

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Khoa Công Nghệ Thông Tin I



Báo Cáo Bài Tập Lớn

Môn học: IoT và ứng dụng

Giáo viên hướng dẫn: Nguyễn Quốc Uy

Họ và tên	: Hoàng Anh Quân
Mã sinh viên	: B21DCCN606
Nhóm lớp	: 06

Hà Nội, 2024

I. Giới thiệu.....	4
1. Đặt vấn đề.....	4
2. Mục tiêu và phạm vi đề tài.....	4
3. Định hướng giải pháp.....	4
4. Bố cục bài tập lớn.....	4
II. Tính năng.....	6
1. Giám sát cảm biến:.....	6
2. Điều khiển thiết bị từ xa:.....	6
3. Tự động hóa dựa trên ngưỡng:.....	6
4. Cập nhật dữ liệu theo thời gian thực:.....	6
5. Chế độ xem chi tiết:.....	6
6. Giao tiếp MQTT:.....	7
7. Giao diện người dùng thân thiện:.....	7
III. Giao diện.....	8
1. Trang Home.....	8
2. Trang Sensor Data.....	9
3. Trang Device History.....	9
4. Trang Profile.....	10
IV. Thiết kế hệ thống.....	11
1. Cơ Sở Dữ Liệu.....	11
2. Hệ Thống.....	11
3. Luồng Dữ Liệu.....	12
4. Các thiết bị phần cứng.....	12
5. MQTT broker.....	12
6. Tài Liệu API.....	13
V. Source code.....	14
1. Arduino:.....	14
2. Front end:.....	19
1. index.html:.....	19
2. lichSu.html:.....	23
3. thongKe.html:.....	28
4. myProfile.html:.....	32
5. Backend:.....	38
VI. Kết quả.....	50
1. Tổng quan.....	50
2. Đo lường các chỉ số như nhiệt độ, độ ẩm, ánh sáng thời gian thực.....	50
3. Điều khiển các thiết bị điện.....	50
4. Lưu trữ và đưa ra dữ liệu cho người dùng.....	51
5. Đánh giá chung.....	51
Kết Luận.....	52

I. Giới thiệu

1. Đặt vấn đề

Với sự phát triển nhanh chóng của công nghệ và nhu cầu kết nối các thiết bị thông minh, Internet of Things (IoT) đã trở thành một lĩnh vực đầy tiềm năng. IoT cho phép các thiết bị vật lý giao tiếp và trao đổi dữ liệu với nhau thông qua internet, giúp tạo ra hệ thống tự động hoá và cải thiện hiệu quả trong nhiều lĩnh vực, từ nhà thông minh đến công nghiệp 4.0. Dự án của chúng tôi được xây dựng nhằm giải quyết một vấn đề thực tiễn liên quan đến việc giám sát và điều khiển từ xa các thiết bị điện tử thông qua nền tảng IoT.

2. Mục tiêu và phạm vi đề tài

Mục tiêu của dự án là thiết kế và triển khai một hệ thống IoT giúp giám sát các thông số môi trường (nhiệt độ, độ ẩm, ánh sáng) và điều khiển các thiết bị điện trong thời gian thực. Phạm vi của đề tài bao gồm:

Xây dựng hệ thống phần cứng bao gồm các cảm biến và thiết bị điều khiển.

Thiết kế giao diện người dùng để hiển thị dữ liệu và điều khiển từ xa.

Tích hợp giao thức truyền thông MQTT, để đảm bảo việc giao tiếp giữa các thành phần hệ thống.

3. Định hướng giải pháp

Dự án sẽ sử dụng các cảm biến phổ biến như DHT11 để đo nhiệt độ và độ ẩm, cảm biến ánh sáng để đo cường độ ánh sáng. Giao tiếp giữa các thành phần sẽ được thực hiện thông qua giao thức MQTT. Để điều khiển và giám sát, một ứng dụng web sẽ được phát triển, cho phép người dùng quản lý thiết bị từ xa thông qua mạng internet.

4. Bố cục bài tập lớn

Bài tập lớn này được chia thành 5 chương:

Chương 1: Giới thiệu về vấn đề, mục tiêu, và phạm vi của đề tài, cũng như giải pháp định hướng.

Chương 2: Thiết kế giao diện và tổng thể hệ thống.

Chương 3: Trình bày chi tiết về các thiết bị phần cứng, giao thức truyền thông, và các thành phần phần mềm của hệ thống.

Chương 4: Phần code, bao gồm code nhúng, code backend, và code frontend.

Chương 5: Kết quả thử nghiệm và đánh giá hệ thống.

II. Tính năng

Hệ thống web IoT được phát triển với nhiều tính năng nổi bật, đáp ứng nhu cầu giám sát và điều khiển thiết bị thông minh của người dùng. Dưới đây là một số tính năng chính:

1. Giám sát cảm biến:

Hệ thống liên tục thu thập và hiển thị dữ liệu từ các cảm biến, bao gồm nhiệt độ, độ ẩm và mức ánh sáng. Người dùng có thể theo dõi các thông số này theo thời gian thực để đưa ra quyết định chính xác.

2. Điều khiển thiết bị từ xa:

Người dùng có thể điều khiển các thiết bị như quạt, điều hòa và đèn từ xa thông qua giao diện web. Việc sử dụng checkbox làm công tắc giúp người dùng dễ dàng bật/tắt thiết bị theo ý muốn.

3. Tự động hóa dựa trên ngưỡng:

Hệ thống tự động điều chỉnh trạng thái của các thiết bị dựa trên các ngưỡng cảm biến đã định trước. Ví dụ, quạt sẽ tự động bật khi độ ẩm vượt quá 80%, điều hòa sẽ kích hoạt khi nhiệt độ vượt quá 35°C, và đèn sẽ sáng khi mức ánh sáng giảm xuống dưới 100.

4. Cập nhật dữ liệu theo thời gian thực:

Với việc sử dụng AJAX, trang web sẽ tự động cập nhật các phần dữ liệu mà không cần làm mới toàn bộ trang. Điều này giúp cải thiện trải nghiệm người dùng và tăng cường tính tương tác của hệ thống.

5. Chế độ xem chi tiết:

Người dùng có thể xem chi tiết các biểu đồ và số liệu thống kê thông qua một cửa sổ modal. Tính năng này cho phép người dùng nắm bắt thông tin một cách rõ ràng và trực quan hơn.

6. Giao tiếp MQTT:

Hệ thống sử dụng giao thức MQTT để giao tiếp hiệu quả giữa máy chủ và thiết bị. Điều này không chỉ đảm bảo độ ổn định trong việc truyền tải dữ liệu mà còn cho phép xử lý lệnh nhanh chóng.

7. Giao diện người dùng thân thiện:

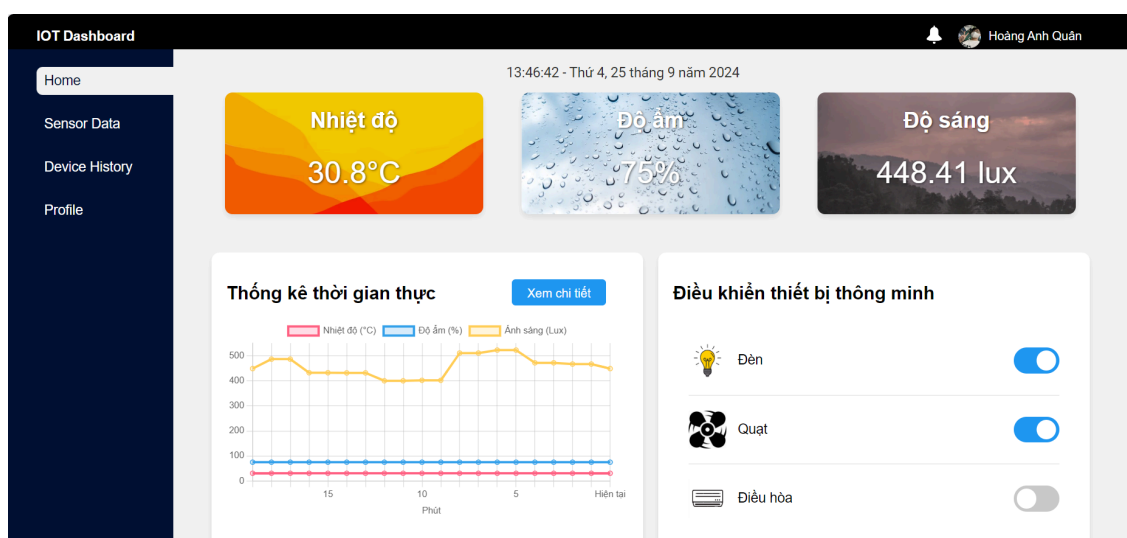
Giao diện web được thiết kế trực quan và dễ sử dụng, giúp người dùng không cần có kiến thức kỹ thuật vẫn có thể dễ dàng tương tác và điều khiển các thiết bị.

III. Giao diện

Giao diện của hệ thống web IoT được thiết kế để cung cấp trải nghiệm người dùng trực quan và dễ dàng trong việc tương tác với các thiết bị thông minh. Hệ thống gồm bốn trang chính: **Home**, **Sensor Data**, **Device History**, và **Profile**. Mỗi trang đều có chức năng riêng biệt, giúp người dùng dễ dàng truy cập và quản lý các thông tin cần thiết.

1. Trang Home

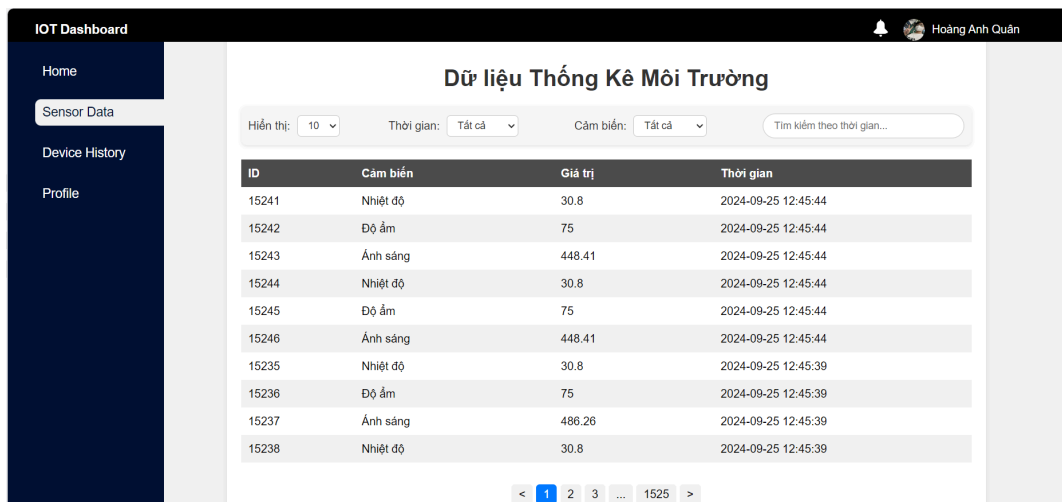
- Trang chính của hệ thống hiển thị tổng quan về tình trạng của các thiết bị và thông số cảm biến hiện tại. Người dùng có thể nhanh chóng thấy các chỉ số như nhiệt độ, độ ẩm, và mức ánh sáng.
- Ở trung tâm, có một biểu đồ thời gian thực hiển thị dữ liệu về nhiệt độ, độ ẩm, và độ sáng, giúp người dùng theo dõi các chỉ số này trong một khoảng thời gian cụ thể. Bên cạnh đó, người dùng cũng có thể xem chi tiết hơn bằng cách nhấn vào nút "Xem chi tiết".
- Phía bên phải giao diện là phần "Điều khiển thiết bị thông minh," nơi người dùng có thể bật/tắt các thiết bị như đèn, quạt, và điều hòa bằng các công tắc chuyển đổi.
- Tổng thể, giao diện này cung cấp một cái nhìn tổng quan và khả năng điều khiển hệ thống IoT một cách hiệu quả và thuận tiện.



Hình 3.1: Hình ảnh giao diện trang Home

2. Trang Sensor Data

- Trang này tập trung vào việc hiển thị dữ liệu từ các cảm biến trong thời gian thực. Các thông số như nhiệt độ, độ ẩm, và mức ánh sáng được trình bày dưới dạng bảng và biểu đồ, giúp người dùng dễ dàng theo dõi và phân tích.
- Tính năng lọc và tìm kiếm cũng được tích hợp để người dùng có thể nhanh chóng tìm kiếm thông tin cần thiết.

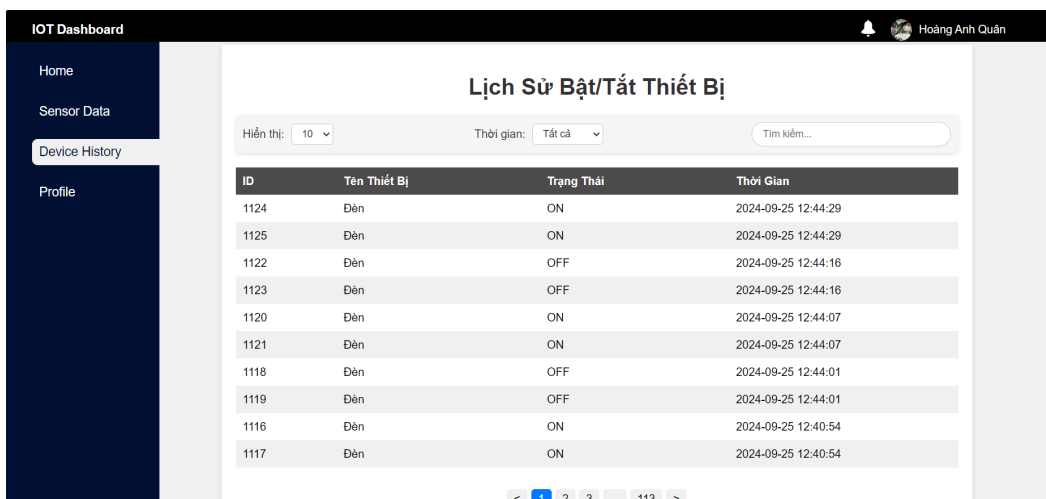


ID	Cảm biến	Giá trị	Thời gian
15241	Nhiệt độ	30.8	2024-09-25 12:45:44
15242	Độ ẩm	75	2024-09-25 12:45:44
15243	Ánh sáng	448.41	2024-09-25 12:45:44
15244	Nhiệt độ	30.8	2024-09-25 12:45:44
15245	Độ ẩm	75	2024-09-25 12:45:44
15246	Ánh sáng	448.41	2024-09-25 12:45:44
15235	Nhiệt độ	30.8	2024-09-25 12:45:39
15236	Độ ẩm	75	2024-09-25 12:45:39
15237	Ánh sáng	486.26	2024-09-25 12:45:39
15238	Nhiệt độ	30.8	2024-09-25 12:45:39

Hình 3.2: Hình ảnh giao diện trang Sensor data

3. Trang Device History

- Trang Device History cho phép người dùng xem lại lịch sử hoạt động của các thiết bị. Các thông tin như thời gian bật/tắt, trạng thái thiết bị, và các sự kiện quan trọng khác được ghi lại và trình bày một cách trực quan.
- Tính năng lọc và tìm kiếm cũng được tích hợp để người dùng có thể nhanh chóng tìm kiếm thông tin cần thiết.



ID	Tên Thiết Bị	Trạng Thái	Thời Gian
1124	Đèn	ON	2024-09-25 12:44:29
1125	Đèn	ON	2024-09-25 12:44:29
1122	Đèn	OFF	2024-09-25 12:44:16
1123	Đèn	OFF	2024-09-25 12:44:16
1120	Đèn	ON	2024-09-25 12:44:07
1121	Đèn	ON	2024-09-25 12:44:07
1118	Đèn	OFF	2024-09-25 12:44:01
1119	Đèn	OFF	2024-09-25 12:44:01
1116	Đèn	ON	2024-09-25 12:40:54
1117	Đèn	ON	2024-09-25 12:40:54

Hình 3.3: Hình ảnh giao diện trang Device history

4. Trang Profile

Trang "Profile" cung cấp thông tin chi tiết về cá nhân người dùng. Giao diện được chia thành ba phần chính, giúp người dùng dễ dàng theo dõi và cập nhật thông tin:

1. Phần thông tin cá nhân (bên trái):

- Hiển thị ảnh đại diện của người dùng cùng với tên và trường học.
- Bên dưới là các liên kết mạng xã hội của người dùng.

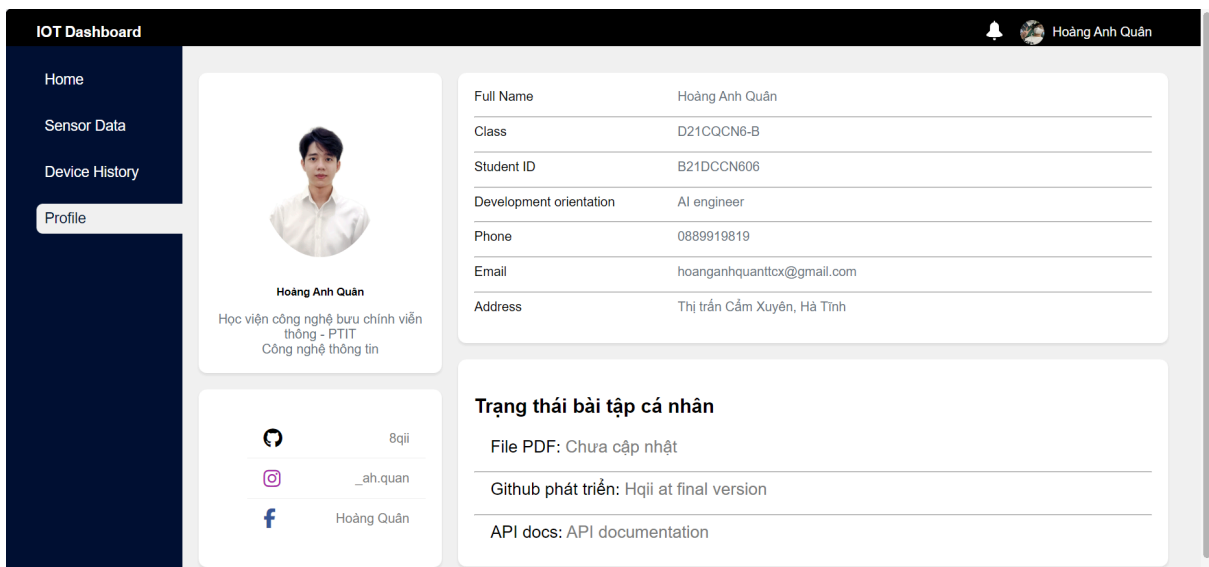
2. Phần thông tin chi tiết (bên phải):

- Hiển thị các thông tin cá nhân cụ thể của người dùng

3. Phần trạng thái bài tập cá nhân (phía dưới bên phải):

- Cho biết tình trạng cập nhật của các tài liệu liên quan đến bài tập cá nhân như file PDF, Github, API docs.

Giao diện này cung cấp một cái nhìn tổng quan và chi tiết về thông tin cá nhân, học vấn, và trạng thái dự án cá nhân của người dùng, giúp quản lý và truy cập thông tin một cách hiệu quả.



Hình 3.4: Hình ảnh giao diện trang Profile

IV. Thiết kế hệ thống

1. Cơ Sở Dữ Liệu

Cơ sở dữ liệu của hệ thống được thiết kế để lưu trữ thông tin về các cảm biến, trạng thái thiết bị, và các bình luận từ Facebook. Sử dụng SQLite để thực hiện cơ sở dữ liệu, với các bảng chính như sau:

- **Bảng `sensors`**: Lưu trữ dữ liệu cảm biến.
 - `id` INTEGER PRIMARY KEY AUTOINCREMENT,
 - `sensor` TEXT NOT NULL,
 - `value` REAL,
 - `time` TIMESTAMP DEFAULT (datetime('now', 'localtime'))
- **Bảng `devices`**: Lưu trữ thông tin về các thiết bị.
 - `id` INTEGER PRIMARY KEY AUTOINCREMENT,
 - `device_name` TEXT NOT NULL,
 - `status` TEXT NOT NULL,
 - `time` TEXT DEFAULT (datetime('now', 'localtime'))
- **Bảng `notification`**: Lưu trữ thông báo hệ thống.
 - `id` INTEGER PRIMARY KEY AUTOINCREMENT,
 - `message` TEXT NOT NULL,
 - `timestamp` DATETIME DEFAULT (DATETIME(CURRENT_TIMESTAMP, '+7 hours'))

2. Hệ Thống

Hệ thống IoT của bạn bao gồm các thành phần chính sau:

- **Frontend**: Được phát triển bằng HTML, CSS, và JavaScript, cho phép người dùng tương tác với hệ thống qua giao diện web. Giao diện này bao gồm:
 - Trang chính (Home): Hiển thị thông tin tổng quan về các thiết bị và cảm biến.
 - Trang dữ liệu cảm biến: Hiển thị dữ liệu thống kê cảm biến.
 - Trang lịch sử thiết bị: Lưu trữ và hiển thị lịch sử hoạt động của các thiết bị.
 - Trang hồ sơ: Quản lý thông tin người dùng.
- **Backend**: Sử dụng Flask để xử lý yêu cầu từ frontend, tương tác với cơ sở dữ liệu, Esp32 qua giao thức MQTT và cung cấp API cho các chức năng của hệ thống.
- **ESP32**: Thiết bị IoT sử dụng để kết nối với các cảm biến và điều khiển các thiết bị (quạt, điều hòa, đèn).

3. Luồng Dữ Liệu

Luồng dữ liệu trong hệ thống diễn ra như sau:

1. **Thu thập dữ liệu cảm biến:** Các cảm biến gửi dữ liệu về nhiệt độ, độ ẩm và ánh sáng tới ESP32.
2. **Gửi dữ liệu tới Backend:** ESP32 sử dụng MQTT để gửi dữ liệu tới máy chủ Flask.
3. **Lưu trữ dữ liệu:** Backend nhận dữ liệu và lưu trữ vào cơ sở dữ liệu SQLite.
4. **Cập nhật Frontend:** Frontend định kỳ gửi yêu cầu tới API để lấy dữ liệu mới nhất và cập nhật giao diện người dùng mà không cần tải lại trang.
5. **Tương tác người dùng:** Người dùng có thể điều khiển thiết bị thông qua giao diện web, gửi yêu cầu bật/tắt thiết bị tới backend, và backend cập nhật trạng thái thiết bị trong cơ sở dữ liệu.

4. Các thiết bị phần cứng

Trong dự án IoT này, hệ thống phần cứng đóng vai trò quan trọng trong việc thu thập dữ liệu từ môi trường và điều khiển các thiết bị điện. Các thiết bị phần cứng được sử dụng bao gồm:

- **Cảm biến DHT11:** Đây là một cảm biến cơ bản dùng để đo nhiệt độ và độ ẩm môi trường. Cảm biến này có khả năng truyền dữ liệu dưới dạng tín hiệu số, giúp quá trình lấy dữ liệu trở nên đơn giản và nhanh chóng.
- **Cảm biến ánh sáng:** Được sử dụng để đo cường độ ánh sáng trong không gian. Điều này có thể giúp hệ thống tự động điều chỉnh độ sáng của đèn hoặc các thiết bị chiếu sáng khác khi cần thiết.
- **Vi điều khiển ESP32:** Đây là một bộ vi điều khiển mạnh mẽ với khả năng kết nối Wi-Fi tích hợp, rất phù hợp cho các ứng dụng IoT. ESP32 sẽ thu thập dữ liệu từ các cảm biến, xử lý và gửi chúng đến hệ thống server thông qua giao thức MQTT.
- **Nguồn cấp điện:** Hệ thống cần một nguồn cấp điện ổn định để đảm bảo các thiết bị hoạt động liên tục. Các module nguồn 5V hoặc pin sạc được sử dụng để cung cấp năng lượng cho các vi điều khiển và cảm biến.

5. MQTT broker

Giao thức MQTT

MQTT (Message Queuing Telemetry Transport) là một giao thức truyền thông nhẹ, được thiết kế tối ưu cho các thiết bị có tài nguyên hạn chế và mạng không ổn định.

Trong dự án này, MQTT được sử dụng để truyền thông tin giữa các thiết bị phần cứng và server, đảm bảo việc thu thập và điều khiển thiết bị diễn ra liên tục, chính xác.

Broker MQTT

Broker MQTT đóng vai trò là một trạm trung gian giữa các thiết bị IoT. Nó sẽ nhận và phân phối các thông điệp từ các thiết bị phát thông tin (publisher) đến các thiết bị nhận thông tin (subscriber). Trong hệ thống của chúng tôi, **Mosquitto** là broker MQTT được lựa chọn, vì nó là một trong những broker phổ biến và dễ triển khai.

Cấu hình hệ thống MQTT

- **Topic:** Trong MQTT, dữ liệu được gửi qua các kênh gọi là "topic". Dự án này sử dụng các topic như:
 - home/sensor/data: Nhận dữ liệu cảm biến được gửi từ ESP32
 - home/device/control: Gửi lệnh điều khiển để bật/tắt các thiết bị.
 - home/device/status: Nhận xác nhận bật/tắt thiết bị từ ESP32

6. Tài Liệu API

Liên kết tới file API documentation Export bởi Postman:
<https://drive.google.com/file/d/10gqtD-T6M8fCym404c--HgRldDxrUIAE/view?usp=sharing>

Dưới đây là tài liệu API cho hệ thống:

V. Source code

1. Arduino:

```
#include <WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
#include "DHT.h"

#define dhtpin 25
#define dhttype DHT11
DHT dht(dhtpin, dhttype);

#define LDR_PIN 32 // Chân ADC để đọc tín hiệu từ LDR

#define FAN_PIN 4 // LED đại diện cho quạt
#define AC_PIN 5 // LED đại diện cho điều hòa
#define LIGHT_PIN 2 // LED đại diện cho đèn

// Thông tin mạng WiFi
const char* ssid = "Hqii";
const char* password = "0123456789";

// MQTT Server
const char* mqtt_server = "172.20.10.3"; // Địa chỉ IP của máy nhận MQTT
const int mqtt_port = 1884;
const char* mqtt_user = "quan"; // Tên người dùng
const char* mqtt_password = "b21dccn606"; // Mật khẩu
const char* mqtt_sensor_topic = "home/sensor/data"; // Topic để gửi dữ liệu cảm biến
const char* mqtt_control_topic = "home/device/control"; // Topic để nhận lệnh điều khiển
const char* mqtt_status_topic = "home/device/status"; // Topic để gửi trạng thái thiết bị

WiFiClient espClient;
PubSubClient client(espClient);

// Hàm callback để xử lý lệnh MQTT
```

```

void callback(char* topic, byte* payload, unsigned int length) {
    // Chuyển payload thành chuỗi
    char message[length + 1];
    strncpy(message, (char*)payload, length);
    message[length] = '\0'; // Kết thúc chuỗi

    Serial.print("Nhận được lệnh MQTT trên topic: ");
    Serial.println(topic);
    Serial.print("Nội dung lệnh: ");
    Serial.println(message);

    // Giải mã chuỗi JSON
    StaticJsonDocument<200> doc;
    DeserializationError error = deserializeJson(doc, message);

    if (error) {
        Serial.print("Lỗi giải mã JSON: ");
        Serial.println(error.c_str());
        return;
    }

    // Lấy giá trị từ JSON
    const char* device = doc["device"];
    const char* status = doc["status"];

    if (String(device) == "all" && String(status) == "off") {
        digitalWrite(LIGHT_PIN, LOW); // Tắt đèn
        digitalWrite(FAN_PIN, LOW); // Tắt quạt
        digitalWrite(AC_PIN, LOW); // Tắt điều hòa

        // Gửi trạng thái thiết bị "all" đã tắt
        client.publish(mqtt_status_topic, "{\"device\": \"fan\", \"status\": \"off\"}");
        client.publish(mqtt_status_topic, "{\"device\": \"light\", \"status\": \"off\"}");
        client.publish(mqtt_status_topic, "{\"device\": \"ac\", \"status\": \"off\"}");
        return; // Thoát khỏi hàm sau khi tắt tất cả
    }

    if (String(device) == "all" && String(status) == "on") {
        digitalWrite(LIGHT_PIN, HIGH); // Tắt đèn
        digitalWrite(FAN_PIN, HIGH); // Tắt quạt
    }
}

```

```

digitalWrite(AC_PIN, HIGH);    // Tắt điều hòa

// Gửi trạng thái thiết bị "all" đã tắt
client.publish(mqtt_status_topic, "{\"device\": \"fan\", \"status\": \"on\"}");
client.publish(mqtt_status_topic, "{\"device\": \"light\", \"status\": \"on\"}");
client.publish(mqtt_status_topic, "{\"device\": \"ac\", \"status\": \"on\"}");
return; // Thoát khỏi hàm sau khi tắt tất cả
}

// Kiểm tra thiết bị và trạng thái để điều khiển LED tương ứng
if (String(device) == "fan") {
  if (String(status) == "on") {
    digitalWrite(FAN_PIN, HIGH);
    client.publish(mqtt_status_topic, "{\"device\": \"fan\", \"status\": \"on\"}");
  } else if (String(status) == "off") {
    digitalWrite(FAN_PIN, LOW);
    client.publish(mqtt_status_topic, "{\"device\": \"fan\", \"status\": \"off\"}");
  }
} else if (String(device) == "ac") {
  if (String(status) == "on") {
    digitalWrite(AC_PIN, HIGH);
    client.publish(mqtt_status_topic, "{\"device\": \"ac\", \"status\": \"on\"}");
  } else if (String(status) == "off") {
    digitalWrite(AC_PIN, LOW);
    client.publish(mqtt_status_topic, "{\"device\": \"ac\", \"status\": \"off\"}");
  }
} else if (String(device) == "light") {
  if (String(status) == "on") {
    digitalWrite(LIGHT_PIN, HIGH);
    client.publish(mqtt_status_topic, "{\"device\": \"light\", \"status\": \"on\"}");
  } else if (String(status) == "off") {
    digitalWrite(LIGHT_PIN, LOW);
    client.publish(mqtt_status_topic, "{\"device\": \"light\", \"status\": \"off\"}");
  }
}
}

void setup() {
  Serial.begin(9600);
  dht.begin();
}

```



```

// Cài đặt chân output cho các thiết bị
pinMode(FAN_PIN, OUTPUT);
pinMode(AC_PIN, OUTPUT);
pinMode(LIGHT_PIN, OUTPUT);

// Kết nối WiFi
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Đang kết nối đến WiFi...");
}
Serial.println("Đã kết nối WiFi");

// Kết nối đến MQTT Broker với xác thực
client.setServer(mqtt_server, mqtt_port);
client.setCallback(callback); // Gán hàm callback
while (!client.connected()) {
    Serial.println("Đang kết nối đến MQTT Broker...");
    if (client.connect("ESP32Client", mqtt_user, mqtt_password)) {
        Serial.println("Đã kết nối đến MQTT");
        client.subscribe(mqtt_control_topic); // Đăng ký nhận lệnh điều khiển
    } else {
        delay(2000);
    }
}

void loop() {
    client.loop(); // Lắng nghe lệnh từ MQTT

    // Đọc dữ liệu cảm biến DHT11 và LDR
    float humidity = dht.readHumidity();
    float temperature = dht.readTemperature();
    int sensorValue = analogRead(LDR_PIN);
    float voltage = sensorValue * (3.3 / 4095.0); // Chuyển đổi giá trị ADC sang điện
        áp (0 - 3.3V)
    float lux = (2500 / voltage - 500) / 3.3; // Công thức chuyển đổi gần đúng từ điện
        áp sang lux
    if (lux > 200000) {

```

```

    lux = 200000;
}
float light = lux;

// Chuẩn bị dữ liệu dưới dạng JSON
String jsonData = "{\"temperature\": " + String(temperature) +
    ", \"humidity\": " + String(humidity) +
    ", \"light\": " + String(light) + "}";

// Gửi dữ liệu cảm biến qua MQTT
client.publish(mqtt_sensor_topic, jsonData.c_str());

// In ra Serial Monitor để kiểm tra
Serial.print("Độ ẩm: ");
Serial.print(humidity);
Serial.print("%, Nhiệt độ: ");
Serial.print(temperature);
Serial.print("°C, Ánh sáng: ");
Serial.print(light);
Serial.println(" lux");

for(int i = 0 ; i < 10 ; i ++){
    delay(500);
    client.loop();
}

// Gửi dữ liệu mỗi 60 giây
}

```

2. Front end:

1. index.html:

```
<!DOCTYPE html>
<html lang="vi">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>IOT Dashboard</title>
  <link rel="stylesheet" href="css/styles.css">
  <link rel="stylesheet" href="css/all.css">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <script src="js/chart.js"></script>
  <script src="js/index.js" defer></script>
  <script src="js/all.js"></script>
  <link href="https://fonts.googleapis.com/css2?family=Poppins&display=swap"
    rel="stylesheet">
  <link
    href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;500;700&
    display=swap" rel="stylesheet">
</head>

<body>
  <nav>
    <div class="navdiv">
      <h2>IOT Dashboard</h2>
      <!-- Phần bên phải của navbar -->
      <div class="navbar-right">
        <div class="notification">
          
          <div id="notificationDropdown" class="notification-dropdown">
            <!-- Các thông báo sẽ được thêm vào đây qua JavaScript -->
          </div>
        </div>
        <div class="user-info" onclick="goToProfile()">
          
        </div>
      </div>
    </div>
  </nav>
</body>
</html>
```

```

        <span>Hoàng Anh Quân</span>
    </div>
</div>
</div>
</nav>
<div id="overlay" class="overlay"></div>

<!-- Cửa sổ chi tiết thông báo -->
<div id="notificationDetail" class="notification-detail">
    <div class="detail-header">
        <span class="detail-title">Notification Detail</span>
        <button id="closeDetail" class="close-btn">X</button>
    </div>
    <div class="detail-content">
        <p class="detail-timestamp"></p>
        <p class="detail-message"></p>
    </div>
</div>

<div class="dashboard">
    <div class="sidebar">
        <ul>
            <li class="active"><a href="index.html">Home</a></li>
            <li><a href="thongKe.html">Sensor Data</a></li>
            <li><a href="lichSu.html">Device History</a></li>
            <li><a href="myProfile.html">Profile</a></li>
        </ul>
    </div>

    <!-- Phần nội dung chính -->
    <div class="content">
        <div class="time-display" id="currentTime"></div>
        <div class="cards">
            <div class="card" id="temperatureCard">
                <h3>Nhiệt độ</h3>
                <p id="temperature">--°C</p>
            </div>
            <div class="card" id="humidityCard">
                <h3>Độ ẩm</h3>
                <p id="humidity">--%</p>
            </div>
        </div>
    </div>
</div>

```

```

</div>
<div class="card" id="lightCard">
  <h3>Độ sáng</h3>
  <p id="light">--lux</p>
</div>
</div>
<div class="side_notification" id="notification">
  <div class="notification__body">
    
    <span id="notificationMessage"></span>
    
  </div>
  <div class="notification__progress"></div>
</div>

```

```

<!-- New sections -->
<div class="bottom-content">
  <div class="bottom-section left-bg">
    <div class="top_left">
      <h2>Thống kê thời gian thực</h2>
      <!-- Nút bấm Xem chi tiết -->
      <button id="detailButton">Xem chi tiết</button>
    </div>
    <canvas id="myChart"></canvas>
  </div>

  <!-- Cửa sổ chi tiết, mặc định ẩn -->
  <div id="detailChartModal" class="detail-chart-modal" style="display:
none;">
    <div class="DetailChart">
      <div>
        <h2>Thống kê thời gian thực</h2>
      </div>
      <div class="tempChartBox">
        <canvas id="lineChart" width="400" height="100"></canvas>
      </div>
    </div>

```

```

        <div class="dryChartBox">
            <canvas id="humidityChart" width="400"
height="100"></canvas>
        </div>
        <div class="lightChartBox">
            <canvas id="lightChart" width="400" height="100"></canvas>
        </div>
        <!-- Nút đóng cửa sổ -->
        <div class="closeButtonContainer">
            <button id="closeButton">Đóng</button>
        </div>
    </div>
</div>

<div class="bottom-section right-bg">
    <div>
        <h2>Điều khiển thiết bị thông minh</h2>
    </div>
    <div class="deviceControl">
        <div class="device">
            
            <span>Đèn</span>
            <label class="switch">
                <input type="checkbox" data-device="light" id="light-switch">
                <span class="slider round"></span>
            </label>
        </div>
        <hr class="divider">
        <div class="device">
            
            <span>Quạt</span>
            <label class="switch">
                <input type="checkbox" data-device="fan" id="fan-switch">
                <span class="slider round"></span>
            </label>
        </div>
        <hr class="divider">
        <div class="device">

```

```

        
        <span>Điều hòa</span>
        <label class="switch">
            <input type="checkbox" data-device="ac" id="ac-switch">
            <span class="slider round"></span>
        </label>
    </div>
</div>

</div>
</div>
</div>
</body>

</html>

```

2. lichSu.html:

```
<!DOCTYPE html>
<html lang="vi">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>IOT Dashboard - Lịch Sử Thiết Bị</title>
  <link rel="stylesheet" href="css/lichSu.css">
  <link rel="stylesheet" href="css/all.css">
  <script src="js/lichSu.js"></script>
  <script src="js/all.js"></script>
  <script src="https://cdn.socket.io/4.5.2/socket.io.min.js"></script>

</head>

<body>
  <nav>
    <div class="navdiv">
      <h2>IOT Dashboard</h2>
      <!-- Phần bên phải của navbar -->
      <div class="navbar-right">
        <div class="notification">
          
          <!-- <span class="badge">99</span> -->
          <div id="notificationDropdown" class="notification-dropdown">
            <!-- Các thông báo sẽ được thêm vào đây qua JavaScript -->
          </div>
        </div>
        <div class="user-info" onclick="goToProfile()">
          
          <span>Hoàng Anh Quân</span>
        </div>
      </div>
    </div>
  </nav>

  <div id="overlay" class="overlay"></div>
```



```

<!-- Cửa sổ chi tiết thông báo -->
<div id="notificationDetail" class="notification-detail">
  <div class="detail-header">
    <span class="detail-title">Notification Detail</span>
    <button id="closeDetail" class="close-btn">X</button>
  </div>
  <div class="detail-content">
    <p class="detail-timestamp"></p>
    <p class="detail-message"></p>
  </div>
</div>
<div class="dashboard">
  <div class="sidebar">
    <ul>
      <li><a href="index.html">Home</a></li>
      <li><a href="thongKe.html">Sensor Data</a></li>
      <li class="active"><a href="lichSu.html">Device History</a></li>
      <li><a href="myProfile.html">Profile</a></li>
    </ul>
  </div>
  <div class="container">
    <h1>Lịch Sử Bật/Tắt Thiết Bị</h1>

    <div class="table-controls">
      <div class="entries-select">
        <label for="entriesPerPage">Hiển thị:</label>
        <select id="entriesPerPage">
          <option value="5">5</option>
          <option value="10" selected>10</option>
          <option value="25">25</option>
        </select>
      </div>

      <div class="filter-controls">
        <label for="filterSelect">Thời gian:</label>
        <select id="filterSelect">
          <option value="all">Tất cả</option>
          <option value="today">Hôm nay</option>
          <option value="7days">7 Ngày</option>
          <option value="1month">1 Tháng</option>
        </select>
      </div>
    </div>
  </div>
</div>

```

```

        <option value="3months">3 Tháng</option>

    </select>
</div>

<!--

<div class="filter-controls"></div>
    <label for="deviceFilterSelect">Thiết bị:</label>
    <select id="deviceFilterSelect">
        <option value="all">Tất cả</option>
        <option value="fan">Quạt</option>
        <option value="ac">Điều hòa</option>
        <option value="light">Đèn</option>
    </select>
</div> -->


<div class="search-bar">
    <input type="text" id="searchInput" placeholder="Tìm kiếm..." />
</div>
</div>

<table id="dataTable">
    <thead>
        <tr>
            <th data-sort="id" class="sortable">ID <span class="sort-arrow asc">▲</span><span class="sort-arrow desc">▼</span></th>
            <th data-sort="device_name" class="sortable">Tên Thiết Bị <span class="sort-arrow asc">▲</span><span class="sort-arrow desc">▼</span></th>
            <th data-sort="status" class="sortable">Trạng Thái <span class="sort-arrow asc">▲</span><span class="sort-arrow desc">▼</span></th>
            <th data-sort="time" class="sortable">Thời Gian <span class="sort-arrow asc">▲</span><span class="sort-arrow desc">▼</span></th>
        </tr>
    </thead>
    <tbody>
        <!-- Dữ liệu sẽ được thêm vào đây bằng JavaScript -->
    </tbody>
</table>

```

```
<div id="pagination">  
  <!-- Các nút phân trang sẽ được thêm vào đây -->  
</div>  
</div>
```

```
</div>
```

```
</body>
```

```
</html>
```

3. thongKe.html:

```
<!DOCTYPE html>
<html lang="vi">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>IOT Dashboard</title>
  <link rel="stylesheet" href="css/thongKe.css">
  <link rel="stylesheet" href="css/all.css">
  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
  <script src="js/thongKe.js"></script>
  <script src="js/all.js"></script>
  <script src="https://cdn.socket.io/4.5.2/socket.io.min.js"></script>

</head>

<body>
  <nav>
    <div class="navdiv">
      <h2>IOT Dashboard</h2>
      <!-- Phần bên phải của navbar -->
      <div class="navbar-right">
        <div class="notification">
          
          <!-- <span class="badge">99</span> -->
          <div id="notificationDropdown" class="notification-dropdown">
            <!-- Các thông báo sẽ được thêm vào đây qua JavaScript -->
          </div>
        </div>
        <div class="user-info" onclick="goToProfile()">
          
          <span>Hoàng Anh Quân</span>
        </div>
      </div>
    </div>
  </nav>
```

```

<div id="overlay" class="overlay"></div>
<!-- Cửa sổ chi tiết thông báo -->
<div id="notificationDetail" class="notification-detail">
  <div class="detail-header">
    <span class="detail-title">Notification Detail</span>
    <button id="closeDetail" class="close-btn">X</button>
  </div>
  <div class="detail-content">
    <p class="detail-timestamp"></p>
    <p class="detail-message"></p>
  </div>
</div>
<div class="dashboard">
  <div class="sidebar">
    <ul>
      <li><a href="index.html">Home</a></li>
      <li class="active"><a href="thongKe.html">Sensor Data</a></li>
      <li><a href="lichSu.html">Device History</a></li>
      <li><a href="myProfile.html">Profile</a></li>
    </ul>
  </div>
  <div class="container">
    <h1>Dữ liệu Thống Kê Môi Trường</h1>

    <div class="table-controls">
      <div class="entries-select">
        <label for="entriesSelect">Hiển thị:</label>
        <select id="entriesSelect">
          <option value="5">5</option>
          <option value="10" selected>10</option>
          <option value="25">25</option>
        </select>
      </div>

      <div class="filter-controls">
        <label for="filterSelect">Thời gian:</label>
        <select id="filterSelect">
          <option value="all">Tất cả</option>
          <option value="today">Hôm nay</option>
          <option value="7days">7 Ngày</option>
        </select>
      </div>
    </div>
  </div>
</div>

```

```

        <option value="1 month">1 Tháng</option>
        <option value="3 months">3 Tháng</option>

    </select>
</div>

<div class="filter-controls">
    <label for="sensorFilterSelect">Cảm biến:</label>
    <select id="sensorFilterSelect">
        <option value="all">Tất cả</option>
        <option value="temperature">Nhiệt độ</option>
        <option value="humidity">Độ ẩm</option>
        <option value="light">Ánh sáng</option>
    </select>
</div>

<div class="search-bar">
    <input type="text" id="searchInput" placeholder="Tìm kiếm..." />
    <select id="searchTypeSelect">
        <option value="time">Tìm kiếm theo thời gian</option>
        <option value="sensor">Tìm kiếm theo cảm biến</option>
    </select>
</div>
</div>

<table id="dataTable">
    <thead>
        <tr>
            <th data-sort="id" class="sortable">ID <span class="sort-arrow
asc">▲</span><span class="sort-arrow desc">▼</span>
            </th>
            <th data-sort="sensor" class="sortable">Cảm biến <span
class="sort-arrow asc">▲</span><span
class="sort-arrow desc">▼</span></th>
            <th data-sort="value" class="sortable">Giá trị <span
class="sort-arrow asc">▲</span><span
class="sort-arrow desc">▼</span></th>
            <th data-sort="time" class="sortable">Thời gian <span
class="sort-arrow asc">▲</span><span
class="sort-arrow desc">▼</span></th>

```

```
        </tr>
    </thead>
    <tbody>
        <!-- Dữ liệu sẽ được thêm vào đây bằng JavaScript -->
    </tbody>
</table>

<div id="pagination">
    <!-- Các nút phân trang sẽ được thêm vào đây -->
</div>

</div>

</body>

</html>
```

4. myProfile.html:

```
<!DOCTYPE html>
<html lang="vi">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>IOT Dashboard</title>
  <link rel="stylesheet" href="css\profile.css">
  <link rel="stylesheet" href="css\all.css">
  <link
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css"
rel="stylesheet">

  <script src="js/all.js"></script>
  <script src="https://cdn.socket.io/4.5.2/socket.io.min.js"></script>

</head>

<body>
  <nav>
    <div class="navdiv">
      <h2>IOT Dashboard</h2>
      <!-- Phần bên phải của navbar -->
      <div class="navbar-right">
        <div class="notification">
          
          <!-- <span class="badge">99</span> -->
          <div id="notificationDropdown" class="notification-dropdown">
            <!-- Các thông báo sẽ được thêm vào đây qua JavaScript -->
          </div>
        </div>
        <div class="user-info" onclick="goToProfile()">
          
          <span>Hoàng Anh Quân</span>
        </div>
      </div>
    </div>
  </nav>
</body>
```



```

    </div>
</nav>

<div id="overlay" class="overlay"></div>
<!-- Cửa sổ chi tiết thông báo -->
<div id="notificationDetail" class="notification-detail">
    <div class="detail-header">
        <span class="detail-title">Notification Detail</span>
        <button id="closeDetail" class="close-btn">X</button>
    </div>
    <div class="detail-content">
        <p class="detail-timestamp"></p>
        <p class="detail-message"></p>
    </div>
</div>
<div class="dashboard">
    <div class="sidebar">
        <ul>
            <li><a href="index.html">Home</a></li>
            <li><a href="thongKe.html">Sensor Data</a></li>
            <li><a href="lichSu.html">Device History</a></li>
            <li class="active"><a href="myProfile.html">Profile</a></li>
        </ul>
    </div>
    <div class="student-info-container">
        <!-- Cột trái -->
        <div class="left-column">
            <!-- Cột con bên trái 1 -->
            <div class="card mb-4">
                <div class="card-body text-center">
                    
                    <h5 class="my-3">Hoàng Anh Quân</h5>
                    <p class="text-muted mb-1">Học viện công nghệ bưu chính viễn
thông - PTIT</p>
                    <p class="text-muted mb-4">Công nghệ thông tin</p>
                </div>
            </div>
            <!-- Cột con bên trái 2 -->
            <div class="card mb-4 mb-lg-0">

```

```

<div class="card-body p-0">
  <ul class="list-group list-group-flush rounded-3">
    <li class="list-group-item d-flex justify-content-between
align-items-center p-3">
      <i class="fab fa-github fa-lg text-body"></i>
      <a href="https://github.com/8qii" target="_blank">8qii</a>
    </li>
    <li class="list-group-item d-flex justify-content-between
align-items-center p-3">
      <i class="fab fa-instagram fa-lg" style="color: #ac2bac;"></i>
      <a href="https://www.instagram.com/_ah.quan/"
target="_blank">_ah.quan</a>
    </li>
    <li class="list-group-item d-flex justify-content-between
align-items-center p-3">
      <i class="fab fa-facebook-f fa-lg" style="color: #3b5998;"></i>
      <a href="https://www.facebook.com/Hqiii38/"
target="_blank">Hoàng Quân</a>
    </li>
  </ul>
</div>
</div>
</div>

```

```

<!-- Cột phải -->
<div class="right-column">
  <!-- Cột con bên phải 1 -->
  <div class="card mb-4">
    <div class="card-body">
      <div class="card-body">
        <div class="row">
          <div class="col-sm-3">
            <p class="mb-0">Full Name</p>
          </div>
          <div class="col-sm-9">
            <p class="text-muted mb-0">Hoàng Anh Quân</p>
          </div>
        </div>
      </div>
    </div>
    <hr>
    <div class="row">

```

```

<div class="col-sm-3">
  <p class="mb-0">Class</p>
</div>
<div class="col-sm-9">
  <p class="text-muted mb-0">D21CQCN6-B</p>
</div>
</div>
<hr>
<div class="row">
  <div class="col-sm-3">
    <p class="mb-0">Student ID</p>
  </div>
  <div class="col-sm-9">
    <p class="text-muted mb-0">B21DCCN606</p>
  </div>
</div>
<hr>
<div class="row">
  <div class="col-sm-3">
    <p class="mb-0">Development orientation</p>
  </div>
  <div class="col-sm-9">
    <p class="text-muted mb-0">AI engineer</p>
  </div>
</div>
<hr>
<div class="row">
  <div class="col-sm-3">
    <p class="mb-0">Phone</p>
  </div>
  <div class="col-sm-9">
    <p class="text-muted mb-0">0889919819</p>
  </div>
</div>
<hr>
<div class="row">
  <div class="col-sm-3">
    <p class="mb-0">Email</p>
  </div>
  <div class="col-sm-9">

```

```

        <p class="text-muted
mb-0">hoanganhquanttcx@gmail.com</p>
    </div>
</div>
<hr>
<div class="row">
    <div class="col-sm-3">
        <p class="mb-0">Address</p>
    </div>
    <div class="col-sm-9">
        <p class="text-muted mb-0">Thị trấn Cẩm Xuyên, Hà
Tĩnh</p>
    </div>
</div>
</div>
</div>
<!-- Các mục khác -->
</div>
</div>
<!-- Cột con bên phải 2 -->
<div class="card mb-4 mb-md-0">
    <div class="card-body">
        <h3 style="font-size: 1.5rem;">Trạng thái bài tập cá nhân </h2>
        <p style="margin-left: 20px; font-size: 1.3rem ; padding-bottom:
10px;">File PDF: <a
href="https://drive.google.com/file/d/1L3RI8SJe4AzrK7XCwgttQEULm5kS6EZ/vie
w?usp=sharing"
        target="_blank">Bấm để mở link</a></p>
    <hr>
    <p style="margin-left: 20px; font-size: 1.3rem ; padding-bottom:
10px;">Github phát triển:
        <a href="https://github.com/8qii/IOT-web/tree/final"
target="_blank">Hqii at final
        version</a>
    </p>
    <hr>
    <p style="margin-left: 20px; font-size: 1.3rem ; padding-bottom:
10px;">API docs: <a

```

href="https://drive.google.com/file/d/10gqtD-T6M8fCym404c--HgRldDxrU1AE/view?usp=sharing"

target="_blank">Bấm để mở link</p>

</div>

</div>

</div>

</div>

</div>

</body>

</html>

5. Backend:

```
import random
from flask import Flask, jsonify, request
from flask_cors import CORS
import sqlite3
from datetime import datetime, timedelta
import pytz
import json
import paho.mqtt.client as mqtt

app = Flask(__name__)
CORS(app) # Kích hoạt CORS

def get_sensor_data():
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()

    # Truy vấn để lấy giá trị mới nhất của mỗi cảm biến
    cursor.execute("""
        SELECT sensor, value, time
        FROM sensors
        WHERE id IN (
            SELECT MAX(id)
            FROM sensors
            WHERE sensor IN ('temperature', 'humidity', 'light')
            GROUP BY sensor
        )
    """)

    rows = cursor.fetchall()
    conn.close()

    # Tạo từ điển để lưu trữ kết quả
    data = {"temperature": "--", "humidity": "--", "light": "--"}

    for row in rows:
        sensor, value, _ = row
        data[sensor] = value
```

```
return data if rows else None
```

```
@app.route('/api/sensor_data', methods=['GET'])
def sensor_data():
    data = get_sensor_data()
    if data:
        return jsonify(data)
    return jsonify({"error": "No data found"}), 404
```

```
def get_chart_data():
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()

    # Truy vấn để lấy 20 mẫu dữ liệu mới nhất của mỗi loại cảm biến
    cursor.execute("""
        SELECT value
        FROM sensors
        WHERE sensor = 'temperature'
        ORDER BY id DESC
        LIMIT 20
    """)
    temperatures = cursor.fetchall()

    cursor.execute("""
        SELECT value
        FROM sensors
        WHERE sensor = 'humidity'
        ORDER BY id DESC
        LIMIT 20
    """)
    humidities = cursor.fetchall()

    cursor.execute("""
        SELECT value
        FROM sensors
        WHERE sensor = 'light'
        ORDER BY id DESC
        LIMIT 20
    """)
```

```

        """)
    lights = cursor.fetchall()

    conn.close()

    # Chuẩn bị dữ liệu trả về dưới dạng danh sách của các danh sách
    data = list(zip(temperatures, humidities, lights))

    # Chuyển đổi từ list các tuple về dạng list của list cho đúng với JS logic
    data = [[temp[0], hum[0], light[0]] for temp, hum, light in data]

    return data

@app.route('/api/chart_data', methods=['GET'])
def chart_data():
    data = get_chart_data()
    return jsonify(data)

#-----

@app.route('/api/device-status', methods=['GET'])
def get_device_status():
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()

    cursor.execute(
        "SELECT device_name, status FROM devices ORDER BY time DESC")
    rows = cursor.fetchall()

    # Lấy trạng thái mới nhất cho mỗi thiết bị
    status_map = {}
    for row in rows:
        if row[0] not in status_map:
            status_map[row[0]] = row[1]

    conn.close()
    return jsonify(status_map)

```



```

#-----notification-----

@app.route('/api/notifications', methods=['GET'])
def get_notifications():
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()

    cursor.execute(
        "SELECT message, timestamp FROM notification ORDER BY id DESC
        LIMIT 10")
    rows = cursor.fetchall()

    notifications = [{'message': row[0], 'timestamp': row[1]} for row in rows]
    conn.close()

    return jsonify(notifications)

def add_notification(message):
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()
    cursor.execute("INSERT INTO notification (message) VALUES (?)", (message,))
    conn.commit()
    conn.close()

@app.route('/api/add-notification', methods=['POST'])
def add_notification_route():
    data = request.get_json()
    message = data.get('message')

    if message:
        add_notification(message)
        return jsonify({'success': True}), 200
    else:
        return jsonify({'success': False, 'error': 'No message provided'}), 400

#-----Thong Ke-----

```

```

@app.route('/api/sensors/all', methods=['GET'])
def get_all_sensors_data():
    conn = sqlite3.connect(r'G:/Coding/database/iot.db')
    cursor = conn.cursor()

    cursor.execute(
        "SELECT id, temperature, humidity, light, time FROM sensors ORDER BY id
        ASC")
    rows = cursor.fetchall()

    data = []
    for row in rows:
        data.append({
            'id': row[0],
            'nhiet_do': row[1],
            'do_am': row[2],
            'do_sang': row[3],
            'time': row[4] # Đảm bảo rằng thời gian được trả về đúng định dạng
        })

    conn.close()
    return jsonify(data)

```

#-----Lich Su-----

```

@app.route('/api/devices', methods=['GET'])
def get_devices_data():
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()

    cursor.execute(
        "SELECT id, device_name, status, time FROM devices ORDER BY id
        DESC")
    rows = cursor.fetchall()

    data = []
    for row in rows:
        data.append({
            'id': row[0],
            'device_name': row[1],

```

```

        'status': row[2],
        'time': row[3]
    })

```

```

conn.close()
return jsonify(data)

```

```

#-----device filter by day-----
@app.route('/api/devices-filter', methods=['GET'])
def get_devices_data_filter():
    # Nhận tham số "filter" từ query string
    filter_param = request.args.get(
        'filter', 'all') # Giá trị mặc định là "all"

    # Kết nối tới cơ sở dữ liệu
    conn = sqlite3.connect('G:/Coding/database/iot.db')
    cursor = conn.cursor()

    # Lấy thời gian hiện tại
    now = datetime.now()

    # Xây dựng câu truy vấn SQL và giá trị bộ lọc
    query = "SELECT id, device_name, status, time FROM devices WHERE 1=1"
    params = []

    if filter_param == 'today':
        # Lọc cho dữ liệu của ngày hôm nay
        start_date = now.strftime('%Y-%m-%d 00:00:00')
        end_date = now.strftime('%Y-%m-%d 23:59:59')
        query += " AND time BETWEEN ? AND ?"
        params.extend([start_date, end_date])

    elif filter_param == '7days':
        # Lọc cho dữ liệu trong 7 ngày qua
        start_date = (now - timedelta(days=7)).strftime('%Y-%m-%d %H:%M:%S')
        query += " AND time >= ?"
        params.append(start_date)

    elif filter_param == '1month':

```

```

# Lọc cho dữ liệu trong 1 tháng qua
start_date = (now - timedelta(days=30)).strftime('%Y-%m-%d %H:%M:%S')
query += " AND time >= ?"
params.append(start_date)

elif filter_param == '3months':
    # Lọc cho dữ liệu trong 3 tháng qua
    start_date = (now - timedelta(days=90)).strftime('%Y-%m-%d %H:%M:%S')
    query += " AND time >= ?"
    params.append(start_date)

# Sắp xếp theo thời gian mới nhất trước
query += " ORDER BY time DESC"

# Thực hiện truy vấn
cursor.execute(query, params)
rows = cursor.fetchall()

# Định dạng dữ liệu kết quả thành danh sách các dictionary
data = []
for row in rows:
    data.append({
        'id': row[0],
        'device_name': row[1],
        'status': row[2],
        'time': row[3]
    })

# Đóng kết nối cơ sở dữ liệu
conn.close()

# Trả về dữ liệu dưới dạng JSON
return jsonify(data)

#-----filter sensor data-----
# Đảm bảo mã API trả về dữ liệu đúng định dạng

@app.route('/api/sensors-filter', methods=['GET'])
def get_sensors_data_filter():

```

```

# Nhận các tham số từ query string
filter_param = request.args.get('filter', 'all')
sensor_filter = request.args.get('sensor', 'all')
search_query = request.args.get('search', '')

# Kết nối tới cơ sở dữ liệu
conn = sqlite3.connect('G:/Coding/database/iot.db')
cursor = conn.cursor()

# Lấy thời gian hiện tại
now = datetime.now()

# Xây dựng câu truy vấn SQL và giá trị bộ lọc
query = "SELECT id, sensor, value, time FROM sensors WHERE 1=1"
params = []

if filter_param == 'today':
    # Lọc cho dữ liệu của ngày hôm nay
    start_date = now.strftime('%Y-%m-%d 00:00:00')
    end_date = now.strftime('%Y-%m-%d 23:59:59')
    query += " AND time BETWEEN ? AND ?"
    params.extend([start_date, end_date])

elif filter_param == '7days':
    # Lọc cho dữ liệu trong 7 ngày qua
    start_date = (now - timedelta(days=7)).strftime('%Y-%m-%d %H:%M:%S')
    query += " AND time >= ?"
    params.append(start_date)

elif filter_param == '1month':
    # Lọc cho dữ liệu trong 1 tháng qua
    start_date = (now - timedelta(days=30)).strftime('%Y-%m-%d %H:%M:%S')
    query += " AND time >= ?"
    params.append(start_date)

elif filter_param == '3months':
    # Lọc cho dữ liệu trong 3 tháng qua
    start_date = (now - timedelta(days=90)).strftime('%Y-%m-%d %H:%M:%S')
    query += " AND time >= ?"
    params.append(start_date)

```

```

if sensor_filter != 'all':
    query += " AND sensor = ?"
    params.append(sensor_filter)

if search_query:
    query += " AND time LIKE ?"
    params.append(f"%{search_query}%")

# Sắp xếp theo thời gian mới nhất trước
query += " ORDER BY time DESC"

# Thực hiện truy vấn
cursor.execute(query, params)
rows = cursor.fetchall()

# Định dạng dữ liệu kết quả thành danh sách các dictionary
data = []
for row in rows:
    data.append({
        'id': row[0],
        'sensor': row[1],
        'value': row[2],
        'time': row[3]
    })

# Đóng kết nối cơ sở dữ liệu
conn.close()

# Trả về dữ liệu dưới dạng JSON
return jsonify(data)

# -----MQTT-----
# Cấu hình MQTT
mqtt_broker = "172.20.10.3" # Địa chỉ IP của máy nhận MQTT
mqtt_port = 1884
mqtt_topic = "home/sensor/data"
mqtt_topic_control = "home/device/control" # Chủ đề điều khiển thiết bị
mqtt_topic_status = "home/device/status" # Chủ đề cập nhật trạng thái thiết bị

```

```
mqtt_user = "quan" # Username cho MQTT
mqtt_password = "b21dccn606" # Password cho MQTT
```

```
# Hàm callback khi có tin nhắn từ MQTT
```

```
def on_message(client, userdata, message):
    try:
        print(f'Received message: {message.payload.decode()}')

        if message.topic == mqtt_topic:
            payload = json.loads(message.payload.decode())
            temperature = payload.get('temperature')
            humidity = payload.get('humidity')
            light = payload.get('light')

            conn = sqlite3.connect('G:/Coding/database/iot.db')
            cursor = conn.cursor()

            if temperature is not None:
                cursor.execute(
                    "INSERT INTO sensors (sensor, value) VALUES (?, ?)",
                    ('temperature', temperature))
            if humidity is not None:
                cursor.execute(
                    "INSERT INTO sensors (sensor, value) VALUES (?, ?)", ('humidity',
                    humidity))
            if light is not None:
                cursor.execute(
                    "INSERT INTO sensors (sensor, value) VALUES (?, ?)", ('light', light))

            conn.commit()
            conn.close()

            print("Dữ liệu cảm biến đã được lưu vào cơ sở dữ liệu")

        elif message.topic == mqtt_topic_status:
            payload = json.loads(message.payload.decode())
            device = payload.get('device')
```

```

status = payload.get('status')

conn = sqlite3.connect('G:/Coding/database/iot.db')
cursor = conn.cursor()

if device is not None and status is not None:
    cursor.execute(
        "INSERT INTO devices (device_name, status) VALUES (?, ?)",
        (device, status))

    conn.commit()
    conn.close()

    print(f"Trạng thái thiết bị {device} đã được cập nhật thành {status}")

except Exception as e:
    print(f"Error while processing message: {e}")

# Khởi tạo MQTT client và cấu hình callback
mqtt_client = mqtt.Client()
mqtt_client.on_message = on_message

# Thêm thông tin đăng nhập MQTT
mqtt_client.username_pw_set(mqtt_user, mqtt_password)

def on_connect(client, userdata, flags, rc):
    if rc == 0:
        print("Kết nối thành công đến MQTT broker")
        client.subscribe(mqtt_topic)
        client.subscribe(mqtt_topic_status)
    else:
        print(f"Kết nối thất bại với mã lỗi: {rc}")

mqtt_client.on_connect = on_connect

# ----- Flask API cho điều khiển thiết bị -----

```



```

@app.route('/api/control-device', methods=['POST'])
def control_device():
    try:
        data = request.get_json()
        device = data.get('device')
        status = data.get('status')

        if not device or not status:
            return jsonify({'success': False, 'message': 'Thiếu thông tin thiết bị hoặc trạng thái'})

        mqtt_payload = json.dumps({
            'device': device,
            'status': status
        })
        mqtt_client.publish(mqtt_topic_control, mqtt_payload)

        return jsonify({'success': True, 'message': f'{device} is now {status}'})
    except Exception as e:
        return jsonify({'success': False, 'message': f'Lỗi: {str(e)}'})

# Chạy Flask server
if __name__ == '__main__':
    mqtt_client.connect(mqtt_broker, mqtt_port)
    mqtt_client.subscribe(mqtt_topic)
    mqtt_client.subscribe(mqtt_topic_status)

    mqtt_client.loop_start()
    app.run(debug=True)

```

VI. Kết quả

1. Tổng quan

Sau khi hoàn thành việc thiết kế và triển khai hệ thống IoT bao gồm các phần cứng, phần mềm, giao diện, và giao thức truyền thông, chúng tôi đã tiến hành thử nghiệm hệ thống để đánh giá hiệu suất và khả năng hoạt động của các thành phần. Các thử nghiệm này nhằm đảm bảo rằng hệ thống có thể thu thập dữ liệu chính xác từ các cảm biến, điều khiển thiết bị điện ổn định, và xử lý thông tin một cách hiệu quả thông qua giao thức MQTT.

2. Đo lường các chỉ số như nhiệt độ, độ ẩm, ánh sáng thời gian thực

Hệ thống đã được thử nghiệm trong một môi trường kiểm soát, với các điều kiện môi trường thay đổi để đánh giá khả năng thu thập dữ liệu từ các cảm biến.

Cảm biến DHT11 (nhiệt độ và độ ẩm):

Nhiệt độ được thu thập dao động từ 25°C đến 35°C với độ chính xác chấp nhận được cho các ứng dụng giám sát thông thường.

Độ ẩm dao động từ 40% đến 95%, và các dữ liệu được cập nhật liên tục với độ trễ không đáng kể, đảm bảo thời gian thực.

Cảm biến ánh sáng:

Cảm biến ánh sáng hoạt động ổn định, phản hồi thay đổi cường độ ánh sáng ngay lập tức khi môi trường sáng hoặc tối đi. Dữ liệu cường độ ánh sáng được biểu diễn chính xác trên giao diện.

Tất cả các thông số này được truyền tải thông qua giao thức MQTT và hiển thị trên dashboard thời gian thực. Các thử nghiệm cho thấy dữ liệu được cập nhật một cách mượt mà và đồng bộ giữa các thành phần của hệ thống.

3. Điều khiển các thiết bị điện

Hệ thống được thử nghiệm với khả năng điều khiển các thiết bị điện (chẳng hạn như đèn hoặc quạt) từ xa thông qua giao diện web. Kết quả cho thấy:

Lệnh bật/tắt các thiết bị điện được thực thi gần như ngay lập tức sau khi người dùng nhấn nút trên giao diện. Độ trễ giữa thời gian phát lệnh và thời gian thực thi dưới 1 giây, đảm bảo yêu cầu của một hệ thống điều khiển từ xa.

Trong các trường hợp mạng Wi-Fi yếu, hệ thống vẫn đảm bảo các lệnh điều khiển được truyền tải chính xác nhờ giao thức MQTT và cơ chế xử lý lại khi mất kết nối tạm thời.

Các thử nghiệm cho thấy hệ thống ổn định, không có lỗi trong quá trình điều khiển các thiết bị điện, và khả năng kết nối mạng tốt.

4. Lưu trữ và đưa ra dữ liệu cho người dùng

Hệ thống sử dụng cơ sở dữ liệu SQLite3 để lưu trữ dữ liệu từ các cảm biến và lịch sử điều khiển. Trong quá trình thử nghiệm:

Dữ liệu cảm biến được lưu trữ liên tục mà không có mất mát dữ liệu.

Hệ thống cho phép người dùng truy xuất lịch sử dữ liệu trong các khoảng thời gian cụ thể (ví dụ: theo ngày, tuần, tháng). Dữ liệu được hiển thị dưới dạng biểu đồ dễ hiểu, giúp người dùng theo dõi các thay đổi theo thời gian.

Các thử nghiệm cho thấy dữ liệu cảm biến được lưu trữ chính xác và hệ thống có thể xử lý truy vấn dữ liệu nhanh chóng từ SQLite, ngay cả khi khối lượng dữ liệu tăng lên.

5. Đánh giá chung

Kết quả thử nghiệm cho thấy hệ thống IoT đã thiết kế và triển khai hoạt động tốt trong môi trường thử nghiệm. Cụ thể:

Các cảm biến đo lường nhiệt độ, độ ẩm, và ánh sáng hoạt động ổn định, cung cấp dữ liệu thời gian thực chính xác.

Khả năng điều khiển từ xa các thiết bị điện thông qua giao diện web hoạt động hiệu quả, với độ trễ thấp và tính ổn định cao.

Hệ thống lưu trữ dữ liệu cảm biến vào CSDL và cho phép truy vấn dữ liệu lịch sử một cách linh hoạt, nhanh chóng.

Giao thức MQTT hoạt động ổn định, đảm bảo việc truyền dữ liệu và lệnh điều khiển trong môi trường mạng không ổn định.

Dù kết quả thử nghiệm rất khả quan, có một số vấn đề nhỏ có thể cải thiện trong các phiên bản tiếp theo của hệ thống, như tối ưu hóa giao diện người dùng để hiển thị dữ liệu trực quan hơn, hoặc tích hợp thêm các chức năng cảnh báo khi các thông số môi trường vượt ngưỡng an toàn.

Kết Luận

Dự án đã thành công trong việc phát triển một hệ thống IoT hoàn chỉnh, có khả năng giám sát và điều khiển thiết bị tự động.