Mastering Python

الدرس # 14

Flask

web development,
one drop at a time

بناء المواقع الالكترونية

By:
Hussam Hourani

V1.0 -DEC 2019

Agenda

- Flask Overview
- "Hello, world!“ Example
- Multi Routes
- HTML Tags & Flask
- Variable Rules
- Blueprints & Files Structure
- Rendering Templates
- More HTML styles
- URL Building
- Using CSS
- Using JavaScript

Flask Overview

Flask is a web application framework written in Python. Flask is also easy to get started with as a beginner. Flask is based on the Werkzeug WSGI (Web Server Gateway Interface) toolkit and Jinja2 template engine

Why Flask?

- easy to use.
- built in development server and debugger
- integrated unit testing support
- RESTful request dispatching
- uses Jinja2 templating
- support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode based
- extensively documented

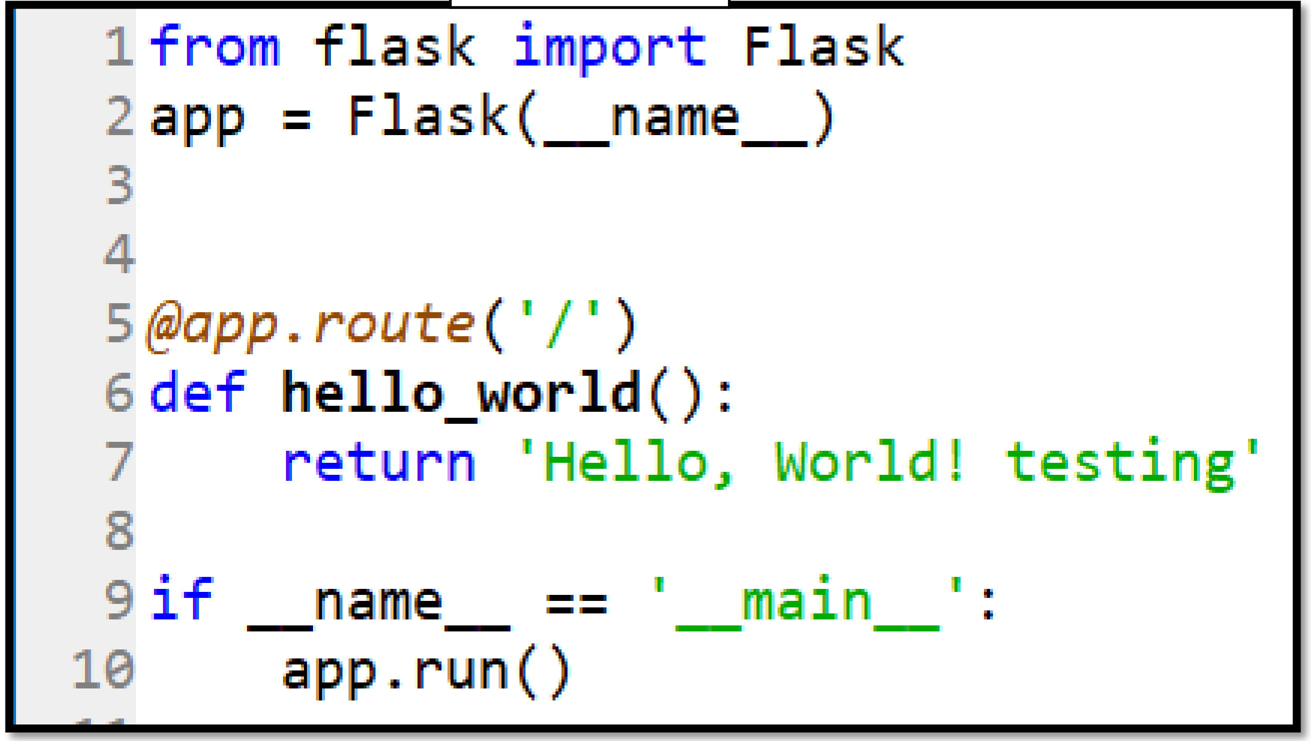
```
pip install flask
```

https://pypi.org/project/googletrans/

https://pythonspot.com/flask-web-app-with-python/

"Hello, world!“ Example

flask0.py



```python
from flask import Flask
app = Flask(__name__)


@app.route('/')
def hello_world():
    return 'Hello, World! testing'

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

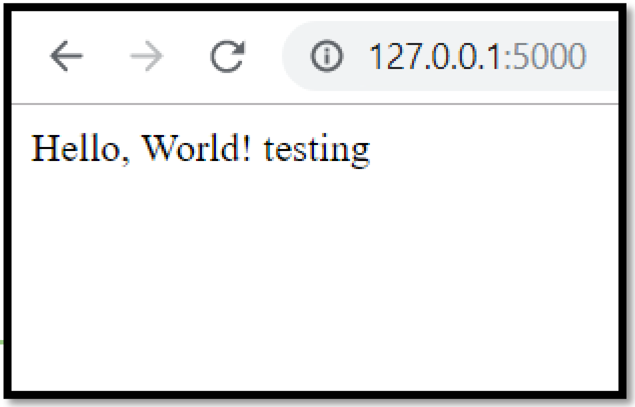
The **route()** function of the Flask class is a decorator, which tells the application which URL should call the associated function.

Here, URL **‘/’** rule is bound to the **hello_world()**

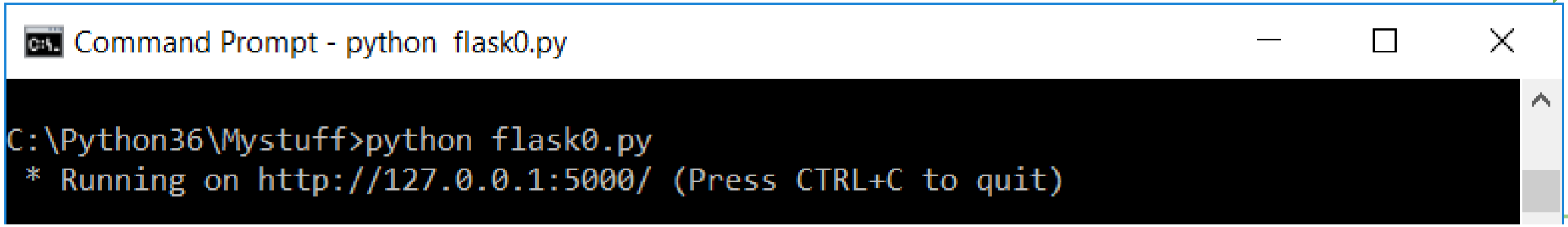
http://127.0.0.1:5000/

http://localhost:5000/

Output



Hello, World! testing



```
C:\Python36\Mystuff>python flask0.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

By : Hussam Hourani

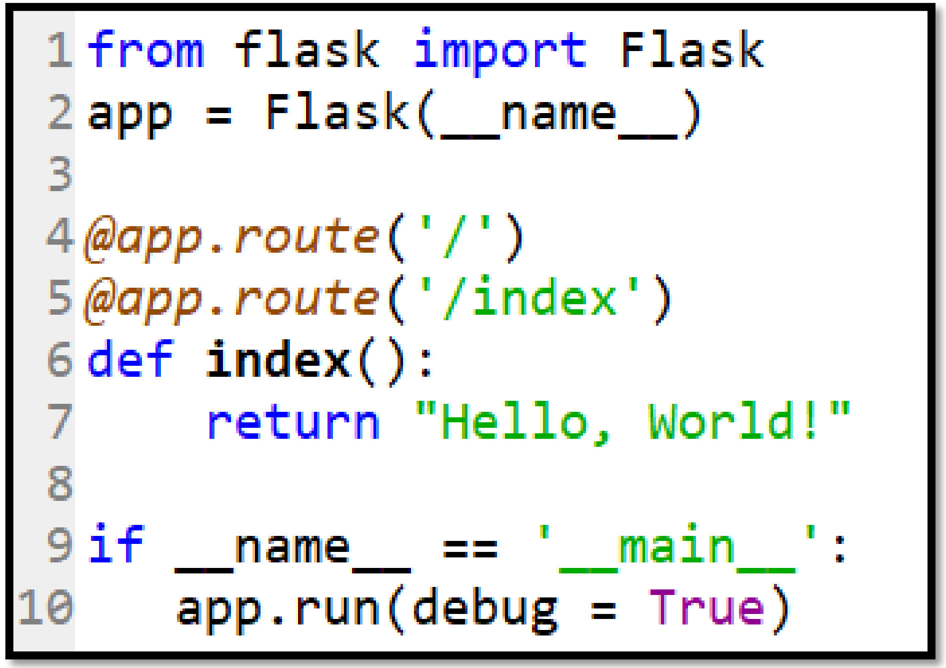
https://www.fullstackpython.com/flask.html

App.run

app.run(host, port, debug, options)

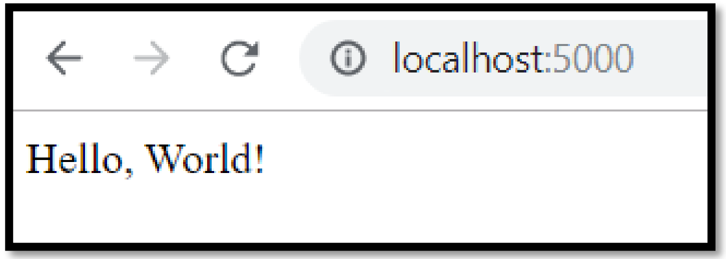
Sr.No.	Parameters & Description
1	**host** Hostname to listen on. Defaults to 127.0.0.1 (localhost). Set to ‘0.0.0.0’ to have server available externally
2	**port** Defaults to 5000
3	**debug** Defaults to false. If set to true, provides a debug information
4	**options** To be forwarded to underlying server.

"Hello, world!“ Example

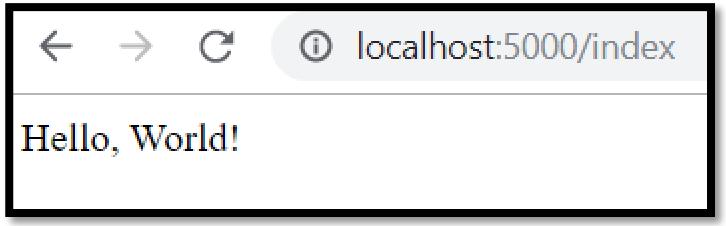


```python
from flask import Flask
app = Flask(__name__)

@app.route('/')
@app.route('/index')
def index():
    return "Hello, World!"

if __name__ == '__main__':
    app.run(debug = True)
```

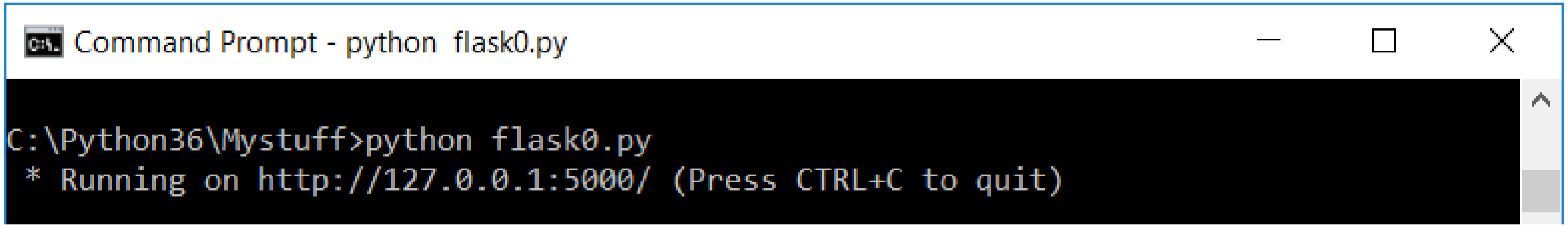
Output



Hello, World!



Hello, World!

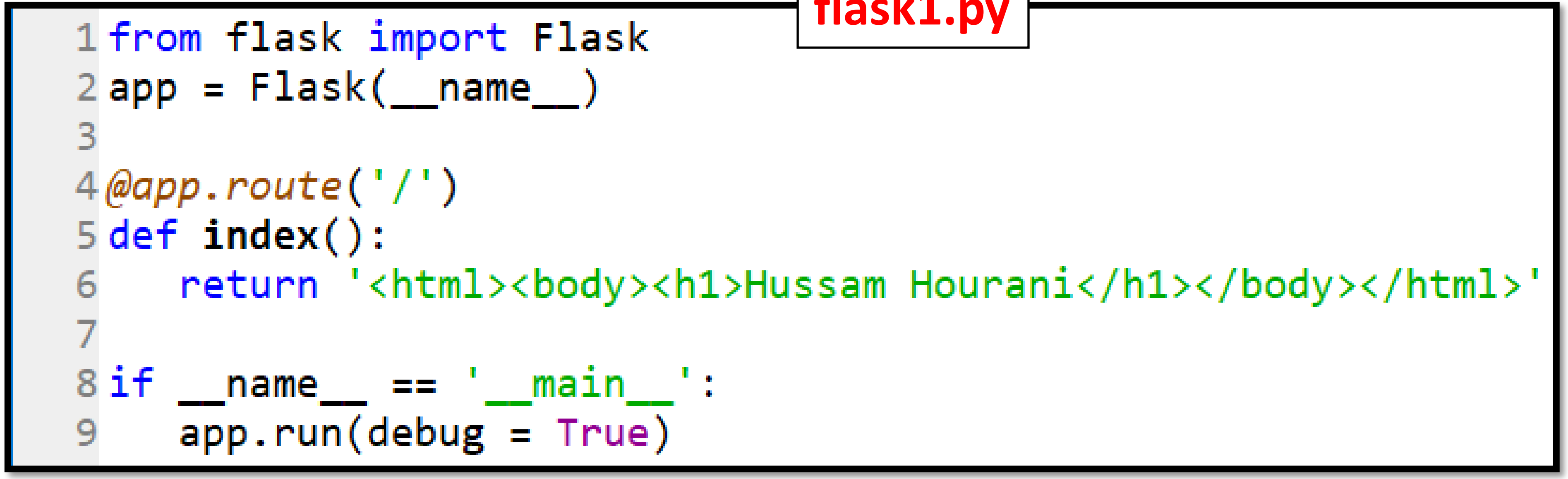


```
C:\Python36\Mystuff>python flask0.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

https://www.fullstackpython.com/flask.html

HTML Tags & Flask

flask1.py

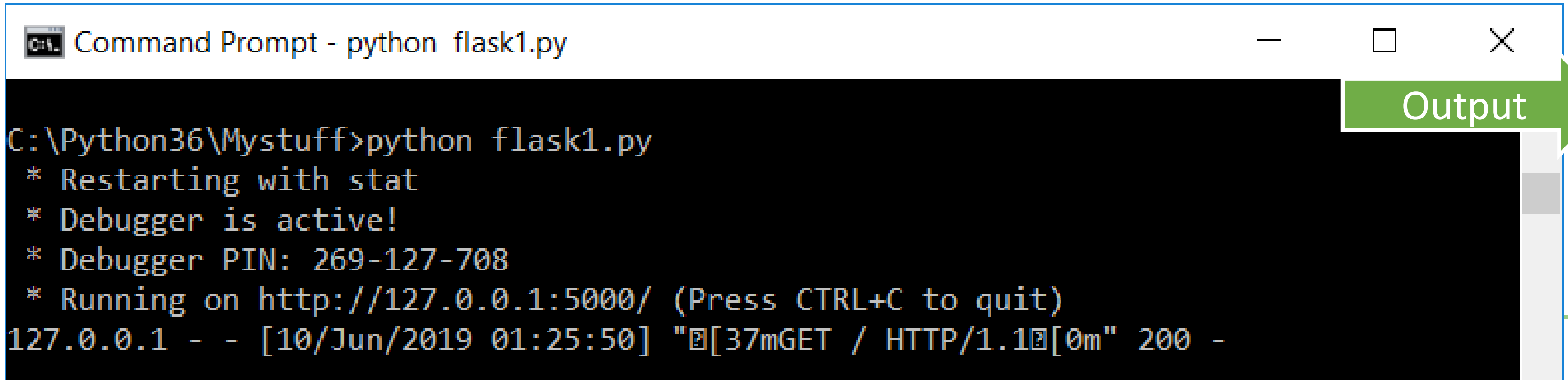


```python
from flask import Flask
app = Flask(__name__)

@app.route('/')
def index():
    return '<html><body><h1>Hussam Hourani</h1></body></html>'

if __name__ == '__main__':
    app.run(debug = True)
```

enable **debug support**, The server will then reload itself if the code changes. It will also provide a useful debugger to track the errors if any, in the application

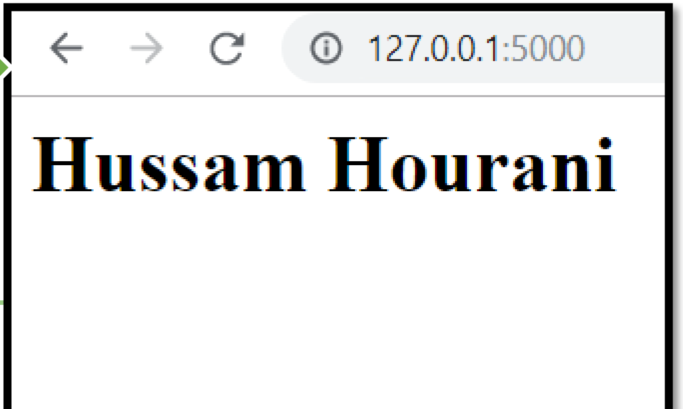
http://127.0.0.1:5000/

http://localhost:5000/



```
C:\Python36\Mystuff>python flask1.py
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 269-127-708
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [10/Jun/2019 01:25:50] "⍰[37mGET / HTTP/1.1⍰[0m" 200 -
```

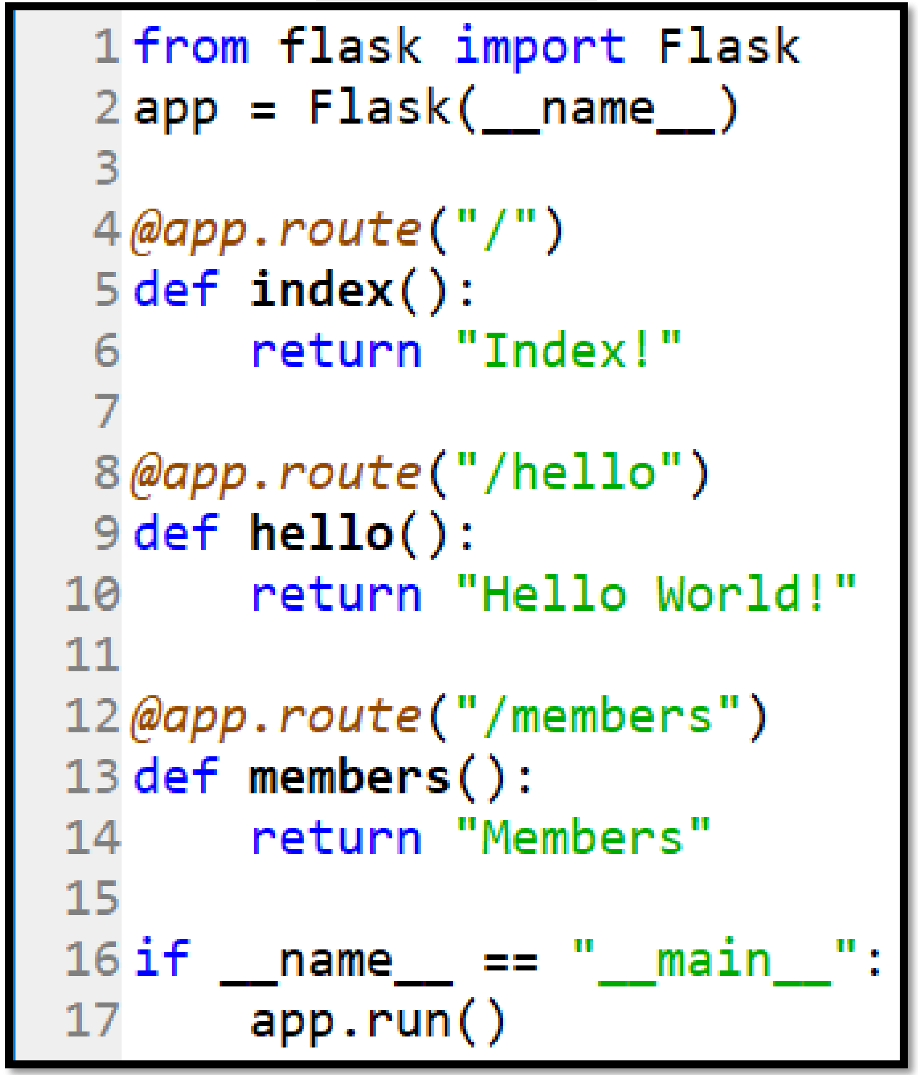
Output



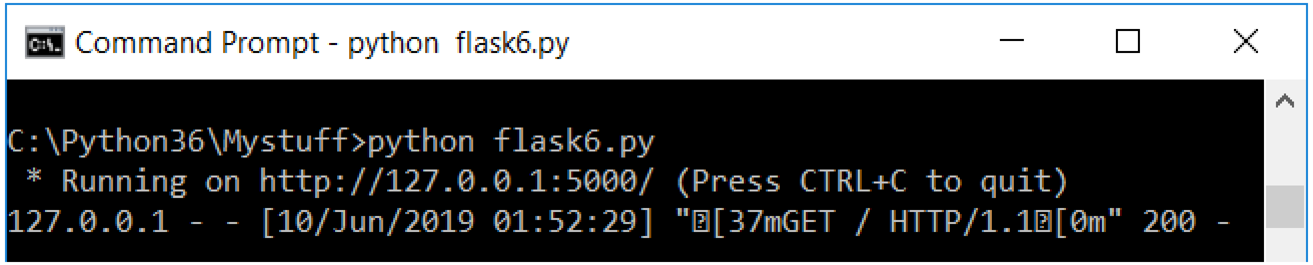
Hussam Hourani

Multi Routes

flask6.py

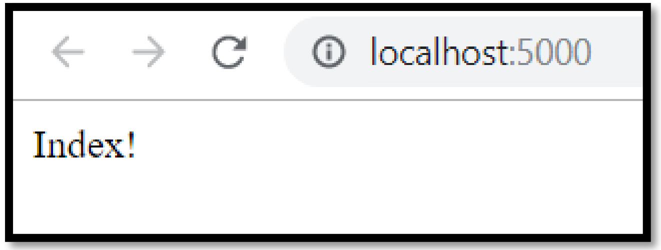


```python
from flask import Flask
app = Flask(__name__)

@app.route("/")
def index():
    return "Index!"

@app.route("/hello")
def hello():
    return "Hello World!"

@app.route("/members")
def members():
    return "Members"

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



```
C:\Python36\Mystuff>python flask6.py
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [10/Jun/2019 01:52:29] "�[37mGET / HTTP/1.1�[0m" 200 -
```

Output

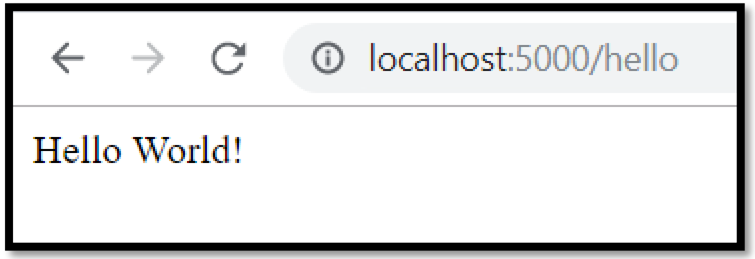


Index!

http://127.0.0.1:5000/

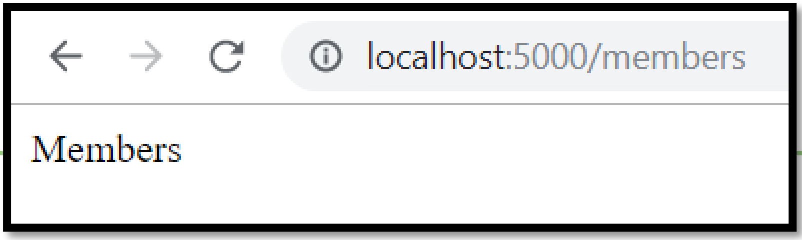
http://localhost:5000/

Output



Hello World!

Output

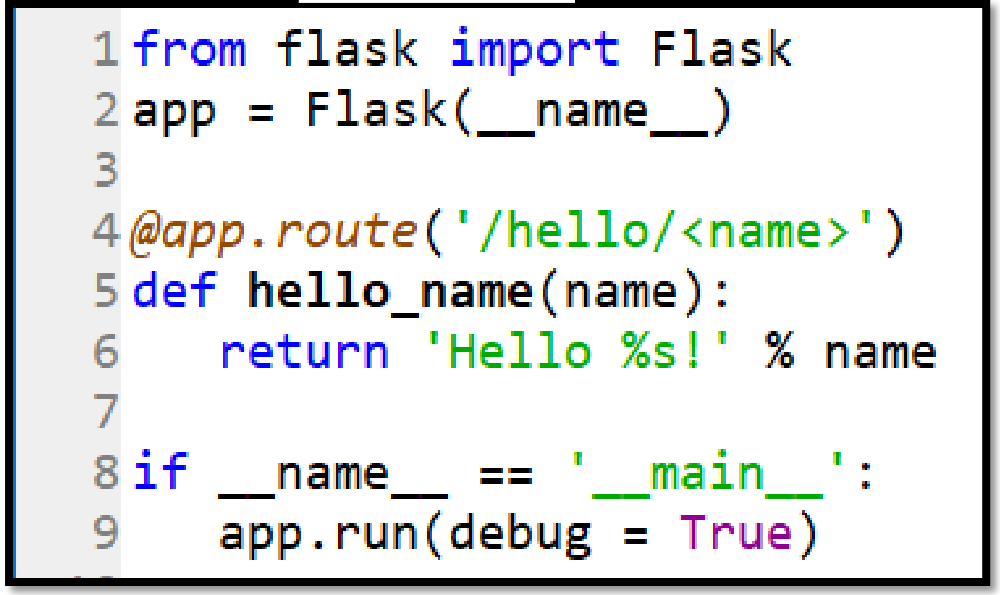


Members

https://pythonspot.com/flask-web-app-with-python/

Variable Rules

flask9.py

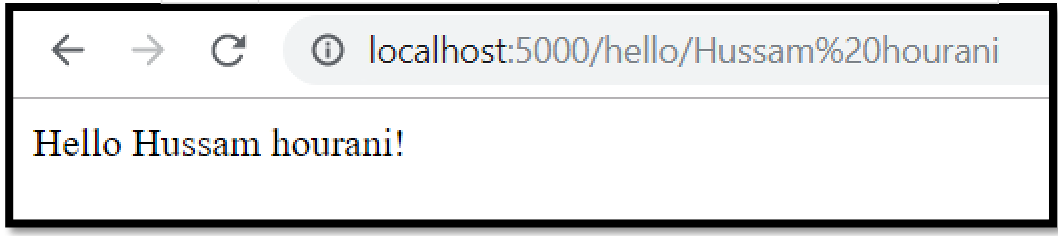


```python
from flask import Flask
app = Flask(__name__)

@app.route('/hello/<name>')
def hello_name(name):
   return 'Hello %s!' % name

if __name__ == '__main__':
   app.run(debug = True)
```

default string variable

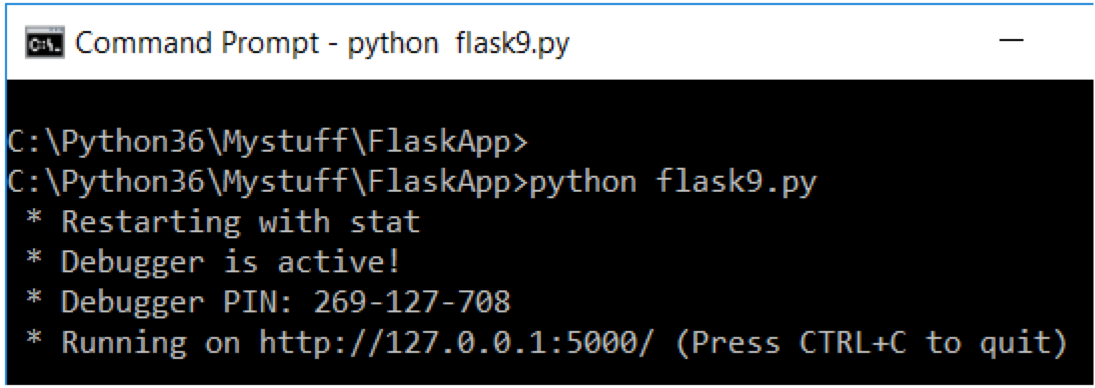
1	**Int** accepts integer
2	**float** For floating point value
3	**path** accepts slashes used as directory separator character

Output



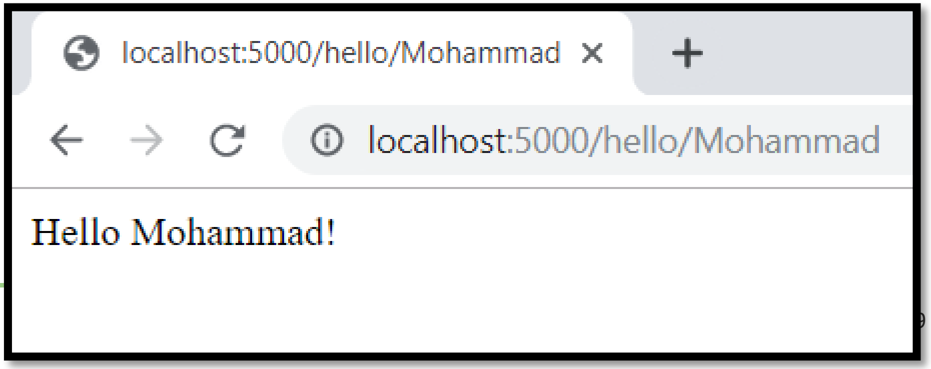
localhost:5000/hello/Hussam%20hourani

Hello Hussam hourani!



```
C:\Python36\Mystuff\FlaskApp>
C:\Python36\Mystuff\FlaskApp>python flask9.py
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 269-127-708
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Output



localhost:5000/hello/Mohammad

Hello Mohammad!

https://www.tutorialspoint.com/flask/flask_quick_guide.htm

Blueprints & Files Structure

Functional Structure

```
yourapp/
    __init__.py
    static/
    templates/
        home/
        control_panel/
        admin/
    views/
        __init__.py
        home.py
        control_panel.py
        admin.py
    models.py
```

organize the pieces of your app by what they do

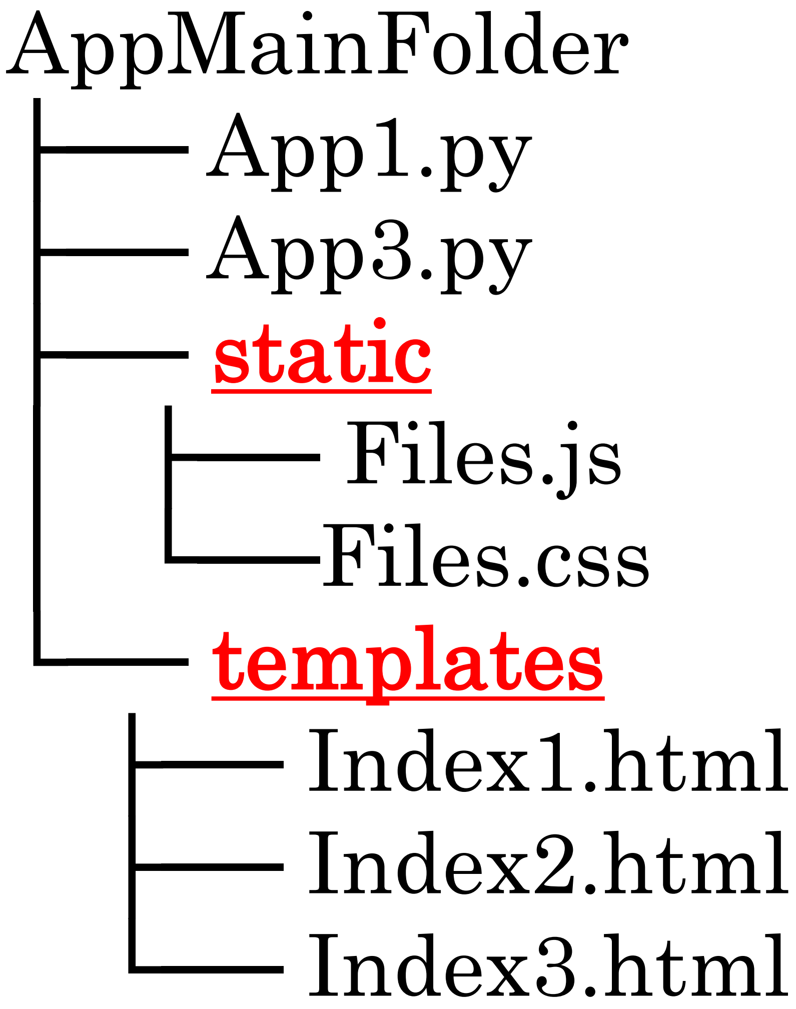
Division Structure

```
yourapp/
    __init__.py
    admin/
        __init__.py
        views.py
        static/
        templates/
    home/
        __init__.py
        views.py
        static/
        templates/
    control_panel/
        __init__.py
        views.py
        static/
        templates/
    models.py
```

organize the pieces of the application based on which part of the app they contribute to

http://exploreflask.com/en/latest/blueprints.html

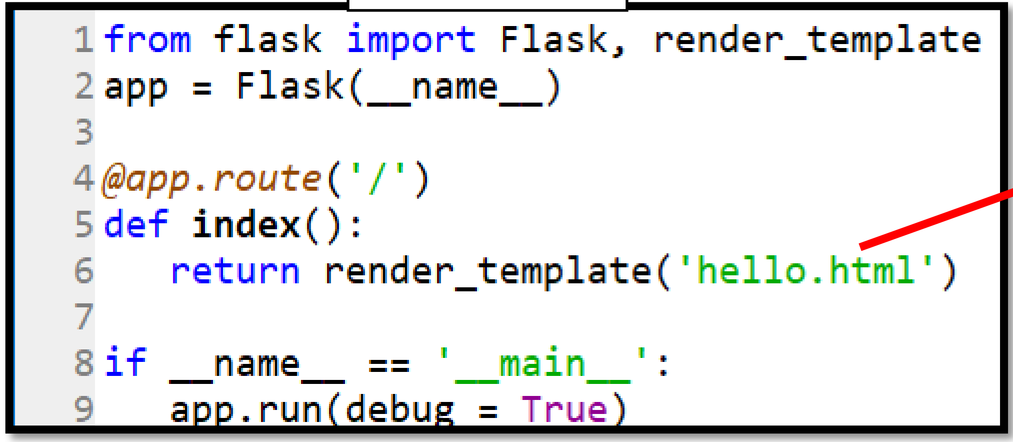
Files Structure



- AppMainFolder
  - App1.py
  - App3.py
  - **static**
    - Files.js
    - Files.css
  - **templates**
    - Index1.html
    - Index2.html
    - Index3.html

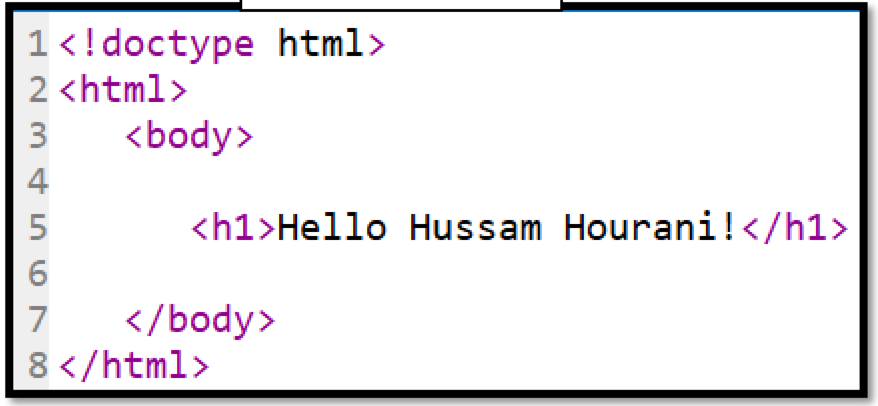
Rendering Templates

flask2.py



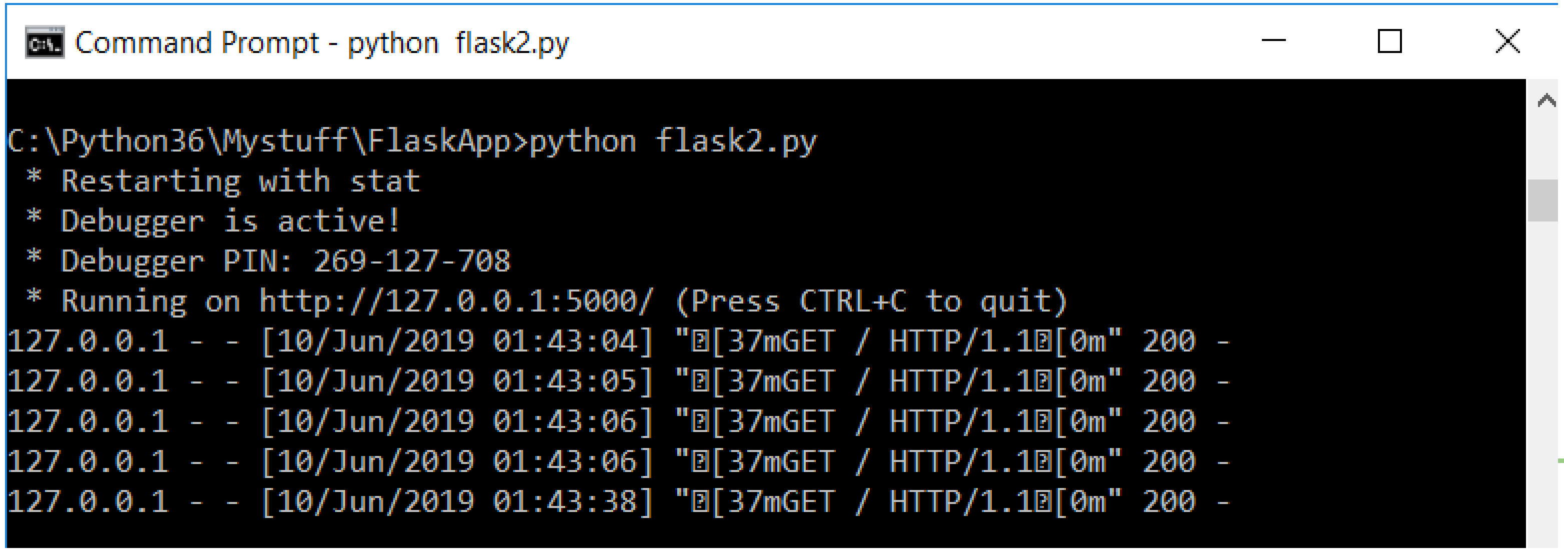
```python
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/')
def index():
    return render_template('hello.html')

if __name__ == '__main__':
    app.run(debug = True)
```

Hello.html



```html
<!doctype html>
<html>
   <body>

      <h1>Hello Hussam Hourani!</h1>

   </body>
</html>
```

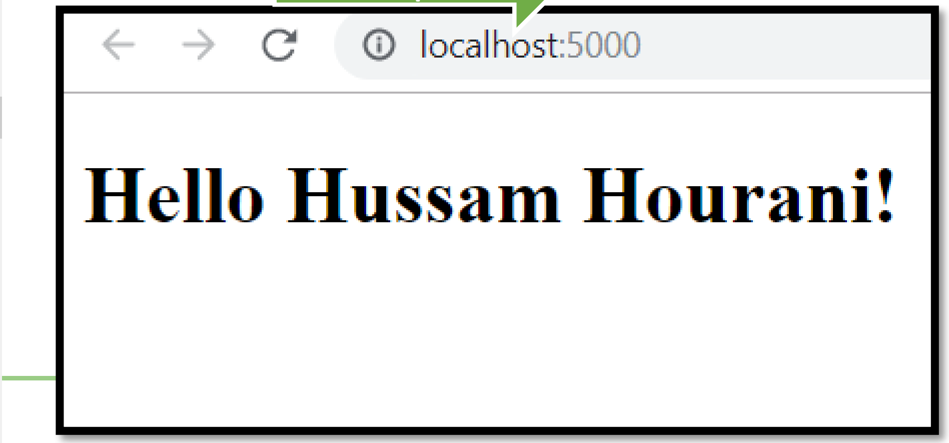
FlaskApp
- -**flask2.py**
- -templates
  - - **hello.html**



Command Prompt - python flask2.py

```
C:\Python36\Mystuff\FlaskApp>python flask2.py
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 269-127-708
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [10/Jun/2019 01:43:04] "[37mGET / HTTP/1.1[0m" 200 -
127.0.0.1 - - [10/Jun/2019 01:43:05] "[37mGET / HTTP/1.1[0m" 200 -
127.0.0.1 - - [10/Jun/2019 01:43:06] "[37mGET / HTTP/1.1[0m" 200 -
127.0.0.1 - - [10/Jun/2019 01:43:06] "[37mGET / HTTP/1.1[0m" 200 -
127.0.0.1 - - [10/Jun/2019 01:43:38] "[37mGET / HTTP/1.1[0m" 200 -
```

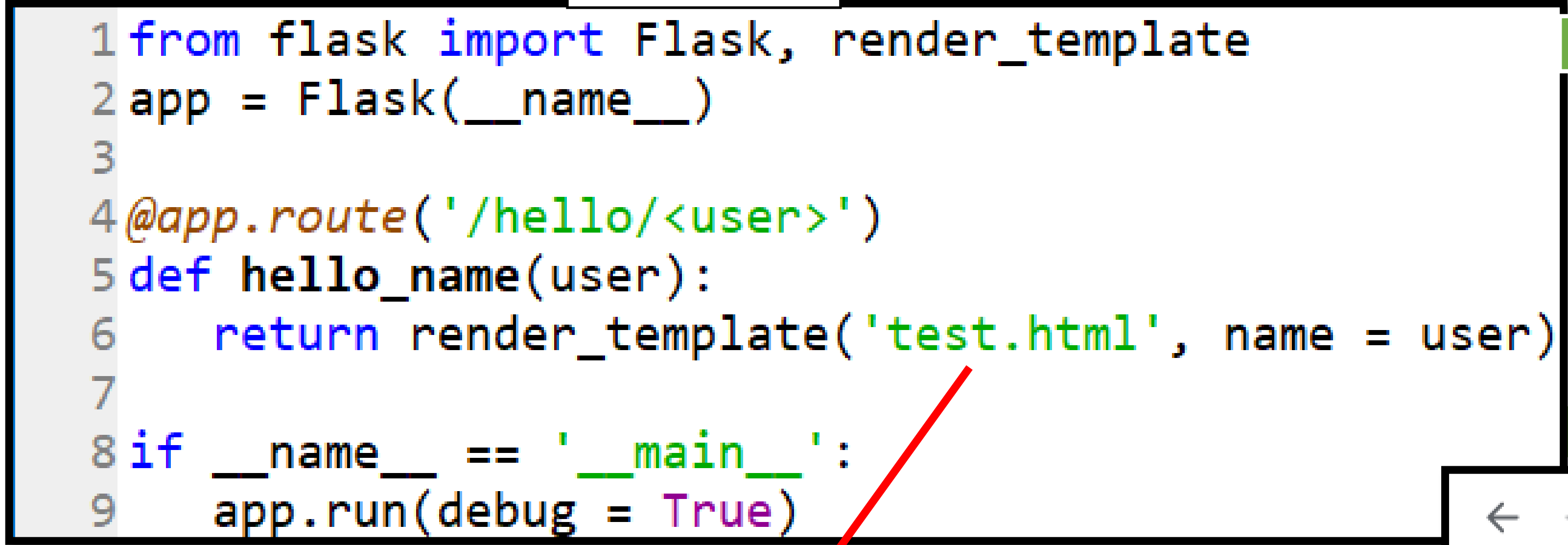
Output



Hello Hussam Hourani!

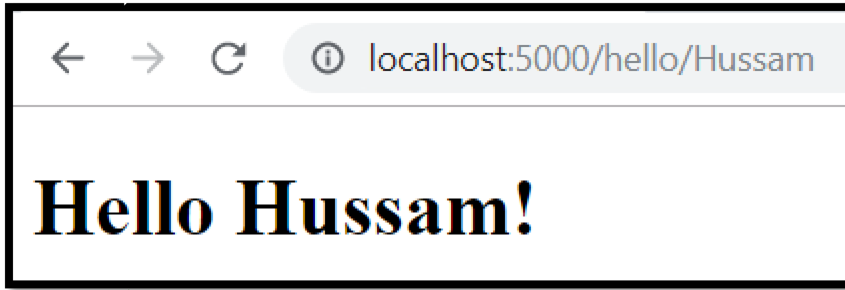
Rendering Templates – Passing Parameters

flask3.py



```python
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/hello/<user>')
def hello_name(user):
   return render_template('test.html', name = user)

if __name__ == '__main__':
   app.run(debug = True)
```

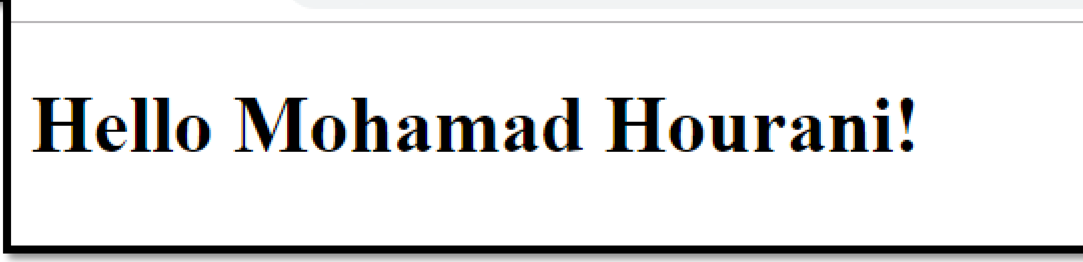
Output



localhost:5000/hello/Hussam

Hello Hussam!

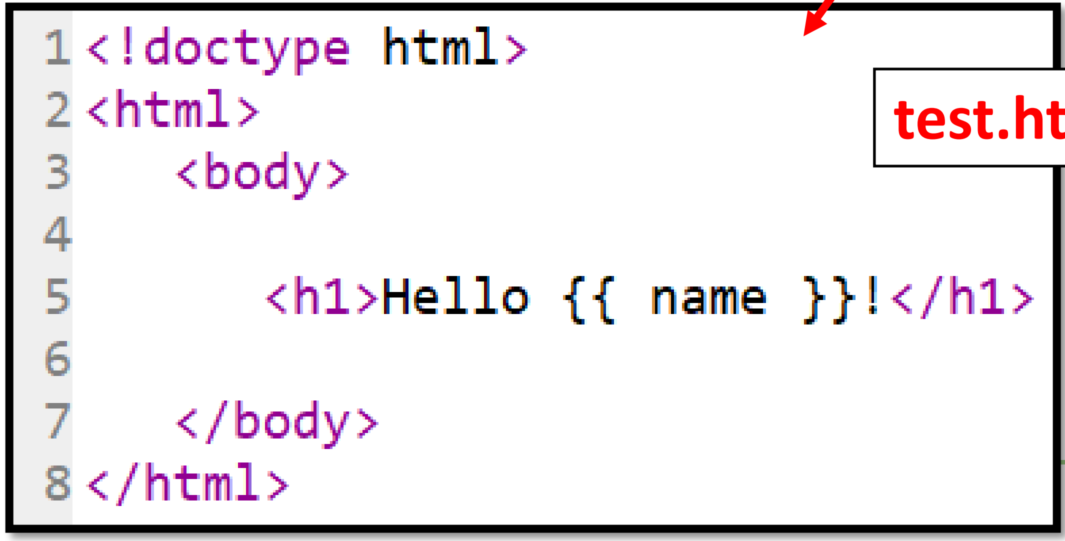
Output



localhost:5000/hello/Mohamad%20Hourani

Hello Mohamad Hourani!

test.html



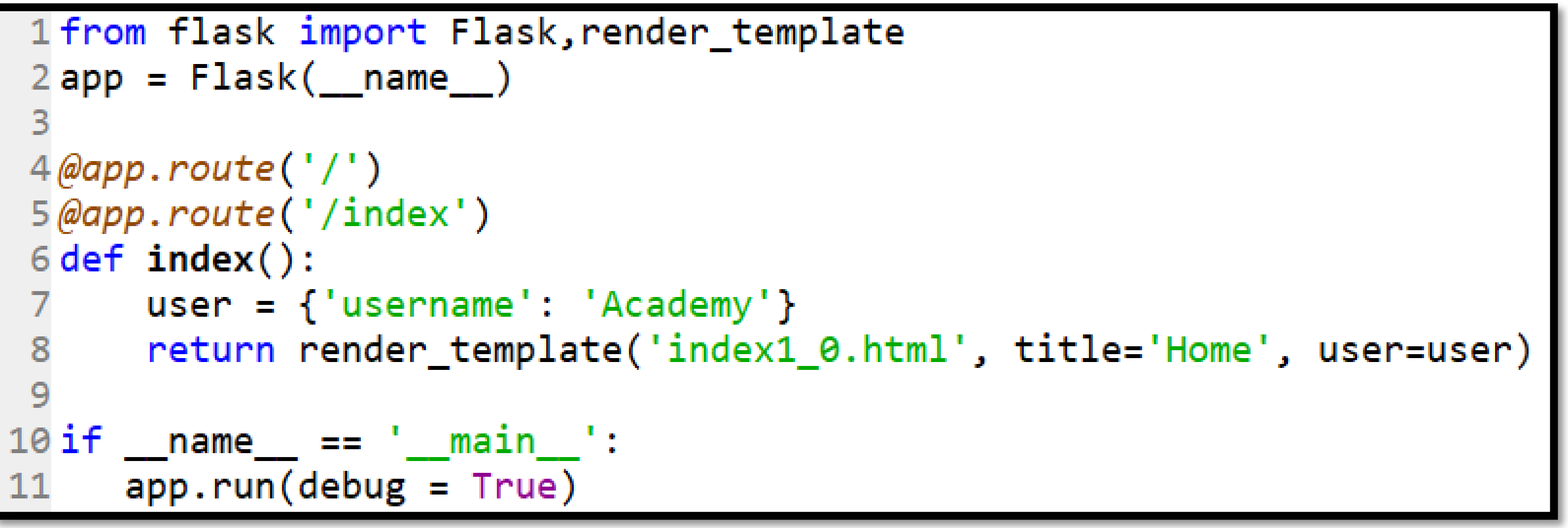
```html
<!doctype html>
<html>
   <body>

      <h1>Hello {{ name }}!</h1>

   </body>
</html>
```

The **Jinja2** template engine uses the following delimiters for escaping from HTML.

{{ ... }} for Expressions to print to the template output

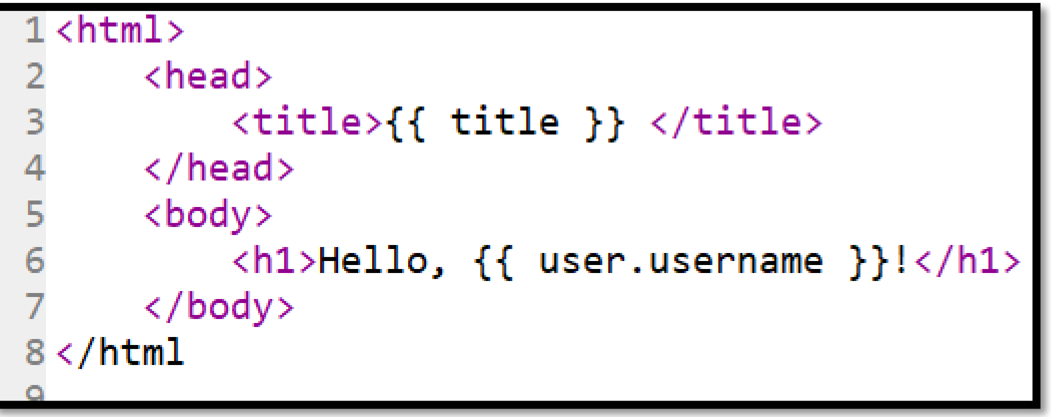
https://www.tutorialspoint.com/flask/flask_quick_guide.htm

Rendering Templates – Passing Parameters

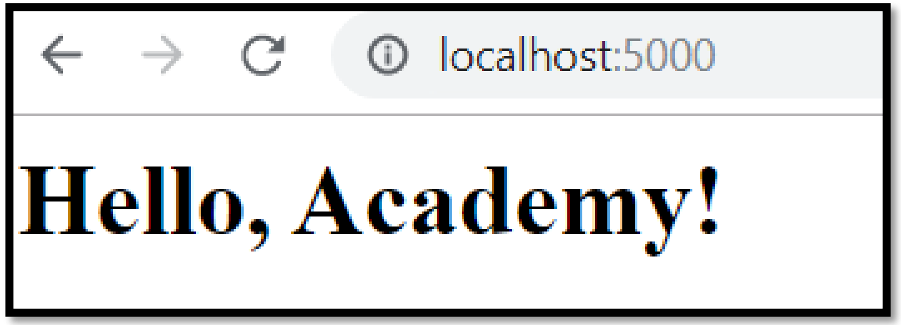


```python
from flask import Flask,render_template
app = Flask(__name__)

@app.route('/')
@app.route('/index')
def index():
    user = {'username': 'Academy'}
    return render_template('index1_0.html', title='Home', user=user)

if __name__ == '__main__':
   app.run(debug = True)
```

Output

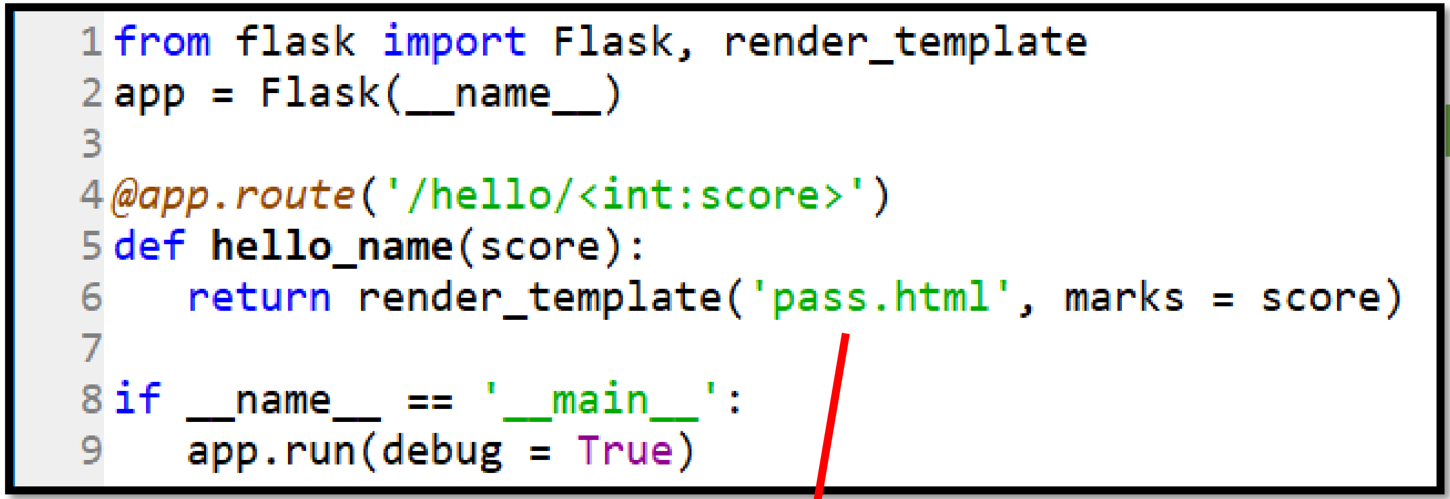


```html
<html>
    <head>
        <title>{{ title }} </title>
    </head>
    <body>
        <h1>Hello, {{ user.username }}!</h1>
    </body>
</html
```



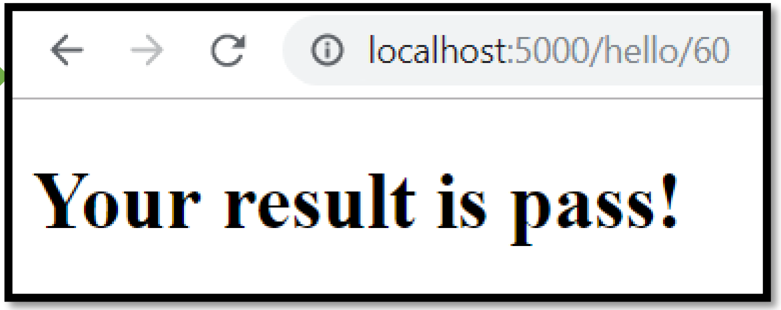
Hello, Academy!

Rendering Templates – Jinja2

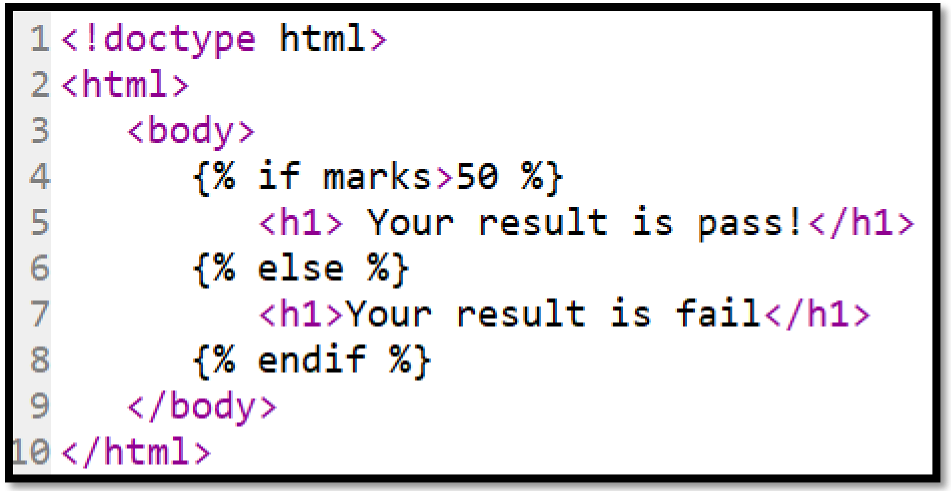


```python
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/hello/<int:score>')
def hello_name(score):
   return render_template('pass.html', marks = score)

if __name__ == '__main__':
   app.run(debug = True)
```

Output

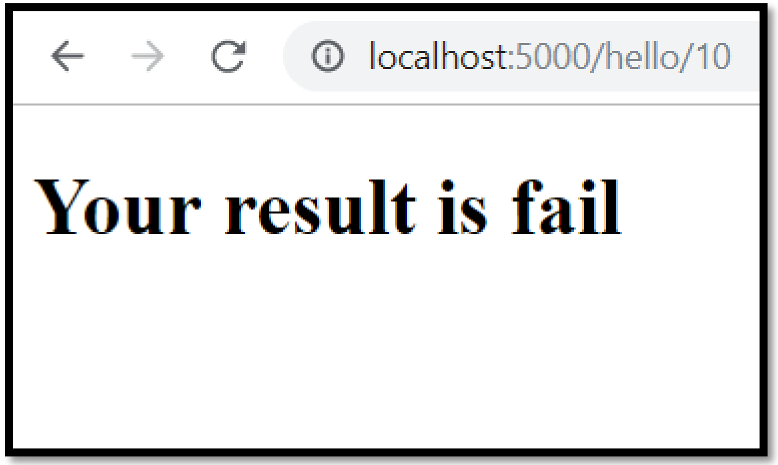


Your result is pass!



```html
<!doctype html>
<html>
   <body>
      {% if marks>50 %}
         <h1> Your result is pass!</h1>
      {% else %}
         <h1>Your result is fail</h1>
      {% endif %}
   </body>
</html>
```

Output



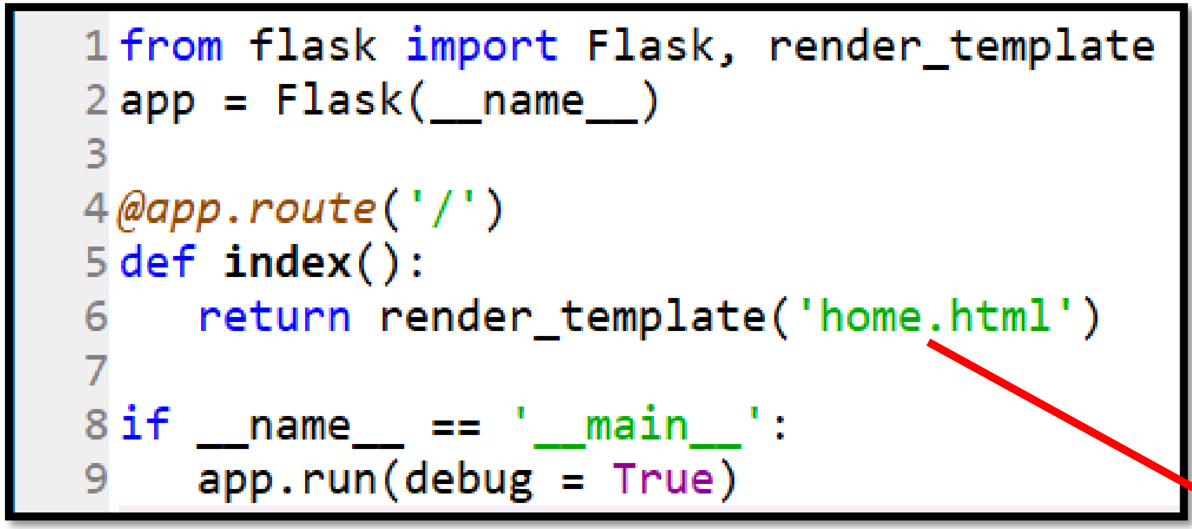
Your result is fail

The **Jinja2** template engine uses the following delimiters for escaping from HTML.

{% ... %} for Statements

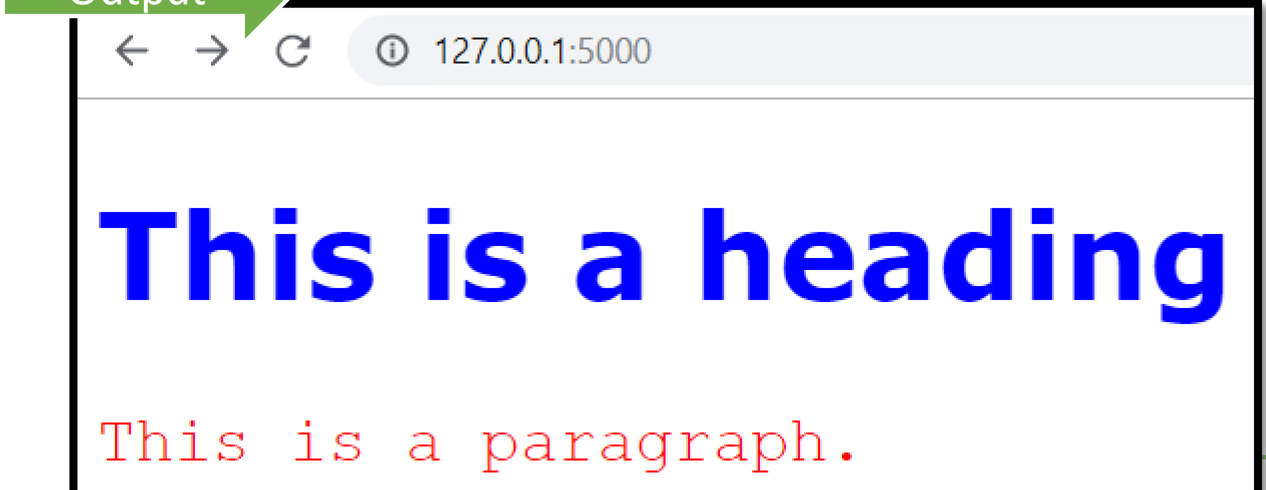
https://www.tutorialspoint.com/flask/flask_quick_guide.htm

More HTML styles



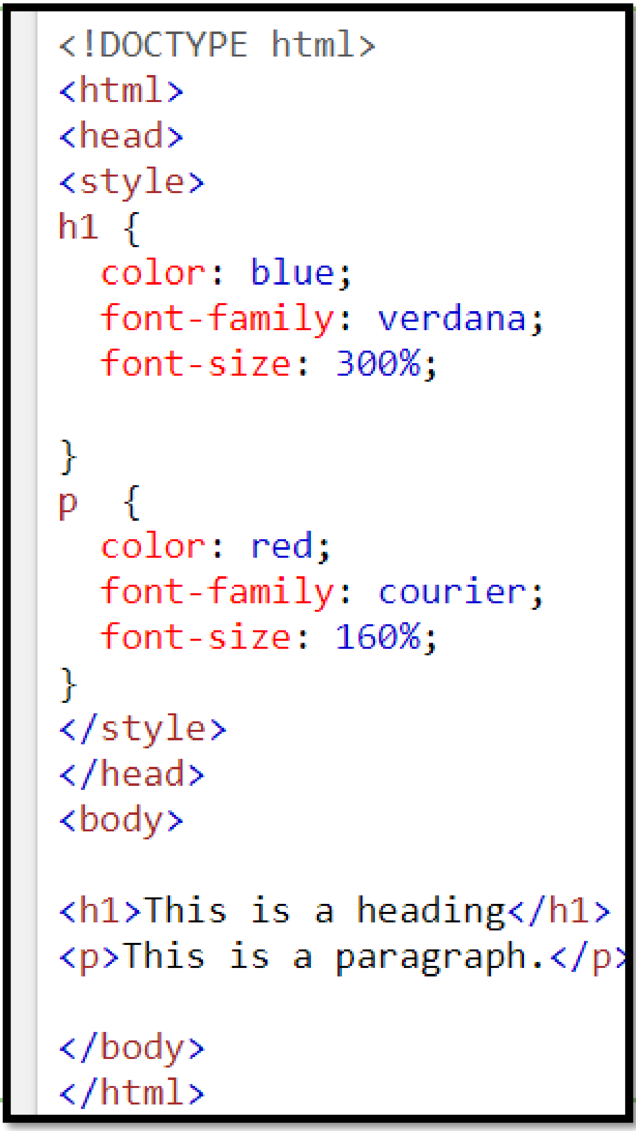
```python
from flask import Flask, render_template
app = Flask(__name__)

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

if __name__ == '__main__':
    app.run(debug = True)
```

Output



This is a heading

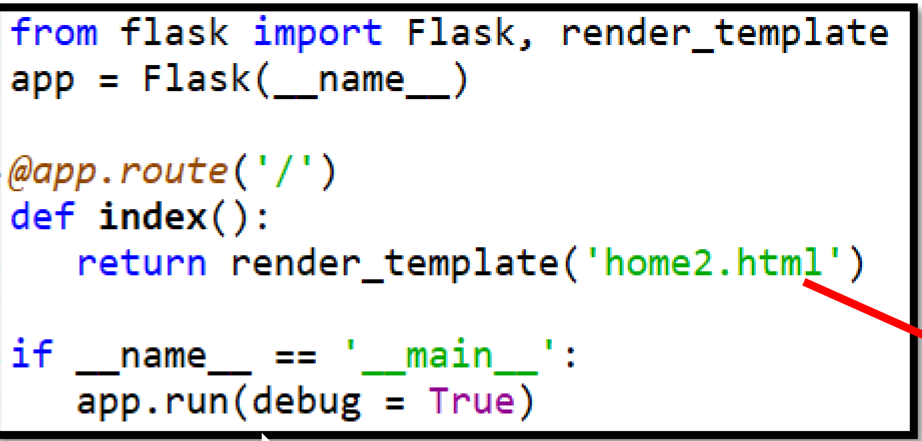
This is a paragraph.



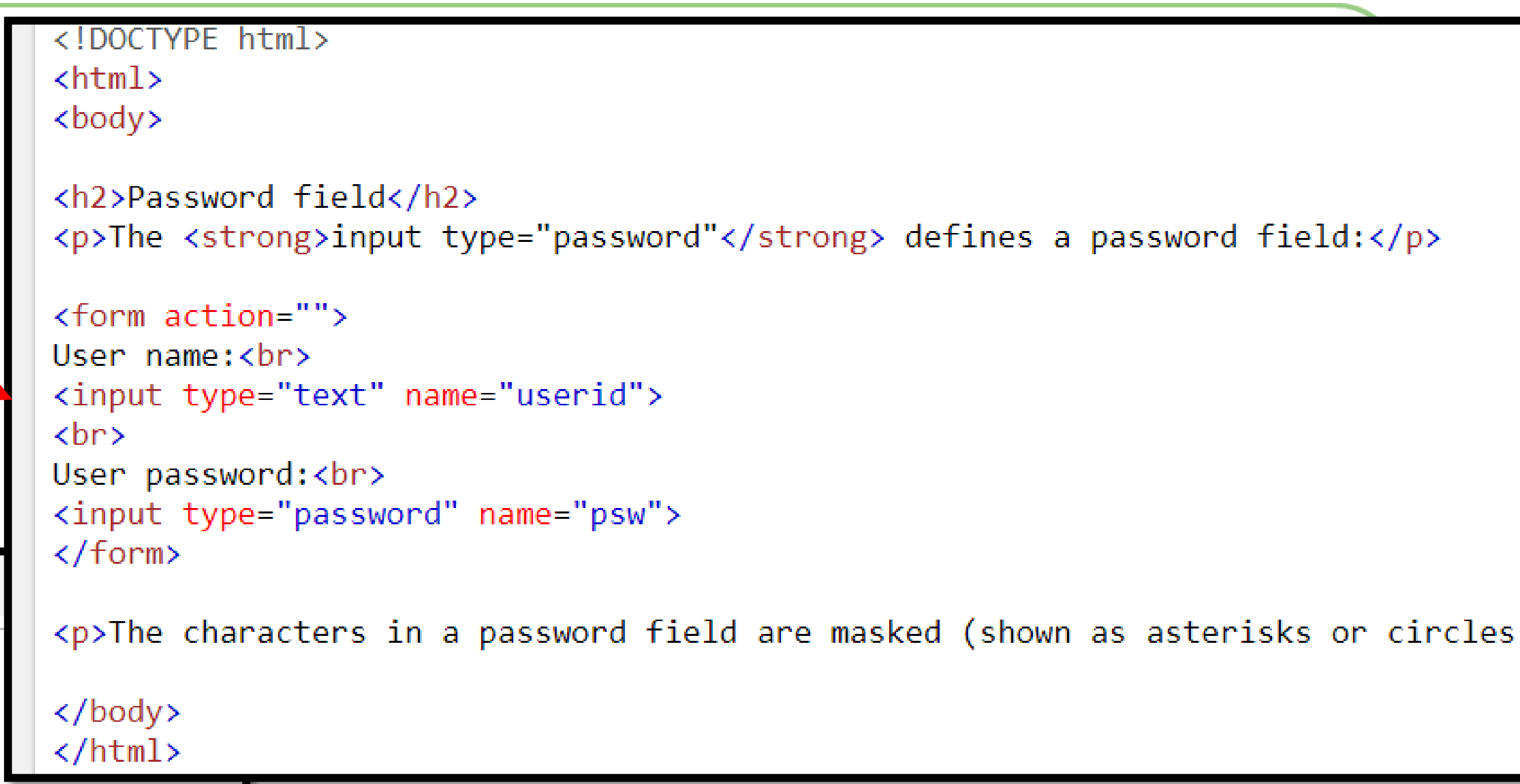
```html
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  color: blue;
  font-family: verdana;
  font-size: 300%;

}
p  {
  color: red;
  font-family: courier;
  font-size: 160%;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

https://www.w3schools.com/html/tryit.asp?filename=tryhtml_css_fonts

More HTML

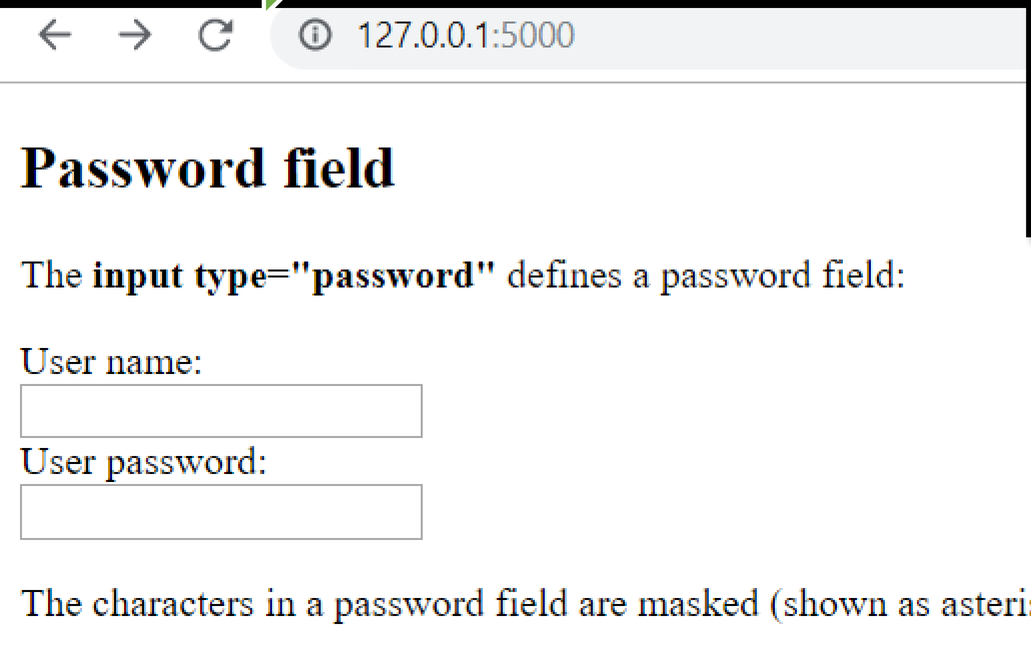


```python
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/')
def index():
   return render_template('home2.html')

if __name__ == '__main__':
   app.run(debug = True)
```



```html
<!DOCTYPE html>
<html>
<body>

<h2>Password field</h2>
<p>The <strong>input type="password"</strong> defines a password field:</p>

<form action="">
User name:<br>
<input type="text" name="userid">
<br>
User password:<br>
<input type="password" name="psw">
</form>

<p>The characters in a password field are masked (shown as asterisks or circles

</body>
</html>
```

Output



127.0.0.1:5000

Password field

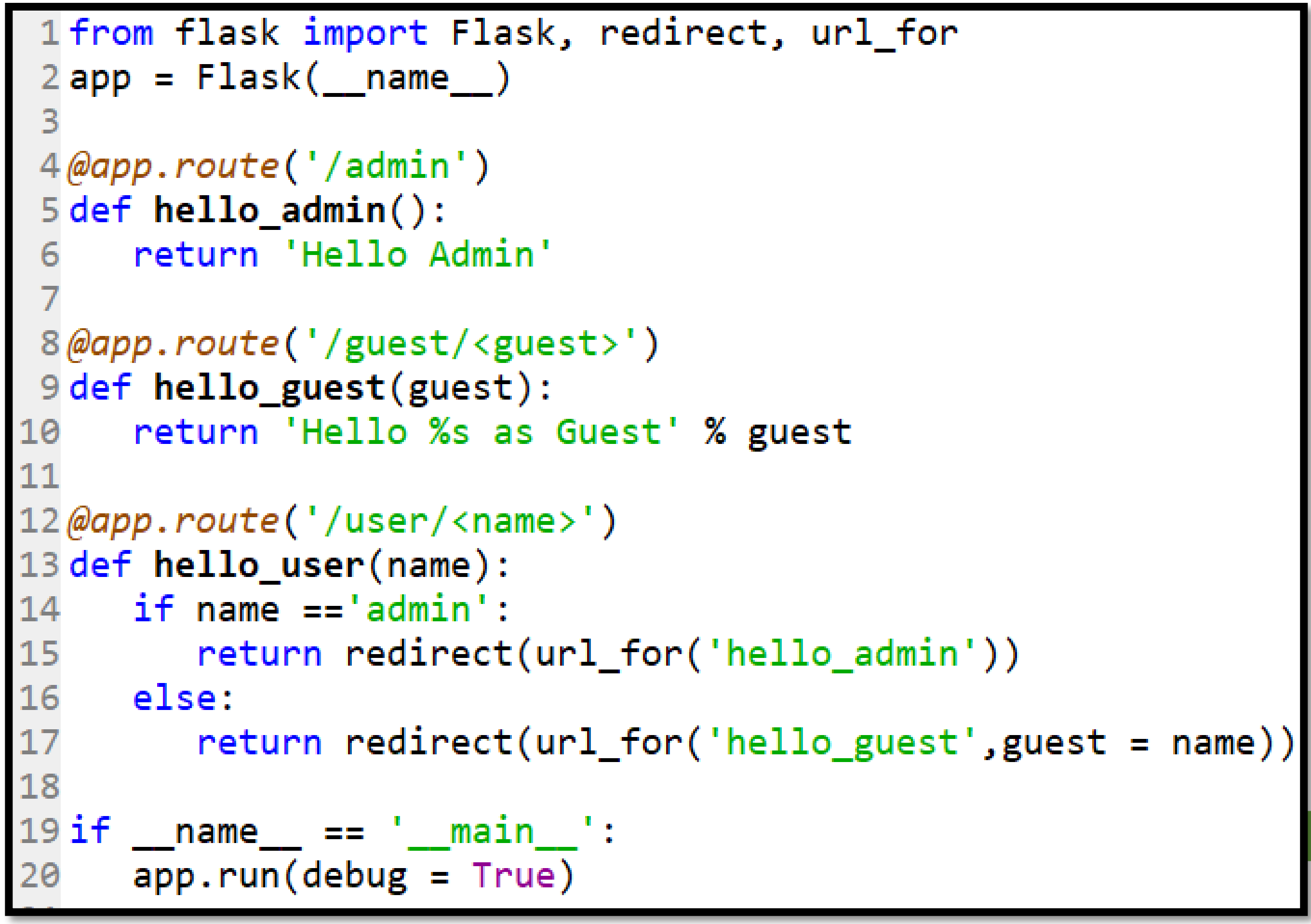
The **input type="password"** defines a password field:

User name:

User password:

The characters in a password field are masked (shown as asterisks or circles).

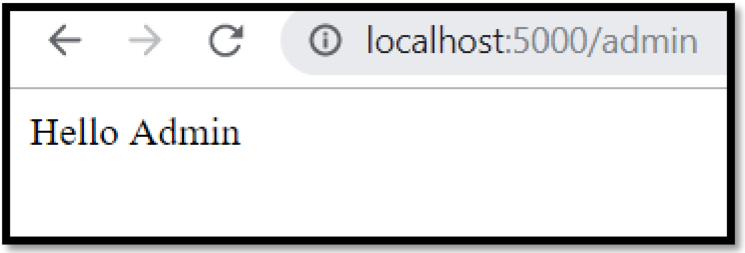
URL Building



```python
from flask import Flask, redirect, url_for
app = Flask(__name__)

@app.route('/admin')
def hello_admin():
   return 'Hello Admin'

@app.route('/guest/<guest>')
def hello_guest(guest):
   return 'Hello %s as Guest' % guest

@app.route('/user/<name>')
def hello_user(name):
   if name =='admin':
      return redirect(url_for('hello_admin'))
   else:
      return redirect(url_for('hello_guest',guest = name))

if __name__ == '__main__':
   app.run(debug = True)
```

http://localhost:5000/user/admin

Output

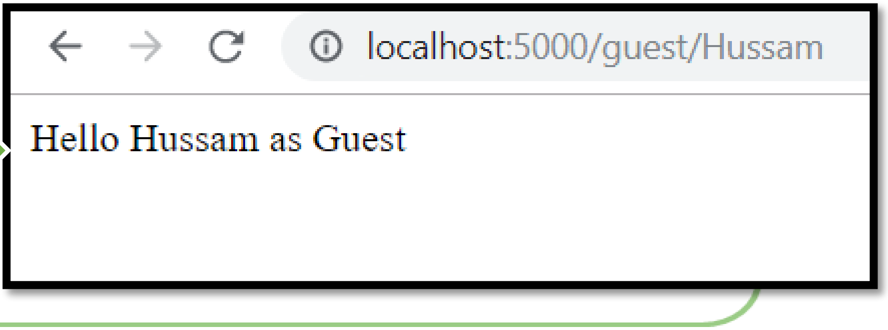


localhost:5000/admin

Hello Admin

http://localhost:5000/user/mvl

Output



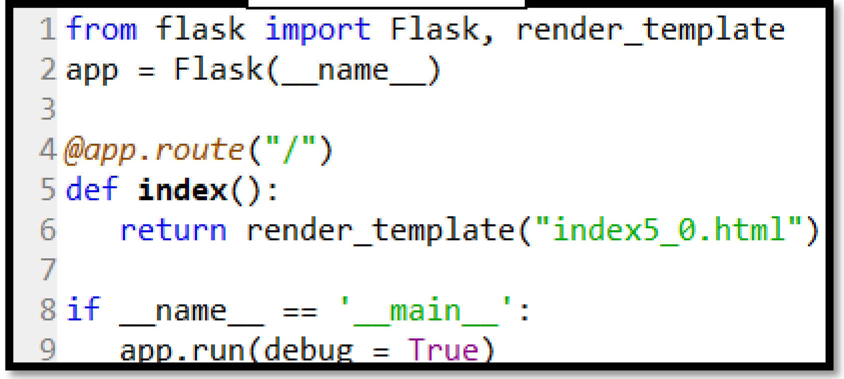
localhost:5000/guest/Hussam

Hello Hussam as Guest

https://www.tutorialspoint.com/flask/flask_quick_guide.htm

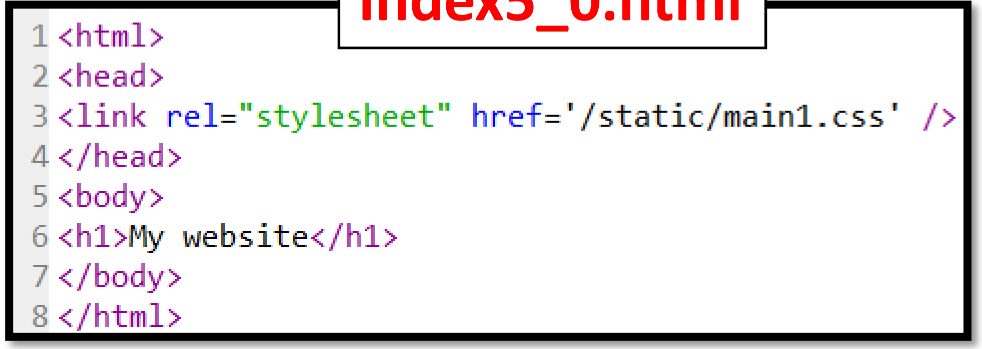
CSS – Project1

flask6.py



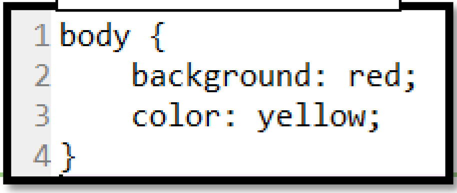
```python
from flask import Flask, render_template
app = Flask(__name__)

@app.route("/")
def index():
    return render_template("index5_0.html")

if __name__ == '__main__':
    app.run(debug = True)
```

Index5_0.html



```html
<html>
<head>
<link rel="stylesheet" href='/static/main1.css' />
</head>
<body>
<h1>My website</h1>
</body>
</html>
```

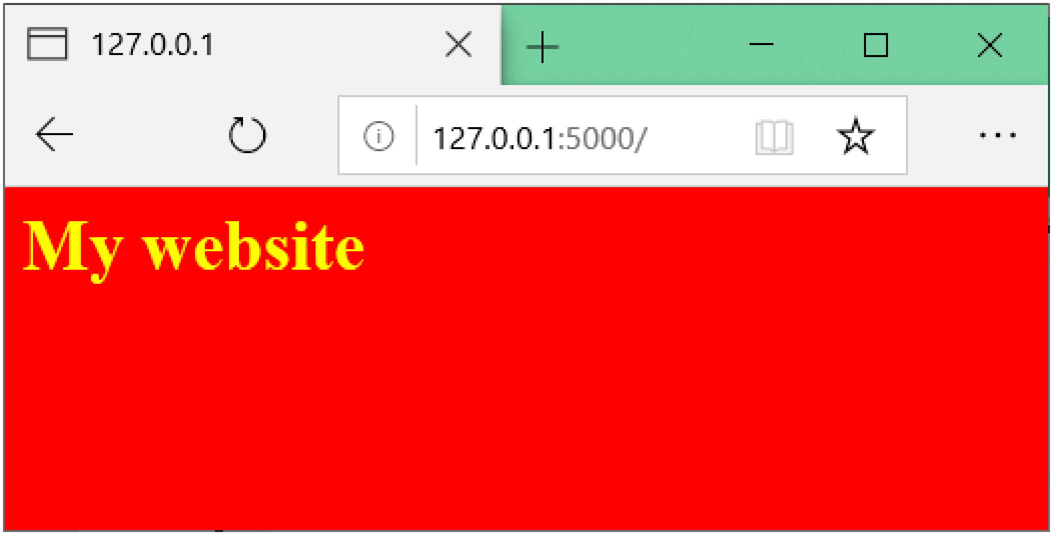
Main1.css



```css
body {
    background: red;
    color: yellow;
}
```

Save under: static subfolder

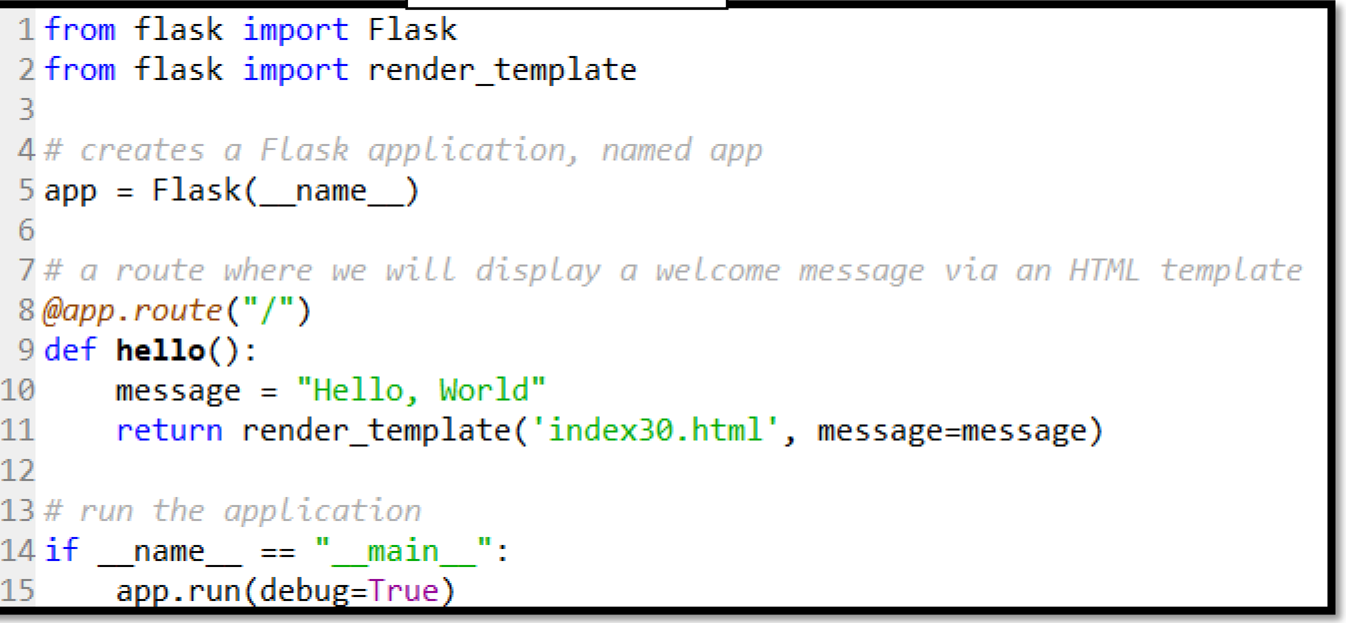
Output



My website

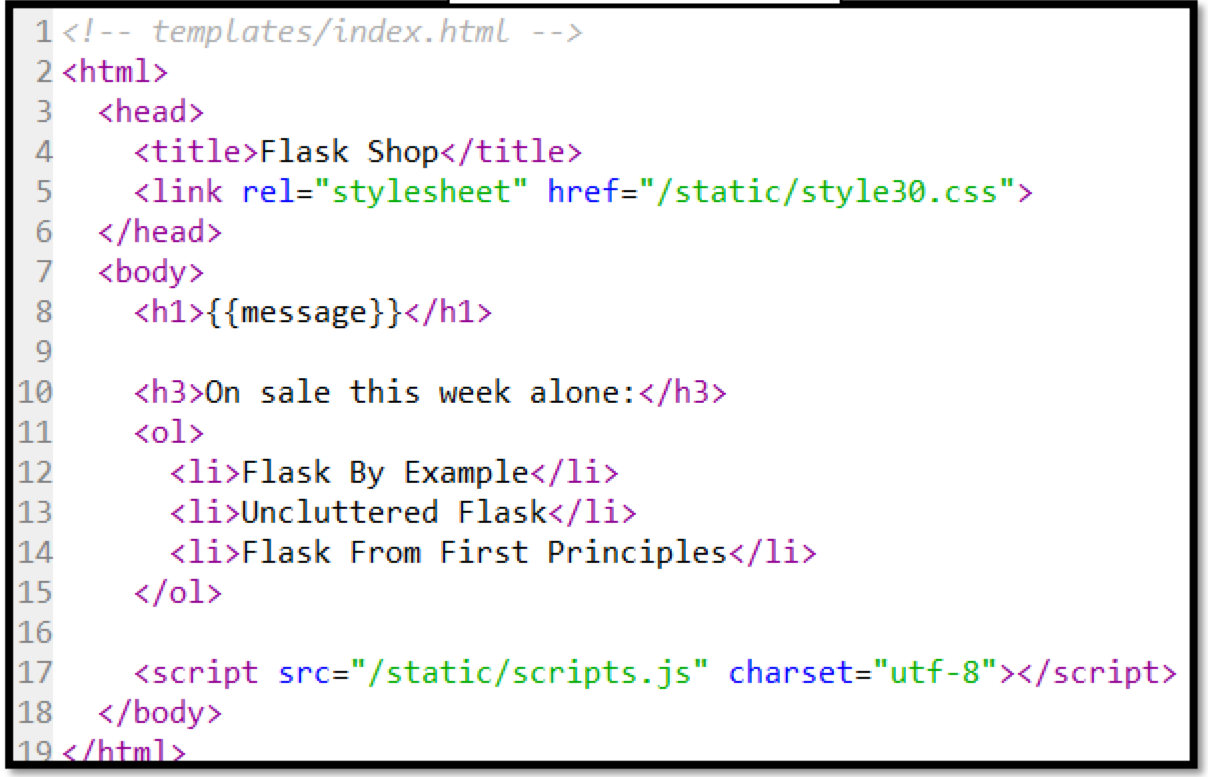
CSS – Project2

Flask30.py



```python
from flask import Flask
from flask import render_template

# creates a Flask application, named app
app = Flask(__name__)

# a route where we will display a welcome message via an HTML template
@app.route("/")
def hello():
    message = "Hello, World"
    return render_template('index30.html', message=message)

# run the application
if __name__ == "__main__":
    app.run(debug=True)
```

index30.html

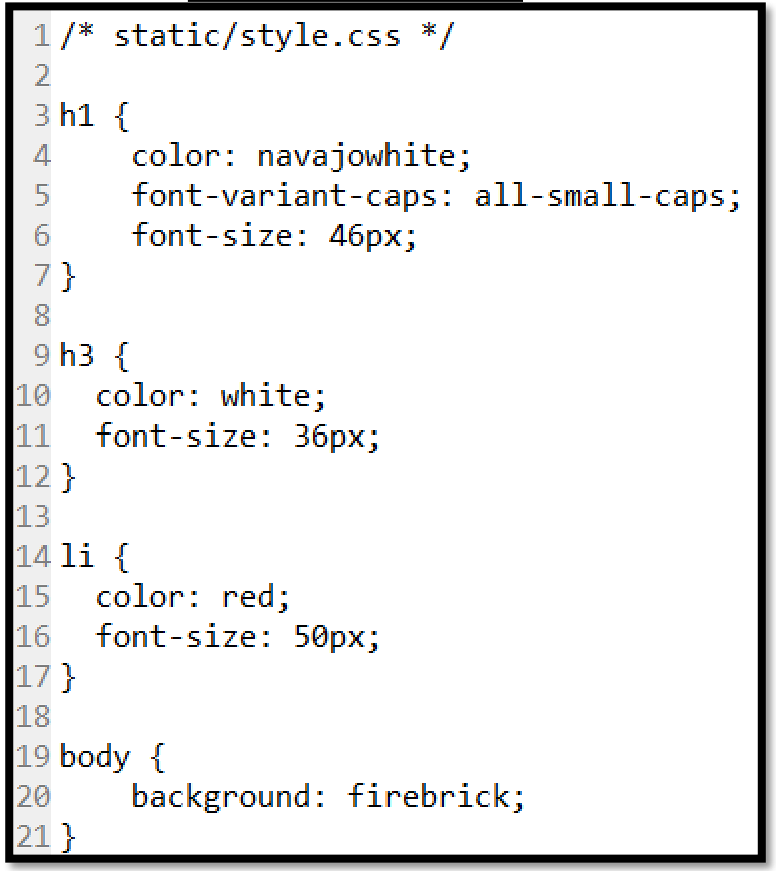


```html
<!-- templates/index.html -->
<html>
  <head>
    <title>Flask Shop</title>
    <link rel="stylesheet" href="/static/style30.css">
  </head>
  <body>
    <h1>{{message}}</h1>

    <h3>On sale this week alone:</h3>
    <ol>
      <li>Flask By Example</li>
      <li>Uncluttered Flask</li>
      <li>Flask From First Principles</li>
    </ol>

    <script src="/static/scripts.js" charset="utf-8"></script>
  </body>
</html>
```

Save under: template subfolder

CSS – Project2

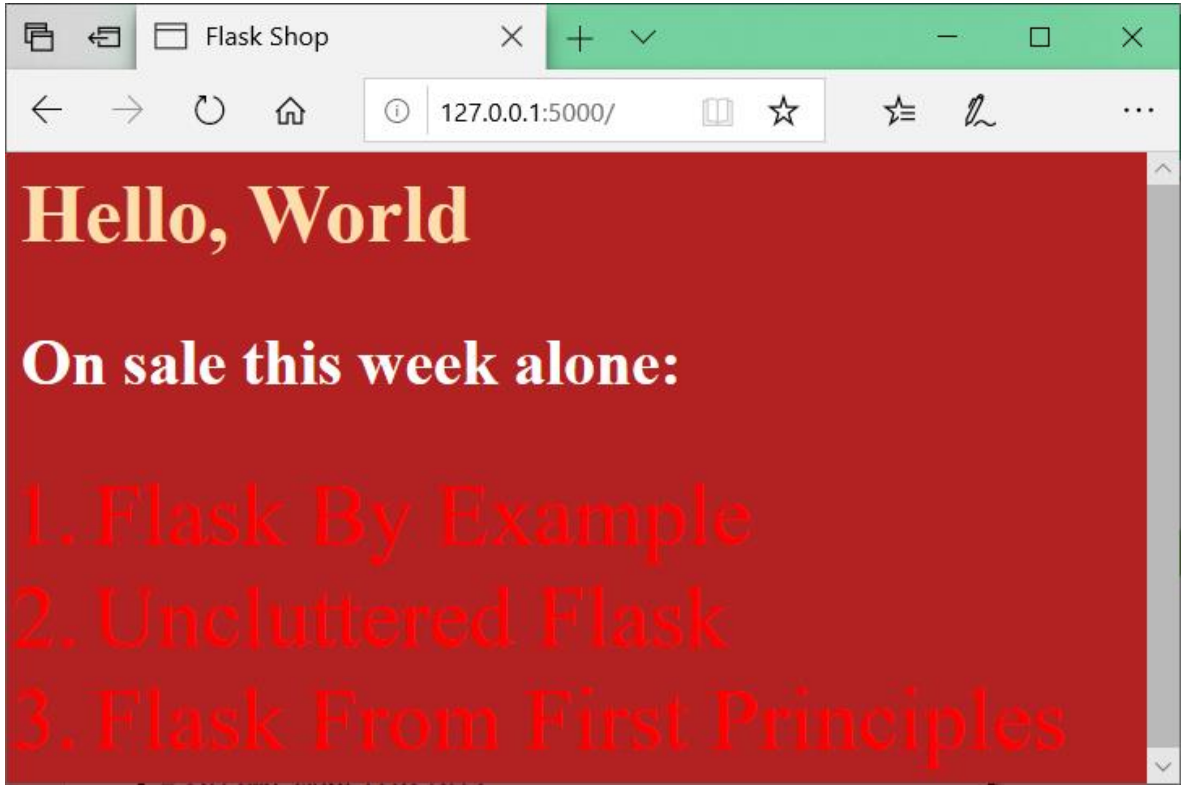
Style30.css



```css
/* static/style.css */

h1 {
     color: navajowhite;
     font-variant-caps: all-small-caps;
     font-size: 46px;
}

h3 {
   color: white;
   font-size: 36px;
}

li {
   color: red;
   font-size: 50px;
}

body {
     background: firebrick;
}
```

Save under: static subfolder

Output



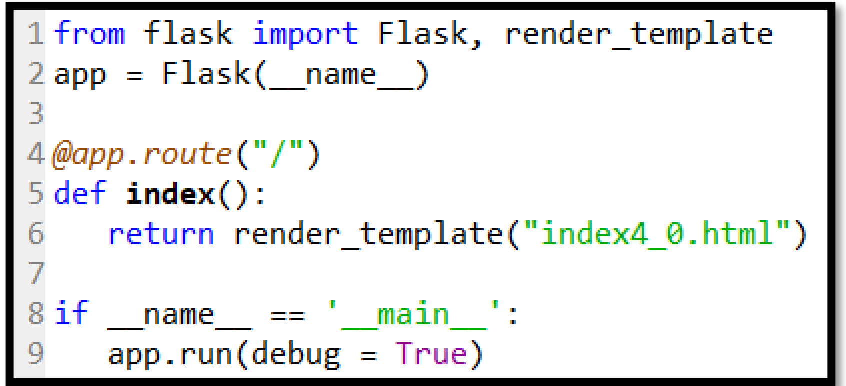
Hello, World

On sale this week alone:

1. Flask By Example
2. Uncluttered Flask
3. Flask From First Principles

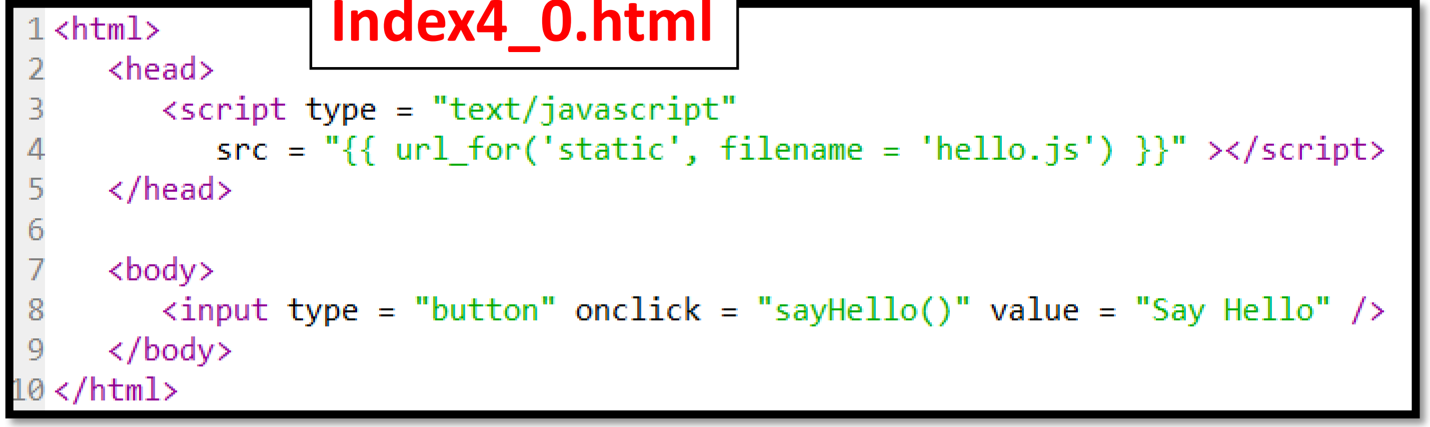
Javascript

flask7.py



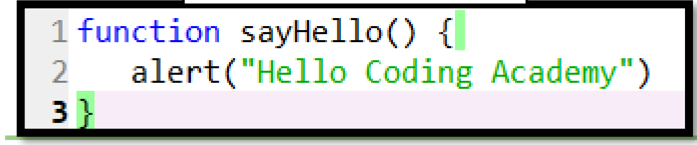
```python
from flask import Flask, render_template
app = Flask(__name__)

@app.route("/")
def index():
    return render_template("index4_0.html")

if __name__ == '__main__':
    app.run(debug = True)
```

Index4_0.html



```html
<html>
   <head>
      <script type = "text/javascript"
         src = "{{ url_for('static', filename = 'hello.js') }}" ></script>
   </head>

   <body>
      <input type = "button" onclick = "sayHello()" value = "Say Hello" />
   </body>
</html>
```

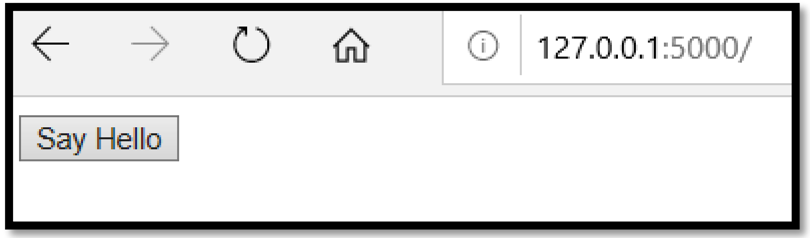
Hello.js

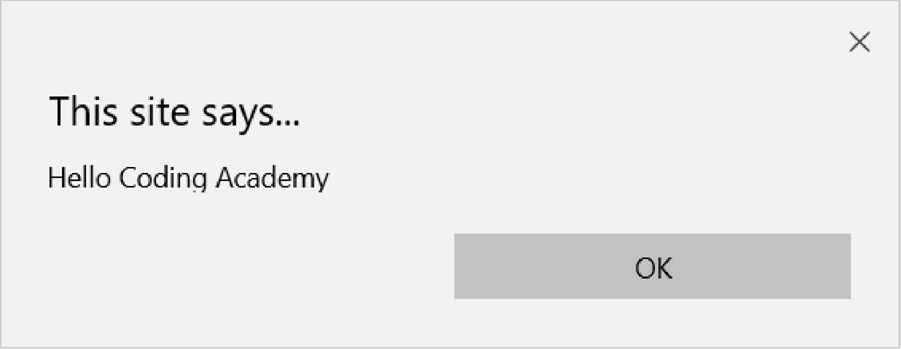


```javascript
function sayHello() {
   alert("Hello Coding Academy")
}
```

Save under: static subfolder

Output





This site says...

Hello Coding Academy

End of the lesson

Thank You

Master in Software Engineering

Hussam Hourani has over 25 years of Organizations Transformation, VROs, PMO, Large Scale and Enterprise Programs Global Delivery, Leadership, Business Development and Management Consulting. His client experience is wide ranging across many sectors but focuses on Performance Enhancement, Transformation, Enterprise Program Management, Artificial Intelligence and Data Science.