

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO MÔN IOT VÀ ỨNG DỤNG

Đề tài: Quản lý và điều khiển thiết bị trong nhà thông minh

Giảng viên hướng dẫn : TS.Nguyễn Quốc Uy

Họ và tên sinh viên : Dương Hải Lưu

Mã sinh viên : B22DCCN513

Lớp : D22HTTT05

Nhóm : 16

Mục lục

I, GIỚI THIỆU TỔNG QUAN	4
1, Giới thiệu đề tài.....	4
2, Mục tiêu đề tài.....	4
3, Mô tả hệ thống	4
4, Các thiết bị sử dụng cho hệ thống.....	5
II. THIẾT KẾ HỆ THỐNG	8
1, Thiết kế kiến trúc	8
2, Thiết kế giao diện.....	10
3, Luồng hoạt động	14
4, Thiết kế cơ sở dữ liệu bằng MySQL.....	16
III. XÂY DỰNG HỆ THỐNG.....	17
1, Code phần cứng trên Arduino IDE	17
2, Api docs.....	22
IV. KẾT LUẬN	26

LỜI CẢM ƠN

Trước tiên em xin gửi lời cảm ơn đến trường Học Viện Công Nghệ Bưu Chính Viễn Thông đã đưa môn IoT và ứng dụng vào trong chương trình giảng dạy. Đặc biệt em xin gửi lời cảm ơn sâu sắc tới thầy Nguyễn Quốc Uy đã tận tình hướng dẫn và truyền đạt những kiến thức quý báu cho em trong suốt thời gian qua để hoàn thành bài báo cáo này. Tuy đã có nhiều cố gắng nhưng do kiến thức và thời gian thực hiện có hạn nên bài báo cáo của em không tránh khỏi những sai sót. Em rất mong nhận được những ý kiến đóng góp, phê bình của thầy. Đó sẽ là hành trang quý giá để em có thể hoàn thiện mình sau này. Em xin chân thành cảm ơn thầy.

I, GIỚI THIỆU TỔNG QUAN

1, Giới thiệu đề tài

Trong bối cảnh cuộc sống hiện đại, công nghệ thông tin đã và đang len lỏi vào từng ngóc ngách của cuộc sống. Cùng với đó thì Internet of Thing (IoT) trở thành một phần không thể thiếu trong cuộc sống. Nó được sử dụng trong nhiều lĩnh vực công nghệ, nông nghiệp, các hoạt động sống hàng ngày,... Đề tài quản lý và điều khiển thiết bị trong nhà thông minh sẽ tập 3 chung xây dựng một hệ thống IoT đơn giản có chức năng là thu thập thông tin từ các cảm biến nhiệt độ, độ ẩm, ánh sáng đồng thời điều khiển các thiết bị quạt, điều hòa, bóng điện trong nhà. Hệ thống này nhằm giúp cho mọi người có thể theo dõi ngôi nhà của mình mọi lúc mọi nơi, chỉ cần có internet.

2, Mục tiêu đề tài

Thu thập dữ liệu môi trường từ các cảm biến: Hệ thống cần có khả năng thu thập dữ liệu từ các cảm biến nhiệt độ, độ ẩm và ánh sáng. Dữ liệu này sẽ được ghi nhận liên tục và cập nhật lên hệ thống để người dùng có thể theo dõi từ xa.

Điều khiển các thiết bị điện trong nhà: Hệ thống có khả năng điều khiển các thiết bị như quạt, điều hòa và bóng điện dựa trên các điều kiện môi trường hoặc theo yêu cầu của người dùng. Người dùng có thể điều khiển thiết bị thông qua giao diện từ xa bằng kết nối internet.

Tích hợp với mạng không dây và kết nối internet: Hệ thống sử dụng kết nối WiFi để truyền tải dữ liệu từ cảm biến và gửi lệnh điều khiển đến các thiết bị. Điều này giúp người dùng dễ dàng theo dõi và điều khiển thiết bị trong nhà từ xa thông qua điện thoại hoặc máy tính.

Lưu trữ và phân tích dữ liệu: Dữ liệu từ cảm biến sẽ được lưu trữ vào cơ sở dữ liệu, cho phép người dùng theo dõi các thay đổi theo thời gian, từ đó đưa ra các quyết định tự động dựa trên điều kiện môi trường (ví dụ, tự động bật điều hòa khi nhiệt độ vượt quá mức nhất định).

3, Mô tả hệ thống

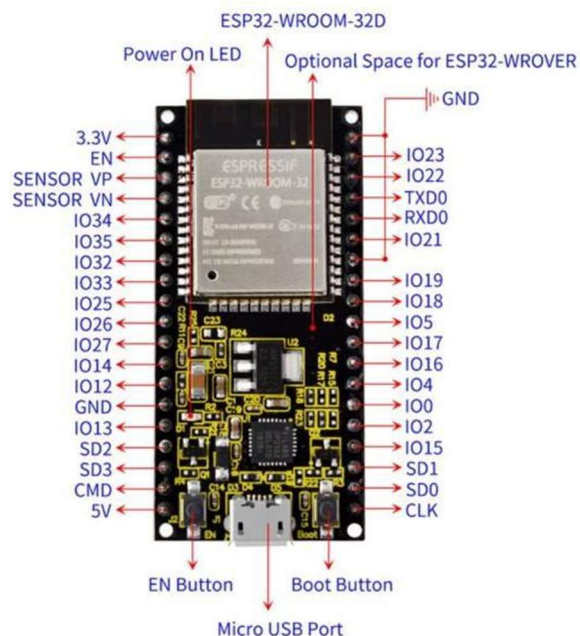
Hệ thống giám sát nhiệt độ, độ ẩm, ánh sáng:

- Cảm biến DHT11: Sử dụng cảm biến DHT11 để đo lường nhiệt độ (từ 0 đến 50°C) và độ ẩm (từ 20 đến 100%).

- Quang trở: Sử dụng quang trở giúp dễ dàng theo dõi sự thay đổi của cường độ
- ánh sáng trong nhà.
- Dữ liệu được cập nhật theo thời gian thực để có thể giúp người dùng cập nhật nhiệt độ, độ ẩm ngay lập tức. Ngoài ra, người dùng cũng có thể theo dõi lịch sử của nhiệt độ, độ ẩm để biết khi nào nhiệt độ cao, khi nào nhiệt độ thấp.
- Hệ thống điều khiển (bật/tắt) các thiết bị trong nhà theo mong muốn mà không cần thao tác trực tiếp với thiết bị. Chỉ cần có internet, người dùng có thể thao tác trực tiếp bật/tắt các thiết bị trên website.

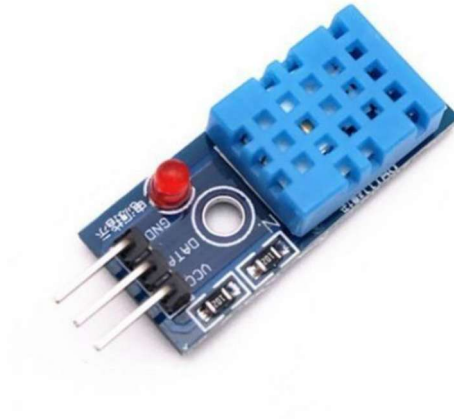
4, Các thiết bị sử dụng cho hệ thống

1. ESP32 WROOM Wifi

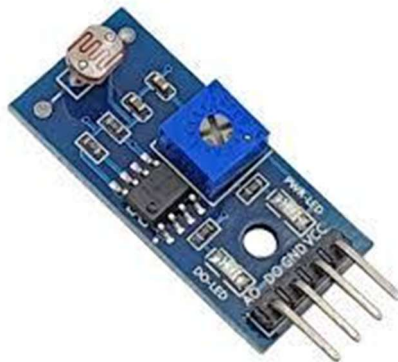


- Thông số kỹ thuật:
- IC chính : Wifi BLE SoC ESP32 ESP-WROOM-32
- Dòng điện áp sử dụng: 2.2V ~ 3.6V
 - Nhân xử lý trung tâm: ESP32-D0WDQ6 Dual-core low power Xtensa® 32-bit LX6 microprocessors.

- ROM: 448Kbytes
2. Cảm biến DHT11 (cảm biến nhiệt độ, độ ẩm)



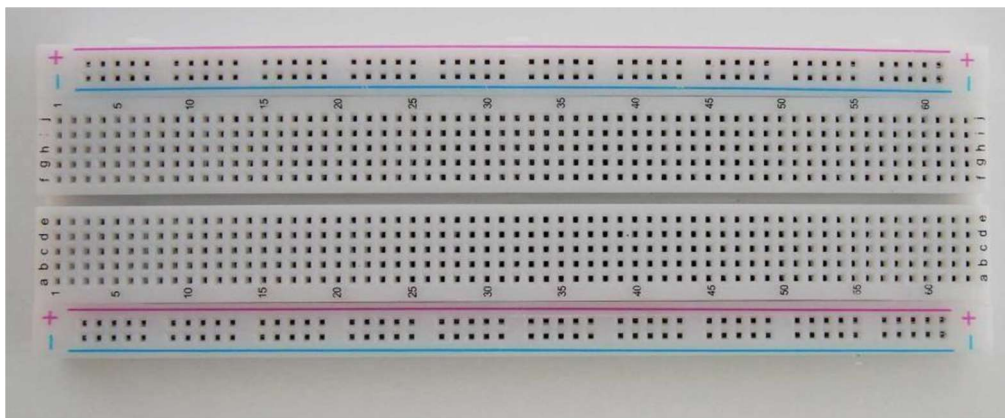
- Thông số kỹ thuật:
 - Điện áp hoạt động: 5V
 - Chuẩn giao tiếp: TTL, 1 wire.
 - Khoảng đo độ ẩm: 20%-80%RH sai số $\pm 5\%RH$
 - Khoảng đo nhiệt độ: 0-50°C sai số $\pm 2^{\circ}C$
 - Tần số lấy mẫu tối đa 1Hz (1 giây / lần) + Kích thước: 28mm x 12mm x 10mm
3. Quang trở



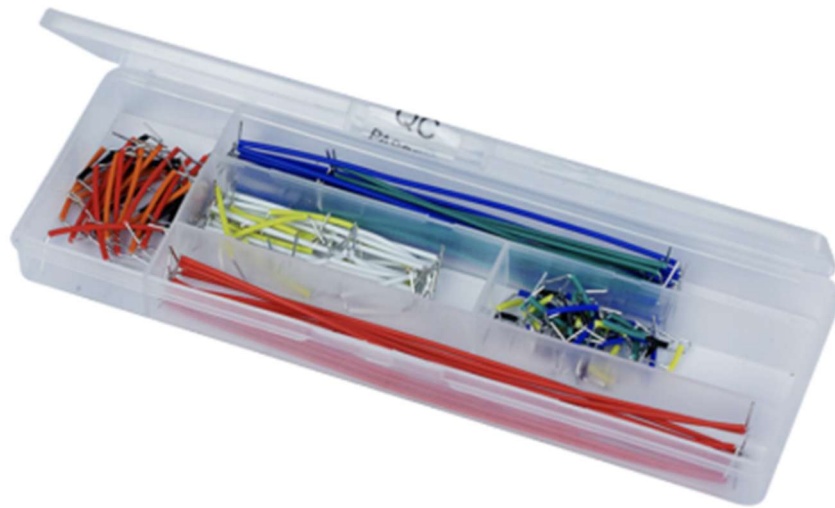
4. Điện trở



5. BreadBoard



6. Dây cáp



7. Các đèn led

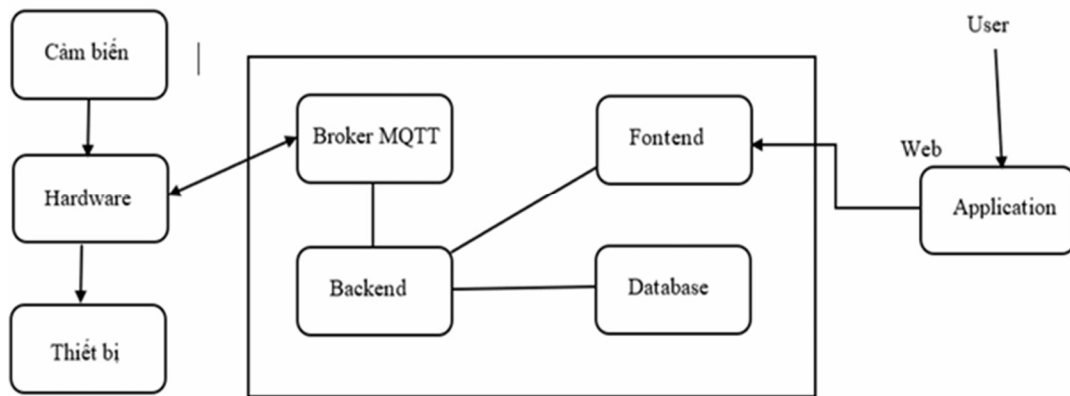


8. Mạch sau khi lắp

II. THIẾT KẾ HỆ THỐNG

1, Thiết kế kiến trúc

- Sơ đồ thiết kế hệ thống

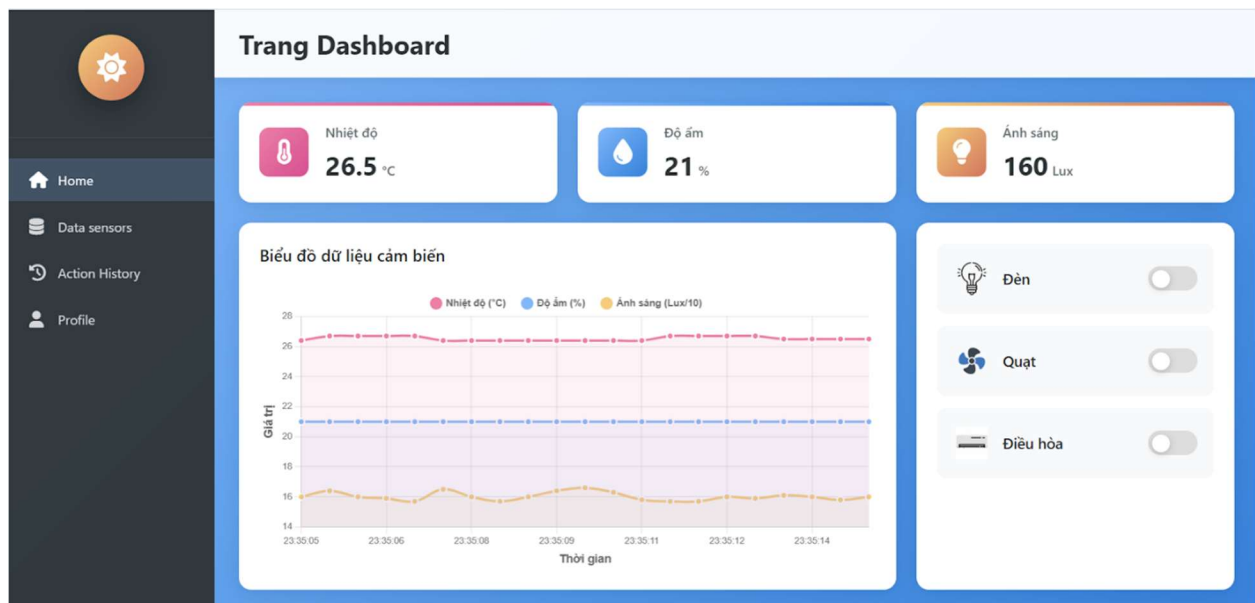


- Kiến trúc hệ thống
 - ESP32
 - Publish dữ liệu cảm biến lên MQTT topic `iot/sensors/data`
 - Subscribe `iot/led/light/command` để nhận lệnh điều khiển thiết bị.
 - Publish trạng thái thiết bị lên `iot/led/light/status`
 - MQTT Broker (trung tâm giao tiếp)
 - Các topic:
 - `iot/sensors/data` → dữ liệu cảm biến.
 - `iot/led/light/command` → lệnh điều khiển.
 - `iot/led/light/status` → trạng thái thiết bị.
- Server Web (ExpressJS+ MySQL + MQTT)
 - `mqtt`: chạy nền, subscribe `sensor/data` và lưu vào bảng `data_sensor` trong cơ sở dữ liệu.
 - API trong `api/data`: trả dữ liệu JSON cho web (bản ghi mới nhất, danh sách, biểu đồ, lịch sử).
 - API trong `api/mqtt`: publish lệnh MQTT, lấy trạng thái thiết bị.
- Web Dashboard
 - Gọi API để hiển thị 3 ô dữ liệu (nhiệt độ, độ ẩm, ánh sáng).
 - Vẽ biểu đồ bằng `Chart.js`
 - Gửi lệnh bật/tắt công tắc → gửi socket: `device-control`
 - Cập nhật icon trạng thái thiết bị

2, Thiết kế giao diện

Giao diện hệ thống gồm 4 trang: Trang chủ (Dashboard), trang theo dõi dữ liệu cảm biến (Data sensors), trang quản lý lịch sử hành động (Action history), trang thông tin cá nhân (Profile) được làm bằng html, css, Javascript. Backend viết bằng ExpressJS.

Trang Dashboard



- Dữ liệu cập nhật 3 ô
 - Nhiệt độ (Temperature): Hiện thị nhiệt độ hiện tại với biểu tượng nhiệt kế.
 - Độ ẩm (Humidity): Hiện thị độ ẩm hiện tại với biểu tượng giọt nước.
 - Ánh sáng (Light): Hiện thị cường độ ánh sáng hiện tại với biểu tượng mặt trời.
- Biểu đồ theo dõi
 - Biểu đồ kết hợp hiển thị dữ liệu nhiệt độ và độ ẩm, ánh sáng theo thời gian.
 - Đường màu đỏ thể hiện nhiệt độ.
 - Đường màu xanh dương thể hiện độ ẩm.
 - Đường màu vàng thể hiện ánh sáng
- Điều khiển thiết bị

- Quạt (Fan): Biểu tượng quạt với công tắc bật/tắt.
- Điều hòa (Air Conditioner): Biểu tượng điều hòa với công tắc bật/tắt.
- Đèn (Light): Biểu tượng bóng đèn với công tắc bật/tắt.

Trang Data Sensor

ID	Nhiệt độ (°C)	Độ ẩm (%)	Ánh sáng (Lux)	Thời gian tạo
4432	27.70	48.00	836	2025/10/03 14:20:38
4433	27.70	48.00	737	2025/10/03 14:20:38
4430	27.00	48.00	800	2025/10/03 14:20:37
4431	27.70	48.00	794	2025/10/03 14:20:37
4428	27.00	48.00	805	2025/10/03 14:20:36
4429	27.00	48.00	799	2025/10/03 14:20:36
4426	27.10	48.00	751	2025/10/03 14:20:35

- Hiện thông tin của nhiệt độ, độ ẩm, ánh sáng với các trường dữ liệu:
 - id: mã của dữ liệu cảm biến
 - Nhiệt độ: Nhiệt độ đọc từ cảm biến
 - Độ ẩm: Độ ẩm đọc từ cảm biến
 - Ánh sáng: Ánh sáng đọc từ quang trở
 - Thời gian: Thời gian thực của dữ liệu
- Các chức năng cho user sử dụng ở trang Data sensor:
 - Tìm kiếm dữ liệu cảm biến theo thời gian
 - Lọc dữ liệu cảm biến theo Nhiệt độ, Độ ẩm, Ánh sáng
 - Thực hiện sắp xếp dữ liệu cảm biến theo chiều tăng dần hoặc giảm dần của các trường dữ liệu
 - Phân trang dữ liệu theo yêu cầu của người dùng

Trang History Action

Trang Action History

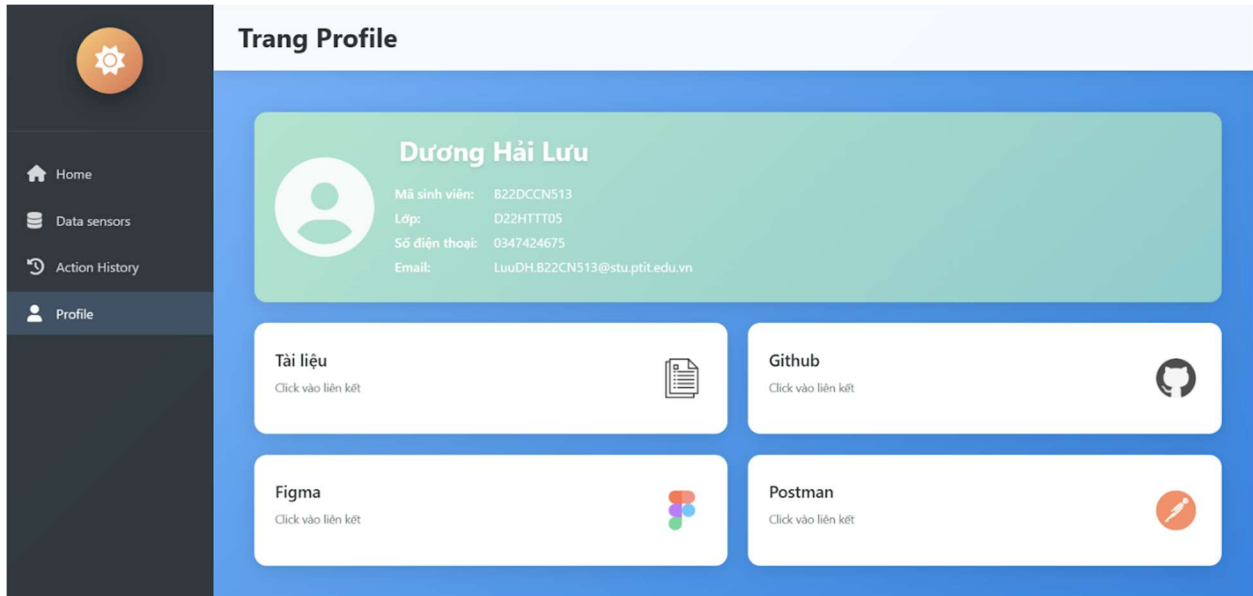
Thiết bị: Tất cả thiết bị | Hành động: Tất cả hành động | **Lọc**

ID	Thiết bị	Hành động	Thời gian tạo
13	Light	on	2025/10/03 13:43:32
12	Light	off	2025/10/03 13:43:30
11	Fan	on	2025/10/03 13:43:27
10	Fan	off	2025/10/03 13:43:26
9	Fan	on	2025/10/03 13:43:24
8	Air condition	on	2025/10/03 13:43:22
7	Light	on	2025/10/03 13:43:15

« < 1 2 > »

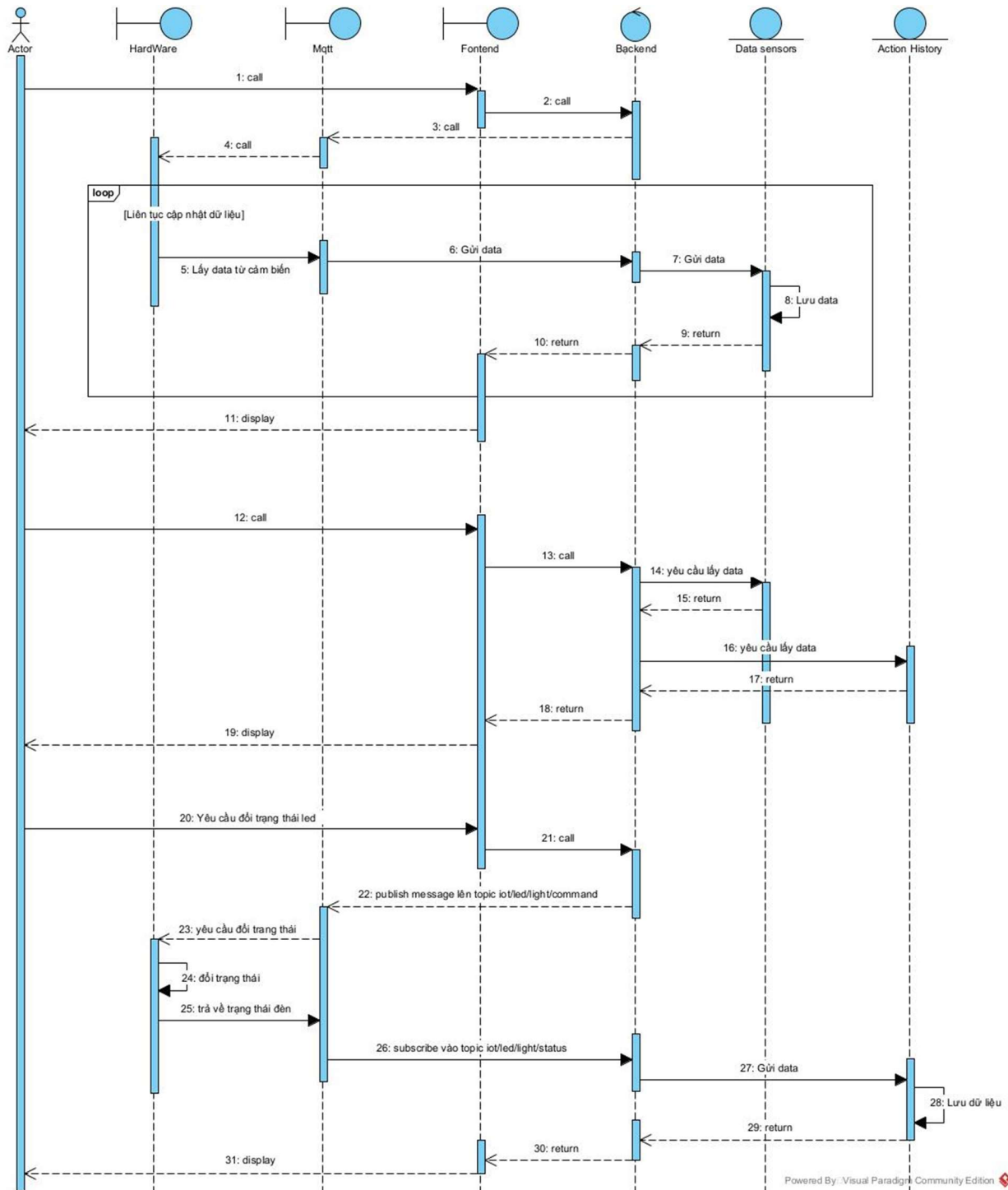
- Hiện thông số các hành động của người dùng:
 - id: Mã lịch sử
 - Thiết bị: Tên thiết bị ghi nhận
 - Hành động: Trạng thái thiết bị Bật/Tắt
 - Ngày giờ: Thời gian ghi nhận trạng thái thiết bị thay đổi Các chức năng cho
- user sử dụng ở trang History Action:
 - Thực hiện tìm kiếm theo thời gian lịch sử bật tắt thiết bị
 - Lọc dữ liệu bật tắt thiết bị theo tên thiết bị
 - Phân trang dữ liệu lịch sử bật tắt thiết bị theo nhu cầu của user

Trang Profile



- Trang Profile này cung cấp:
 - Thông tin cá nhân và học tập của người dùng.
 - Liên kết đến tài khoản GitHub cá nhân.
 - Khả năng tải xuống tài liệu PDF của dự án.
 - Truy cập nhanh đến tài liệu API của dự án.

3, Luồng hoạt động



- Luồng Cập nhật Dữ liệu liên tục

- Khởi tạo kết nối: Hardware (Thiết bị) gọi (1) Mqtt, sau đó Mqtt gọi (2) Backend, Backend gọi (3) Frontend, và Mqtt nhận được phản hồi (4).
- Vòng lặp (Loop) Cập nhật:
 - Hardware lấy dữ liệu từ cảm biến (5).
 - Hardware gửi dữ liệu (6) đến Mqtt.
 - Mqtt gửi dữ liệu (7) đến Data sensors.
 - Data sensors lưu dữ liệu (8) và phản hồi (9) lại Mqtt.
 - Mqtt phản hồi (10) lại Frontend.
 - Frontend hiển thị dữ liệu mới (11).
- Luồng Yêu cầu và Truy xuất Dữ liệu Lịch sử
 - Frontend gọi (12) Backend để yêu cầu dữ liệu lịch sử.
 - Backend gọi (13) Data sensors.
 - Data sensors yêu cầu (14) và nhận phản hồi (15) từ Action History
 - Data sensors tiếp tục yêu cầu (16) và nhận phản hồi (17) từ Action History (có thể là để xác nhận hoặc lấy thêm thông tin).
 - Data sensors phản hồi (18) dữ liệu đã truy xuất cho Backend.
 - Backend phản hồi (19) lại Frontend.
 - Frontend hiển thị dữ liệu lịch sử (20).
- Luồng Điều khiển và Cập nhật Trạng thái Thiết bị (LED)
 - Frontend gửi lệnh Yêu cầu đổi trạng thái led (21) đến Backend.
 - Backend gọi (22) Mqtt.
 - Mqtt xuất bản (publish message) (23) lên kênh điều khiển (iot/led/light/command).
 - Hardware nhận lệnh và thực hiện:
 - Yêu cầu đổi trạng thái (24).

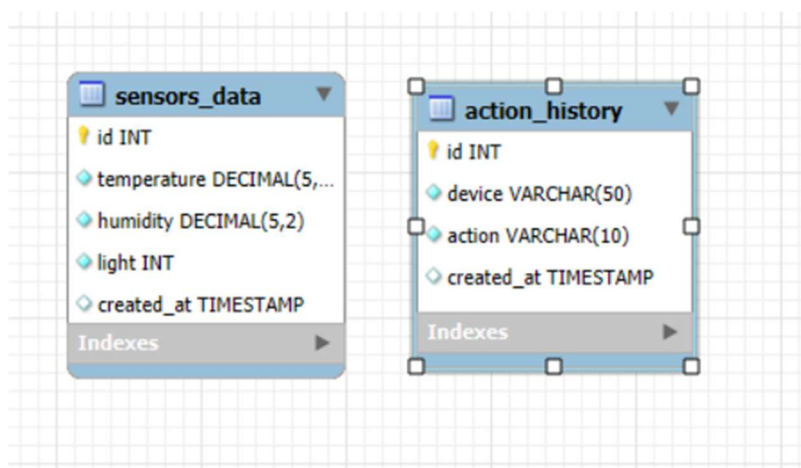
- Đổi trạng thái đèn (25).
- Trả về trạng thái đèn (26) cho Mqtt.
- Cập nhật Trạng thái:
 - Backend đã đăng ký (subscribe) (27) kênh trạng thái (iot/led/light/status).
 - Mqtt gửi dữ liệu trạng thái (28) đến Action History.
 - Action History lưu dữ liệu (29) và phản hồi (30) lại Mqtt.
 - Mqtt phản hồi (31) trạng thái cập nhật cho Backend.
 - Backend phản hồi (32) lại Frontend.
 - Frontend hiển thị trạng thái mới (33) của thiết bị.

4, Thiết kế cơ sở dữ liệu bằng MySQL

Database gồm có hai bảng là bảng data_sensor và history_aton:

- Data_sensor: lưu thông tin của các thông số nhiệt độ, độ ẩm, ánh sáng theo thời gian (cứ 2 giây lại đẩy dữ liệu lên một lần).
- history_action: lưu lại các trạng thái bật, tắt thiết bị theo thời gian thực của người dùng.

Sử dụng MySQL để tạo cơ sở dữ liệu:



III. XÂY DỰNG HỆ THỐNG

1, Code phần cứng trên Adruino IDE

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

- Wifi.h: thư viện kết nối với Wifi
- PubSubClient.h: thư viện sử dụng MQTT
- DHT.h: thư viện để đọc cảm biến DHT11

```
// Cấu hình Chân cắm
#define DHTPIN 25
#define DHTTYPE DHT11
#define LDR_PIN 34
#define LED_LIGHT_PIN 13
#define FAN_PIN 14
#define AC_PIN 27
```

- DHTPIN: Định nghĩa chân kết nối cảm biến DHT11.
- LIGHT_PIN: Chân đọc giá trị từ cảm biến ánh sáng (LDR)
- LED_LIGHT_PIN, FAN_PIN, AC_PIN: Chân điều khiển các thiết bị (đèn, quạt, điều hòa).

```

const char* ssid = "IamHaiLuu";
const char* password = "09042004";

// Cấu hình MQTT
const char* mqtt_server = "192.168.1.15";
const int mqtt_port = 1883;
const char* mqtt_user = "user";
const char* mqtt_password = "123";
const char* mqtt_client_id = "ESP32Client_1";

```

- Khai báo các thông tin kết nối MQTT vs Wifi

```

// Khởi tạo đối tượng
WiFiClient espClient;
PubSubClient client(espClient);
DHT dht(DHTPIN, DHTTYPE);

```

- Khởi tạo các đối tượng:
 - espClient: Đối tượng đại diện cho kết nối Wi-Fi của ESP32.
 - client: Đối tượng MQTT sử dụng kết nối Wi-Fi từ espClient.
 - dht: Đối tượng DHT để đọc dữ liệu từ cảm biến nhiệt độ và độ ẩm

```

void setup() {
    Serial.begin(115200);
    pinMode(LED_LIGHT_PIN, OUTPUT);
    pinMode(FAN_PIN, OUTPUT);
    pinMode(AC_PIN, OUTPUT);

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

```

Hàm thiết lập ban đầu:

- `Serial.begin(115200)`: Bắt đầu kết nối Serial để in thông tin debug.
- `dht.begin()`: Khởi tạo cảm biến DHT11.
- `pinMode`: Thiết lập chế độ cho các chân GPIO. 12
- `WiFi.begin(ssid, password)`: Kết nối Wi-Fi với tên mạng và mật khẩu đã định nghĩa.
- `client.setServer()`: Thiết lập MQTT server và cổng.
- `client.setCallback(callback)`: Đăng ký hàm callback để xử lý khi nhận dữ liệu từ MQTT

```

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
  unsigned long now = millis();
  if (now - lastMsg > PUBLISH_INTERVAL) {
    lastMsg = now;
    // Đọc dữ liệu từ cảm biến
    float humidity = dht.readHumidity();
    float temperature = dht.readTemperature();
    int light = analogRead(LDR_PIN);
    light = map(light, 0, 4095, 0, 1000);
    if (isnan(humidity) || isnan(temperature)) {
      Serial.println("Failed to read from DHT sensor!");
      return;
    }
    // Tạo JSON payload
    StaticJsonDocument<200> doc;
    doc["humidity"] = humidity;
    doc["temperature"] = temperature;
    doc["light"] = light;
    char jsonBuffer[200];
    serializeJson(doc, jsonBuffer);
    // Gửi dữ liệu lên MQTT broker
    client.publish(mqtt_publish_topic, jsonBuffer);
    Serial.print("Published message: ");
    Serial.println(jsonBuffer);
  }
}

```

- Đọc dữ liệu nhiệt độ, độ ẩm từ cảm biến DHT11 (dht.readHumidity() và dht.readTemperature()).
- Đọc giá trị ánh sáng từ cảm biến LDR (analogRead(LIGHT_PIN)).

- `client.publish(mqtt_publish_topic, jsonBuffer)`: Gửi dữ liệu nhiệt độ, độ ẩm, ánh sáng qua MQTT với topic là `sensor/data`.
- `delay(2000)`: Chờ 2 giây trước khi thực hiện lại.

2, Api docs

Sensor APIs:

- GET /api/dashboard: Lấy dữ liệu cảm biến, trạng thái thiết bị và dữ liệu biểu đồ cho dashboard.

HTTP IOT / dashboard

GET http://localhost:3000/api/dashboard

Params Authorization Headers (6) Body Scripts Settings


Body Cookies Headers (8) Test Results


{ } JSON Preview Visualize


```
1 {
2   "success": true,
3   "data": {
4     "sensorData": {
5       "id": 6303,
6       "temperature": "26.50",
7       "humidity": "48.00",
8       "light": 610,
9       "created_at": "2025-10-03T07:36:27.000Z"
10    },
11    "devices": {
12      "light": true,
13      "fan": true,
14      "ac": true
15    },
16    "chartData": [
17      {
18        "temperature": "26.30",
19        "humidity": "48.00",
20        "light": 603,
21        "created_at": "2025-10-03T07:36:19.000Z"
22      },
23      {
24        "temperature": "26.80",
25        "humidity": "48.00",
26        "light": 589,
27        "created_at": "2025-10-03T07:36:19.000Z"
28      }
29    ]
30  }
31 }
```


localhost:3000


Trang Dashboard


 Home


 Data sensors

 Action History

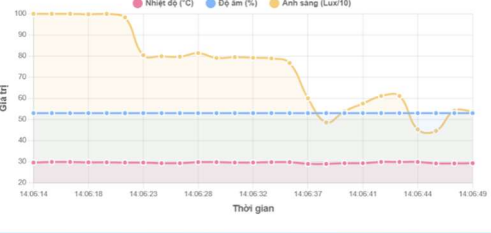
 Profile

 Nhiệt độ
29.3 °C

 Độ ẩm
53 %


 Ánh sáng
538 Lux


Biểu đồ dữ liệu cảm biến




Giá trị

Thời gian

 Đèn ☐

 Quạt ☐

 Điều hòa ☐

- GET /api/data-sensors - Lấy dữ liệu cảm biến với khả năng phân trang, sắp xếp và tìm kiếm

HTTP IOT / sensors

GET http://localhost:3000/api/data-sensors?page=1&limit=5&sortBy=temperature&sortOrder=ASC

key	value	message
page	1	
limit	5	
sortBy	temperature	
sortOrder	ASC	

Body Cookies Headers (8) Test Results (1/1) 200 OK

{ } JSON Preview Visualize

```

1 {
2   "success": true,
3   "data": {
4     "data": [
5       {
6         "id": 2963,
7         "temperature": "25.00",
8         "humidity": "35.00",
9         "light": 540,
10        "created_at": "2025-10-07T14:57:28.000Z"
11      },
12      {
13        "id": 2964,
14        "temperature": "25.00",
15        "humidity": "35.00",
16        "light": 563,
17        "created_at": "2025-10-07T14:57:29.000Z"
18      },
19    ]
20  }
21 }

```

localhost:3000/data-sensors?page=1&limit=5&sortBy=temperature&sortOrder=ASC

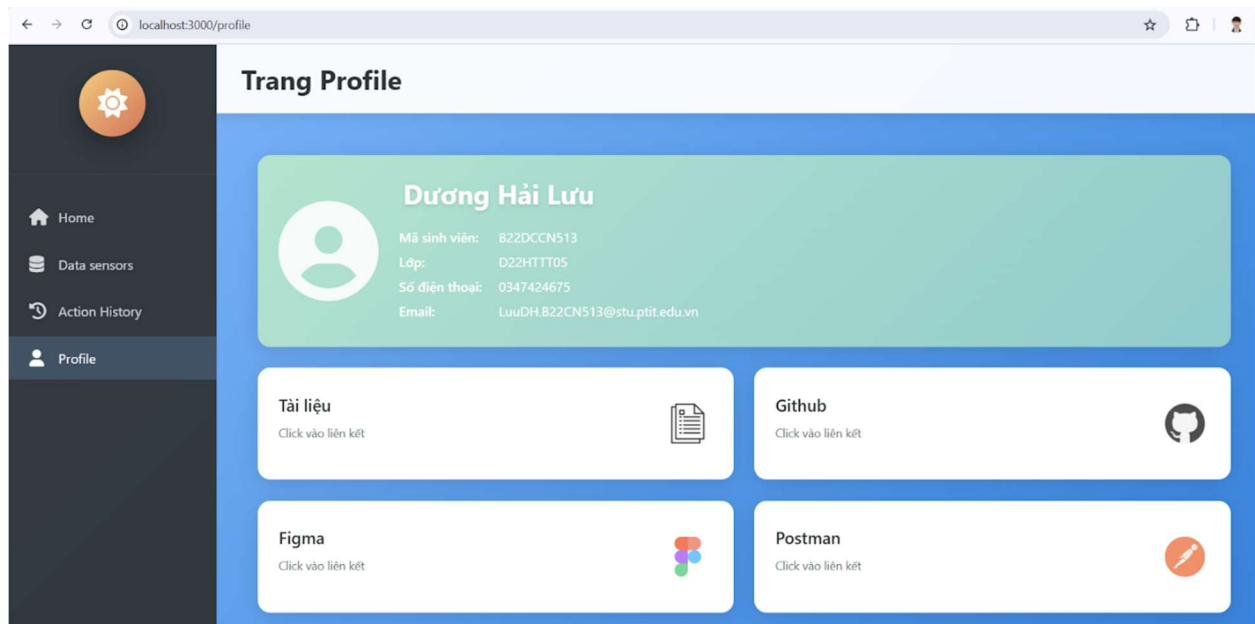
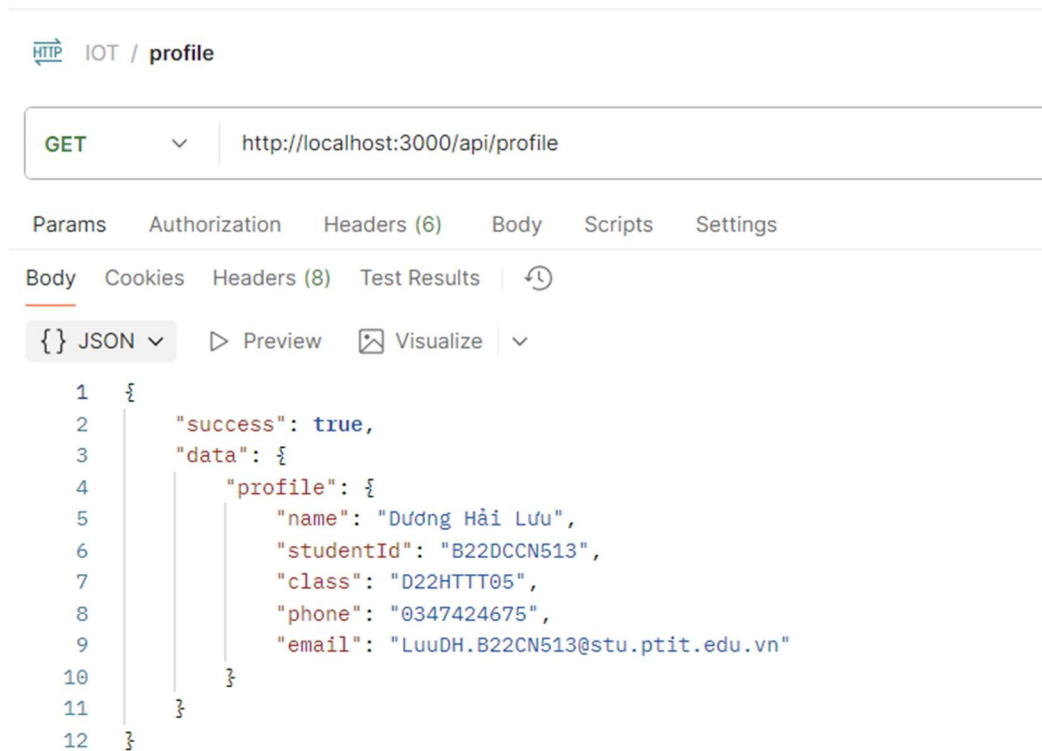
Trang Data Sensors

ID	Nhiệt độ (°C)	Độ ẩm (%)	Ánh sáng (Lux)	Thời gian tạo
2963	25.00	35.00	540	2025/10/07 21:57:28
2964	25.00	35.00	563	2025/10/07 21:57:29
2973	25.00	35.00	740	2025/10/07 21:57:43
2974	25.00	35.00	816	2025/10/07 21:57:44
2961	25.10	35.00	811	2025/10/07 21:57:25

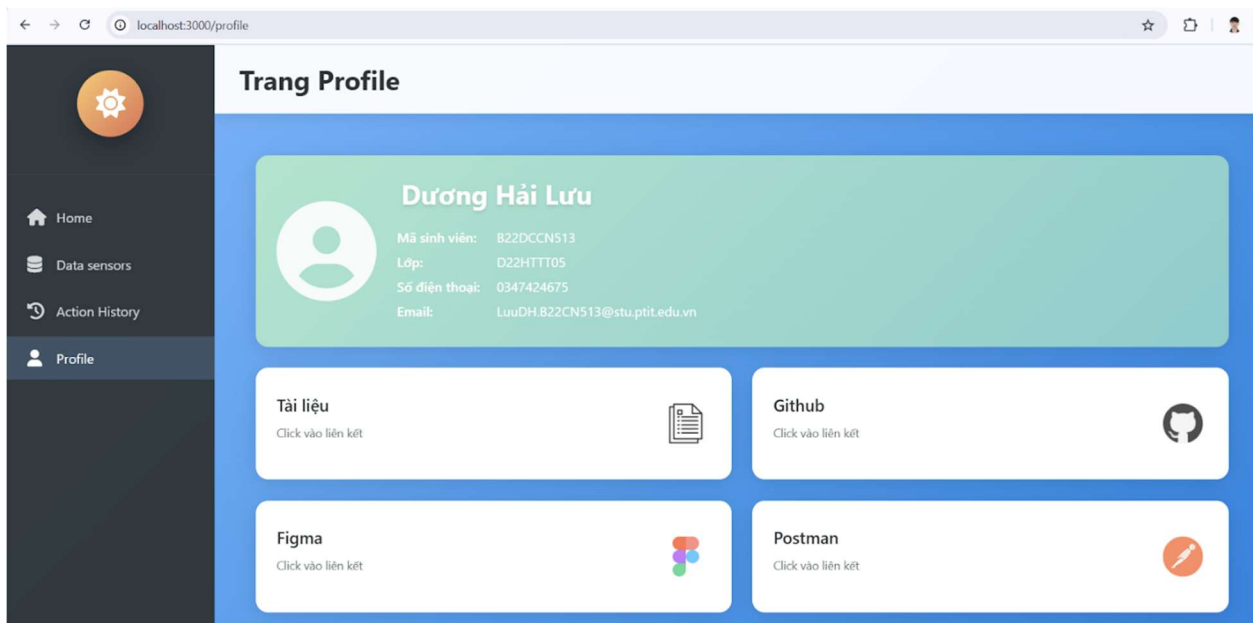
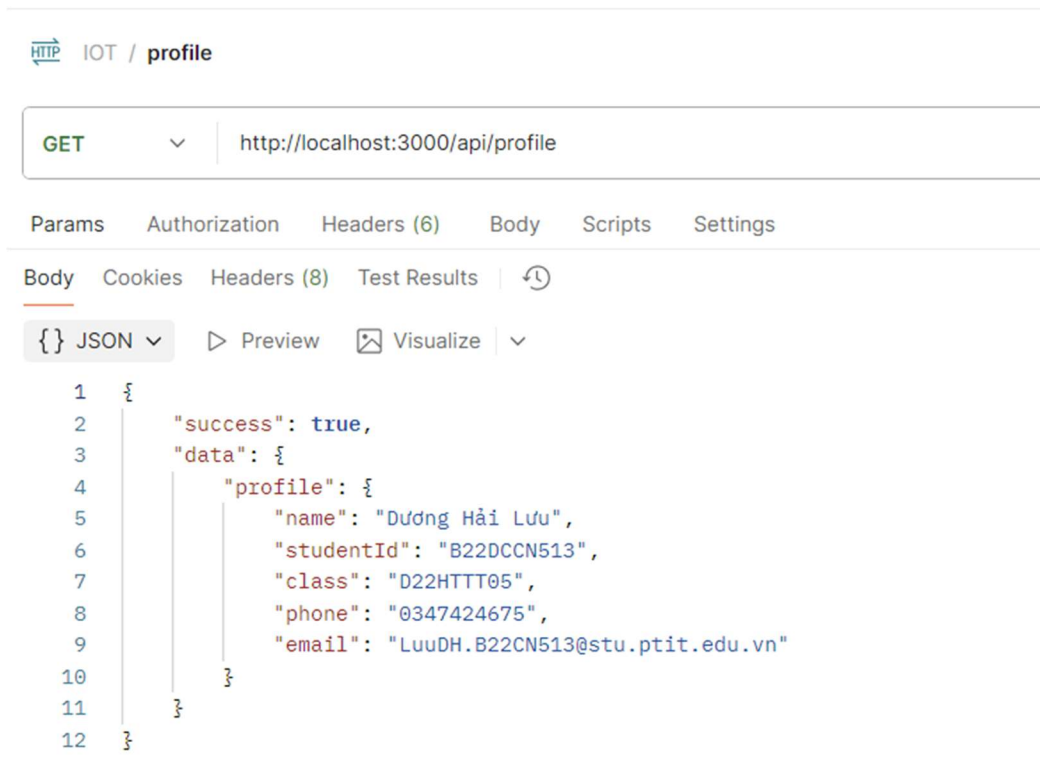
Hiện thị: dòng/trang

<< < 1 2 3 > >>

- GET /api/action-history: Lấy lịch sử hành động thiết bị với khả năng phân trang, sắp xếp, tìm kiếm và lọc.



- GET /api/profile: Lấy thông tin hồ sơ người dùng



IV. KẾT LUẬN

Trong đề tài này, em đã xây dựng thành công một hệ thống IoT giám sát và điều khiển thiết bị gia đình dựa trên vi điều khiển ESP32, giao thức MQTT, cơ sở dữ liệu MySQL và giao diện web. Hệ thống được thiết kế với kiến trúc phân tầng rõ ràng, bao gồm: lớp cảm biến và thiết bị chấp hành (ESP32 và các cảm biến), lớp truyền thông (MQTT Broker), lớp xử lý dữ liệu (API PHP, MySQL) và lớp giao diện (trình duyệt web).

Về mặt giám sát, hệ thống thu thập dữ liệu cảm biến (nhiệt độ, độ ẩm, ánh sáng) từ ESP32, truyền về MQTT Broker, sau đó được một tiến trình nền xử lý và lưu trữ trực tiếp vào cơ sở dữ liệu. Người dùng có thể theo dõi dữ liệu này trên giao diện web thông qua ba ô thông tin trực quan và biểu đồ biến thiên theo thời gian, đồng thời có thể xem lại toàn bộ lịch sử dữ liệu dưới dạng bảng với chức năng tìm kiếm, lọc và phân trang.

Về mặt điều khiển, hệ thống cho phép người dùng thao tác trực tiếp trên giao diện web để bật/tắt các thiết bị điện như đèn, quạt, điều hòa. Các lệnh điều khiển được gửi thông qua API đến MQTT Broker và được ESP32 nhận về để thực hiện. Nhờ vậy, hệ thống đảm bảo khả năng phản hồi gần thời gian thực, giúp người dùng tương tác nhanh chóng và thuận tiện.

Hệ thống được xây dựng với ưu điểm:

- Phân tách rõ ràng giữa giao diện, API xử lý dữ liệu, và API giao tiếp MQTT → dễ dàng bảo trì, mở rộng.
- Cập nhật dữ liệu liên tục mà không cần tải lại trang nhờ socketIO.
- Ứng dụng thực tế trong mô hình nhà thông minh, giám sát môi trường, tiết kiệm chi phí và có tính khả thi cao.

Tuy nhiên, hệ thống vẫn còn một số hạn chế:

- Mới dừng ở mức thử nghiệm với local, chưa triển khai trên server thực tế.
- Chưa có cơ chế bảo mật nâng cao cho MQTT và API.
- Giao diện web còn đơn giản, chưa có phân quyền người dùng.
- Chưa có ứng dụng di động để hỗ trợ điều khiển từ xa thuận tiện hơn.

Trong tương lai, em định hướng sẽ mở rộng và hoàn thiện hệ thống theo các hướng:

- Bổ sung thêm nhiều loại cảm biến (khí gas, khói, chuyển động) để nâng cao tính ứng dụng.
- Tích hợp bảo mật MQTT với SSL/TLS và xác thực tài khoản người dùng.
- Phát triển ứng dụng di động (Android/iOS) để giám sát và điều khiển mọi lúc, mọi nơi. Kết hợp trí tuệ nhân tạo (AI) để dự đoán và tự động điều chỉnh thiết bị theo điều kiện môi trường.