Case Study: Giao thức MQTT trong IoT

BƯỚC 1 – GIỚI THIỆU TỔNG QUÁT VỀ MQTT TRONG MÔ HÌNH IOT

1.1. Mục tiêu

Sinh viên hiểu được vai trò của giao thức **MQTT** trong hệ thống IoT, nắm mô hình **Publish/Subscribe**, và biết các **thành phần bắt buộc** cần có khi triển khai.

1.2. Kiến thức cần nắm

Khái niệm	Ý nghĩa
MQTT (Message Queuing Telemetry Transport)	Giao thức truyền thông nhẹ, dùng để gửi dữ liệu từ thiết bị IoT lên server
Broker	Trung gian phân phối dữ liệu (ví dụ Mosquitto)
Publisher	Thiết bị gửi dữ liệu (ví dụ ESP32)
Subscriber	Thiết bị hoặc ứng dụng nhận dữ liệu
Topic	Đường dẫn dữ liệu trung gian giữa Pub và Sub

1.3. Mô hình luồng dữ liệu trong lớp học

```
[ESP32 giả lập trên Wokwi] --(MQTT pub)--> [Mosquitto broker trên máy tính]
                                                        ↑
                                              [Sub: MQTT Explorer / Python]
```

1.4. Thành phần tối thiểu để thực hành mô phỏng

Thành phần	Mô tả	Ghi chú
ESP32 (Wokwi)	Vi điều khiển giả lập, đóng vai trò publisher	Không cần thiết bị thật
Máy tính cá nhân	Cài Mosquitto broker, đóng vai broker	Bắt buộc, dùng localhost
MQTT Explorer / Python	Dùng để subscribe dữ liệu từ ESP32	Tùy chọn nâng cao

1.5. Kết quả học tập dự kiến

Sau bước này, sinh viên sẽ:

- Nhận biết các thành phần trong hệ thống MQTT.
- Phân biệt được vai trò của từng thiết bị (Pub / Broker / Sub).
- Biết mục tiêu tiếp theo là dựng broker để ESP32 có chỗ gửi dữ liệu.

BƯỚC 2 – CÀI ĐẶT VÀ KIỂM TRA MOSQUITTO BROKER TRÊN MÁY TÍNH

2.1. Mục tiêu

Sinh viên cài đặt và chạy thành công **Mosquitto Broker** trên **máy tính cá nhân** (Windows hoặc Ubuntu) ở chế độ mặc định, để từ đó có thể nhận dữ liệu từ thiết bị ESP32 mô phỏng trong bước tiếp theo.

2.2. Các yêu cầu trước khi thực hiện

Mục	Trạng thái cần thiết
Kết nối Internet	Phải có✅
Có quyền cài phần mềm	Có quyền admin✅
Biết mở Terminal (hoặc CMD)	✅

2.3. Hướng dẫn cho Hệ điều hành Windows

Bước 1 – Tải và cài đặt Mosquitto

1. Truy cập: **https://mosquitto.org/download**
2. Chọn phần "Windows installer" (dạng `.exe`, ví dụ: `mosquitto-2.0.18-install-windows-x64.exe`)
3. Cài đặt như bình thường (Next → Next), **nên bỏ tùy chọn "Run as service" nếu không cần chạy ngầm mỗi lần khởi động**.

Bước 2 – Kiểm tra đã cài xong chưa

4. Mở **Command Prompt (CMD)**
5. Gõ:

```
mosquitto -v
```

Nếu thấy dòng như `mosquitto version 2.0.x starting`, nghĩa là OK.

2.4. Hướng dẫn cho Ubuntu / WSL / Debian-based Linux

Bước 1 – Cài Mosquitto và client

```
sudo apt update
sudo apt install mosquitto mosquitto-clients
```

Bước 2 – Kiểm tra broker

```
mosquitto -v
```

2.5. Chạy thử Mosquitto ở chế độ foreground (hiển thị log)

Lệnh này chạy broker trực tiếp để chúng ta nhìn thấy có ai kết nối/gửi dữ liệu.

```
mosquitto -v
```

Broker sẽ chạy ở:

- Địa chỉ: `localhost`
- Port mặc định: `1883`
- Không có mật khẩu, không mã hóa (dùng cho lớp học nội bộ)

Lưu ý: Giữ cửa sổ terminal/cmd này **luôn mở** trong suốt quá trình chạy các thực hành tiếp theo!

2.6. Kiểm tra broker hoạt động với lệnh CLI

Mục tiêu

Xác nhận rằng Mosquitto broker đang chạy ổn định trên máy tính cá nhân, có khả năng nhận và phân phối dữ liệu theo cơ chế publish/subscribe, thông qua lệnh dòng (CLI) và giao diện đồ họa (MQTT Explorer).
Mục này hoàn toàn độc lập với ESP32 – chỉ kiểm tra chức năng của broker.
Lưu ý:

- Cổng sử dụng mặc định: 1883
- Broker đang lắng nghe tại localhost

2.6.1. Kiểm tra bằng lệnh dòng (CLI)

a. Mở **2 cửa sổ terminal** (Terminal A và Terminal B)

Terminal A: chạy `subscriber`

```
mosquitto_sub -h localhost -t iot/test
```

Terminal B: chạy `publisher`

```
mosquitto_pub -h localhost -t iot/test -m "Xin chào MQTT"
```

Terminal A Terminal B

→ Cho thấy broker đã hoạt động đúng, nhận và phân phối thông điệp thành công.

2.6.2. Kiểm tra bằng MQTT Explorer

a. Cài và mở MQTT Explorer

Tải tại: https://mqtt-elxplorer.com
Cài đặt như phần mềm thông thường

b. Tạo kết nối

- Click "+ Add new connection"
- Nhập thông tin:
  - **Name**: Local Test
  - **Broker address**: localhost
  - **Port**: 1883
- Nhấn **Connect**

c. Gửi một gói tin thử

Trên một cửa sổ terminal (hoặc PowerShell):

Thực hiện lệnh:

```
mosquitto_pub -h localhost -t iot/classroom/temp -m "{\"temperature\":27.2,\"humidity\":58.6}"
```

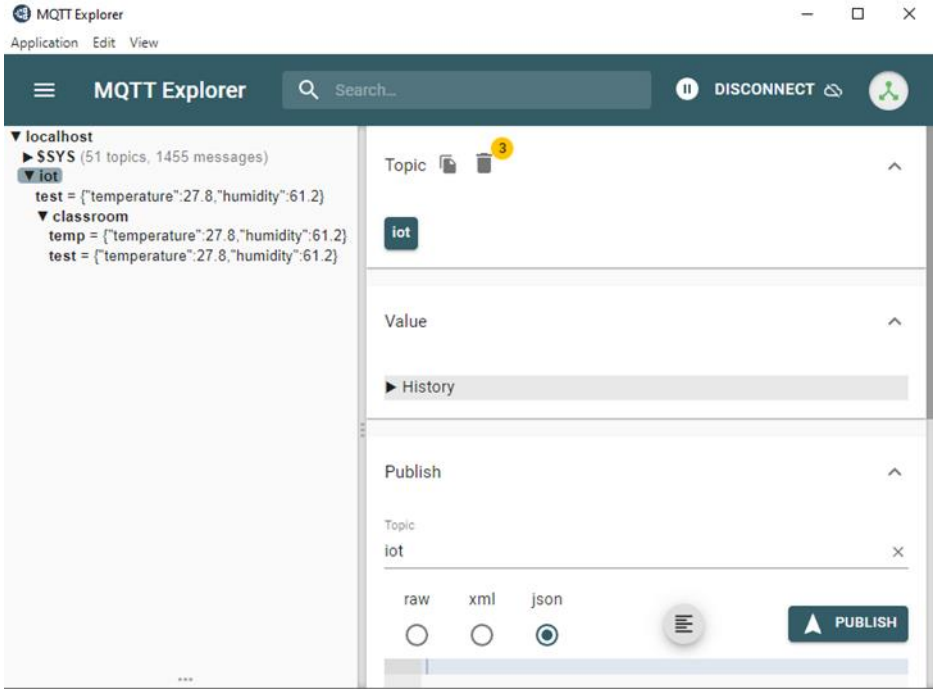
Kết quả:

Quan sát trong MQTT Explorer, chúng ta sẽ thấy:

iot
└── classroom
  └── temp

→ {"temperature":27.8,"humidity":61.2}

Như vậy, → Broker hiển thị đúng topic và dữ liệu được publish.



Lưu ý:

Nội dung	Mục đích
CLI (lệnh dòng)	Cho sinh viên hiểu rõ cách broker hoạt động nội bộ
MQTT Explorer (giao diện đồ họa)	Trực quan, dễ trình bày và theo dõi dữ liệu theo thời gian thực

2.7. Ghi chú cho sinh viên

Câu hỏi kiểm tra kiến thức	Gợi ý trả lời
localhost nghĩa là gì?	Máy tính đang sử dụng
Cổng 1883 là gì?	Cổng mặc định của giao thức MQTT
Có cần Internet để broker hoạt động không?	Không cần, vì ta đang dùng trong mạng nội bộ

Kết thúc Bước 2

Sau bước này, chúng ta đã sẵn sàng để **nhận dữ liệu từ ESP32 (giả lập)** qua MQTT.

Bước 3 – MÔ PHỎNG ESP32 GỬI DỮ LIỆU ĐẾN BROKER

3.1. Mục tiêu

Sinh viên sử dụng ESP32 giả lập trong Wokwi, viết mã MicroPython để:

- Kết nối Wi-Fi mô phỏng (SSID: `Wokwi-GUEST`);
- Kết nối đến MQTT Broker (địa chỉ ánh xạ `10.0.2.2`);
- Gửi dữ liệu giả lập nhiệt độ/độ ẩm dưới dạng JSON qua MQTT topic.

Chuỗi hoạt động cần đạt được
ESP32 (Wokwi, MicroPython)
└──[Publish JSON]──> Mosquitto Broker (localhost, máy thật)

3.2. Chuẩn bị môi trường (Wokwi)

Mục tiêu

Thiết lập một môi trường mô phỏng sử dụng **bo mạch ESP32** trên nền tảng **Wokwi**, sẵn sàng để viết và thực thi chương trình MicroPython gửi dữ liệu MQTT.

3.2.1. Giới thiệu về Wokwi

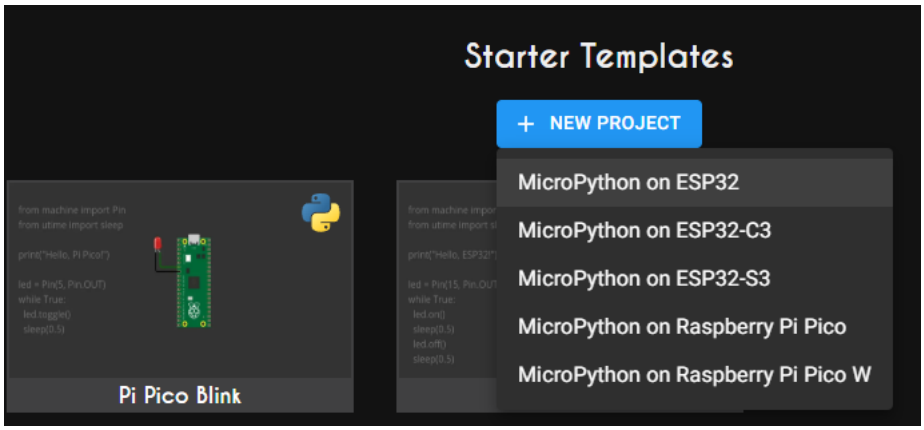
- **Wokwi** là nền tảng mô phỏng phần cứng trực tuyến cho phép chạy code trên bo mạch như ESP32, ESP8266, Arduino... ngay trong trình duyệt.
- Wokwi hỗ trợ **MicroPython cho ESP32**, cho phép sinh viên mô phỏng hoàn toàn mà không cần thiết bị vật lý.
- Các thành phần như **cảm biến, đèn LED, Wi-Fi** đều được giả lập.

3.2.2. Thành phần cần có trong mô hình

Thành phần	Vai trò mô phỏng
Bo mạch ESP32	Thiết bị chính để lập trình MicroPython và gửi MQTT
Wi-Fi mô phỏng	Sử dụng mạng giả lập "Wokwi-GUEST" để kết nối mạng
Không cần cảm biến vật lý	Dữ liệu nhiệt độ/độ ẩm được tạo giả lập bằng hàm Python

3.2.3. Tạo mới dự án trên Wokwi

1. Truy cập: https://wokwi.com
2. Nhấn nút **"New Project"**
3. Chọn mẫu **"MicroPython for ESP32"**
4. Dự án sẽ tạo sẵn một file `main.py`
5. Xóa nội dung mặc định và chờ dán đoạn mã ở mục 3.4



3.2.4. Cấu trúc tệp dự án (tự động tạo)

Tệp / Thư mục	Mô tả

main.py	File chính chứa mã MicroPython
diagram.json	(Tự sinh ra) chứa cấu hình phần cứng mô phỏng
project-name.txt	(Tuỳ chọn) tên hiển thị của dự án

3.2.5. Yêu cầu kết nối đến broker (máy của sinh viên)

Để mô phỏng có thể gửi dữ liệu đến broker thật (Mosquitto đang chạy trên máy tính cá nhân), cần cấu hình đúng **địa chỉ IP đặc biệt**:

mqtt_server = "10.0.2.2"

Tham số	Giải thích
10.0.2.2	Là địa chỉ nội bộ ánh xạ từ Wokwi đến localhost của máy thật
1883	Là cổng mặc định của Mosquitto broker

Đây là kỹ thuật "loopback routing" **của Wokwi để ESP32 mô phỏng** liên lạc với máy thật**.**

Lưu ý:

Gợi ý triển khai	**Mục đích**
Sinh viên chỉ cần máy có trình duyệt	Không phụ thuộc vào thiết bị thật
Không cần cài IDE hay thư viện ngoại	Triển khai tập trung vào logic mạng
Dùng Wokwi-GUEST để tránh cấu hình Wi-Fi	Đơn giản hoá quy trình kết nối ban đầu

Sau khi hoàn thành bước này, sinh viên sẽ:

- Có một môi trường mô phỏng sẵn sàng chạy code ESP32,
- Kết nối được đến broker thật trên máy của mình,
- Sẵn sàng sang **mục 3.3 – Cấu hình Wi-Fi và MQTT trong mã MicroPython.**

Code: Project '**MicroPython MQTT DHT22 (ESP32) CT**'

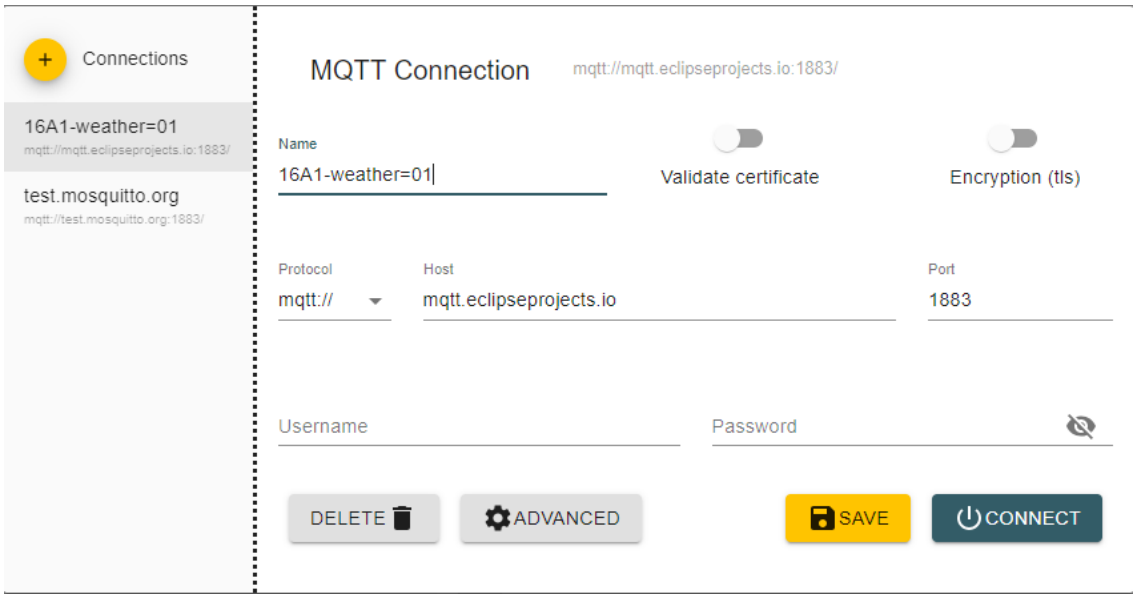
```
1  import network                  # Thư viện kết nối Wi-Fi
2  import time                     # Dùng để delay và đếm thời gian
3  import json                     # Định dạng dữ liệu JSON
4  from machine import Pin         # Giao tiếp chân GPIO
5  from umqtt.simple import MQTTClient  # Thư viện giao thức MQTT
6  import dht                      # Thư viện cảm biến DHT22
7
8  # === Khởi tạo cảm biến DHT22 trên chân GPIO15 ===
9  sensor = dht.DHT22(Pin(15))
10
11 # === Thông tin Wi-Fi (mô phỏng trên Wokwi) ===
12 ssid = "Wokwi-GUEST"
13 password = ""
```

```
14
15  # === Kết nối Wi-Fi ===
16  def connect_wifi():
17      wlan = network.WLAN(network.STA_IF)
18      wlan.active(True)
19      wlan.connect(ssid, password)
20      print("Đang kết nối Wi-Fi...")
21      for _ in range(10):
22          if wlan.isconnected():
23              break
24          time.sleep(1)
25      if wlan.isconnected():
26          print("Wi-Fi OK:", wlan.ifconfig())
27      else:
28          print("Kết nối Wi-Fi thất bại.")
29
30  # === Cấu hình MQTT Broker ===
31  MQTT_CLIENT_ID = "16A1-weather-01"          # Tên thiết bị (không trùng)
32  MQTT_BROKER = "mqtt.eclipseprojects.io"   # Broker công cộng
33  MQTT_TOPIC = b"16A1-demo/temp"              # Chủ đề gửi dữ liệu
34
35  # === Kết nối MQTT Broker ===
36  def connect_mqtt():
37      try:
38          client = MQTTClient(MQTT_CLIENT_ID, MQTT_BROKER, port=1883)
39          client.connect()
40          print("Đã kết nối MQTT tới", MQTT_BROKER)
41          return client
42      except Exception as e:
43          print("Lỗi MQTT:", e)
44          return None
45
46  # === Khởi động hệ thống ===
47  connect_wifi()
48  client = connect_mqtt()
49
50  count = 1  # Bộ đếm số lần gửi dữ liệu
51
52  # === Vòng lặp chính: đọc cảm biến và gửi MQTT định kỳ ===
53  while True:
54      try:
55          sensor.measure()                    # Đọc cảm biến
56          temp = sensor.temperature()         # Nhiệt độ (°C)
57          hum = sensor.humidity()             # Độ ẩm (%)
58
59          # Tạo chuỗi JSON kèm số đếm
60          payload = json.dumps({
61              "temperature": temp,
62              "humidity": hum,
63              "count": count
64          })
```

```
65
66          # Gửi dữ liệu lên MQTT
67          if client:
68              client.publish(MQTT_TOPIC, payload)
69              print(f"Đã gửi (#{count}):", payload)
70          else:
71              print("Không thể gửi – MQTT chưa kết nối.")
72
73          count += 1  # Tăng số lần gửi
74
75      except Exception as e:
76          print("Lỗi cảm biến:", e)
77
78      time.sleep(5)  # Đợi 5 giây trước lần gửi tiếp theo
```

Khi chạy chương trình, quan sát trên MQTT explore:

Thiết lập kết nối với MQTT Broker như sau:



Thêm topic: 16A1-demo/temp