

tutorial2.py

```python
from flask import Flask, render_template

app = Flask(__name__)

@app.route("/")
def home():
    return render_template("index.html", content="Testing")

if __name__ == "__main__":
    app.run()
```

Templates Continued

When we run the script and navigate to the home page we get the following.



```
False :(
```

Templates Continued

Here's a quick example of the syntax for loops.

index.html

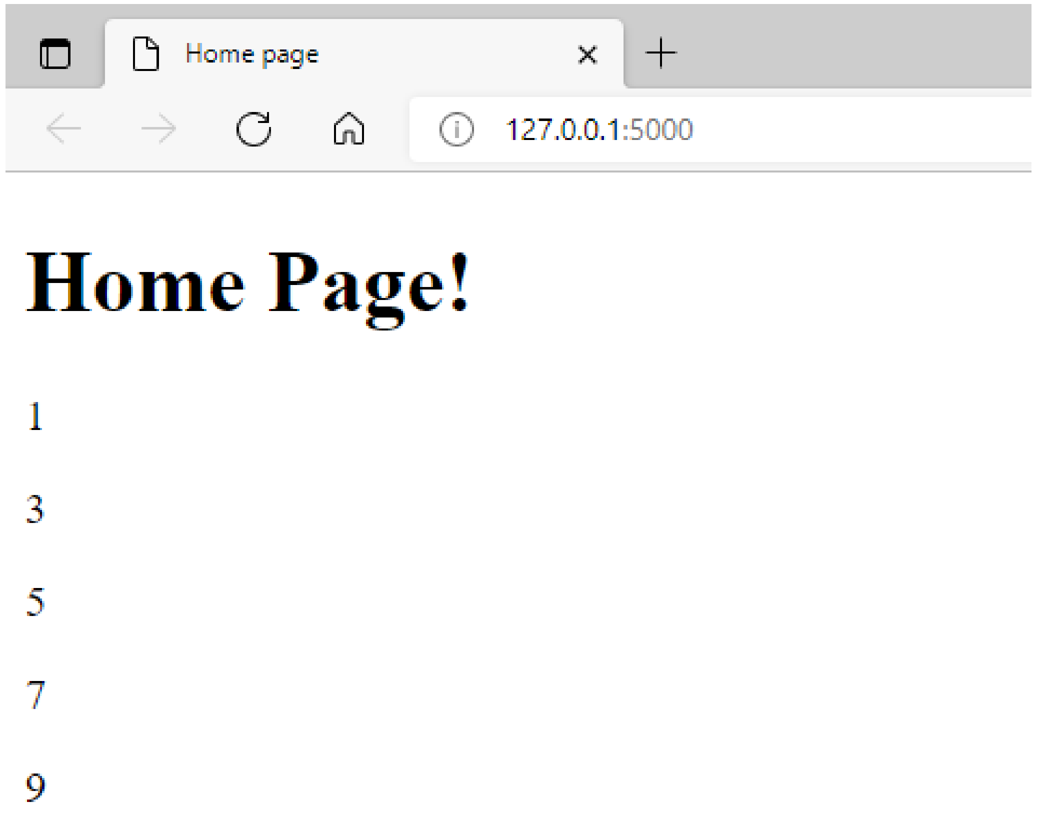
```
<!doctype html>
<html>
<head>
    <title>Home page</title>
    </head>
    <body>
      {% for x in range(10) %}
          {% if x % 2 ==1 %}
             <p>{{x}}</p>
          {% endif %}
       {% endfor %}
    </body>
</html>
```

tutorial2.py

```
from flask import Flask, render_template


app = Flask(__name__)


@app.route("/")
def home():
    return render_template("index.html", content="Testing")


if __name__ == "__main__":
    app.run()
```

Templates Continued

When we run the script and navigate to the home page we get the following.



```
Home Page!

1

3

5

7

9
```

Templates Continued

Here's a quick example of the syntax for loops.

index.html

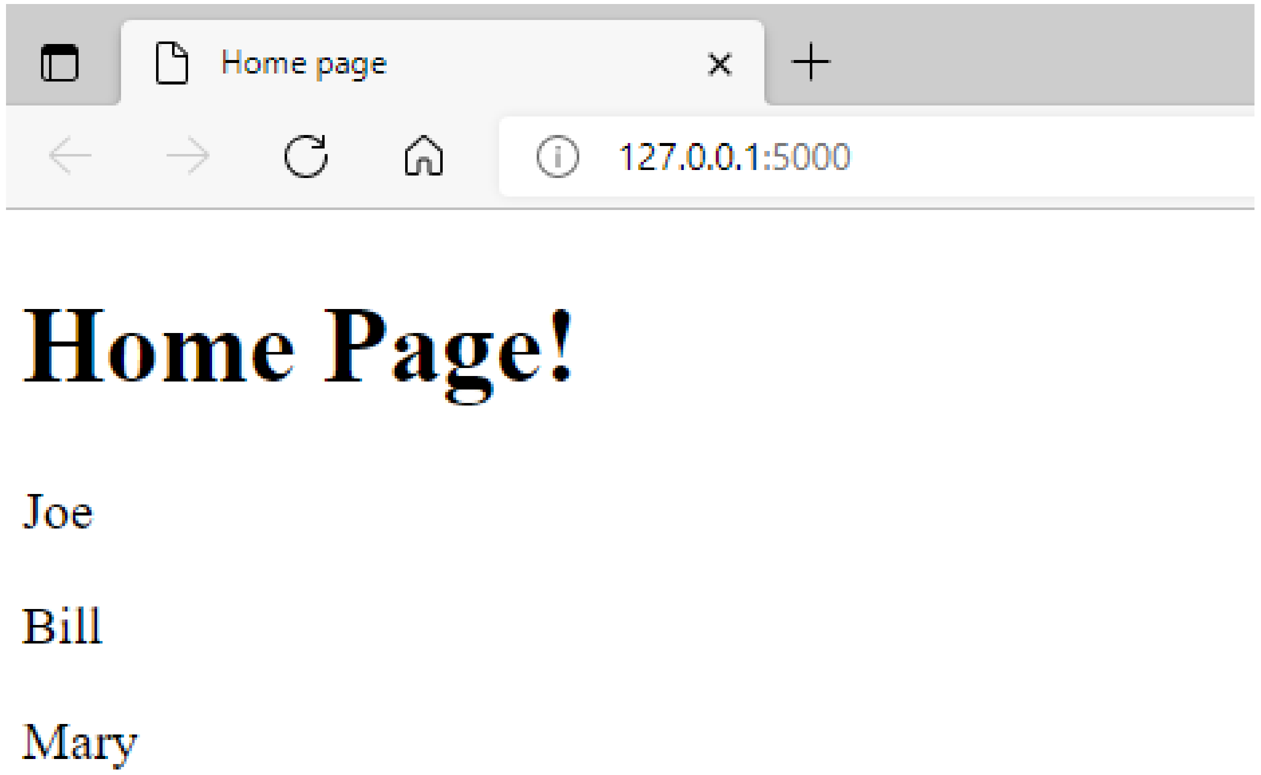
```
<!doctype html>
<html>
<head>
    <title>Home page</title>
    </head>
    <body>
        <h1>Home Page!</h1>
        {% for x in content %}
           <p>{{x}}</p>
        {% endfor %}
    </body>
</html>
```

tutorial2.py

```
from flask import Flask, render_template


app = Flask(__name__)


@app.route("/")
def home():
  return render_template("index.html",
content=['Joe', 'Bill', 'Mary'])

if __name__ == "__main__":
  app.run()
```

Templates Continued

When we run the script and navigate to the home page we get the following.



Home Page!

Joe

Bill

Mary

Passing Multiple Values

Just a quick note here to let you know that you can pass multiple values to your HTML files by defining more keyword arguments in your render_template function or by passing in things like dicts or lists.

Passing Multiple Values

Just a quick note here to let you know that you can pass multiple values to your HTML files by defining more keyword arguments in your render_template function or by passing in things like dicts or lists.

```
@app.route("/")
def home():
    return render_template("index.html", content="Testing", x=4)
```

```
@app.route("/")
def home():
    return render_template("index.html", content={"a":2, "b":"hello"})
```