

LIA1 - TH1NG AB

Uppgift nr2: Bygg en enkel IoT-server med Flask

- **Mål:** Skapa en lokal server på din dator som tar emot och visar sensordata.
- **Verktyg:** Python och Flask (ett lättviktigt webbramverk för Python).
- **Steg:**
 1. Installera Flask om du inte redan har det (pip install Flask).
 2. Skapa en Flask-app som tar emot simulerad sensordata via POST-anrop (t.ex. temperatur och luftfuktighet).
 3. Bygg en enkel webbsida där du visar datan som tas emot.
- **Vad du lär dig:** Webbutveckling för IoT, REST API-bygge och grundläggande serverkommunikation.

STEG:

Vi kan börja med att ge en kort introduktion om vad "Flask" är för något. Flask är ett lättviktigt webbramverk för Python som gör det enkelt att bygga webbapplikationer och REST API, den ger oss möjligheten att skapa en server som kan ta emot och svara på förfrågningar över HTTP-protokollet tillskillnad från den vanliga Web API som förlitar sig på flera kommunikationsprotokoll som SOAP, XML-RPC och JSON-RPC.

-

Börja med att öppna terminalen på din dator och installera Flask, kör följande kommando → pip install Flask

-

Skapa en mapp på din dator och namnge den "IoT-Server_Flask"

Öppna Visual Studio Code (bifogar en länk om man inte har det) → <https://code.visualstudio.com/download>

Gå till fliken "extensions" och installera "python" & "HTML"

Öppna din mapp i Visual Studio Code och skapa en ny fil, namnge den för app.py (koden finns att hämta på min GitHub)

-

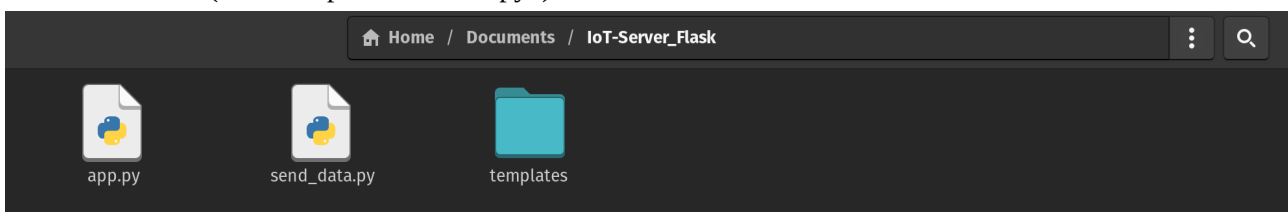
Öppna din mapp "IoT-Server_Flask" och skapa en ny mapp där, namnge den "templates"

I visual studio code skapa en ny fil och namnge den, index.html (OBS!! Denna fil ska ligga i templates mappen)

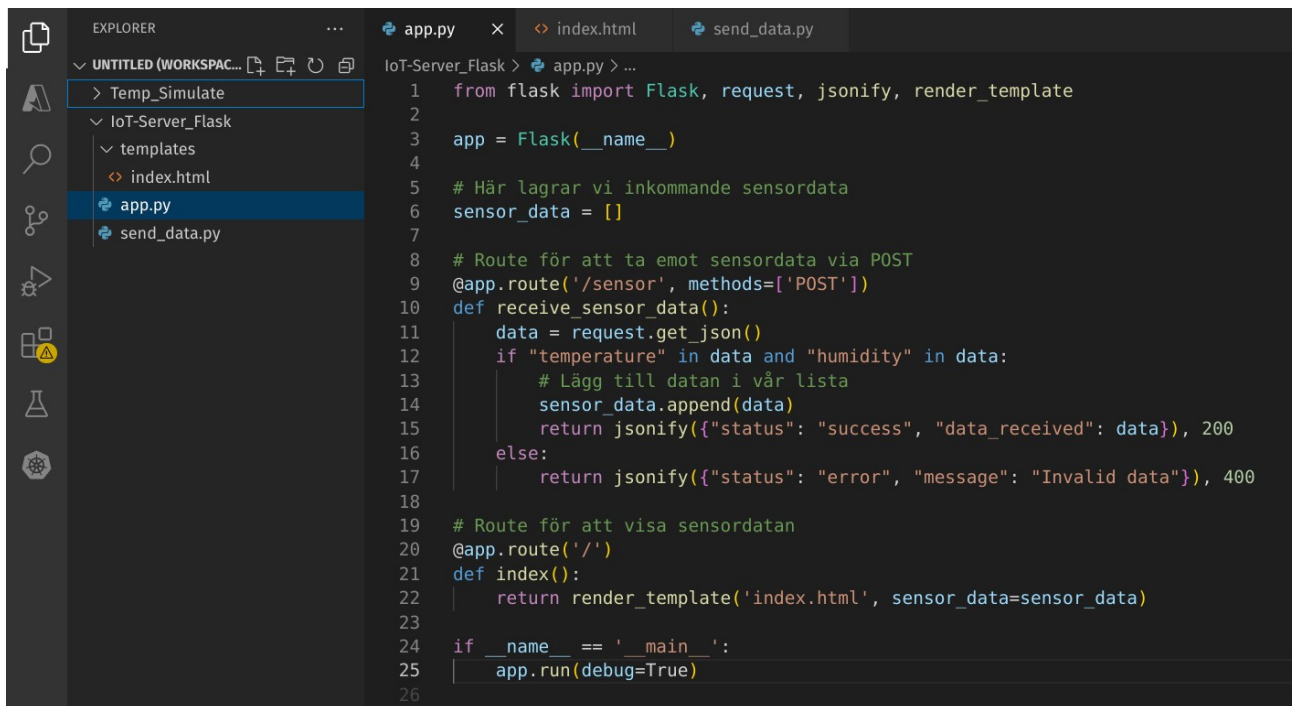
```
/IoT-Server_Flask
├── app.py
└── templates
    └── index.html
```

-

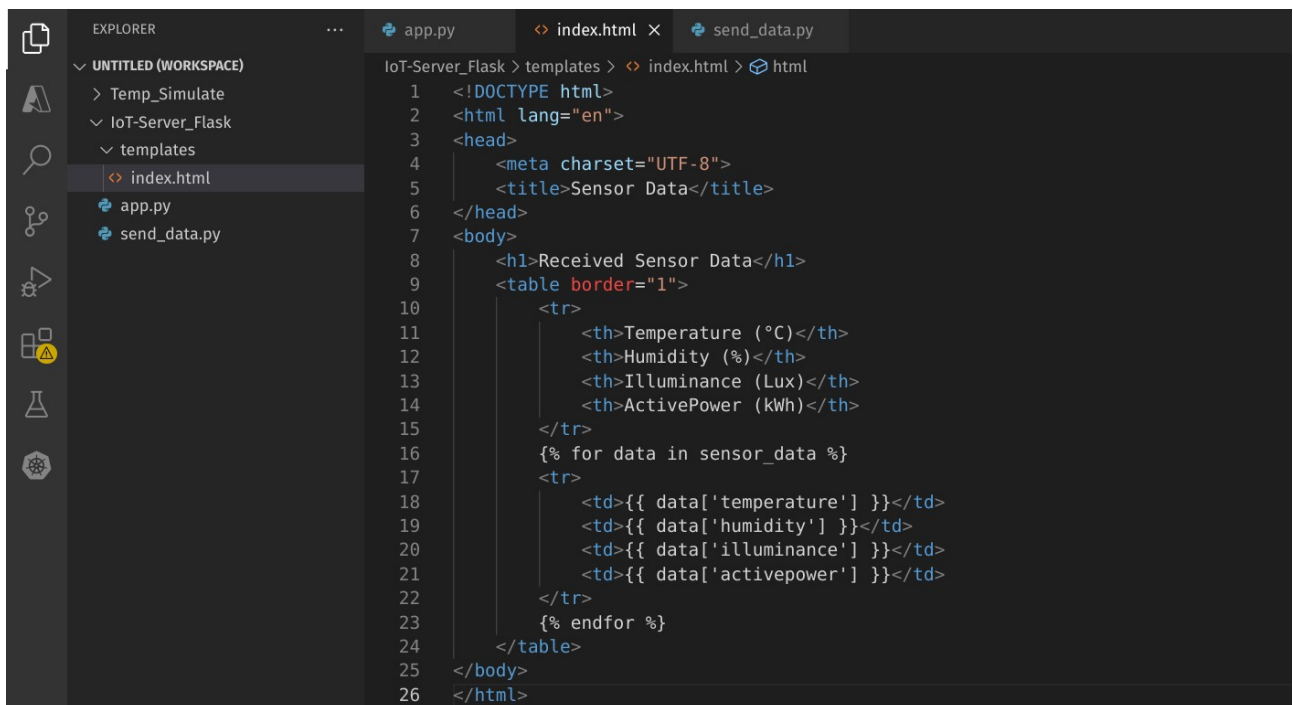
Så bör det se ut (tänk inte på "send_data.py")



Om du har gjort allt rätt i Visual Studio Code bör du ha samma vy som mig:



```
1 from flask import Flask, request, jsonify, render_template
2
3 app = Flask(__name__)
4
5 # Här lagrar vi inkommande sensordata
6 sensor_data = []
7
8 # Route för att ta emot sensordata via POST
9 @app.route('/sensor', methods=['POST'])
10 def receive_sensor_data():
11     data = request.get_json()
12     if "temperature" in data and "humidity" in data:
13         # Lägg till datan i vår lista
14         sensor_data.append(data)
15         return jsonify({"status": "success", "data_received": data}), 200
16     else:
17         return jsonify({"status": "error", "message": "Invalid data"}), 400
18
19 # Route för att visa sensordatan
20 @app.route('/')
21 def index():
22     return render_template('index.html', sensor_data=sensor_data)
23
24 if __name__ == '__main__':
25     app.run(debug=True)
26
```



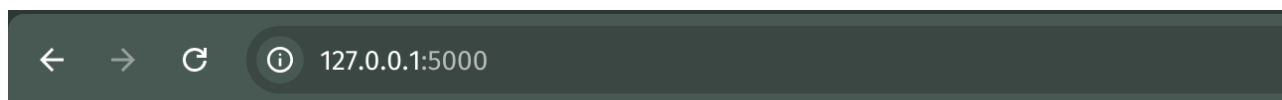
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Sensor Data</title>
6 </head>
7 <body>
8     <h1>Received Sensor Data</h1>
9     <table border="1">
10         <tr>
11             <th>Temperature (°C)</th>
12             <th>Humidity (%)</th>
13             <th>Illuminance (Lux)</th>
14             <th>ActivePower (kWh)</th>
15         </tr>
16         {% for data in sensor_data %}
17         <tr>
18             <td>{{ data['temperature'] }}</td>
19             <td>{{ data['humidity'] }}</td>
20             <td>{{ data['illuminance'] }}</td>
21             <td>{{ data['activepower'] }}</td>
22         </tr>
23         {% endfor %}
24     </table>
25 </body>
26 </html>
```

-
Nu kan du öppna en terminal i Visual Studio Code och köra följande kommando → `python3 app.py`

Då kommer följande text upp (som på bilden), detta betyder att din Flask-server startade korrekt !

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE
fabriziomugni@fabrizios-dator:~/Documents/IoT-Server_Flask$ python3 app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 133-192-538
```

-
Öppna en webbläsare på din dator och skriv följande → <http://127.0.0.1:5000/>
Detta borde visa "index.html" med en tom tabell (om inga sensordata ännu har skickats).



Received Sensor Data

Temperature (°C)	Humidity (%)	Illuminance (Lux)	ActivePower (kWh)
------------------	--------------	-------------------	-------------------

-
Skapa en ny fil på "IoT-Server_Flask" och namge den "send_data.py" (koden finns på min GitHub)

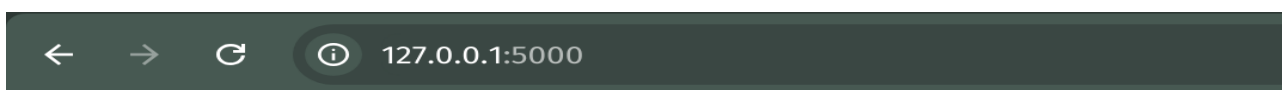
Öppna en ny terminal utan att stänga den första terminalen vi öppnade !
Kör följande kommando → `python3 send_data.py`

Detta kommer att skicka ett POST-anrop till din Flask-server och skriva ut svaret i terminalen.

-
Såhär bör det se ut för dig om du har gjort allt rätt !

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 133-192-538
127.0.0.1 - - [05/Nov/2024 15:33:09] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [05/Nov/2024 15:42:13] "POST /sensor HTTP/1.1" 200 -

fabriziomugni@fabrizios-dator:~/Documents/IoT-Server_Flask$ python3 send_data.py
{'data_received': {'activepower': 0.9, 'humidity': 60, 'illuminance': 15.0, 'temperature': 22.5}, 'status': 'success'}
```



Received Sensor Data

Temperature (°C)	Humidity (%)	Illuminance (Lux)	ActivePower (kWh)
22.5	60	15.0	0.9

-

Om man nu vill utveckla projektet vidare, här är några idéer:

- **Lägg till fler sensorer:** Utöka med andra typer av sensordata, som lufttryck eller ljusstyrka, och anpassa koden för att ta emot och visa dem.
- **Automatisk uppdatering:** Använd JavaScript för att automatiskt uppdatera sidan när ny data kommer in, utan att behöva ladda om.
- **Databaslagring:** Istället för att lagra data i en lista kan du använda en databas, som SQLite, för att spara data mer permanent.
- **Visualisering:** Lägg till grafer för att visualisera sensordatan över tid.

Lycka till ! :)