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

1. Mục tiêu

Viết chương trình Python để:

- Nhận dữ liệu từ MQTT broker (topic `iot/khdl/esp32`).
- Ghi dữ liệu vào file CSV hoặc cơ sở dữ liệu SQLite.
- Vẽ biểu đồ nhiệt độ và độ ẩm real-time (hoặc sau khi thu thập đủ).

2. Yêu cầu

- **Thiết bị phát dữ liệu (Publisher)**: ESP32 mô phỏng DHT22 trên Wokwi hoặc script Python phát JSON {temperature, humidity, timestamp}.
- **Broker**: Sử dụng broker.hivemq.com (public broker).
- **Client**: Máy tính cài Python (để subscribe và lưu trữ dữ liệu).

3. Cài đặt thư viện

```
pip install paho-mqtt matplotlib
```

4. Viết mã code mô phỏng

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

Mã nguồn: iot_data_logger.py (phiên bản CSV + biểu đồ)

```python
import paho.mqtt.client as mqtt
import csv
import time
import json
import matplotlib.pyplot as plt

CSV_FILE = "sensor_data.csv"
MQTT_BROKER = "broker.hivemq.com"
MQTT_PORT = 1883
MQTT_TOPIC = "iot/khdl/esp32"

# Tạo file CSV nếu chưa có
with open(CSV_FILE, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["timestamp", "temperature", "humidity"])

temps, hums, times = [], [], []

def on_connect(client, userdata, flags, rc):
    if rc == 0:
        print("✅ Đã kết nối MQTT broker.")
        client.subscribe(MQTT_TOPIC)
    else:
        print("❌ Kết nối thất bại, mã lỗi:", rc)

def on_message(client, userdata, msg):
```

```
    try:
        data = json.loads(msg.payload.decode())
        timestamp = data.get("timestamp", time.time())
        temp = data.get("temperature", 0)
        hum = data.get("humidity", 0)

        print(f"📥 Dữ liệu nhận được: {timestamp}, {temp},
{hum}")

        with open(CSV_FILE, mode='a', newline='') as file:
            writer = csv.writer(file)
            writer.writerow([timestamp, temp, hum])

        temps.append(temp)
        hums.append(hum)
        times.append(timestamp)

        # Vẽ biểu đồ sau mỗi 10 bản ghi
        if len(temps) % 10 == 0:
            plt.clf()
            plt.subplot(2,1,1)
            plt.plot(times, temps, 'r-', label='Nhiệt độ (°C)')
            plt.legend()
            plt.subplot(2,1,2)
            plt.plot(times, hums, 'b-', label='Độ ẩm (%)')
            plt.legend()
            plt.pause(0.1)

    except Exception as e:
        print("⚠ Lỗi xử lý dữ liệu:", e)

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect(MQTT_BROKER, MQTT_PORT, 60)
plt.ion()  # Bật chế độ vẽ tương tác
client.loop_forever()
```

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 2 giây có 1 bản tin JSON gửi lên.

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

Bước 3: Kiểm tra file CSV

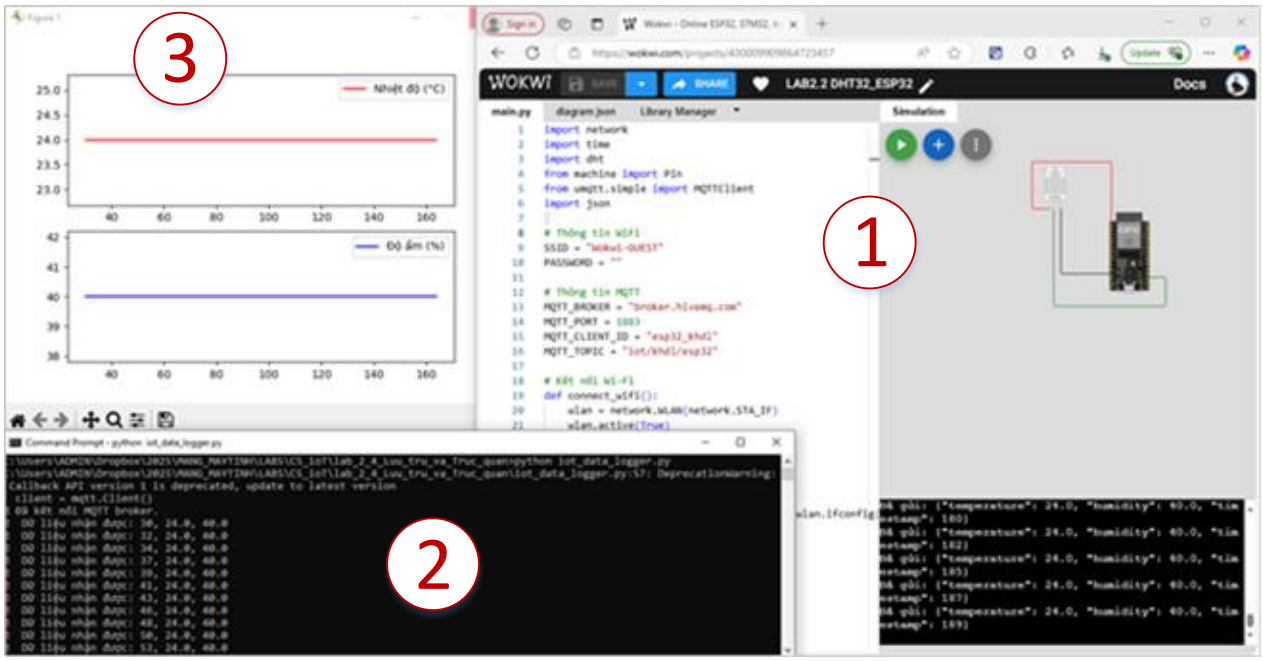
- Mở file `sensor_data.csv` bằng Excel hoặc Notepad
- Kiểm tra dữ liệu có được ghi dòng mới đúng định dạng không

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

Bước 5: Dừng thử nghiệm

- Nhấn Ctrl+C trong terminal để dừng script



Hình 1. Kết quả thực hiện lab 2.4.a

Nhận xét: Do thực hiện giả lập cảm biến DHT22 + ESP 32 trong wokwi, nên dữ liệu về nhiệt độ và độ ẩm không đổi ("temperature": 24.0, "humidity") → mặc dù thu thập và Public lên MQTT Broker với chu kỳ 2 giây, nhưng kết quả đồ thị biểu diễn ở mục 3 trên hình 1 là 2 đường thẳng. Do vậy tiếp theo chúng ta sẽ giả lập dữ liệu ngẫu nhiên ngay trong publisher để thay cho DHT 22 trong wokwi)

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

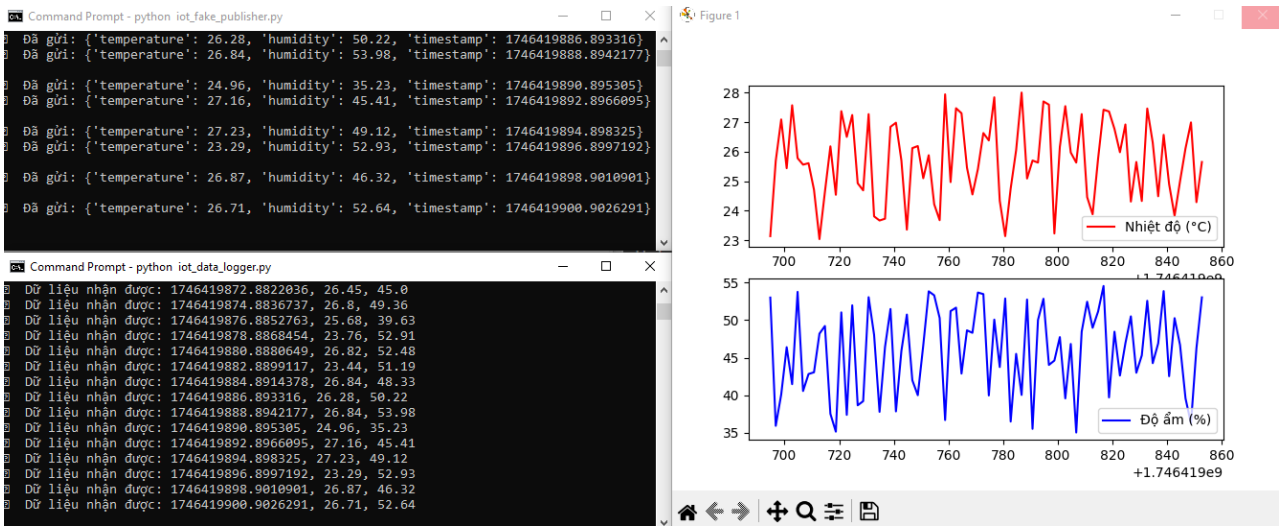
Viết mã code Python mô phỏng thiết bị iot như sau:

Mã nguồn: iot_fake_publisher.py

```python
import paho.mqtt.client as mqtt
import json
import time
import random

broker = "broker.hivemq.com"
port = 1883
topic = "iot/khdl/esp32"

client = mqtt.Client()
client.connect(broker, port, 60)

while True:
    data = {
        "temperature": round(random.uniform(23, 28), 2),
        "humidity": round(random.uniform(35, 55), 2),
        "timestamp": time.time()
    }
    client.publish(topic, json.dumps(data))
    print("📤 Đã gửi:", data)
    time.sleep(2)
```

Kiểm thử Lab 2.4b:

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**
Các bước còn lại thực hiện như Lab2.4a. → Kết quả chúng ta có như hình 2.



Hình 2. Kết quả thực hiện lab 2.4.b

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

Yêu cầu

- Ghi dữ liệu vào **SQLite database** sensor_data.db thay vì CSV.
- Biểu đồ vẫn cập nhật realtime mỗi 2 bản tin.
- Cột dữ liệu: `timestamp`, `temperature`, `humidity`

Viết mã lưu dữ liệu cảm biến vào SQLite và vẽ biểu đồ realtime:

Code: iot_data_logger_sqlite.py

```
1  import paho.mqtt.client as mqtt
2  import sqlite3
3  import time
4  import json
5  import matplotlib.pyplot as plt
6
7  DB_FILE = "sensor_data.db"
8  MQTT_BROKER = "broker.hivemq.com"
9  MQTT_PORT = 1883
10 MQTT_TOPIC = "iot/khdl/esp32"
11
12 # Kết nối SQLite và tạo bảng nếu chưa có
13 conn = sqlite3.connect(DB_FILE)
14 cursor = conn.cursor()
15 cursor.execute("""
16     CREATE TABLE IF NOT EXISTS sensor_data (
17         timestamp REAL,
18         temperature REAL,
19         humidity REAL
20     )
21 """)
22 conn.commit()
23
24 temps, hums, times = [], [], []
25
26 def on_connect(client, userdata, flags, rc):
27     if rc == 0:
28         print("✅ Đã kết nối MQTT broker.")
29         client.subscribe(MQTT_TOPIC)
30     else:
31         print("❌ Kết nối thất bại, mã lỗi:", rc)
32
33 def on_message(client, userdata, msg):
34     try:
35         data = json.loads(msg.payload.decode())
```

```
        timestamp = data.get("timestamp", time.time())
        temp = data.get("temperature", 0)
        hum = data.get("humidity", 0)

        print(f"📥 Dữ liệu nhận: {timestamp}, {temp}, {hum}")

        # Ghi vào SQLite
        cursor.execute("INSERT INTO sensor_data VALUES (?, ?, ?)",
(timestamp, temp, hum))
        conn.commit()

        temps.append(temp)
        hums.append(hum)
        times.append(timestamp)

        if len(temps) % 2 == 0:
            plt.clf()
            plt.subplot(2, 1, 1)
            plt.plot(times, temps, 'r-', label='Nhiệt độ (°C)')
            plt.legend()
            plt.subplot(2, 1, 2)
            plt.plot(times, hums, 'b-', label='Độ ẩm (%)')
            plt.legend()
            plt.pause(0.1)

    except Exception as e:
        print("⚠️ Lỗi xử lý:", e)

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message

client.connect(MQTT_BROKER, MQTT_PORT, 60)
plt.ion()
client.loop_forever()
```

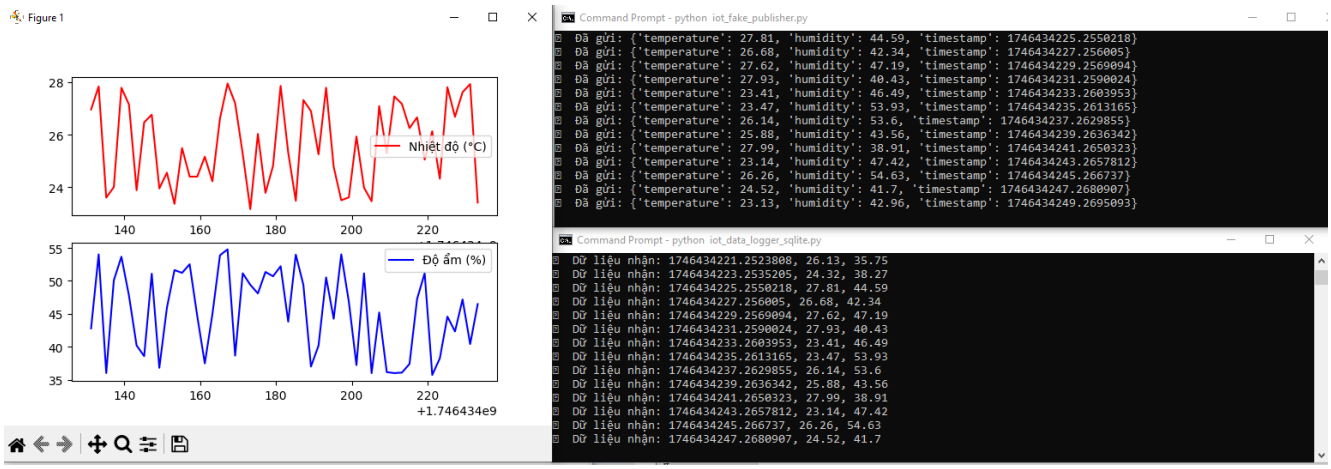
Kiểm thử Lab 2.4c:

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

Kết quả trực quan như hình 3.



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
```

Code: read_data_from_csv.py

```python
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
# Hàm đọc dữ liệu từ file CSV
def load_data(file_path="sensor_data.csv"):
    # Đọc file CSV, chuyển đổi cột 'timestamp' thành kiểu datetime
    try:
        data = pd.read_csv(file_path, parse_dates=['timestamp'])
        return data
    except Exception as e:
        print("Lỗi đọc dữ liệu:", e)
        return None
# Hàm cập nhật biểu đồ, được gọi định kỳ bởi FuncAnimation
def animate(i):
    data = load_data()
    if data is None or data.empty:
        print("Không có dữ liệu")
        return
    # Lấy dữ liệu thời gian, nhiệt độ và độ ẩm
    x = data['timestamp']
    y_temp = data['temperature']
    y_humid = data['humidity']
    # Xóa biểu đồ cũ
    plt.cla()
```

```
    # Vẽ biểu đồ nhiệt độ
    plt.plot(x, y_temp, label='Temperature (°C)', color='red', marker='o')
    # Vẽ biểu đồ độ ẩm
    plt.plot(x, y_humid, label='Humidity (%)', color='blue', marker='x')
    # Định dạng trục x cho dễ đọc: xoay nhãn thời gian
    plt.xticks(rotation=45, ha="right")
    plt.xlabel("Time")
    plt.ylabel("Value")
    plt.title("Real-Time IoT Sensor Data")
    plt.legend(loc="upper left")
    plt.tight_layout()  # Điều chỉnh bố cục cho gọn
# Tạo figure cho biểu đồ
fig = plt.figure()
# Thiết lập animation: mỗi 5000 ms (5 giây) cập nhật lại biểu đồ
ani = FuncAnimation(fig, animate, interval=5000)
plt.show()
```

Kết quả thực hiện chương trình: