BÁO CÁO THỰC HÀNH

**Bài thực hành số xx: Tên bài thực hành**

**Môn học:** <Tên môn học>

**Lớp:** NT…

**THÀNH VIÊN THỰC HIỆN (Nhóm xx):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** |  | **Điểm tự đánh giá** |
| 1 | Phạm Thiều Gia Khang | 21520967 |  |
| 2 |  |  |
|  |  |  |  |

**ĐÁNH GIÁ KHÁC:**

|  |  |
| --- | --- |
| Tổng thời gian thực hiện |  |
| Phân chia công việc |  |
| Ý kiến *(nếu có)*  + Khó khăn  + Đề xuất, kiến nghị |  |

Phần bên dưới của báo cáo này là báo cáo chi tiết của nhóm thực hiện

MỤC LỤC

[A. BÁO CÁO CHI TIẾT 2](#_Toc177398261)

[1. Nội dung 1 2](#_Toc177398262)

[a. Nội dung a 2](#_Toc177398263)

[b. Nội dung b 3](#_Toc177398264)

[2. Nội dung 2 3](#_Toc177398265)

[a. Nội dung a 3](#_Toc177398266)

[B. TÀI LIỆU THAM KHẢO 3](#_Toc177398267)

# BÁO CÁO CHI TIẾT

## Trao đổi dữ liệu sử dụng MQTT

### Cài đặt:

Khởi tạo một máy ảo với hệ điều hành Ubuntu 22.04 làm MQTT Broker, cài đặt Mosquitto

MQTT trên máy ảo.

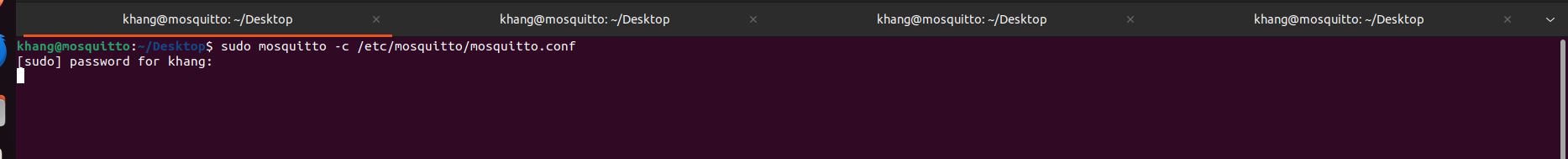
Thêm vào trong file mosquitto.conf để Broker có thể nghe được các các kết nối từ bên ngoài từ port 1883:

listener 1883

allow\_anonymos true

Chạy Mosquitto Broker qua lệnh:

sudo mosquito –c /etc/mosquito/mosquito.conf



Khởi tạo 3 topic để nhận dữ liệu lần lượt từ Node 1 (wemos1), Node 2 (wemos2) và topic led để nhận thông tin bật tắt đèn:

mosquito\_sub –t “wemos1/sensor”

mosquito\_sub –t “wemos2/sensor”

mosquito\_sub –t “wemos/led”

Code (Tương tự nhau ở hay node chỉ thay đổi tên topic):

#include <ESP8266WiFi.h>

#include <PubSubClient.h>

#include <DHT.h>

#include <BH1750.h>

#include <Wire.h>

#define LED1 D4

#define LED2 D5

#define LED3 D6

#define LED4 D7

#define LED5 D8

#define BTN D0

bool state = true;

#define DHTPIN D3

#define DHTTYPE DHT22

bool ledState = false;  // Trạng thái LED

bool buttonPressed = false;

unsigned long buttonPressTime = 0;  // Lưu thời điểm bắt đầu nhấn

const long holdTime = 3000;

const char\* ssid = "WemosTest";

const char\* password = "01072003";

# Địa chỉ của MQTT Broker

const char\* mqtt\_server = "192.168.43.143";

const int mqtt\_port = 1883;

const char\* mqtt\_client\_id = "wemos\_d1";

WiFiClient espClient;

PubSubClient client(espClient);

DHT dht(DHTPIN, DHTTYPE);

BH1750 lightMeter;

# Hàm kết nối với wifi

void setup\_wifi() {

    delay(10);

    Serial.print("Connecting WiFi...");

    WiFi.begin(ssid, password);

    while (WiFi.status() != WL\_CONNECTED) {

        delay(500);

        Serial.print(".");

    }

    Serial.println("\nWiFi connected!");

    Serial.print("IP: ");

    Serial.println(WiFi.localIP());

}

#Hàm nhận phản hồi từ Broker

void callback(char\* topic, byte\* payload, unsigned int length) {

    String message;

    for (int i = 0; i < length; i++) {

        message += (char)payload[i];

    }

    Serial.print("MQTT Received from wemos1: ");

    Serial.print(topic);

    Serial.print(" - Content: ");

    Serial.println(message);

  // Đèn vàng sáng khi có gói tin gửi tới

    digitalWrite(LED4, HIGH);

    delay(500);

    digitalWrite(LED4, LOW);

    // Xử lý MQTT để bật/tắt LED từ xa

    if (String(topic) == "wemos/led") {

        if (message == "ON") {

            digitalWrite(LED5, HIGH);

            ledState = true;

        } else if (message == "OFF") {

            digitalWrite(LED5, LOW);

            ledState = false;

        }

    }

}

# Hàm kết nối với Broker và khởi tạo lại kết nối khi mất kết nối

void reconnect() {

    while (!client.connected()) {

        Serial.print("Connecting to MQTT...");

        if (client.connect(mqtt\_client\_id)) {

            Serial.println("MQTT connected!");

            client.subscribe("wemos2/sensor");

            client.subscribe("wemos/led");

        } else {

            Serial.print("Failed, error code: ");

            Serial.print(client.state());

            Serial.println(" Try again in 5 seconds.");

            delay(5000);

        }

    }

}

void setup() {

    Serial.begin(9600);

    setup\_wifi();

    client.setServer(mqtt\_server, mqtt\_port);

    client.setCallback(callback);

    pinMode(LED1, OUTPUT);

    pinMode(LED2, OUTPUT);

    pinMode(LED3, OUTPUT);

    pinMode(LED4, OUTPUT);

    pinMode(LED5, OUTPUT);

    pinMode(BTN, INPUT\_PULLUP);

    dht.begin();

    Wire.begin(D2, D1);

    lightMeter.begin();

    Serial.println(F("BH1750 Test begin"));

}

void loop() {

    if (!client.connected()) {

        reconnect();

    }

    client.loop();

        bool buttonState = digitalRead(BTN);

    if (buttonState == HIGH) {

        if (!buttonPressed) {

            buttonPressed = true;

            buttonPressTime = millis();

        }

        // Nếu giữ nút >= 1 giây thì đổi trạng thái LED

        else if (millis() - buttonPressTime >= holdTime) {

            ledState = !ledState;

            digitalWrite(LED5, ledState ? HIGH : LOW);

            Serial.print("LED ");

            Serial.println(ledState ? "ON" : "OFF");

            client.publish("wemos/led", ledState ? "ON" : "OFF");

            buttonPressed = false;

        }

    }

    else {

        buttonPressed = false;

    }

    # Đọc dữ liệu từ cảm biến

    float h = dht.readHumidity();

    float t = dht.readTemperature();

    float lux = lightMeter.readLightLevel();

    Serial.print(F("Humidity: "));

    Serial.print(h);

    Serial.print(F("%  Temperature: "));

    Serial.print(t);

    Serial.print("°C  Light: ");

    Serial.print(lux);

    Serial.println(" lx");

    digitalWrite(LED1, LOW);

    digitalWrite(LED2, HIGH);

# Khi độ sáng quá thấp thì bật đèn trắng (LED1)

    if (lux <= 50) {

        digitalWrite(LED1, HIGH);

    }

# Nhiệt độ và độ ẩm ngoài ngưỡng an toàn thì bật đèn đỏ (LED2)

    if (h >= 60 && h <= 70 && t >= 23 && t <= 28) {

        digitalWrite(LED2, LOW);

    }

    // Gửi dữ liệu cảm biến lên MQTT khi gửi thành công thì đèn LED3 sáng

digitalWrite(LED3, LOW);

    char payload[50];

    sprintf(payload, "{\"humidity\": %.2f, \"temperature\": %.2f, \"light\": %.2f}", h, t, lux);

    client.publish("wemos1/sensor", payload);

    digitalWrite(LED3, HIGH);

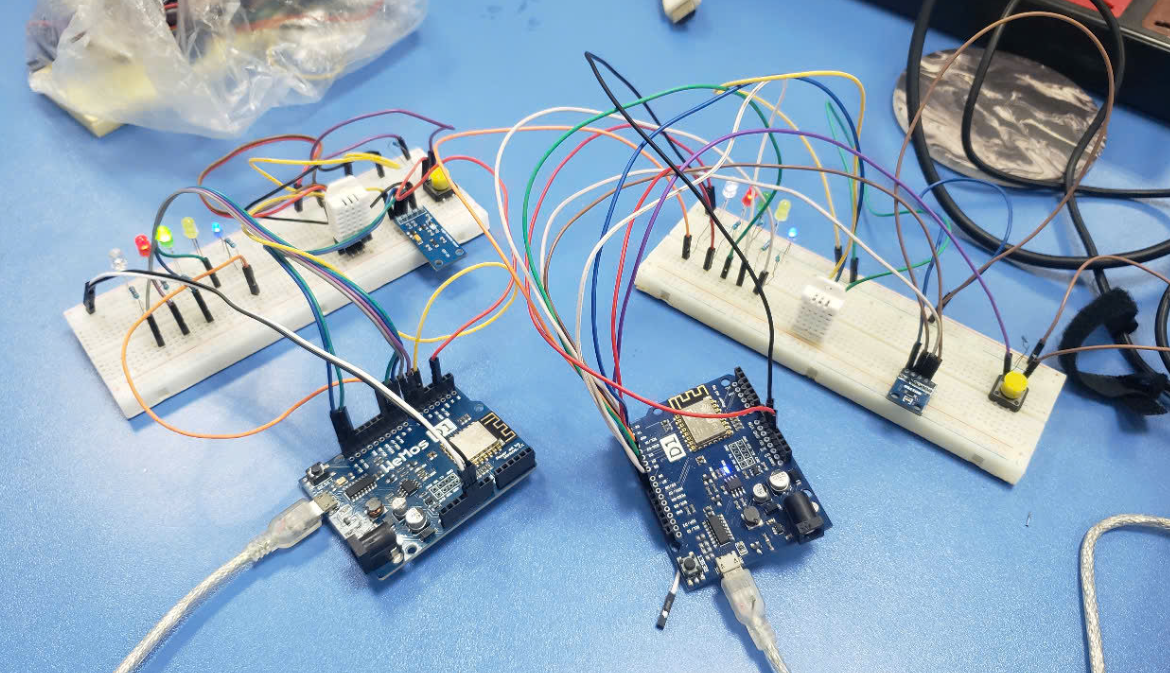
# 3 giây gửi 1 lần

    delay(3000);

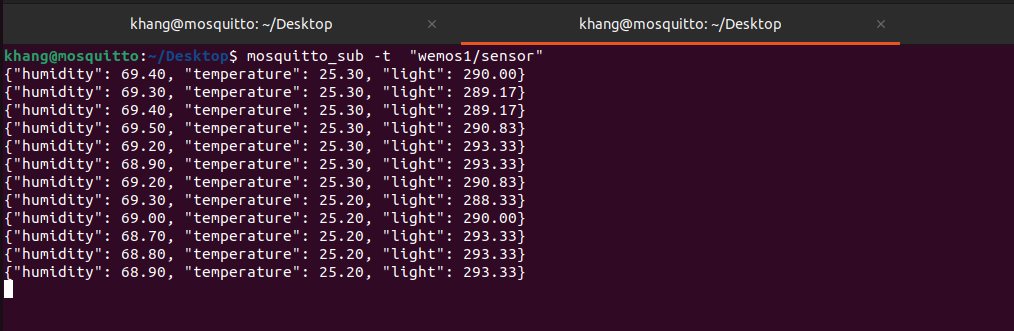
}

### Demo

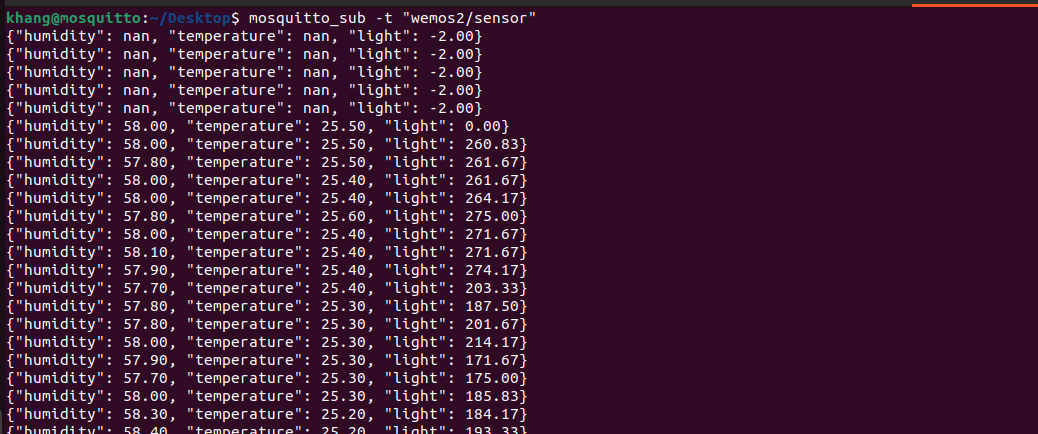
Mô hình:



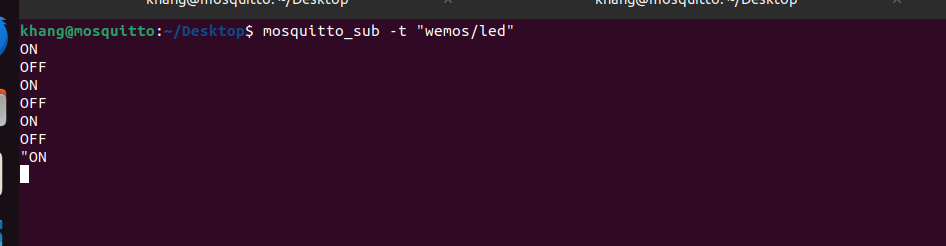
Topic wemos1/sensor:



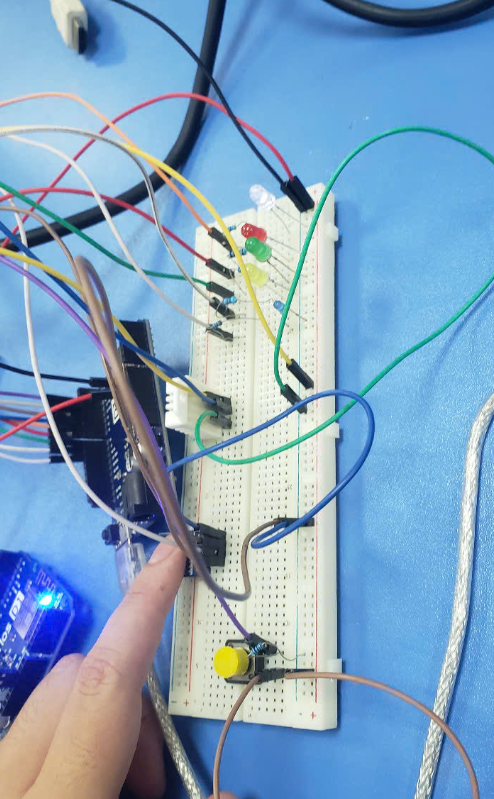
Topic wemos2/sensor:



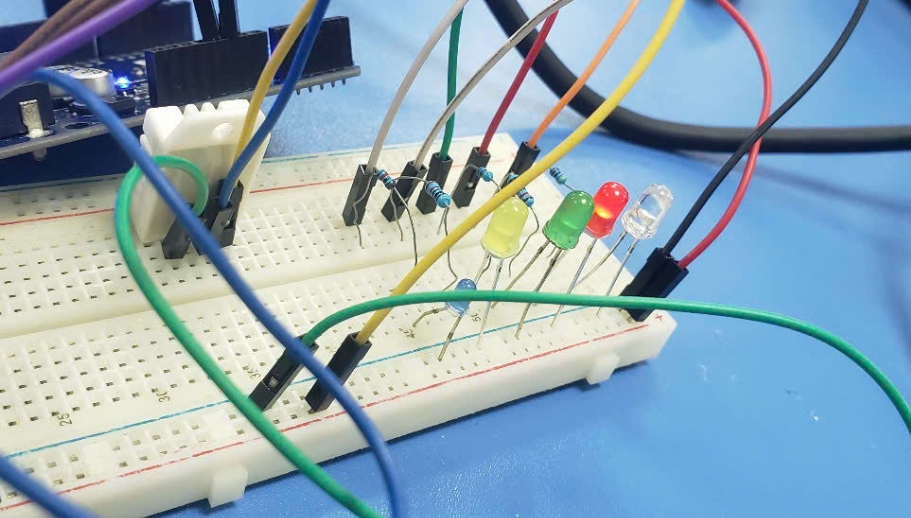
Topic wemos/led:



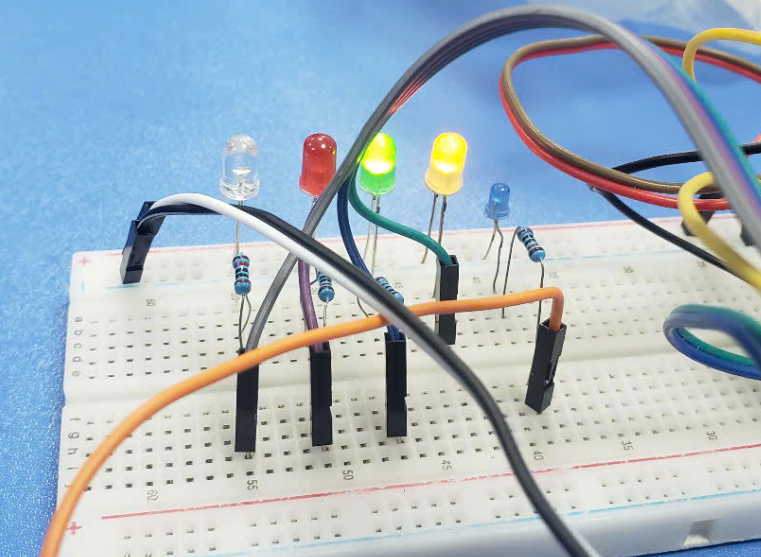
Trường hợp 1: Khi độ sang qua thấp, đèn trắng sang:



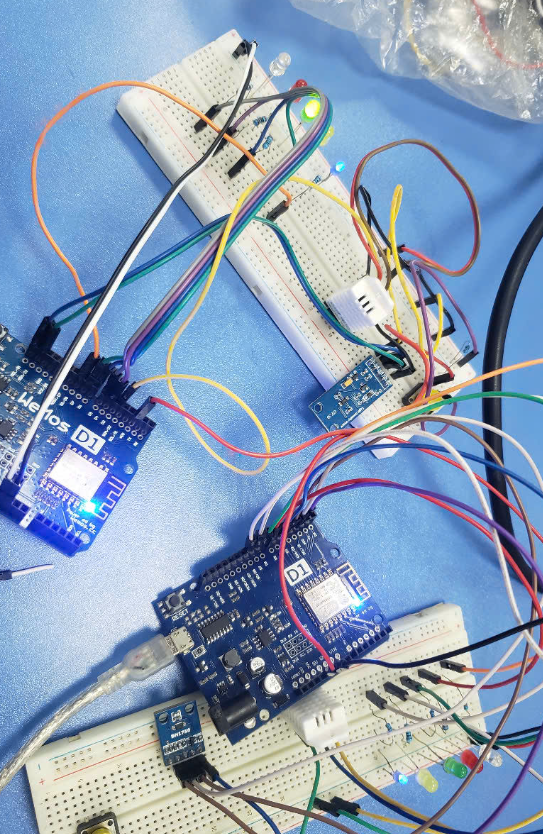
Trường hợp 2: Đèn đỏ sang khi nhiệt độ ẩm nằm ngoài ngưỡng an toàn:



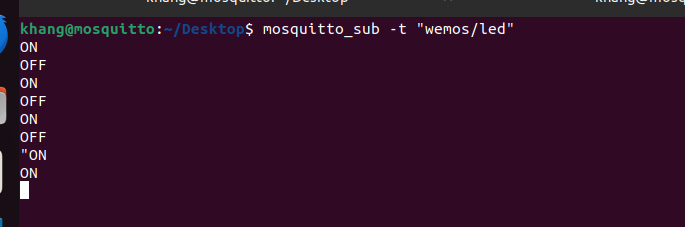
Trường hợp 3 và 4: Đèn xanh sang khi thiết bị đang gửi tín hiệu, đèn vàng sẽ sáng khi có tín hiệu gửi tới:



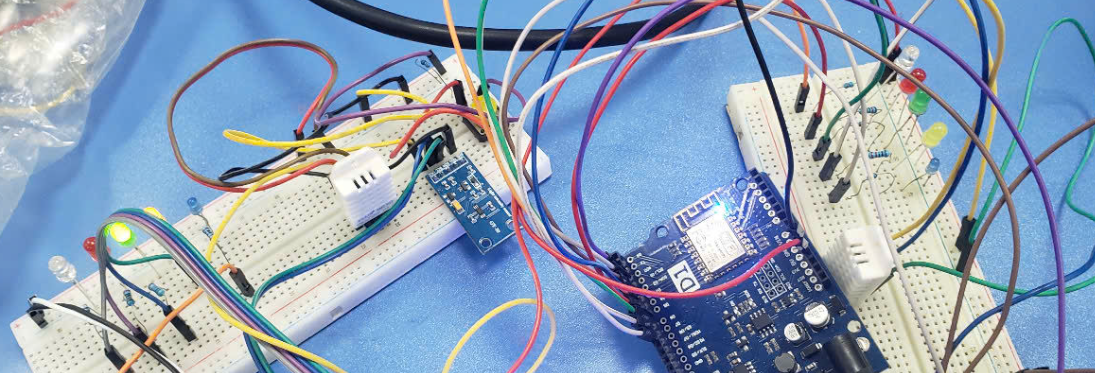
Nút bấm đồng bộ giữa 2 Node, khi ledState là true, cả đèn cùng sáng:



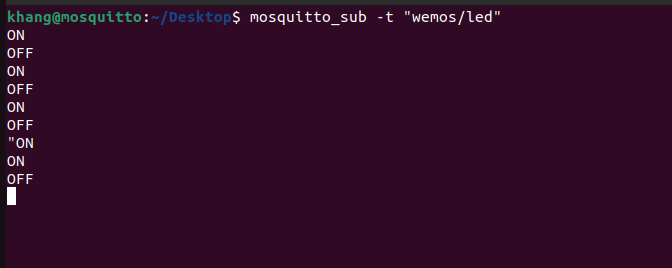
Borker nhận tín hiệu ON từ 1 Node:



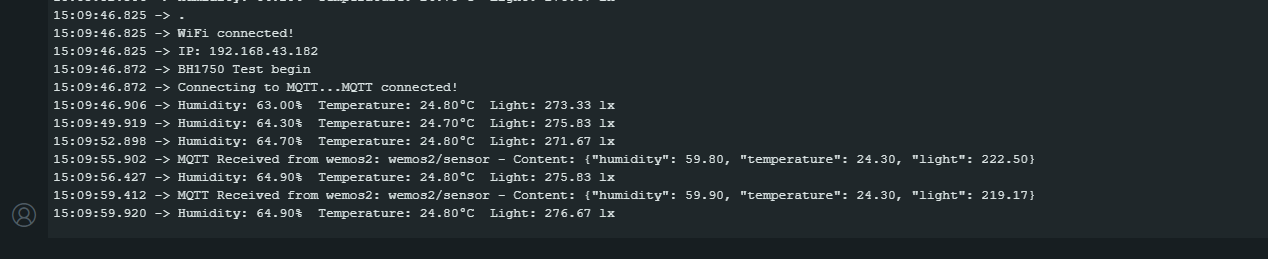
Nhấn giữ tiếp 3s để tắt đèn từ 1 trong 2 Node:

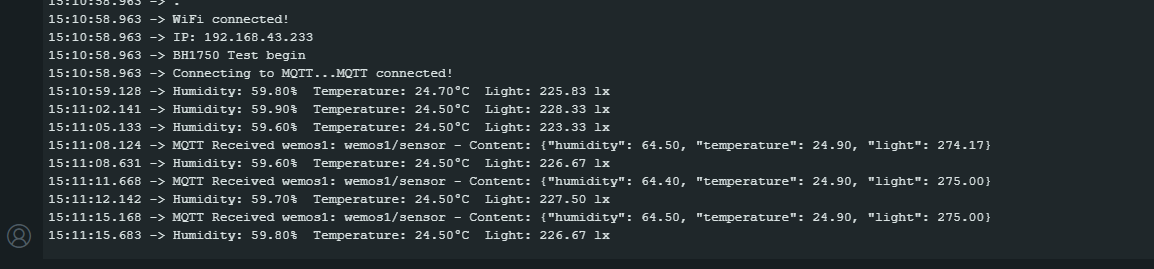


Broker hiển thị lệnh OFF:



Serial của từng Node nhận callback từ Broker và hiển thị thông tin Sensor của nó:





## Nội dung 2

### Nội dung a

1. Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old.

# TÀI LIỆU THAM KHẢO