

# THỰC HÀNH ĐIỀU KHIỂN QUA SMART PHONE, WEB

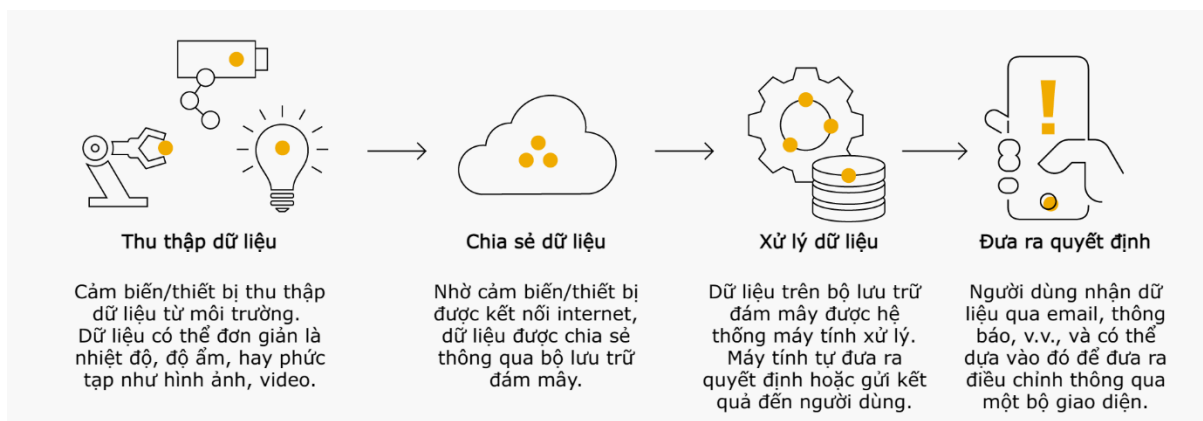
## Lý thuyết: Tổng quan về hệ thống IoT

Hệ thống IoT, viết tắt của Internet of Things (mạng lưới internet của các vật), là một mạng lưới của các thiết bị và cảm biến được kết nối với internet và có khả năng trao đổi dữ liệu với nhau mà không cần sự can thiệp của con người.

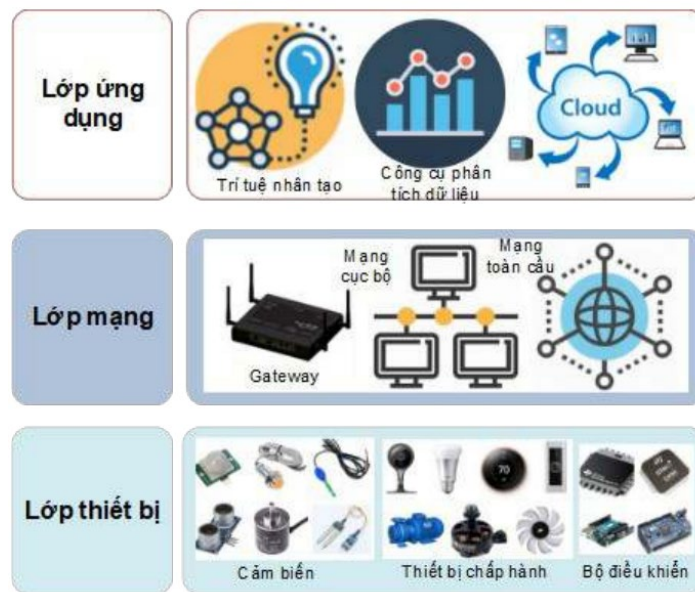
Một hệ thống IoT sẽ bao gồm 4 thành phần chính:

- **Thiết bị hay còn được gọi là Things:** gồm các thiết bị cuối tham gia vào mạng IoT (đồng hồ đeo tay, tủ lạnh, smartphone,...), mỗi thiết bị sẽ được tích hợp một cảm biến không dây thông minh hoặc các thiết bị có thể nhận lệnh trực tiếp từ người dùng
- **Trạm kết nối hay cổng kết nối (Gateways):** Là cầu nối giữa các công nghệ truyền thông khác nhau. IoT Gateway là một máy tính nhúng làm cầu nối kết nối giữa các cảm biến (Sensors), tác nhân (Actors) tới mạng Internet hoặc mạng nội bộ Intranet. Các công nghệ được tích hợp trong một IoT Gateway là Bluetooth, Wifi, BLE, Zigbee, Z-wave, 6LoWPAN, NFC, WiFi Direct, GSM, LTE, LoRa, NB-IoT và LTE-M... để giao tiếp với mạng Intranet hoặc mạng Internet.
- **Hạ tầng mạng hay các điện toán đám mây (Network and cloud):** Các trung tâm dữ liệu và hạ tầng điện toán đám mây gồm một hệ thống lớn các máy chủ, hệ thống lưu trữ và mạng ảo hóa được kết nối. Cloud đóng vai trò như “bộ não” của mô hình IoT vì chúng chịu trách nhiệm xử lý, chỉ huy và phân tích các dữ liệu thu thập được.
- **Bộ phân tích và xử lý dữ liệu ( Services-creation and Solution Layers):** dữ liệu thô sẽ được thu thập, phân tích và chuyển đổi thành các thông tin hữu ích, có khả năng hỗ trợ người dùng đưa ra các quyết định quan trọng.

## Nguyên lý hoạt động cơ bản của IoT:



## Kiến trúc IoT cơ bản:

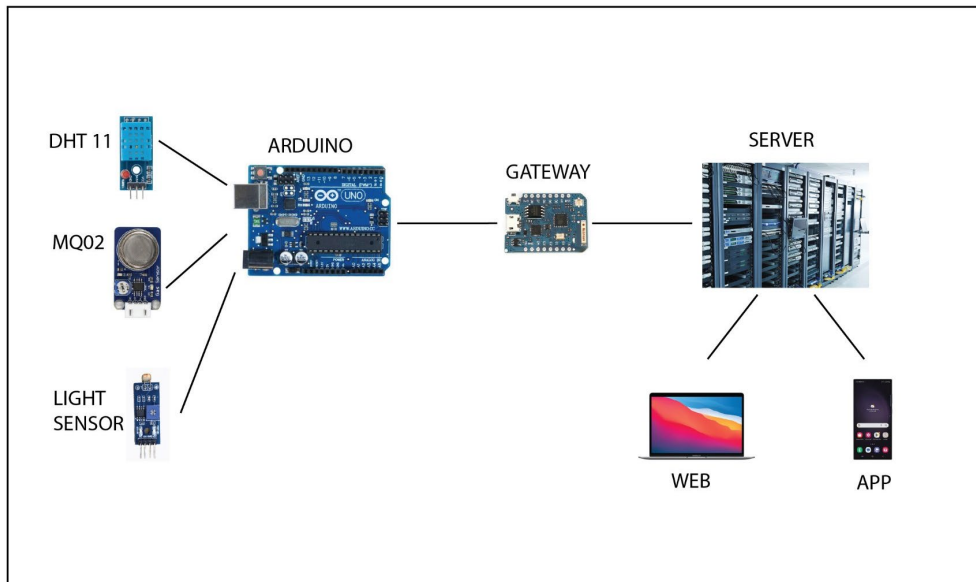


- **Lớp thiết bị:** Lớp này bao gồm các cảm biến, thiết bị chấp hành và các bộ điều khiển như vi xử lý/vi điều khiển, PLC, FPGA và các máy tính nhúng. Các thiết bị này thực hiện việc đo lường và thu thập dữ liệu về các đại lượng vật lý thông qua các cảm biến, điều khiển các thiết bị chấp hành và có khả năng truyền và nhận dữ liệu từ các thiết bị khác qua mạng.
- **Lớp mạng:** Lớp này xác định các giao thức truyền thông và thiết bị liên kết mạng để kết nối các thiết bị IoT với nhau và với các hệ thống ngoại vi. Lớp mạng bao gồm các thiết bị như Hub, Switch, Router, Gateway và các thiết bị có khả năng lưu trữ và xử lý dữ liệu cục bộ trước khi gửi đến trung tâm dữ liệu. Các thiết bị ở lớp thiết bị kết nối với thiết bị Gateway ở lớp mạng thông qua các mạng cục bộ như Wifi, Zigbee, Bluetooth, LoRaWAN hoặc các mạng có dây như CAN Modbus, Profibus, RS485, Ethernet. Sau đó, dữ liệu được gửi lên trung tâm thông qua các kết nối mạng toàn cầu như Internet, 3G/4G/LTE, GSM.
- **Lớp ứng dụng:** Đây là trung tâm lưu trữ dữ liệu và xử lý dữ liệu hay còn gọi là đám mây điện tử. Lớp này thu nhận dữ liệu từ lớp mạng, lưu trữ và xử lý dữ liệu, và đưa ra quyết định dựa trên các thuật toán AI/ML hoặc các công cụ phân tích dữ liệu hiện đại.

## Nội dung 1: Thực hành lấy dữ liệu từ server hiển thị lên web/app

**Yêu cầu:** Lấy dữ liệu(nhiệt độ, độ ẩm, khí gas) từ server(openlab) và hiển thị lên web/app .

**Nội dung đưa dữ liệu từ kit lên server đã thực hiện ở buổi 6**



- **Các cảm biến (Sensors):** Đây là các thiết bị như cảm biến nhiệt độ, độ ẩm, ánh sáng, hoặc bất kỳ loại cảm biến nào khác được sử dụng để thu thập dữ liệu từ môi trường.
- **Vi điều khiển Arduino:** Arduino là một bo mạch điện tử mạnh mẽ và linh hoạt có thể được lập trình để thu thập dữ liệu từ cảm biến và gửi dữ liệu đến gateway. Arduino thường được sử dụng để kiểm soát và thu thập dữ liệu từ các thiết bị cảm biến.
- **Gateway là chip WiFi LOLIN(WEMOS) D1 mini Lite:** Được sử dụng để thu thập dữ liệu từ các thiết bị Arduino và chuyển tiếp dữ liệu đến máy chủ broker.
- **Máy chủ Broker:** Máy chủ Broker là nơi các dữ liệu từ các thiết bị cảm biến được gửi đến và lưu trữ. Trong kiến trúc này, MQTT broker thường được sử dụng, nơi mà Gateway gửi dữ liệu từ các thiết bị cảm biến qua giao thức MQTT.
- **Trang web hiển thị:** Trang web được sử dụng để hiển thị dữ liệu từ các cảm biến. Thông qua trang web này, người dùng có thể theo dõi và quản lý dữ liệu thu thập từ các cảm biến. Trang web có thể được xây dựng bằng các ngôn ngữ lập trình như HTML, CSS, JavaScript và kết nối đến máy chủ broker để lấy dữ liệu thông qua MQTT.

Tóm lại, trong kiến trúc này, các cảm biến gửi dữ liệu đến vi điều khiển Arduino, sau đó Arduino gửi dữ liệu đến Gateway thông qua Wi-Fi. Gateway chuyển tiếp dữ liệu đến máy chủ broker thông qua giao thức MQTT. Máy chủ broker lưu trữ dữ liệu và cung cấp nó cho trang web hiển thị thông qua MQTT.

## Chuẩn bị phần cứng

- Cảm biến DHT11/DHT22, cảm biến khí gas MQ-2
- Board Arduino Uno
- Chip WiFi: LOLIN(WEMOS) D1 mini Lite
- Dây cáp nạp code cho arduino và chip WiFi
- Bus

## Sơ đồ đấu nối

- Sơ đồ đấu nối Arduino - DHT11/22

Arduino	DHT11/22
Vcc	5V
GND	GND
D8	Data

- Sơ đồ đấu nối Arduino – MQ-2

Arduino	MQ-2
Vcc	5V
GND	GND
A1	Data

- Sơ đồ đấu nối Arduino - chip WiFi: **LOLIN(WEMOS) D1 mini Lite**

Arduino	D1 Mini
Vcc	5V
GND	GND
D10	D4
D11	D3

## Code tham khảo

### Code nạp cho Arduino:

```
#include <DHT.h>
#include <ArduinoJson.h>
#include <SoftwareSerial.h>

#define MQ_Data A1    // Data Gas
#define DHT_Data 8    // Data DHT

DHT dht(DHT_Data, DHT22); // DHT11

#define RX_PIN 10
#define TX_PIN 11
```

```

SoftwareSerial mySerial(RX_PIN, TX_PIN);

void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);
  while (!Serial) delay(1);

  dht.begin();
}

unsigned long timeUpdata = millis();

DynamicJsonDocument doc(1024);
DynamicJsonDocument docBtn(1024);

void loop() {
  if (millis() - timeUpdata > 2000) {
    float hum = dht.readHumidity();
    float temp = dht.readTemperature();
    float gas = analogRead(MQ_Data);

    doc["temperature"] = temp;
    doc["humidity"] = hum;
    doc["gas"] = gas;

    // gửi data thông qua cổng giao tiếp mềm cho Chip WiFi
    serializeJson(doc, mySerial);
    mySerial.println();

    // In ra màn hình Serial
    serializeJson(doc, Serial);
    Serial.println();

    timeUpdata = millis();
  }
}

```

**Code nạp cho chip WiFi: LOLIN(WEMOS) D1 mini Lite:**

**Khi nạp nhớ chọn đúng board LOLIN(WEMOS) D1 mini Lite**

```

#ifdef ESP8266
#include <ESP8266WiFi.h> /* WiFi library for ESP8266 */
#else
#include <WiFi.h> /* WiFi library for ESP32 */

```

```
#endif
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <PubSubClient.h>

// Khai báo các chân RX và TX cho espsoftwareserial
#define RX_PIN D3
#define TX_PIN D4

// Tạo một đối tượng espsoftwareserial với tốc độ baud là 9600
SoftwareSerial mySerial(RX_PIN, TX_PIN);

#define wifi_ssid "IoT LAB"
#define wifi_password "kvt1ptit"
#define mqtt_server "10.170.69.245"
#define mqtt_port 1883

WiFiClient espClient;
PubSubClient client(espClient);

// Kết nối tới WiFi
void setup_wifi() {
    delay(10);
    Serial.println();
    Serial.print("Connecting to ");
    Serial.println(wifi_ssid);

    WiFi.begin(wifi_ssid, wifi_password);

    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}

// Kết nối tới MQTT Broker
void reconnect() {
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
```

```

    if (client.connect("IoTLABClient")) { // tự đặt ID name cho Client . Ví dụ
"Client1"
        Serial.println("connected");
    } else {
        Serial.print("failed, rc=");
        Serial.print(client.state());
        Serial.println(" try again in 5 seconds");
        delay(5000);
    }
}
}

//-----Method for Publishing MQTT Messages-----
void publishMessage(const char* topic, String payload, boolean retained) {
    if (client.publish(topic, payload.c_str(), true))
        Serial.println("Message published [" + String(topic) + "]: " + payload);
}

void setup() {
    // Khởi tạo cổng nối tiếp phần cứng với tốc độ baud là 115200 (đây là Baudrate của
    // cổng Serial trên Arduino IDE)
    Serial.begin(115200);

    // Khởi tạo cổng nối tiếp phần mềm với tốc độ baud là 9600 ( đây là Baudrate của
    // cổng mySerial để giao tiếp với board Arduino Uno)
    mySerial.begin(9600);

    setup_wifi();
    client.setServer(mqtt_server, mqtt_port);
}

DynamicJsonDocument doc(1024);

void loop() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
    // nhận data từ Arduino thông qua Tx-Rx và sau đó publish data lên MQTT Broker
    if (mySerial.available()) {
        String data = mySerial.readStringUntil('\n');
        DeserializationError error = deserializeJson(doc, data);
        char mqtt_message[1024];

        serializeJson(doc, mqtt_message);
    }
}

```

```
    publishMessage("Ví dụ Group1", mqtt_message, true); // tự đặt topic Pub theo
nhóm của bạn. Ví dụ "Group1".
}
}
```



Sau khi nạp xong tiến hành lấy dữ liệu từ Broker về trang web để hiển thị lên DashBoard

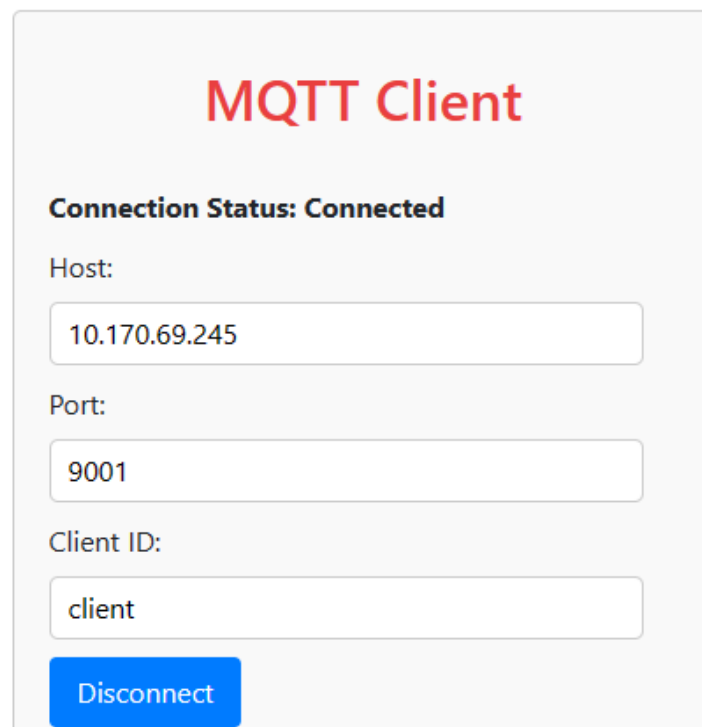
Vào mục **BrokerMQTT** trên trang web của OpenLAB

**Bước 1: Kết nối tới Broker MQTT**

**Host: 10.170.69.245**

**Port: 9001**

**Client ID: Tự đặt cho riêng nhóm**



The screenshot shows a web interface titled "MQTT Client" in red text. Below the title, it displays "Connection Status: Connected". There are three input fields: "Host" with the value "10.170.69.245", "Port" with the value "9001", and "Client ID" with the value "client". At the bottom, there is a blue button labeled "Disconnect".

**Bước 2: Subscribe topic để lấy data từ Broker:**

**Ví dụ: Topic Publish của nhóm là Group1**

```
if (mySerial.available()) {  
    String data = mySerial.readStringUntil('\n');  
    DeserializationError error = deserializeJson(doc, data);  
    char mqtt_message[1024];  
  
    serializeJson(doc, mqtt_message);  
    publishMessage("Group1", mqtt_message, true); // tự đặt topic Pub  
}
```

Ta sẽ điền vào mục Topic to Subscribe là: Group1

Và sau đó bấm Subscribe để lấy data từ Broker

Topic to Subscribe:

SubscribeUnsubscribe


Data Subscribed:

```
{ "temperature": null, "humidity": null, "gas": 87, "led": 0, "buzzer": 0, "fan": 0 }
```

Sau khi Subscribe topic ta nhận được 1 chuỗi json về giá trị các thông số môi trường như nhiệt độ, độ ẩm, gas.


Có thể theo dõi các thông số 1 cách trực quan qua Dashboard:

### Nhóm Group1




Còi

Bật




Quạt

Bật



Led


Bật



Servo

Bật


### Nhóm Group1



Nhiệt độ

Topic: Group1


Value: 28 °C



Độ ẩm

Topic: Group1


Value: 81 %



Khí gas

Topic: Group1

Value: 62 ppm



Ánh sáng

Topic: Group1

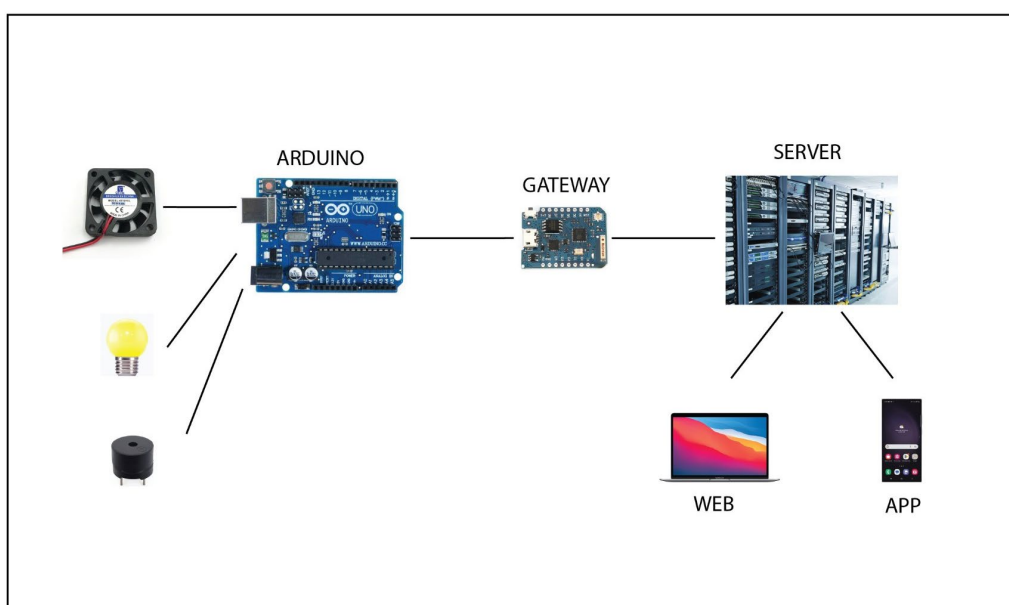
Value: undefined lux

## Biểu đồ



## Nội dung 2: Điều khiển thiết bị qua web

**Mô tả:** Thực hiện điều khiển thiết bị chấp hành bên dưới thông qua web



- **Các thiết bị chấp hành:** Các thiết bị này có thể bao gồm đèn LED, còi, quạt, cửa cuốn, hoặc bất kỳ thiết bị nào mà bạn muốn có khả năng điều khiển từ xa thông qua mạng Internet. Vi điều khiển Arduino sẽ gửi các tín hiệu điều khiển đến các thiết bị này dựa trên các lệnh từ trang web hiển thị, cho phép người dùng điều khiển các chức năng hoặc trạng thái của chúng từ xa.
- **Vi điều khiển Arduino:** Arduino là một bo mạch điện tử mạnh mẽ và linh hoạt sử dụng để kết nối với các cảm biến hoặc thiết bị điện tử khác và thực hiện các hành động dựa trên các tín hiệu từ các cảm biến hoặc lệnh điều khiển từ người dùng thông qua giao diện web.
- **Gateway là chip WiFi LOLIN(WEMOS) D1 mini Lite:** Gateway này giúp việc kết nối các thiết bị nhúng như Arduino với mạng Internet thông qua Wi-Fi, tạo điều kiện cho việc điều khiển từ xa thông qua trang web..
- **Máy chủ Broker:** Máy chủ Broker đảm bảo việc truyền thông đáng tin cậy giữa các thiết bị nhúng và trang web hiển thị. Nó nhận các lệnh từ trang web và chuyển tiếp chúng đến vi điều khiển Arduino để thực thi, và ngược lại.
- **Trang web hiển thị:** trang web này cung cấp giao diện để sử dụng cho người dùng để điều khiển và theo dõi trạng thái của các thiết bị nhúng kết nối với vi điều khiển Arduino thông qua gateway. Người dùng có thể gửi lệnh điều khiển từ xa thông qua trang web, và nhận phản hồi về trạng thái của các thiết bị.

Tóm lại, trong kiến trúc này, các thiết bị chấp hành (như đèn LED, bơm nước, quạt, cửa cuốn và các thiết bị khác) sẽ nhận các lệnh điều khiển từ vi điều khiển Arduino. Vi điều khiển Arduino sẽ nhận các tín hiệu điều khiển từ trang web hiển thị thông qua máy chủ Broker, sau đó chuyển tiếp các lệnh này đến các thiết bị chấp hành. Các thiết bị chấp hành sẽ thực hiện các hành động tương ứng dựa trên các lệnh điều khiển mà

chúng nhận được từ vi điều khiển Arduino. Điều này cho phép người dùng điều khiển các chức năng hoặc trạng thái của các thiết bị từ xa thông qua trang web hiển thị..

### Chuẩn bị phần cứng

- Các thiết bị chấp hành: LED hoặc còi hoặc quạt ( chọn 1 trong 3)
- Board Arduino Uno
- Chip WiFi: LOLIN(WEMOS) D1 mini Lite
- Dây cáp nạp code cho arduino và chip WiFi
- Bus

### Sơ đồ đấu nối

- Sơ đồ đấu nối Arduino – Led / Còi / Quạt

Arduino	LED/Còi/ Quạt
Vcc	5V
GND	GND
D7 (led) D8 (còi) D9 (quạt)	Data

- Sơ đồ đấu nối Arduino - chip WiFi: **LOLIN(WEMOS) D1 mini Lite**

Arduino	D1 Mini
Vcc	5V
GND	GND
D10	D4
D11	D3

### Code tham khảo

#### Code nạp cho chip WiFi: LOLIN(WEMOS) D1 mini Lite

#### **Khi nạp nhớ chọn đúng board LOLIN(WEMOS) D1 mini Lite**

Trong chương trình mã này, cần chú ý mục sau:

Trong hàm **reconnect()**: thay đổi chuỗi trong 2 lệnh sau

**client.connect(“đây là ID name, tự đặt”);** Ví dụ: “Client1”

**client.subscribe(“đây là topic subscribe từ web/app”);** Ví dụ: “button/#”

**\* Chú thích:** button(để phân biệt topic giữa các nhóm) /# (để bắt tất cả các topic con của button, trong bài này ta có 3 topic gồm: button/led, button/buzzer, button/fan).

**Các bạn chú ý đặt topic theo nhóm. Ví dụ: button1/# hoặc button2/#**

Mục đích của topic Sub này là nhận data mà web/app Pub để điều khiển led/buzzer/fan

```

// Kết nối tới MQTT Broker
void reconnect() {
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");

        if (client.connect("IoTLABClient")) { // tự đặt ID name
            Serial.println("connected");
            client.subscribe("button/#"); // tự đặt topic Sub th
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");
            delay(5000);
        }
    }
}
}

```

**Code nạp cho chip WiFi: LOLIN(WEMOS) D1 mini Lite:**

```

#ifdef ESP8266
#include <ESP8266WiFi.h> /* WiFi library for ESP8266 */
#else
#include <WiFi.h> /* WiFi library for ESP32 */
#endif
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <PubSubClient.h>

// Khai báo các chân RX và TX cho espsoftwareserial
#define RX_PIN D3
#define TX_PIN D4

// Tạo một đối tượng espsoftwareserial với tốc độ baud là 9600
SoftwareSerial mySerial(RX_PIN, TX_PIN);

#define wifi_ssid "IoT LAB"
#define wifi_password "kvt1ptit"
#define mqtt_server "10.170.69.245"
#define mqtt_port 1883

WiFiClient espClient;
PubSubClient client(espClient);

// Kết nối tới WiFi
void setup_wifi() {
    delay(10);

```

```

Serial.println();
Serial.print("Connecting to ");
Serial.println(wifi_ssid);

WiFi.begin(wifi_ssid, wifi_password);

while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
}

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}

// Kết nối tới MQTT Broker
void reconnect() {
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");

        if (client.connect("IoTLABClient")) { // tự đặt ID name cho Client . Ví dụ
"Client1 "
            Serial.println("connected");
            client.subscribe("Ví dụ button1/#"); // tự đặt topic Sub theo nhóm của bạn. Ví dụ
"button1/#". Mục đích để subscribe data bật/tắt led/còi từ web/app
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");
            delay(5000);
        }
    }
}

//-----Call back Method for Receiving MQTT message and Switch LED-----
void callback(char* topic, byte* payload, unsigned int length) {
    String incommingMessage = "";
    for (int i = 0; i < length; i++) incommingMessage += (char)payload[i];
    Serial.println("Message arived [" + String(topic) + "]: " + incommingMessage);

    DynamicJsonDocument docSub(1024);

    docSub["topic"] = topic;
    docSub["message"] = incommingMessage;

```

```

serializeJson(docSub, Serial);
Serial.println();

serializeJson(docSub, mySerial);
mySerial.println();
}

//-----Method for Publishing MQTT Messages-----
void publishMessage(const char* topic, String payload, boolean retained) {
    if (client.publish(topic, payload.c_str(), true))
        Serial.println("Message published [" + String(topic) + "]: " + payload);
}

void setup() {
    // Khởi tạo cổng nối tiếp phần cứng với tốc độ baud là 115200 (đây là Baudrate của
    // cổng Serial trên Arduino IDE)
    Serial.begin(115200);

    // Khởi tạo cổng nối tiếp phần mềm với tốc độ baud là 9600 ( đây là Baudrate của
    // cổng mySerial để giao tiếp với board Arduino Uno)
    mySerial.begin(9600);

    setup_wifi();
    client.setServer(mqtt_server, mqtt_port);
    client.setCallback(callback);
}

DynamicJsonDocument doc(1024);

void loop() {
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
    // nhận data từ Arduino thông qua Tx-Rx và sau đó publish data lên MQTT Broker
    if (mySerial.available()) {
        String data = mySerial.readStringUntil('\n');
        DeserializationError error = deserializeJson(doc, data);
        char mqtt_message[1024];

        serializeJson(doc, mqtt_message);
        publishMessage("Ví dụ Group1", mqtt_message, true); // tự đặt topic Pub theo
        nhóm của bạn. Ví dụ "Group1".
    }
}

```





## Code nạp cho Arduino Uno:

Có nhiệm vụ là nhận data từ Chip WiFi D1 mini Lite lấy từ MQTT Broker để bật tắt Led/ Còi/Quạt thông qua giao tiếp USART

### Trong chương trình mã này, cần chú ý nội dung sau:

Trong hàm **loop()** khi arduino nhận data từ chip WiFi ta phân tích bản tin để có thể tiến hành điều khiển các thiết bị chấp hành như: bật/ tắt Led/ Còi/Quạt.

**Trong code nạp cho chip WiFi là: topicSub : button/#**

```
if (client.connect("IoTLABClient")) { // tự đặt ID
    Serial.println("connected");
    client.subscribe("button/#"); // tự đặt topic Sub
} else {
```

Thì ở code nạp cho Arduino ta thực hiện điền như sau:

```
if (!error) {
    String topic = docBtn["topic"];
    String message = docBtn["message"];

    if (topic == "button/led") { // ta sửa theo top
    }

    if (topic == "button/buzzer") { // ta sửa theo
    }

    if (topic == "button/fan") { // ta sửa theo to
    }
} else {
    Serial.print("deserializeJson() failed: ");
    Serial.println(error.c_str());
}
```

**Nếu Topic subscribe là: button1/#**

**Thì ta có các topic con là: button1/led, button1/buzzer , button1/fan**

### Code tham khảo:

```
#include <DHT.h>
#include <ArduinoJson.h>
#include <SoftwareSerial.h>

#define LED 7      // Data LED
#define Buzzer 8   // Data Buzzer
#define Fan 9      // Data Fan

DHT dht(DHT_Data, DHT22); // DHT11

#define RX_PIN 10
#define TX_PIN 11
SoftwareSerial mySerial(RX_PIN, TX_PIN);

void setup() {
    Serial.begin(9600);
    mySerial.begin(9600);
    while (!Serial) delay(1);

    pinMode(LED, OUTPUT);
    pinMode(Buzzer, OUTPUT);
    pinMode(Fan, OUTPUT);

    dht.begin();
}

unsigned long timeUpdata = millis();

DynamicJsonDocument doc(1024);
DynamicJsonDocument docBtn(1024);

void loop() {
    if (millis() - timeUpdata > 2000) {
        doc["led"] = digitalRead(LED);
        doc["buzzer"] = digitalRead(Buzzer);
        doc["fan"] = digitalRead(Fan);

        // gửi data thông qua cổng giao tiếp mềm cho Chip WiFi
        serializeJson(doc, mySerial);
        mySerial.println();

        // In ra màn hình Serial
        serializeJson(doc, Serial);
        Serial.println();
    }
}
```

```

    timeUpdata = millis();
}
if (mySerial.available()) {
    String data = mySerial.readStringUntil('\n');
    DeserializationError error = deserializeJson(docBtn, data);

    serializeJson(docBtn, Serial);
    Serial.println();

    if (!error) {
        String topic = docBtn["topic"];
        String message = docBtn["message"];

        if (topic == "Ví dụ button1/led") { // ta sửa theo topic sub bên code cho chip
WiFi
            if (message == "1") {
                // Thực hiện hành động bật đèn LED
                digitalWrite(LED, HIGH);
            } else if (message == "0") {
                // Thực hiện hành động tắt đèn LED
                digitalWrite(LED, LOW);
            }
            Serial.println("Bật/tắt đèn LED: " + String(message));
        }
        if (topic == "Ví dụ button1/buzzer") { // ta sửa theo topic sub bên code cho chip
WiFi
            if (message == "1") {
                // Thực hiện hành động bật còi
                digitalWrite(Buzzer, HIGH);
            } else if (message == "0") {
                // Thực hiện hành động tắt còi
                digitalWrite(Buzzer, LOW);
            }
            Serial.println("Bật/tắt buzzer: " + String(message));
        }

        if (topic == "Ví dụ button1/fan") { // ta sửa theo topic sub bên code cho chip
WiFi
            if (message == "1") {
                // Thực hiện hành động bật còi
                digitalWrite(Fan, HIGH);
            } else if (message == "0") {
                // Thực hiện hành động tắt còi
                digitalWrite(Fan, LOW);
            }
            Serial.println("Bật/tắt fan: " + String(message));
        }
    }
}

```

```

    }
  } else {
    Serial.print("deserializeJson() failed: ");
    Serial.println(error.c_str());
  }
}
}
}

```

Tiếp theo thực hiện điều khiển các thiết bị chấp hành thông qua web:

**Bật đèn:** Tiến hành pub 1 bản tin từ web với nội dung bật đèn:

**Chú ý topic Pub từ web phải giống với topic Sub ở code mà chip WiFi đã cài đặt:**

Trước tiên ta sẽ điền vào mục Topic to Subscribe là: Group1

Và sau đó bấm Subscribe để lấy data từ Broker để hiển thị trạng thái của các thiết bị chấp hành.

Topic to Subscribe:

Group1

Subscribe Unsubscribe

Data Subscribed:

{"temperature":null,"humidity":null,"gas":87,"led":0,"buzzer":0,"fan":0}

Sau khi Subscribe topic ta nhận được 1 chuỗi json về giá trị các thông số trạng thái của led, buzzer, fan: với các trạng thái 1 là bật và 0 là tắt.

Tiếp đến ta thực hiện điều khiển các thiết bị chấp hành.

Ví dụ: trong code nạp cho chip WiFi để topic Sub là : button/#

```

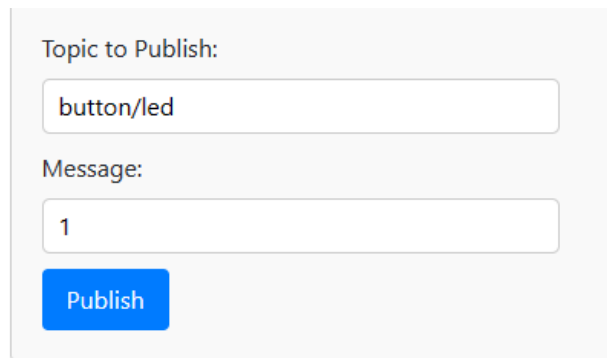
if (client.connect("IoTLABClient"))
  Serial.println("connected");
  client.subscribe("button/#"); //

```

Tương tự trong code nạp cho Arduino :

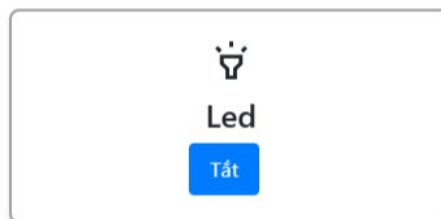
```
if (topic == "button/led") { // ta sử dụng
  if (message == "1") {
    // Thực hiện hành động bật đèn LED
  }
}
```

Thì trên trang web các bạn nhập vào ô Topic to Publish là : **button/led** và message: **1 (value)**



Sau khi nhập xong ta bấm Publish

Trạng thái của đèn khi bật trên Dashboard



**Bật còi:** Tiến hành pub 1 bản tin từ web với nội dung bật còi:

Trong code nạp cho Arduino:

```
if (topic == "button/buzzer") {
  if (message == "1") {
    // Thực hiện hành động bật còi
  }
}
```

Thì trên trang web các bạn nhập vào ô Topic to Publish là : **button/buzzer** và message: **1 (value)**

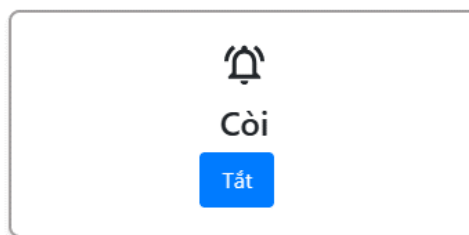
Topic to Publish:

Message:

**Publish**

Sau khi nhập xong ta bấm Publish

Trạng thái của còi khi bật trên DashBoard



**Bật còi:** Tiến hành pub 1 bản tin từ web với nội dung bật quạt: ( **Làm tương tự** )

Thì trên trang web các bạn nhập vào ô Topic to Publish là : button/fan và message: 1 (value)

Topic to Publish:

Message:

**Publish**