

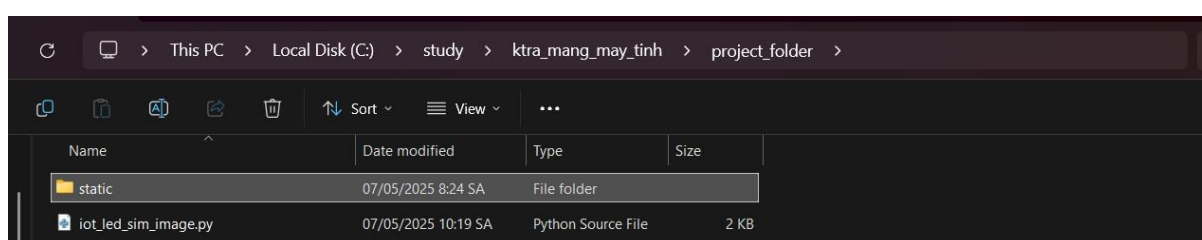
Hoàng Sỹ Việt– 22174600094 – DHKL16A1HN

BÁO CÁO

MẠNG MÁY TÍNH VÀ TRUYỀN SỐ LIỆU

BÀI LAB 1.2: MÔ PHỎNG ĐIỀU KHIỂN LED QUA HTTP BẰNG FRAMEWORK FLASK

Sau khi tạo thư mục có cấu trúc sau:



Sau khi chạy file code `iot_led_sim_image` ta có kết quả sau:

```
iot_led_sim_image.py > ...
17 <br><br>
18 <a href="/?led=on"><button style="padding:10px 20px;">Turn ON</button></a>
19 <a href="/?led=off"><button style="padding:10px 20px;">Turn OFF</button></a>
20 </body>
21 </html>
22 """
23
24 @app.route('/')
25 def index():
26     led = request.args.get("led")
27     if led == "on":
28         led_state["value"] = True
29     elif led == "off":
30         led_state["value"] = False
31
32     image_file = "led_on_white_120.png" if led_state["value"] else "led_off_white_120.png"
33     return render_template_string(
34         HTML_TEMPLATE,
35         state="ON" if led_state["value"] else "OFF",
36         image_file=image_file
37     )
38
39 if __name__ == '__main__':
40     app.run(host='0.0.0.0', port=5000)
```

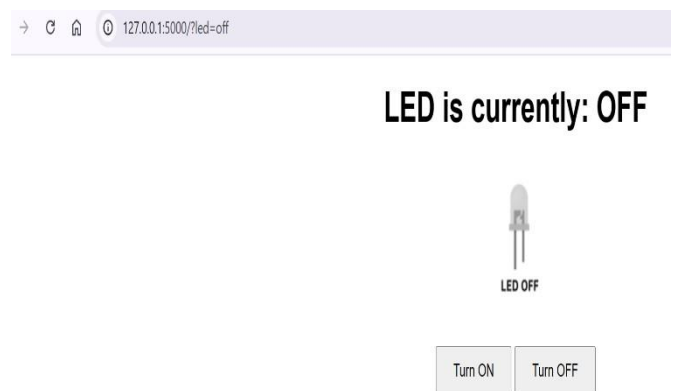
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULTS (PREVIEW)
* Serving Flask app 'iot_led_sim_image'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.30.100:5000
Press CTRL+C to quit
```

Kết quả chạy trên cửa sổ CMD:

```
Microsoft Windows [Version 10.0.26100.3915]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hsvie>python C:\study\ktra_mang_may_tinh\project_folder\iot_led_sim_image.py
* Serving Flask app 'iot_led_sim_image'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.30.100:5000
Press CTRL+C to quit
127.0.0.1 -- [14/May/2025 08:29:11] "GET / HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:11] "GET /static/led_off_white_120.png HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:12] "GET / HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:12] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 -- [14/May/2025 08:29:13] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:13] "GET /static/led_on_white_120.png HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:13] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /?led=on HTTP/1.1" 200 -
```

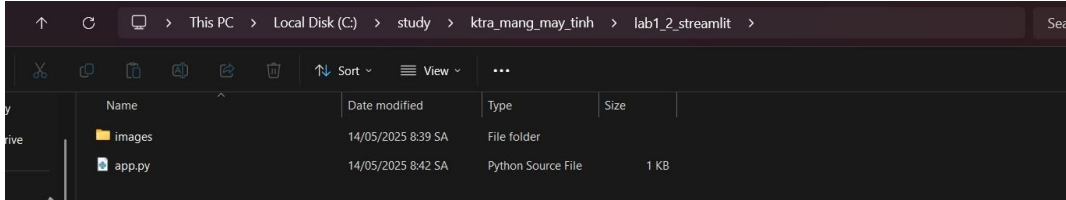
Quan sát giao diện web hiển thị trạng thái LED và các nút điều khiển:



```
Press CTRL+C to quit
127.0.0.1 -- [14/May/2025 08:29:11] "GET / HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:11] "GET /static/led_off_white_120.png HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:12] "GET / HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:12] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 -- [14/May/2025 08:29:13] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:13] "GET /static/led_on_white_120.png HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:13] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:14] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /?led=on HTTP/1.1" 200 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /static/led_on_white_120.png HTTP/1.1" 304 -
127.0.0.1 -- [14/May/2025 08:29:15] "GET /?led=on HTTP/1.1" 200 -
```

BÀI LAB 1.2 PHIÊN BẢN MÔ PHỎNG ĐIỀU KHIỂN LED QUA HTTP BẰNG STREAMLIT

Dưới đây là cấu trúc thư mục được tạo để tiến hành thực hiện bài tập thực hành:



Sau khi chạy đoạn mã app.py ta có kết quả sau:

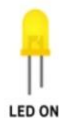
```
app.py X
lab1_2_streamlit > app.py > ...
1 import streamlit as st
2 from PIL import Image
3 # Trạng thái LED mô phỏng
4 if "led_on" not in st.session_state:
5     st.session_state.led_on = False
6 # Xử lý sự kiện nút trước khi render
7 col1, col2 = st.columns(2)
8 with col1:
9     if st.button("BẬT LED"):
10         st.session_state.led_on = True
11 with col2:
12     if st.button("TẮT LED"):
13         st.session_state.led_on = False
14 # Sau khi cập nhật trạng thái + render ảnh
15 st.title("Điều khiển LED mô phỏng bằng Streamlit")
16 if st.session_state.led_on:
17     image_file = "C:\\study\\ktra_mang_may_tinh\\lab1_2_streamlit\\images\\led_on_white_120.png"
18 else:
19     image_file = "C:\\study\\ktra_mang_may_tinh\\lab1_2_streamlit\\images\\led_off_white_120.png"
20 image = Image.open(image_file)
21 st.image(image, caption="LED is ON"
22         if st.session_state.led_on
23         else "LED is OFF", width=200)
```



BẬT LED

TẮT LED

Điều khiển LED mô phỏng bằng Streamlit



LED ON

LED is ON



BẬT LED

TẮT LED

Điều khiển LED mô phỏng bằng Streamlit



LED OFF

LED is OFF

Sau khi chạy trên cửa sổ CMD bằng lệnh ipconfig ta có kết quả sau:

```
Windows IP Configuration

Unknown adapter VPN111 - VPN Client:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Ethernet adapter vietdzai:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix  . : DHCP HOST
    Link-local IPv6 Address . . . . . : fe80::9576:812a:3206:7fa%14
    IPv4 Address. . . . . : 192.168.30.100
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.30.1
```

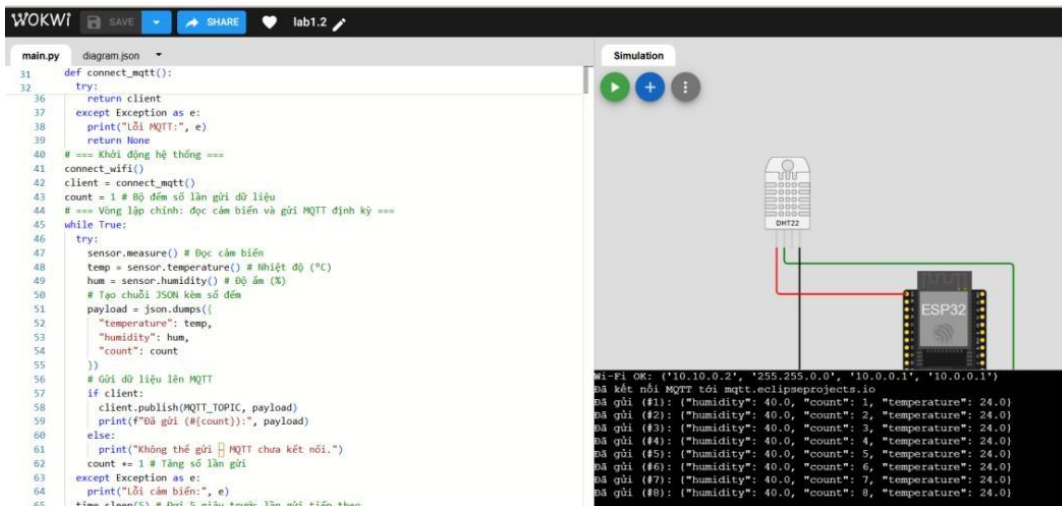
Ta có địa chỉ Ipv4 là: 192.168.30.100

LAB 2.2a – GỬI DỮ LIỆU CẢM BIẾN LÊN MQTT BROKER

Dưới đây là ESP32 mô phỏng (trên Wokwi) đọc dữ liệu từ cảm biến DHT22, kết nối Wifi và gửi kết quả dữ liệu như hình sau:

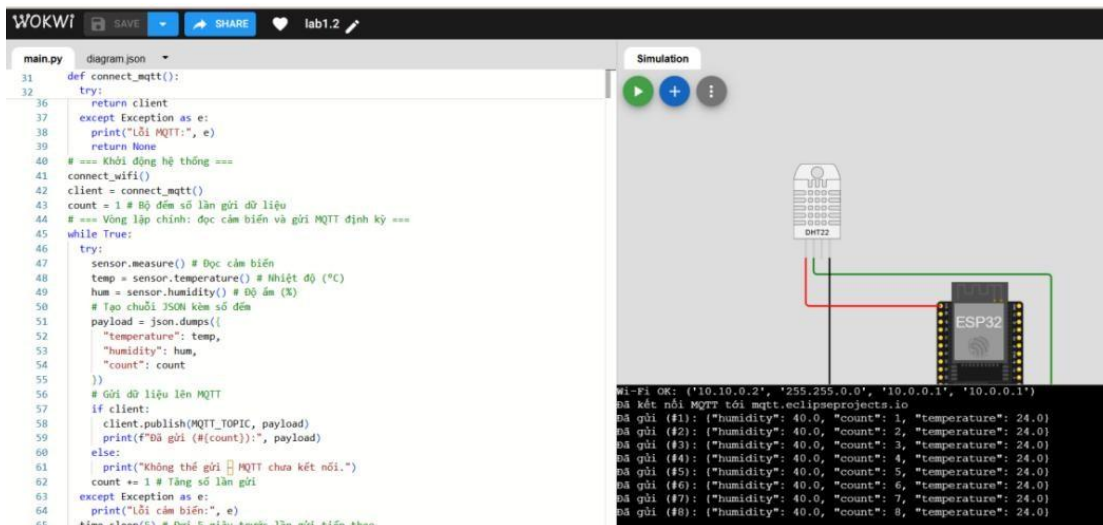
```
lab2.2 > subscriber.py > ...
1
2 import paho.mqtt.client as mqtt
3
4 def on_connect(client, userdata, flags, rc):
5     if rc == 0:
6         print("✅ Kết nối MQTT thành công!")
7         client.subscribe("iot/khdl/esp32")
8     else:
9         print("❌ Lỗi kết nối, mã lỗi:", rc)
10 def on_message(client, userdata, msg):
11     print(f"{msg.topic}: {msg.payload.decode()}")
12
13 client = mqtt.Client()
14
15 client.on_connect = on_connect
16 client.on_message = on_message
17
18 client.connect("broker.hivemq.com", 1883, 60)
19 client.loop_forever()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\mang_may_tinh> & "C:/Program Files/Python312/python.exe" d:/mang_may_tinh/lab2.2/subscriber.py
d:\mang_may_tinh\lab2.2\subscriber.py:13: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
client = mqtt.Client()
✅ Kết nối MQTT thành công!
```

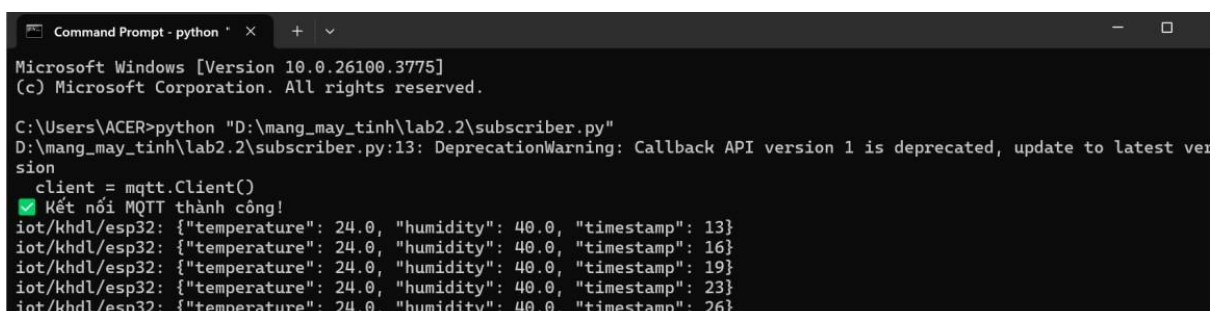


LAB 2.2b – SUBSCRIBE DỮ LIỆU CẢM BIẾN TỪ MQTT BẰNG PYTHON TRÊN PC

Dưới đây là mô phỏng ESP32+DHT22 trong wokwi:



Tiếp theo, ta sẽ thấy dòng JSON xuất hiện liên tục mỗi 5 giây khi mở terminal hoặc CMD và chạy lệnh: python subscriber.py



LAB 2.4: LƯU TRỮ VÀ TRỰC QUAN HÓA DỮ LIỆU IoT

Lab 2.4a: Mô phỏng với wokwi (DHT22+ESP 32)

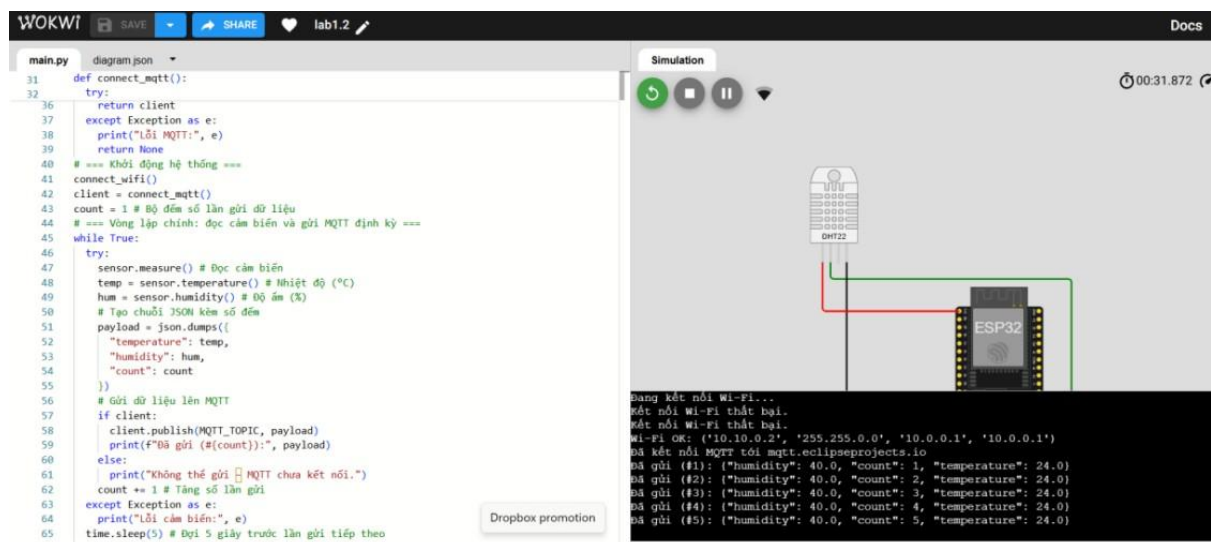
Kiểm thử Lab 2.4a

Bước 1: Khởi động Publisher

Chạy mô phỏng trong Wokwi với ESP32 + DHT22 (main.py đang publish JSON lên `iot/khdl/esp32`)

Đảm bảo mỗi 5 giây có 1 bản tin JSON gửi lên.

Dưới đây là kết quả:



The screenshot shows the Wokwi simulation environment. On the left, a Python script named `main.py` is displayed. It defines a `connect_mqtt()` function and a main loop that reads data from a DHT22 sensor and publishes it to an MQTT topic. The script includes error handling for MQTT connection failures and a 5-second delay between publishes. On the right, the simulation window shows a DHT22 sensor connected to an ESP32 microcontroller. Below the simulation, a terminal window displays the output of the script, showing the MQTT connection status and the JSON payloads being sent.

```
main.py
31 def connect_mqtt():
32     try:
33         return client
34     except Exception as e:
35         print("Lỗi MQTT:", e)
36         return None
37
38 # === Khởi động hệ thống ===
39 connect_wifi()
40 client = connect_mqtt()
41 count = 1 # Bộ đếm số lần gửi dữ liệu
42 # === Vòng lặp chính: đọc cảm biến và gửi MQTT định kỳ ===
43 while True:
44     try:
45         sensor.measure() # Đọc cảm biến
46         temp = sensor.temperature() # Nhiệt độ (°C)
47         hum = sensor.humidity() # Độ ẩm (%)
48         # Tạo chuỗi JSON kèm số đếm
49         payload = json.dumps({
50             "temperature": temp,
51             "humidity": hum,
52             "count": count
53         })
54         # Gửi dữ liệu lên MQTT
55         if client:
56             client.publish(MQTT_TOPIC, payload)
57             print(f"Đã gửi ({count}):", payload)
58         else:
59             print("Không thể gửi MQTT chưa kết nối.")
60         count += 1 # Tăng số lần gửi
61     except Exception as e:
62         print("Lỗi cảm biến:", e)
63     time.sleep(5) # Delay 5 giây trước lần gửi tiếp theo
```

Simulation

00:31.872

Đang kết nối Wi-Fi...
Kết nối Wi-Fi thất bại.
Kết nối Wi-Fi thất bại.
Wi-Fi OK: ('10.10.0.2', '255.255.0.0', '10.0.0.1', '10.0.0.1')
Đã kết nối MQTT tới mqtt.eclipseprojects.io
Đã gửi (#1): {"humidity": 40.0, "count": 1, "temperature": 24.0}
Đã gửi (#2): {"humidity": 40.0, "count": 2, "temperature": 24.0}
Đã gửi (#3): {"humidity": 40.0, "count": 3, "temperature": 24.0}
Đã gửi (#4): {"humidity": 40.0, "count": 4, "temperature": 24.0}
Đã gửi (#5): {"humidity": 40.0, "count": 5, "temperature": 24.0}

Bước 2: Kiểm tra dữ liệu hiển thị trong terminal

Chạy `iot_data_logger.py` Vào thư mục `~\lab2.4\` có chứa file `iot_data_logger.py` Gõ

lệnh: `python iot_data_logger.py`

Quan sát log hiển thị: Dữ liệu nhận được: timestamp, nhiệt độ, độ ẩm

Nếu có lỗi, kiểm tra định dạng JSON hoặc kết nối MQTT

Dưới đây là kết quả:

```
Command Prompt - python * x + v
Microsoft Windows [Version 10.0.26100.3775]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ACER>python "D:\mang_may_tinh\Lab2.4a\iot_data_logger.py"
D:\mang_may_tinh\Lab2.4a\iot_data_logger.py:59: DeprecationWarning: Callback API version 1 is deprecated, update to latest version
client = mqtt.Client()
Đã kết nối MQTT broker.
Dữ liệu nhận được: 23, 24.0, 40.0
Dữ liệu nhận được: 27, 24.0, 40.0
Dữ liệu nhận được: 30, 24.0, 40.0
Dữ liệu nhận được: 33, 24.0, 40.0
Dữ liệu nhận được: 37, 24.0, 40.0
```

Bước 3: Kiểm tra file CSV Mở file sensor_data.csv bằng Excel hoặc Notepad Kiểm tra dữ liệu có được ghi dòng mới đúng định dạng không Dưới đây là kết quả:

| | | |
|----|----|----|
| 23 | 24 | 40 |
| 27 | 24 | 40 |
| 30 | 24 | 40 |
| 33 | 24 | 40 |
| 37 | 24 | 40 |

Bước 4: Kiểm tra vẽ biểu đồ

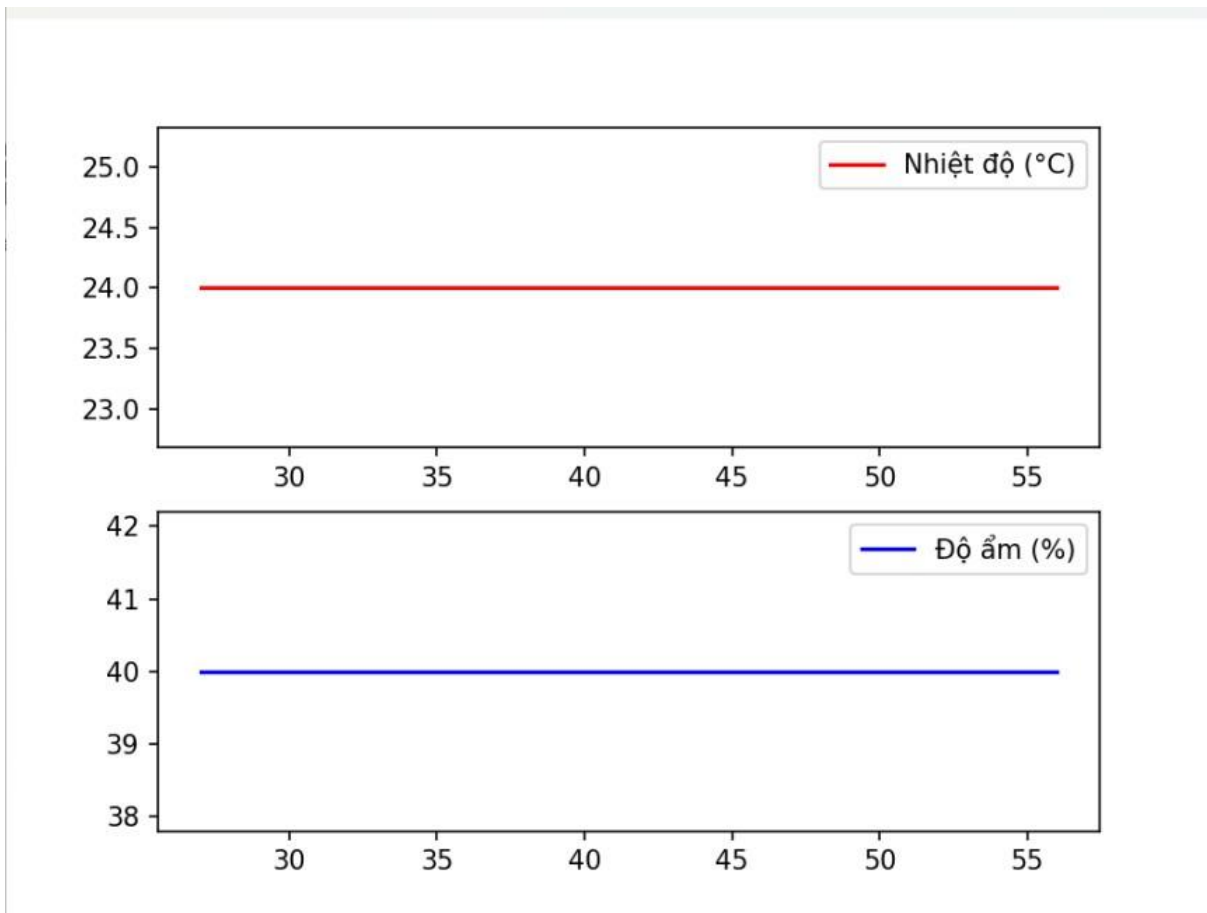
Sau 10 bản tin: xuất hiện cửa sổ vẽ matplotlib

Biểu đồ gồm 2 phần:

Trên: nhiệt độ theo thời gian

Dưới: độ ẩm theo thời gian

Dưới đây là kết quả:



Bước 5: Dừng thử nghiệm

Nhấn Ctrl+C trong terminal để dừng script

Lab 2.4b: Mô phỏng dữ liệu ngẫu nhiên ngay trong publisher

Bước 1 Khởi động Publisher: Thay vì chạy wokwi, chúng ta chạy một cửa sổ terminal:

terminal 1: chạy: `python iot_fake_publisher.py`

Mở Terminal 2: chạy: `python iot_data_logger.py` Dưới

đây là kết quả:

```

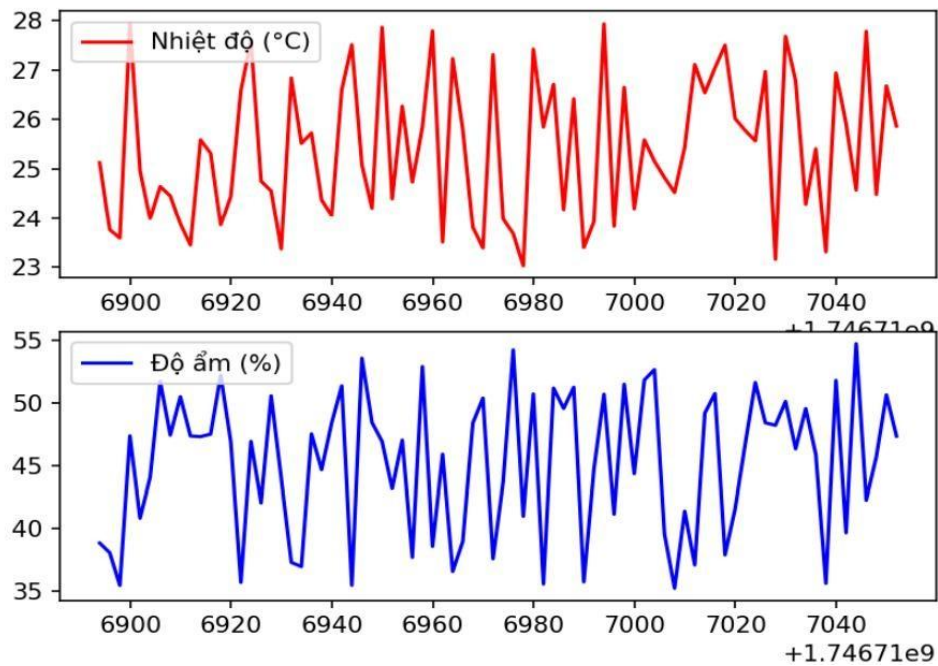
Command Prompt - python
\mang_may_tinh\Lab2.4a\iot_data_logger.py:59: Deprecat
nWarning: Callback API version 1 is deprecated, update
o latest version
client = mqtt.Client()
Đã kết nối MQTT broker.
Dữ liệu nhận được: 1746716893.926052, 25.13, 38.84
Dữ liệu nhận được: 1746716895.9268363, 23.76, 38.06
Dữ liệu nhận được: 1746716897.9279227, 23.59, 35.44

```

```

Command Prompt - python
p': 1746716891.9253924}
Đã gửi: {'temperature': 25.13, 'humidity': 38.84, 'timesta
mp': 1746716893.926052}
Đã gửi: {'temperature': 23.76, 'humidity': 38.06, 'timesta
mp': 1746716895.9268363}
Đã gửi: {'temperature': 23.59, 'humidity': 35.44, 'timesta
mp': 1746716897.9279227}
Đã gửi: {'temperature': 27.94, 'humidity': 47.35, 'timesta
mp': 1746716899.9289556}

```

Lab 2.4c. Lưu dữ liệu cảm biến vào SQLite và vẽ biểu đồ realtime

Bước 1: Terminal 1: `python iot_fake_publisher.py`

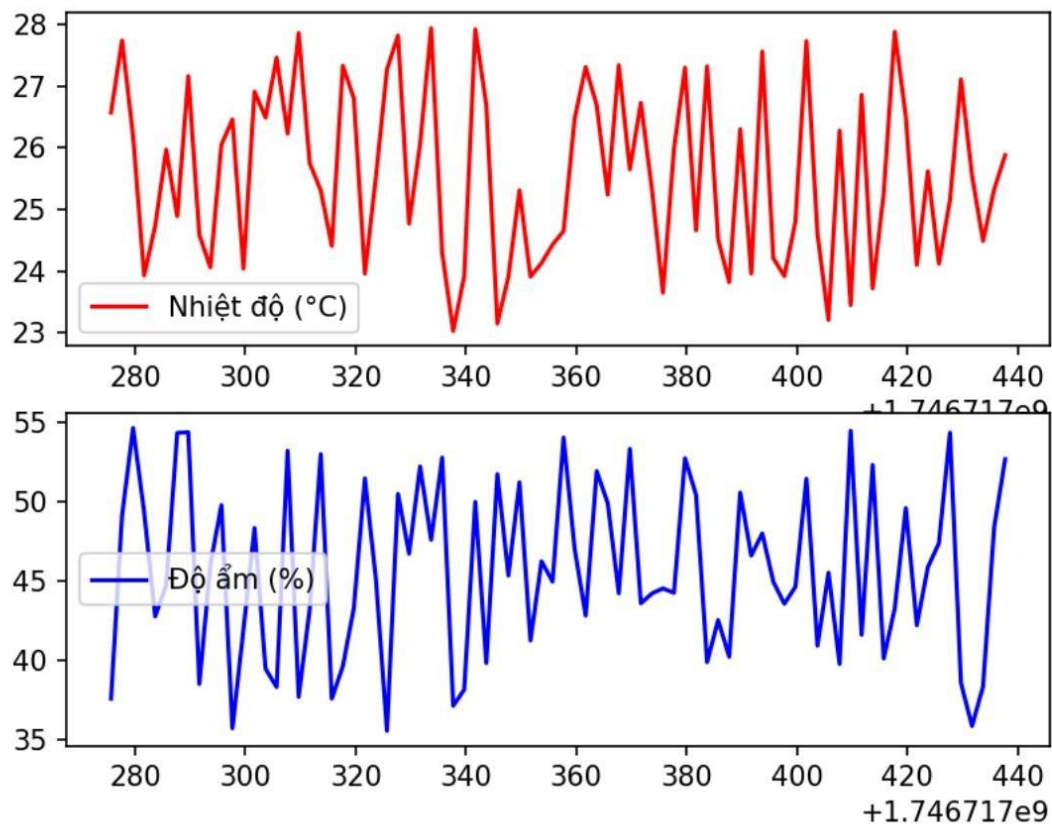
Terminal 2: `python iot_data_logger_sqlite.py`

Các bước còn lại thực hiện tương tự Lab 2.4a

```

Command Prompt - python * x + v - □ x
📦 Đã gửi: {'temperature': 27.15, 'humidity': 39.03,
'timestamp': 1746717463.755332}
📦 Đã gửi: {'temperature': 26.72, 'humidity': 53.06,
'timestamp': 1746717465.7563977}
📦 Đã gửi: {'temperature': 25.75, 'humidity': 36.44,
'timestamp': 1746717467.7572796}
📦 Đã gửi: {'temperature': 24.71, 'humidity': 49.43,
'timestamp': 1746717469.7583687}

Command Prompt - python * x + v - □ x
Dữ liệu nhận: 1746717453.7489748, 26.65, 39.03
Dữ liệu nhận: 1746717455.7498722, 24.2, 44.17
Dữ liệu nhận: 1746717457.7507973, 24.25, 39.82
Dữ liệu nhận: 1746717459.753787, 25.64, 36.12
Dữ liệu nhận: 1746717461.7545996, 23.94, 48.06
Dữ liệu nhận: 1746717463.755332, 27.15, 39.03
Dữ liệu nhận: 1746717465.7563977, 26.72, 53.06
Dữ liệu nhận: 1746717467.7572796, 25.75, 36.44
Dữ liệu nhận: 1746717469.7583687, 24.71, 49.43
  
```



Lab 2.4. d. Đọc dữ liệu từ file .csv

Giả sử file `sensor_data.csv` có nội dung như sau:

| | |
|---|--------------------------------|
| 1 | timestamp,temperature,humidity |
| 2 | 2023-07-15 10:30:00,25.5,60.0 |
| 3 | 2023-07-15 10:31:00,25.7,59.8 |

Sau khi chạy đoạn code `read_data_from_csv.py` ta có kết quả sau:

