

HANDS_ON FLASK-03 : Handling Routes and Templates with Flask Web Application and If-For Structure

- Purpose of the this hands_on training is to give the students introductory knowledge of how to handle routes and use html templates within a Flask web application on Amazon Linux 2 EC2 instance.

Learning Outcomes

At the end of the hands-on training, students will be able to;

- install Python and Flask framework on Amazon Linux 2 EC2 instance
- build a simple web application with Flask framework.
- understand the HTTP request-response cycle and structure of URL.
- create routes (or views) with Flask.
- serve static content and files using Flask.
- serve dynamic content using the html templates.
- write html templates using Jinja Templating Engine.

Outline

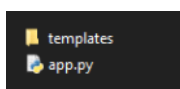
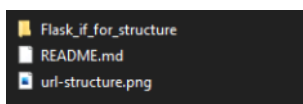
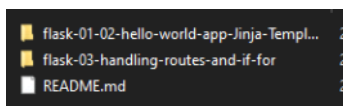
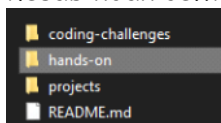
- Part 1 - Getting to know routing and HTTP URLs.
- Part 2 - Write a Web Application using If conditions and for loops
- Part 3 - Write a Web Application with Sample Routings and Templating on GitHub Repo
- Part 4 - Install Python and Flask framework Amazon Linux 2 EC2 Instance and Run the Hello World App on EC2 Instance

Getting to know routing and HTTP URLs.

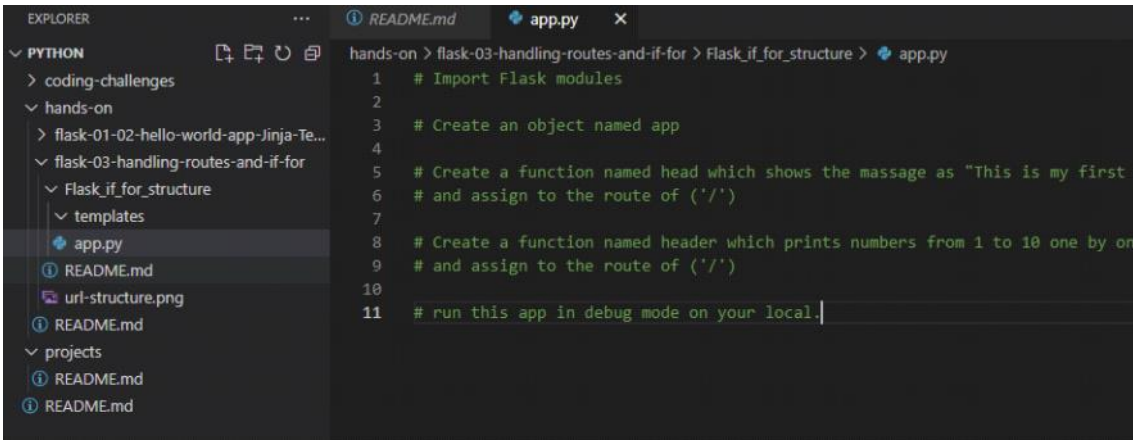
HTTP (Hypertext Transfer Protocol) is a request-response protocol. A client on one side (web browser) asks or requests something from a server and the server on the other side sends a response to that client. When we open our browser and write down the URL (Uniform Resource Locator), we are requesting a resource from a server and the URL is the address of that resource. The structure of typical URL is as the following.

Oncelikle asagidaki görüntüyü (klasör ve dosyaları olusturuyoruz)

Asagida gorulen sablonun bir kismini ve icerisindeki readme dosyasini clarusway-aws-8-21 hesabindan cektik. Bu görüntüyü VS cod da alalim. Templates klasoru kucuk harfle yazilmali

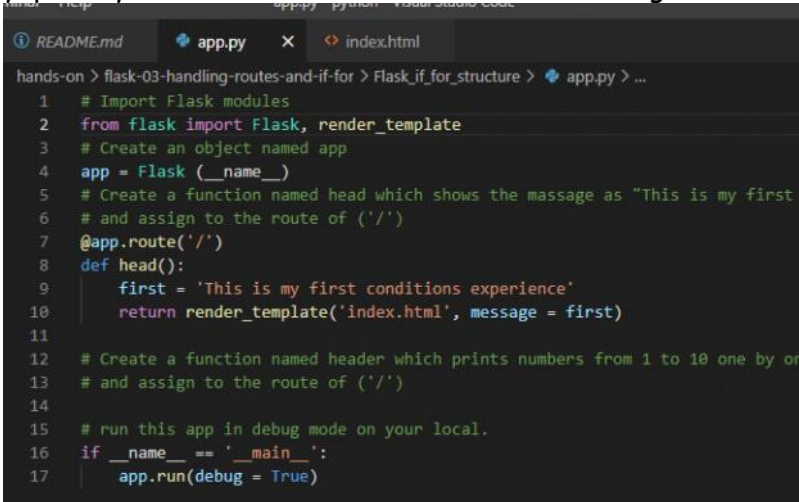


Son olarak readme dosyasi icerisindeki asagidaki yazilari app.py icerisine yapistiralim



```
1 # Import Flask modules
2
3 # Create an object named app
4
5 # Create a function named head which shows the message as "This is my first conditions experience"
6 # and assign to the route of ('/')
7
8 # Create a function named header which prints numbers from 1 to 10 one by one
9 # and assign to the route of ('/')
10
11 # run this app in debug mode on your local.
```

Oncelikle Flask i import ediyoruz ve template kiralayacagimiz icin render_template
Sonra object olusturuyoruz. Templates kalsoru icerisinde index.html dosyasi olusturuyoruz ve eki yapistiriyoruz. Hatalarimizi bulabilmesi icin debug modda calistiriyoruz

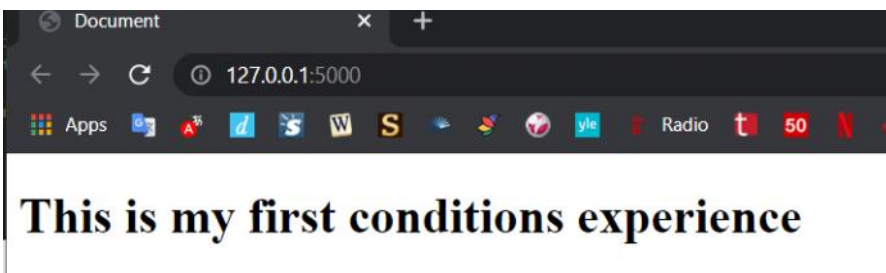


```
1 # Import Flask modules
2 from flask import Flask, render_template
3 # Create an object named app
4 app = Flask(__name__)
5 # Create a function named head which shows the message as "This is my first conditions experience"
6 # and assign to the route of ('/')
7 @app.route('/')
8 def head():
9     first = 'This is my first conditions experience'
10     return render_template('index.html', message = first)
11
12 # Create a function named header which prints numbers from 1 to 10 one by one
13 # and assign to the route of ('/')
14
15 # run this app in debug mode on your local.
16 if __name__ == '__main__':
17     app.run(debug = True)
```

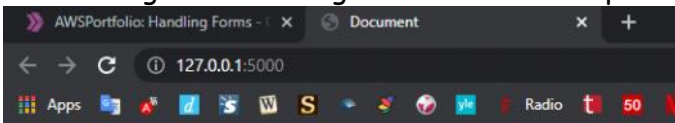
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  {% if message %}
    <h1> {{ message }} </h1>
  {% else %}
    <h1> "There is no message in here..."</h1>
  {% endif %}
</body>
</html>
```

* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

► Eger flask dosyasini calistirmazsa 'pip3 install flask' yazin



Return blogundaki message = first kismini silip tekrar calistiriyoruz



"There is no message in here..."

Yukarida sildigimiz kismi tekrar ekliyoruz.

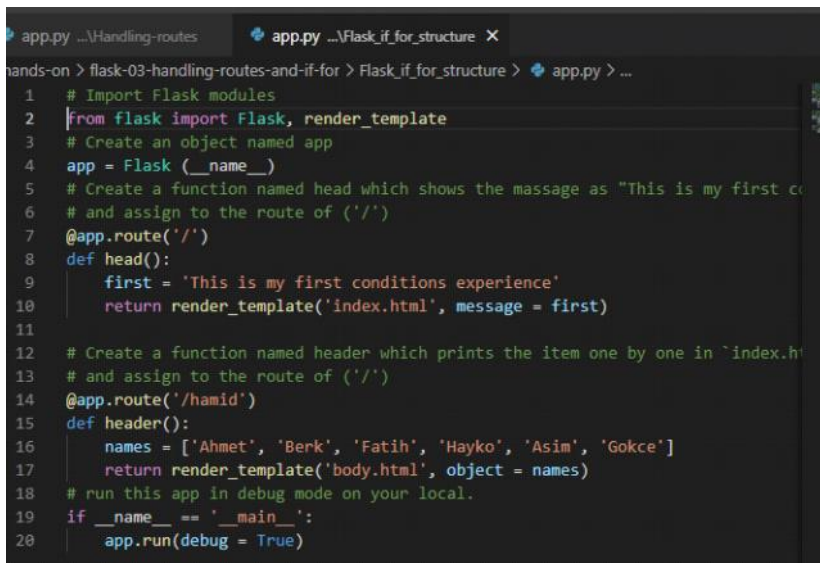
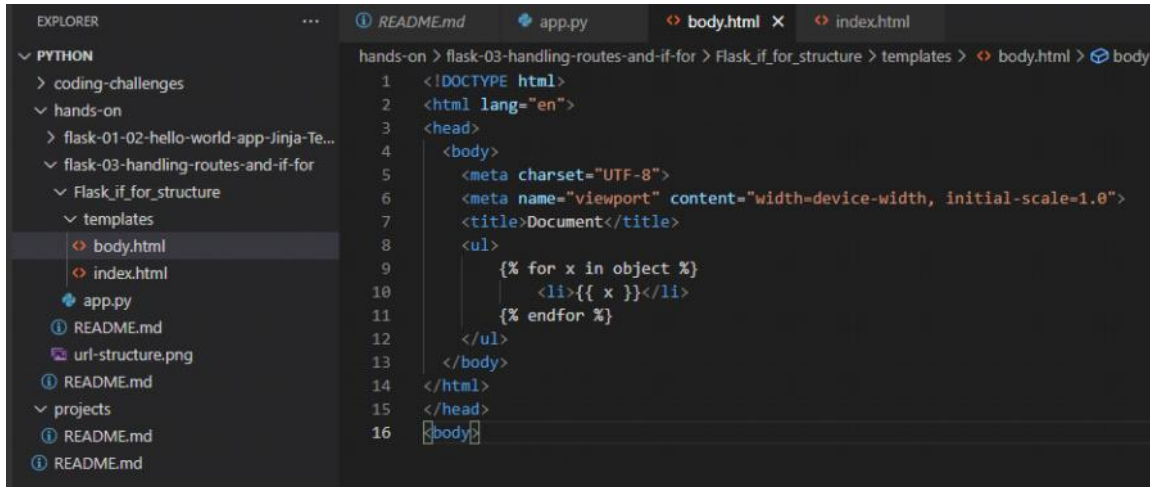
Templates in icerisine body.html dosyasi olusturuyoruz ve asagidaki kismi yapistiriyoruz. (app.py 15-18 arasi yazildi)

```
<!DOCTYPE html>
<html lang="en">
```

```

<head>
<body>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <ul>
    {% for x in object %}
      <li>{{ x }}</li>
    {% endfor %}
  </ul>
</body>
</html>
</head>
<body>

```

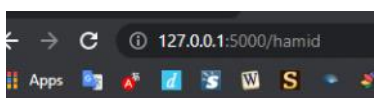


Python kodunu calistirinca asagidaki gorseli aliyoruz(/hamid ==> unutmayin)

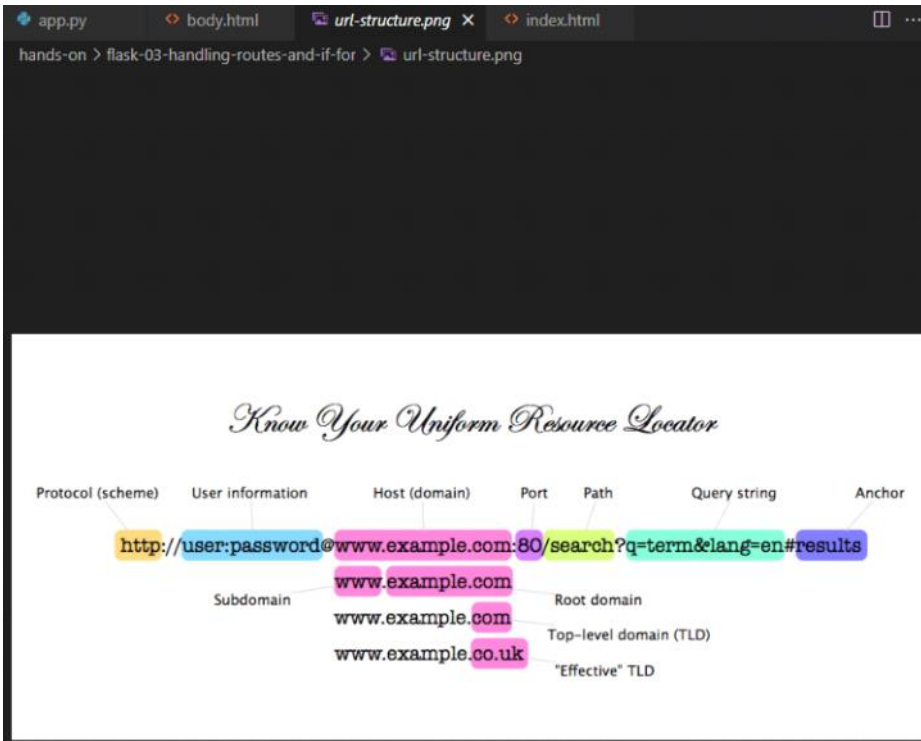
```

* Debugger is
* Debugger PI Follow link (ctrl + click)
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

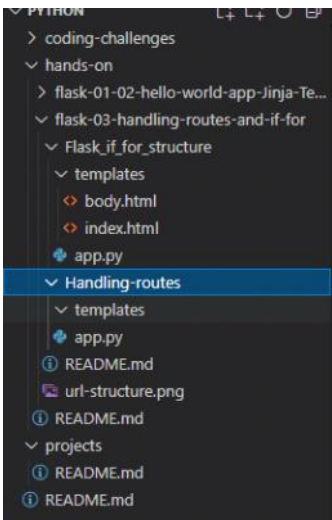
```

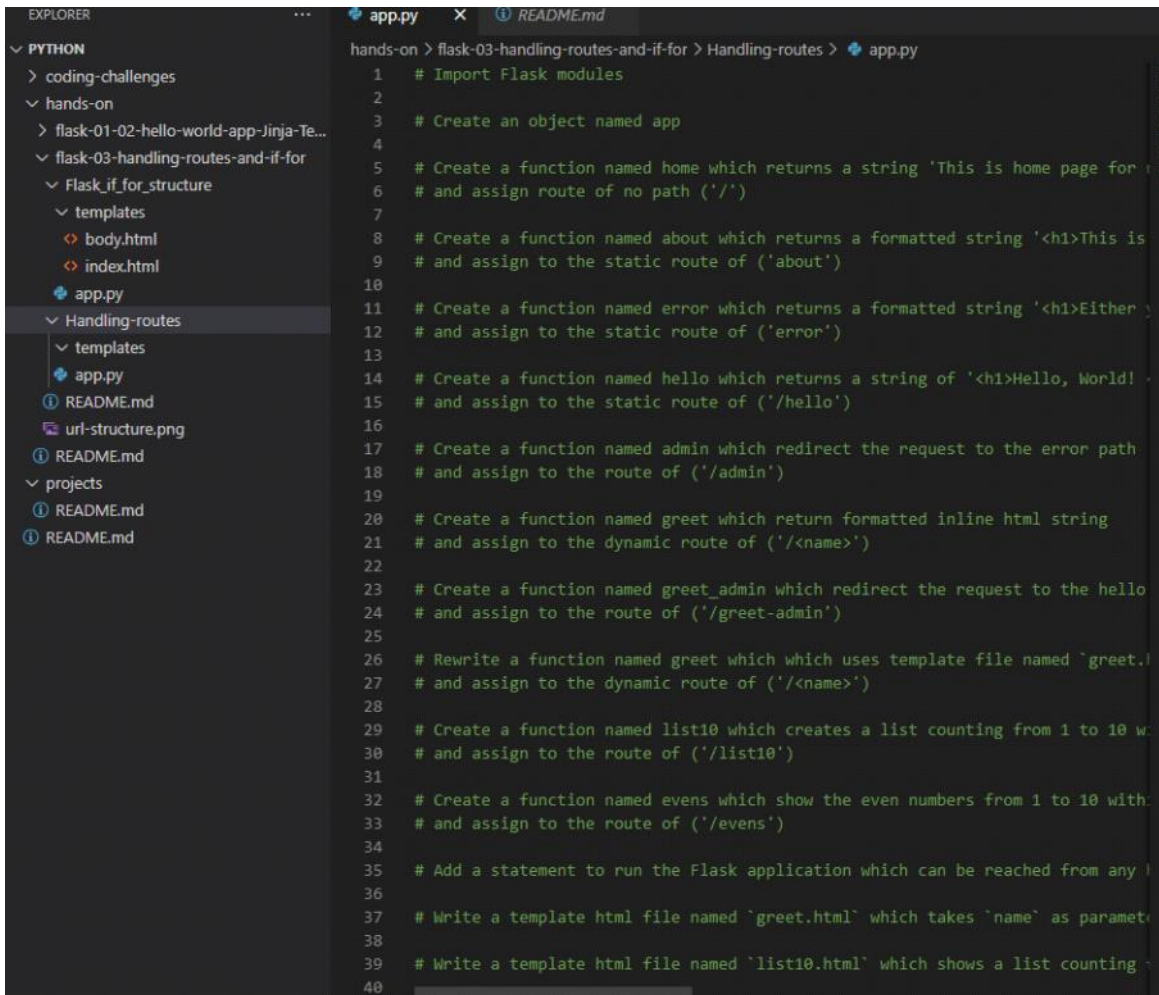


- Ahmet
- Berk
- Fatih
- Hayko
- Asim
- Gokce



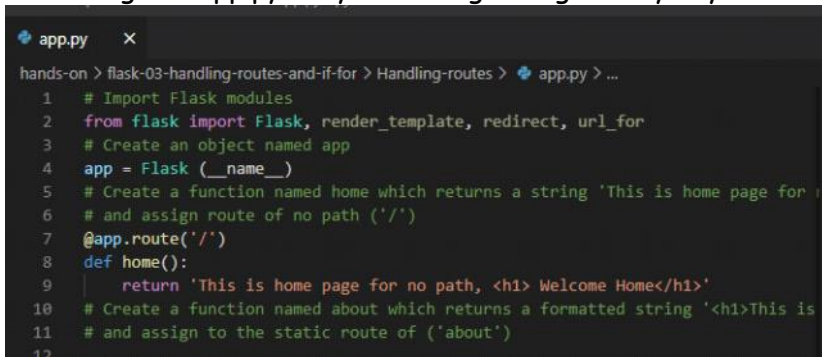
Eski actigimiz sayfaları kapatıyoruz. Flask-03 klasorunun icerisine gorsele gore dizayn ediyoruz. Handling-routes klasoru olusturuyoruz icerisine templates klasoru ve ayni konuma app.py dosyasi olusturup bu dosyaya readme deki ilgili yorum satirini yapistiriyoruz.



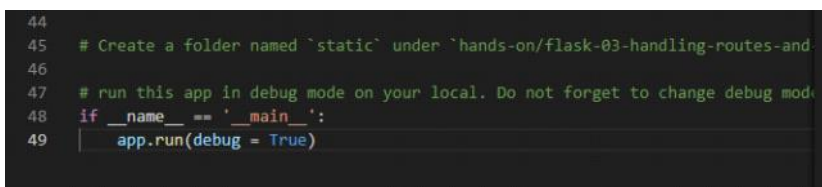


```
1 # Import Flask modules
2
3 # Create an object named app
4
5 # Create a function named home which returns a string 'This is home page for no path'
6 # and assign route of no path ('/')
7
8 # Create a function named about which returns a formatted string '<h1>This is about page for no path'
9 # and assign to the static route of ('about')
10
11 # Create a function named error which returns a formatted string '<h1>Either you did not provide the correct path'
12 # and assign to the static route of ('error')
13
14 # Create a function named hello which returns a string of '<h1>Hello, World!'
15 # and assign to the static route of ('/hello')
16
17 # Create a function named admin which redirect the request to the error path
18 # and assign to the route of ('/admin')
19
20 # Create a function named greet which return formatted inline html string
21 # and assign to the dynamic route of ('/<name>')
22
23 # Create a function named greet_admin which redirect the request to the hello
24 # and assign to the route of ('/greet-admin')
25
26 # Rewrite a function named greet which which uses template file named 'greet.html'
27 # and assign to the dynamic route of ('/<name>')
28
29 # Create a function named list10 which creates a list counting from 1 to 10 with numbers in string format
30 # and assign to the route of ('/list10')
31
32 # Create a function named evens which show the even numbers from 1 to 10 with numbers in string format
33 # and assign to the route of ('/evens')
34
35 # Add a statement to run the Flask application which can be reached from any path
36
37 # Write a template html file named 'greet.html' which takes 'name' as parameter and uses the format '<h1>Hello, <name>!'
38
39 # Write a template html file named 'list10.html' which shows a list counting from 1 to 10 with numbers in string format
40
```

Yani actigimiz app.py dosyasina asagidaki gorseli yaziyoruz

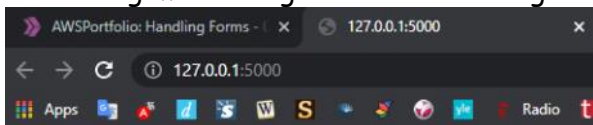


```
1 # Import Flask modules
2 from flask import Flask, render_template, redirect, url_for
3 # Create an object named app
4 app = Flask(__name__)
5 # Create a function named home which returns a string 'This is home page for no path'
6 # and assign route of no path ('/')
7 @app.route('/')
8 def home():
9     return 'This is home page for no path, <h1> Welcome Home</h1>'
10 # Create a function named about which returns a formatted string '<h1>This is about page for no path'
11 # and assign to the static route of ('about')
12
```



```
44
45 # Create a folder named 'static' under 'hands-on/flask-03-handling-routes-and-if-for'
46
47 # run this app in debug mode on your local. Do not forget to change debug mode to True
48 if __name__ == '__main__':
49     app.run(debug = True)
```

Calistirdigimizda asagidaki sonucu alacagiz



This is home page for no path,

Welcome Home

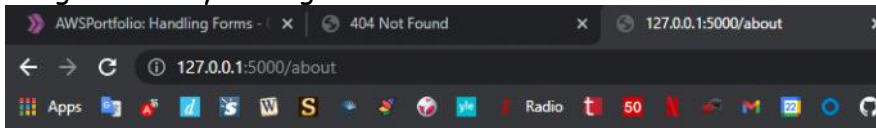
Gorseli uygulayalim


```

7  @app.route('/')
8  def home():
9      return 'This is home page for no path, <h1> Welcome Home</h1>'
10 # Create a function named about which returns a formatted string '<h1>This
11 # and assign to the static route of ('about')
12 @app.route('/about')
13 def about():
14     return '<h1>This is my about page </h1>'
15 # Create a function named error which returns a formatted string '<h1>Eit

```

Asagidaki ciktiyi alacagiz



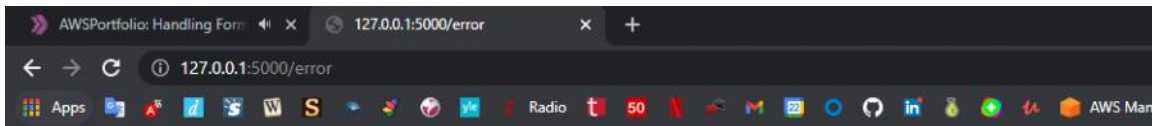
This is my about page

Gorselleri sirayla uygulayalim ve asagidaki ciktilari alacagiz

```

# and assign to the static route of ('error')
@app.route('/error')
def error():
    return '<h1>Either you encountered an error or you are not authorized.</h1>'
# Create a function named hello which returns a string of '<h1>Hello, World! <

```

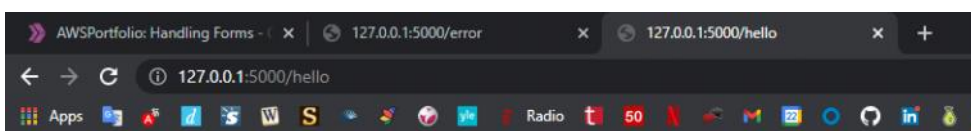


Either you encountered an error or you are not authorized.

```

19     return '<h1>Either you encountered an error or you are not authorized
20 # Create a function named hello which returns a string of '<h1>Hello, Wor
21 # and assign to the static route of ('/hello')
22 @app.route('/hello')
23 def hello():
24     return f'<h1>Hello, World! </h1>'
25 # Create a function named admin which redirect the request to the error p

```



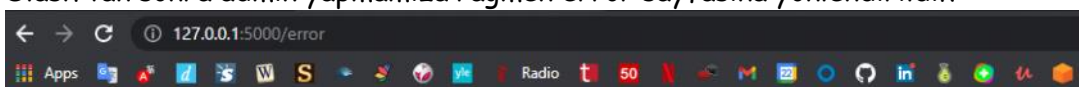
Hello, World!

```

# and assign to the route of ('/admin')
@app.route('/admin')
def admin():
    return redirect(url_for('error'))
# Create a function named greet which return format

```

Slash tan sonra admin yapmamiza ragmen error sayfasina yonlendirildik



Either you encountered an error or you are not authorized.

```

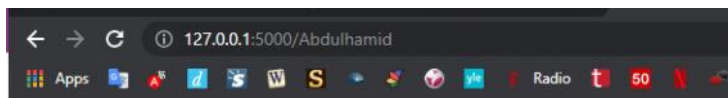
30 # Create a function named greet which return formatted inline html string
31 # and assign to the dynamic route of ('/<name>')
32 @app.route('/<name>')
33 def greet(name):
34     greet_format= f"""
35     <!DOCTYPE html>
36     <html>
37     <head>
38         <title>Greeting Page</title>
39     </head>
40     <body>
41         <h1>Hello, { name }!</h1>
42         <h1>Welcome to my Greeting Page</h1>
43     </body>
44     </html>
45     """
46     return greet_format

```



Hello, !

Welcome to my Greeting Page



Hello, Abdulhamid!

Welcome to my Greeting Page

```

f"""
<!DOCTYPE html>
<html>
<head>
<title>Greeting Page</title>
</head>
<body>
<h1>Hello, { name }!</h1>
<h1>Welcome to my Greeting Page</h1>
</body>
</html>
"""

```

templates klasorunun altina greet.html dosyasi olusturuyoruz. Asagidaki metni yapistiriyoruz.

```

<!DOCTYPE html>
<html>
<head>
<title>Greeting Page</title>
</head>
<body>
<h1>Hello, {{ name }}!</h1>
<h1>Welcome to my Greeting Page</h1>
</body>
</html>

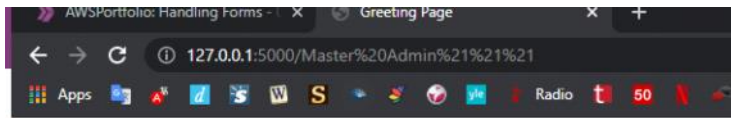
```

App.py kismina goruntudekiler yaziliyor.

```

47 # Create a function named greet_admin which redirect the request to the hell
48 # and assign to the route of ('/greet-admin')
49 @app.route('/greet-admin')
50 def greet_admin():
51     return redirect(url_for('greet', name = 'Master Admin!!!'))
52 # Rewrite a function named greet which which uses template file named 'greet

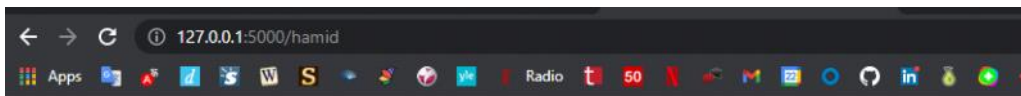
```



Hello, Master Admin!!!!

Welcome to my Greeting Page

```
52 # Rewrite a function named greet which uses template file named `greet.html`
53 # and assign to the dynamic route of `/{<name>}`
54 @app.route('/<name>')
55 def greeting(name):
56     return render_template('greet.html', name=name)
57 # Create a function named list10 which creates a list counting from 1 to 10 with
```



Hello, hamid!

Welcome to my Greeting Page

Templates icerisine list10.html ve evens.html adinda iki file olusturduk ve asagidakileri icerlerine attik

LIST10

```
<!DOCTYPE html>
<html>
<head>
  <title>List 10</title>
</head>
<body>
  <h1>Created 10 List Items</h1>
  <ul>
    {% for x in range(1,11) %}
    <li>List item {{ x }} </li>
    {% endfor %}
  </ul>
</body>
</html>
```

EVENS

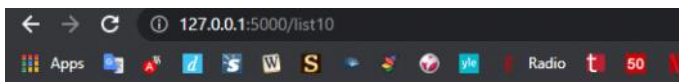
```
<!DOCTYPE html>
<html>
<head>
  <title>List Evens</title>
</head>
<body>
  <h1>Showing Even Number from 1 to 10</h1>
  <ul>
    {% for x in range(1,11) %}
    {% if x % 2 == 0 %}
    <li>Number {{ x }} is even</li>
    {% endif %}
    {% endfor %}
  </ul>
</body>
</html>
```



```

56 |     return render_template('greet.html', name = name)
57 | # Create a function named list10 which creates a list counting from 1 to 10
58 | # and assign to the route of ('/list10')
59 | @app.route('/list10')
60 | def list10():
61 |     return render_template('list10.html')
62 | # Create a function named evens which show the even numbers from 1 to 10 with

```



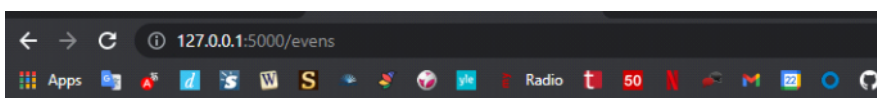
Created 10 List Items

- List item 1
- List item 2
- List item 3
- List item 4
- List item 5
- List item 6
- List item 7
- List item 8
- List item 9
- List item 10

```

62 | # Create a function named evens which show the even numbers from 1 to 10 with
63 | # and assign to the route of ('/evens')
64 | @app.route('/evens')
65 | def evens():
66 |     return render_template('evens.html')
67 | # Add a statement to run the Flask application which can be reached from any h

```



Showing Even Number from 1 to 10

- Number 2 is even
- Number 4 is even
- Number 6 is even
- Number 8 is even
- Number 10 is even

Yazdigimiz app.py leri EC2 da calistiracagiz. Her iki app.py dosyasi sonunu gorseldeki gibi yazacagiz.

```

18 | # run this app in debug mode on your local.
19 | if __name__ == '__main__':
20 |     #app.run(debug=True)
21 |     app.run(host='0.0.0.0', port=80)

```

Yaptigimiz butun degisiklileri github hesabimiza push ediyoruz.

Ec2 makinesi aciyoruz

Key	Value
name	flask_test

Add another tag (Up to 50 tags maximum)

80 ve 22 ye izin veren bir port secilecek

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
HTTP	TCP	80	0.0.0.0/0
HTTP	TCP	80	::/0
SSH	TCP	22	0.0.0.0/0
SSH	TCP	22	::/0

Search for services, features, marketplace products, and docs [Alt+S]							
Instances (1/1) Info							
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability	
flask_test	i-09642fd20ddc27955	Pending	t2.micro	-	No alarms	us-east-1e	

Ssh kopyalanıyor

Connect to instance Info

Connect to your instance i-09642fd20ddc27955 using any of these options

EC2 Instance Connect | Session Manager | **SSH client** | EC2 Serial Console

Instance ID
i-09642fd20ddc27955

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is EC2_key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 EC2_key.pem
4. Connect to your instance using its Public DNS:
ec2-54-162-49-69.compute-1.amazonaws.com

Command copied

ssh -i "EC2_key.pem" ec2-user@ec2-54-162-49-69.compute-1.amazonaws.com

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Cancel

Pc de key.pm lerin bulunduđu dizine git bash den geciyoruz ve cihazı calıstırıyoruz

```
hamid@LAPTOP-U8P8504G MINGW64 ~/ssh
$ ssh -i "EC2_key.pem" ec2-user@ec2-54-162-49-69.compute-1.amazonaws.com
The authenticity of host 'ec2-54-162-49-69.compute-1.amazonaws.com (54.162.49.69)'
can't be established.
ED25519 key fingerprint is SHA256:hV5uDsQGAnONpdSVh1S2c4JDhwjVSP+UTpQ6dI4XwIY.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-162-49-69.compute-1.amazonaws.com' (ED25519) to
the list of known hosts.

 _ _ | _ _ | _ )
 _ | ( _ _ | / Amazon Linux 2 AMI
 _ _ | \ _ _ | _ _ |

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-53-66 ~]$ ^C
[ec2-user@ip-172-31-53-66 ~]$
```

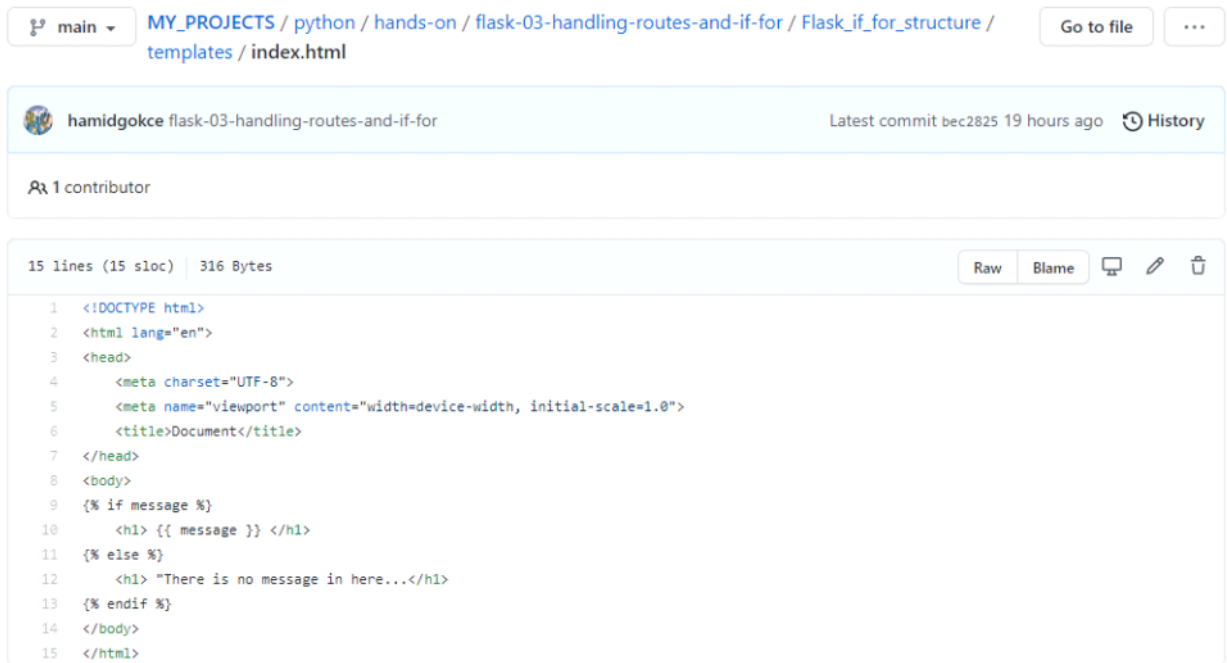
Sırayla asagıdaki komutları yazıyoruz. (bas harfler küçük)

- Sudo yum update -y
- Sudo yum install python3 -
- Python3 --version
- Sudo pip3 install flask
- Pip3 list
- Mkdir iffor
- Mkdir routing

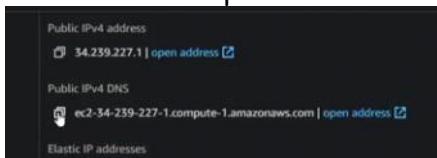
```
[ec2-user@ip-172-31-53-66 ~]$ ls
iffor routing
[ec2-user@ip-172-31-53-66 ~]$
```

- Iffor icerisinde templates klasoru olustur

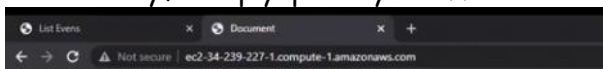
- Asagida bulunan komuta gidip raw i tikliyoruz ve acilan pencerenin linkini kopyalıyoruz
- Wget link(kopyaladigimiz) yazip enter



- Yukaridaki islemi body.html icinde yapiyoruz
- Ihtiyacimiz olan app.py dosyasi icin bir ust klasore cikip raw ve wget islemini tekrar uyguluyoruz
- Python klasorunun bulunduğu dizinde 'sudo python3 app.py' yazip enter
- Ec2 makinasinin public IP4 DNS adresini kopyalıyoruz



- Yeni bir sayfa acip yapistiriyoruz. / ve sonrasini deneyebiliriz



This is my first conditions experience

- Routing klasorune geliyoruz

```

[ec2-user@ip-172-31-8-44 ~]$ ls
iffor routing
[ec2-user@ip-172-31-8-44 ~]$ cd |

```

- Komutlari sirayla uygulayabiliriz. Yukaridaki islemi tekrarlayacagiz

```

[ec2-user@ip-172-31-8-44 ~]$ cd routing/
[ec2-user@ip-172-31-8-44 routing]$ mkdir templates
[ec2-user@ip-172-31-8-44 routing]$ cd templates/
[ec2-user@ip-172-31-8-44 templates]$ |

```

- List10, evens ve greet.html ye uygulayip bir ust klasordeki app.py yi aliyoruz
- Sudo python3 app.py yazip internet sayfamizi aciyoruz
- Sudoyum install tree
- Tree ==> enter

```
[ec2-user@ip-172-31-8-44 ~]$ tree
.
├── iffor
│   ├── app.py
│   └── templates
│       ├── body.html
│       └── index.html
└── routing
    ├── app.py
    └── templates
        ├── evens.html
        ├── greet.html
        └── list10.html

4 directories, 7 files
[ec2-user@ip-172-31-8-44 ~]$
```

- Yeni bir bash acip `curl -v public IP4 DNS` yapistirip calistirdigimizda makinamizin dogru bir sekilde calistigini gorebiliyoruz.

```
lenovo@vincenzo MINGW64 /
$ curl -v ec2-34-239-227-1.compute-1.amazonaws.com
* Trying 34.239.227.1:80...
* Connected to ec2-34-239-227-1.compute-1.amazonaws.com (34.239.227.1) port 80 (#0)
> GET / HTTP/1.1
Host: ec2-34-239-227-1.compute-1.amazonaws.com
User-Agent: curl/7.73.0
Accept: */*

* Mark bundle as not supporting multiuse
* HTTP 1.0, assume close after body
< HTTP/1.0 200 OK
< Content-Type: text/html; charset=utf-8 I
< Content-Length: 257
< Server: Werkzeug/2.0.1 Python/3.7.9
< Date: Tue, 22 Jun 2021 20:00:07 GMT
<
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
```

