Python WebServer With Flask and Raspberry Pi

How to setup a simple WebServer using FLASK and Python.

Installing FLASK

The first thing to do is to install Flask on your Raspberry Pi. Go to Terminal and enter:

```
sudo apt-get install python3-flask
```

Folder structur

Then create a folder (*flaskdemo*) where to have your files organized.

```
mkdir flaskdemo
```

Create 2 sub-folders:

- static *for CSS files*
- templates *for HTML files*

The final folder *tree* will be:

```
/flaskdemo
    /static
    /templates
```

Python WebServer Application

Create the Python *WebServer* with Flask.

- Create a file (*app.py*) in the *flaskdemo* folder
- Insert this code

```
    from flask import Flask

    app = Flask(__name__)

    @app.route('/')
    def index():
        return 'Hello 3. semester'
```

```
    if __name__ == '__main__':
        app.run(debug=True, port=5000, host='0.0.0.0')
```

Lets break it down

1. Load the Flask module into your Python script:

```
    from flask import Flask
```

2. Create a Flask object called app:

```
    app = Flask(__name__)
```

3. Run the **index()** function when someone accesses the root URL (*/*) of the server. In this case, only send the text *Hello World!* to the web browser thrue *return*

```
    def index():
    return "Hello Word"
```

4. Once this script is running from the command line at the terminal, the server starts to *listen* on port *5000*, reporting any errors (*debug=True*):

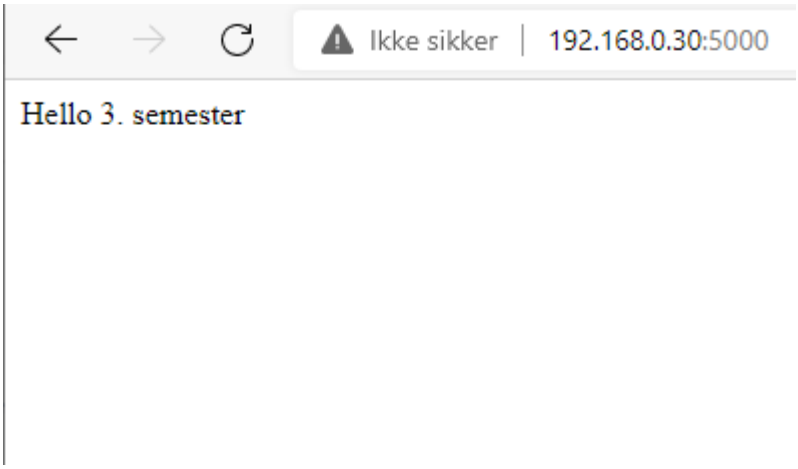
```
    if __name__ == '__main__':
    app.run(debug=True, port=5000, host='0.0.0.0')
```

Run the application - app.py

```
python3 app.py
```

Point your browser to the IP address of the Raspberry PI and port *5000*

```
http://raspberryip_ip:5000
```



index.html

Create a HTML template and a CSS file for styling you page.

This is, in fact, important, otherwise, you would complicate the Python Script putting all on it.

Create a file named **index.html**, save it in the **/templates** folder.

```
<!DOCTYPE html>
   <head>
      <title>{{ time }}</title>
      <link rel="stylesheet" href="../static/style.css/">
   </head>
   <body>
      <h1>Sense Hat Output</h1>
      <h2>The temperatur is: {{ temperatur }}</h2>
      <h2>The pressure is: {{ pressure }}</h2>
      <h2>The humidity is: {{ humidity }}</h2>
   </body>
</html>
```

Anything in double curly braces within the HTML template is interpreted as a **variable** that would be passed to it from the Python script via the **render_template** function.

Create a new Python script - **senesdata.py**

```
from flask import Flask, render_template
import datetime
from sense_hat import SenseHat

sense = SenseHat()

app = Flask(__name__)

@app.route("/")
def templetdata():
    now = datetime.datetime.now()
```

```
    timeString = now.strftime("%d-%m-%Y %H:%M")
    temperaturString = round(sense.get_temperature(),1)
    pressureString = round(sense.get_pressure(),1)
    humidityString = round(sense.get_humidity(),1)

    templateData = {
        'time': timeString,
        'temperatur': temperaturString,
        'pressure': pressureString,
        'humidity': humidityString
    }
    return render_template('index.html', **templateData)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000, debug=True)
```

Execute the Python script

```
python3 senesdata.py
```

Point your browser to the IP address of the Raspberry PI and port *5000*

```
http://raspberryip_ip:5000
```

CSS styling

Include some styling on our page, creating a **CSS** file and stored it on **/static/style.css**

```
body {
    background: blue;
}
h1 {
    color:white;
}
```

Modify the index.html file to inform it to look for the style.css file. You do this inserting the *link* at *head*

```
<!DOCTYPE html>
   <head>
      <title>Sense Data</title>
      <link rel="stylesheet" href="../static/style.css/">
   </head>

   <body>
        <h1>Date and time: {{ time }}</h1>
        <br>
        <h2>Sense Hat Output</h2>
        <h2>The temperatur is: {{ temperatur }}</h2>
        <h2>The pressure is: {{ pressure }}</h2>
        <h2>The humidity is: {{ humidity }}</h2>
   </body>
</html>
```

Execute the Python script

```
python3 senesdata.py
```

Point your browser to the IP address of the Raspberry PI and port *5000*

```
http://raspberryip_ip:5000
```

Creating multiple routes

To add a new route, simply call the *route()* function again with the desired path and create a view function for it - here the:

```
@app.route("/database")
def databasedata():
```

Inside that you have the database connection and the **SELECT** from tha table:

```
conn = mysql.connector.connect(
    host="localhost",
    user="pi",
    password="YouMySQLPassword",
    database="sensedata"
    )

    mycursor = conn.cursor()
    mycursor.execute("SELECT * FROM hatdata")
    data = mycursor.fetchall()
```

Final senesdata.py file

```
from flask import Flask, render_template
import datetime
from sense_hat import SenseHat
import mysql.connector

# Sense data
sense = SenseHat()

# Flask
app = Flask(__name__)

@app.route("/")
def templetdata():
    now = datetime.datetime.now()
    timeString = now.strftime("%d-%m-%Y %H:%M")
    temperaturString = round(sense.get_temperature(),1)
    pressureString = round(sense.get_pressure(),1)
    humidityString = round(sense.get_humidity(),1)

    templateData = {
        'time': timeString,
        'temperatur': temperaturString,
        'pressure': pressureString,
        'humidity': humidityString
    }
    return render_template('index.html', **templateData)

@app.route("/database")
def databasedata():
    try:
        conn = mysql.connector.connect(
            host="localhost",
            user="pi",
            password="YouMySQLPassword",
            database="sensedata"
```

```
        )

        mycursor = conn.cursor()
        mycursor.execute("SELECT * FROM hatdata")
        data = mycursor.fetchall()

        return render_template("database.html", data=data)

    except Exception as e:
        return (str(e))

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=5000, debug=True)
```

You have to create the html file **database.html**. To show the data you have to "run" true all the rows in the SELECT, it is done in a *for loop*

```
{% for row in data %}
    <tr>
    {% for d in row %}
        <td>{{ d }}</td>
    {% endfor %}
    </tr>
{% endfor %}
```

The data are shown in a HTML table.

Final database.html

```
<!DOCTYPE html>
   <head>
      <title>Database</title>
      <link rel="stylesheet" href="../static/style.css/">
   </head>

   <body>
        <h1>Database info</h1>
        <br>
        <table border="1" cellpadding="5" cellspacing="5">
            {% for row in data %}
                <tr>
                {% for d in row %}
                    <td>{{ d }}</td>
                {% endfor %}
                </tr>
            {% endfor %}
        </table>
   </body>
</html>
```

Database info

1	2021-08-06 13:47:51	34.92308044433594	38.67486572265625	0.0
2	2021-08-06 13:47:56	34.903846740722656	38.48683547973633	1008.412353515625
3	2021-08-06 13:48:01	34.96154022216797	38.56541442871094	1008.391845703125
4	2021-08-06 13:48:06	34.88461685180664	38.32125473022461	1008.3671875
5	2021-08-06 13:48:11	34.846153259277344	38.478416442871094	1008.3564453125
6	2021-08-06 13:50:07	34.96154022216797	38.2398681640625	1008.40087890625
7	2021-08-06 13:51:35	35.17308044433594	38.077091217041016	1008.419921875
8	2021-08-06 13:54:50	35.67308044433594	37.03870391845703	1008.384765625
9	2021-08-06 13:56:19	35.42308044433594	38.10515594482422	1008.366455078125
10	2021-08-06 13:56:37	35.596153259277344	37.61122131347656	1008.32763671875
11	2021-08-06 13:56:56	35.5384635925293	37.3193473815918	1008.32958984375
12	2021-08-06 13:57:14	35.653846740722656	37.3782844543457	1008.37109375
13	2021-08-06 13:57:33	35.75	37.209896087646484	1008.32421875
14	2021-08-06 13:57:52	35.615386962890625	37.20147705078125	1008.37109375
15	2021-09-14 16:46:40	27.865385055541992	49.76318359375	0.0
16	2021-09-14 16:46:56	27.846155166625977	50.46479797363281	1019.104736328125
17	2021-09-14 16:47:14	27.903846740722656	50.53215408325195	1019.09912109375

Finaly structure

You have one folders with 2 subfolders and 4 files, *1 Python, 1 css and 2 html*.

```
/flaskdemo
    senesdata.py

    /static
        style.css
```

```
    /templates
        index.html
        database.html
```