

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN I**

# Đề tài :

**Theo dõi thông số trong nhà: ánh sáng, nhiệt độ, độ ẩm và điều khiển các thiết bị**

**Tên sinh viên: Lương Ngọc Yên Mã sinh viên: B21DCCN809 Giảng viên: Nguyễn Quốc Uy** Hà nội, tháng 9/2024

# I. Giới thiệu

# Đặt vấn đề

Một hệ thống thời gian thực cho phép truyền dữ liệu từ cảm biến để hiển thị cho người dùng. Người dùng có thể tương tác ngược lại cho thiết bị. Một hệ thống IoT đơn giản.

# Mục tiêu

Hệ thống sử dụng hai cảm biến để thông báo nhiệt độ và độ ẩm, ánh sáng cho người dùng thông qua trang web có thể truy cập với mạng LAN. Trang web sẽ cung cấp nhiệt độ và độ ẩm, ánh sáng với dashboard hiển thị dễ nhìn, cho phép người dùng có

thể xem thông tin nhiệt độ và độ ẩm trực tiếp ở thời gian thực và có thể điều khiển led.

# Các thiết bị được sử dụng

* 1. **CHIP WIFI ESP WROOM32**

1. **Mô tả CHIP WIFI ESP WROOM32:**

Chip WiFi ESP WROOM32 là một module phát triển mạnh mẽ dựa trên vi xử lý ESP32, thiết kế nhằm cung cấp giải pháp kết nối và giao tiếp không dây qua mạng WiFi hoặc Bluetooth cho các ứng dụng IoT (Internet of Things) và các ứng dụng không dây khác. Với sự tích hợp cao và tính năng đa dạng, ESP WROOM32 là lựa chọn lý tưởng cho những dự án cần khả năng kết nối mạnh mẽ và hiệu quả.

Chip ESP WROOM32 được tích hợp sẵn khả năng kết nối WiFi theo chuẩn 802.11 b/g/n và giao tiếp qua giao thức TCP/IP, hỗ

trợ cả Bluetooth và Bluetooth Low Energy (BLE).

Ngoài ra, nó cũng hỗ trợ giao thức MQTT (Message Queuing Telemetry Transport), giúp dễ dàng trao đổi dữ liệu trong các ứng dụng IoT. Được thiết kế với nhiều giao tiếp ngoại vi và bộ nhớ lớn, ESP WROOM32 đáp ứng nhiều nhu cầu đa dạng trong các ứng dụng phát triển.



# Tính năng và thông số kỹ thuật của ESP WROOM32:

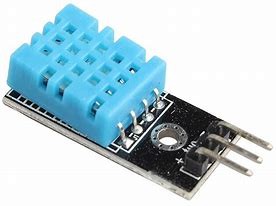
* + Vi xử lý: ESP32, dựa trên kiến trúc Xtensa dual-core 32-bit LX6.
  + Tần số hoạt động: 2.4 GHz cho WiFi, hỗ trợ Bluetooth và BLE.
  + Giao tiếp: WiFi 802.11 b/g/n, Bluetooth 4.2, BLE, hỗ trợ giao thức TCP/IP.
  + Giao tiếp ngoại vi: GPIO (General Purpose Input/Output), UART, I2C, SPI, DAC (Digital-to-Analog Converter), ADC (Analog-to-Digital Converter), PWM (Pulse-Width Modulation).
  + Bộ nhớ: Flash tích hợp lên đến 4MB, RAM 520KB.
  + Điện áp hoạt động: 3.3V.
  + Tiêu thụ năng lượng: Tiết kiệm điện năng với các chế độ sleep.
  + Kích thước: Thông thường là 18 mm x 25.5 mm.

# Cảm biến nhiệt độ độ ẩm DHT11 a, Mô tả

Cảm biến nhiệt độ và độ ẩm DHT11 là một cảm biến giá rẻ và phổ biến trong các dự án liên quan đến đo nhiệt độ và độ ẩm môi trường. Nó được sử dụng để đo và cung cấp thông tin về nhiệt độ và độ ẩm hiện tại trong một môi trường cụ thể

Cảm biến DHT11 sử dụng giao thức 1-wire đơn giản để giao tiếp với vi điều khiển. Nó truyền dữ liệu nhiệt độ và độ ẩm dưới dạng tín hiệu kỹ thuật số qua chân Data.

Cảm biến DHT11 thường được sử dụng trong các ứng dụng như đo nhiệt độ và độ ẩm trong phòng, điều khiển tự động, hệ thống quản lý môi trường và các dự án IoT. Để sử dụng cảm biến, bạn cần cài đặt thư viện DHT11 cho vi điều khiển hoặc vi xử lý mà bạn đang sử dụng và thực hiện các lệnh đọc dữ liệu từ cảm biến thông qua chân Data



# b, DHT11 có một thiết kế đơn giản với ba chân cắm

* + - VCC: Chân nguồn dương, được kết nối với nguồn cung cấp

3.3V - 5V.

* Data: Chân dữ liệu, được sử dụng để giao tiếp với vi điều khiển hoặc vi xử lý để truyền dữ liệu nhiệt độ và độ ẩm.
* GND: Chân nguồn âm, được kết nối với chân GND của nguồn cung cấp và vi điều khiển.

c, Đặc điểm kỹ thuật chính của cảm biến nhiệt độ và độ ẩm DHT11 bao gồm:

* Phạm vi đo nhiệt độ: 0°C đến 50°C với độ chính xác ±2°C. ● Phạm vi đo độ ẩm: 20% đến 90% RH với độ chính xác ±5% RH.
* Điện áp hoạt động: 3.3V - 5V.
* Điện áp logic: 3.3V - 5V (hỗ trợ giao tiếp với các vi điều khiển 3.3V và 5V).
* Tốc độ truyền dữ liệu: Tối đa 1Hz (1 lần đo dữ liệu mỗi giây).
* Kích thước: Thường có kích thước nhỏ gọn và hình dạng hình hộp chữ nhật.

# Cảm biến quang trở

Cảm biến quang trở (LDR - Light Dependent Resistor) là một loại cảm biến ánh sáng thụ động, hoạt động dựa trên nguyên lý thay đổi điện trở khi cường độ ánh sáng chiếu vào nó thay đổi. Cụ thể:

Nguyên lý hoạt động: Khi ánh sáng mạnh chiếu vào, điện trở của quang trở giảm xuống. Ngược lại, khi ánh sáng yếu hoặc không có ánh sáng, điện trở của quang trở tăng lên. Điều này cho phép quang trở đo được mức độ cường độ ánh sáng môi trường.

Ứng dụng: Cảm biến quang trở thường được sử dụng trong các thiết bị như đèn tự động bật/tắt theo ánh sáng môi trường, hệ thống đo cường độ ánh sáng, hoặc các ứng dụng điều khiển ánh sáng dựa vào điều kiện ánh sáng hiện tại.

Cấu tạo: Quang trở thường được làm từ các chất bán dẫn như cadmium sulfide (CdS). Khi ánh sáng chiếu vào, các electron trong vật liệu bán dẫn được kích thích, làm giảm điện trở của cảm biến.

Tính chất: Quang trở có đặc tính phi tuyến tính, có nghĩa là sự thay đổi điện trở không tỉ lệ thuận với sự thay đổi của cường độ ánh sáng.



# Led

LED 2 chân (Light Emitting Diode) là một loại diode phát sáng khi có dòng điện đi qua theo chiều thuận. Nó là một linh kiện điện tử phổ biến, được sử dụng rộng rãi trong các thiết bị điện tử để chiếu sáng hoặc làm đèn báo hiệu.

Cấu tạo:

2 chân:

* + - Chân dương (Anode): Đây là chân dài hơn, được kết nối với cực dương (+) của nguồn điện.
    - Chân âm (Cathode): Chân này ngắn hơn và được kết nối với cực âm (-) của nguồn điện.

Nguyên lý hoạt động:

Khi có dòng điện đi từ Anode sang Cathode (theo chiều thuận của diode), các electron sẽ chuyển động và tái hợp với các lỗ trống trong cấu trúc vật liệu bán dẫn của LED. Quá trình này giải phóng năng lượng dưới dạng ánh sáng.

LED là một diode nên chỉ phát sáng khi dòng điện đi theo một hướng nhất định (từ chân dương sang chân âm). Nếu kết nối

ngược chiều, LED sẽ không sáng. Đặc điểm:

Tiêu thụ năng lượng thấp: LED sử dụng ít năng lượng hơn so với các loại đèn khác.

Tuổi thọ cao: LED có tuổi thọ dài, có thể hoạt động hàng nghìn giờ.

Kích thước nhỏ gọn: Thường rất nhỏ và dễ dàng tích hợp vào các mạch điện tử.

Ánh sáng đa dạng: LED có thể phát ra nhiều màu sắc khác nhau (đỏ, xanh lá, xanh dương, trắng, vàng, v.v.) tùy thuộc vào loại vật liệu bán dẫn được sử dụng.

Ứng dụng: Được sử dụng trong đèn chỉ báo, màn hình LED, đèn nền cho màn hình hiển thị, thiết bị điện tử gia dụng, hệ thống chiếu sáng và nhiều ứng dụng khác.



# Công nghệ sử dụng

* 1. NestJs

Tổng quan về NestJS :

NestJS là một framework phát triển backend dựa trên Node.js, được xây dựng dựa trên các nguyên tắc của TypeScript và kết hợp nhiều mô hình từ các framework hiện đại như Angular, giúp việc phát triển ứng dụng dễ dàng và có tính module hóa cao. NestJS hỗ trợ đầy đủ các tính năng như Dependency Injection (DI), MVC (Model-View-Controller), và các middleware phổ biến, giúp ứng dụng có tính mở rộng và quản lý dễ dàng.

Các tính năng chính:

1. Hỗ trợ TypeScript: Viết mã dễ đọc, an toàn và có tính module hóa cao.
2. Modularity: Cho phép chia nhỏ ứng dụng thành các module riêng lẻ.
3. Routing: Định nghĩa các routes để điều hướng yêu cầu từ client đến các controllers.
4. Middleware và Guards: Hỗ trợ các lớp middleware và guard để quản lý luồng dữ liệu, bảo mật.
5. Dependency Injection: Đảm bảo khả năng mở rộng, giảm sự phụ thuộc của các module với nhau.
6. Hỗ trợ WebSockets và GraphQL: Tích hợp dễ dàng với các công nghệ mới.

**Cài đặt NestJS**

Để cài đặt NestJS, bạn cần cài đặt Node.js và npm trước, sau đó sử dụng CLI của NestJS để tạo và quản lý ứng dụng

1. Cài đặt Node.js và npm Trước tiên, bạn cần cài Node.js nếu chưa cài đặt: - Truy cập trang

[Node.js](https://nodejs.org/en/) để tải và cài đặt Node.js (kèm npm).

1. Cài đặt NestJS CLI Sử dụng lệnh dưới đây để cài đặt CLI của NestJS toàn cục trên hệ thống của bạn: npm

install -g @nestjs/cli

1. Tạo dự án mới với NestJS Sau khi cài đặt CLI, bạn có thể tạo một dự án NestJS mới bằng lệnh: nest new project-name

- Sau khi chạy lệnh này, CLI sẽ yêu cầu bạn chọn môi trường quản lý package (npm, yarn, pnpm), chọn một tùy chọn và chờ CLI tạo cấu trúc dự án cho bạn.

1. Cấu trúc thư mục Khi dự án được tạo, bạn sẽ thấy cấu trúc thư mục giống như sau:

src/

├── app.controller.ts

├── app.controller.spec.ts

├── app.module.ts

├── app.service.ts

└── main.t

* app.controller.ts: Nơi định nghĩa các route và phương thức xử lý yêu cầu.
* app.module.ts: Định nghĩa các module trong ứng dụng.
* app.service.ts: Chứa logic nghiệp vụ của ứng dụng.
* main.ts: File khởi tạo ứng dụng.

1. Chạy ứng dụng

Để chạy ứng dụng, sử dụng lệnh:

npm run start

Ứng dụng sẽ chạy trên `http://localhost:3000` theo mặc định.

Bạn có thể kiểm tra bằng cách truy cập vào URL đó.

1. Cài đặt thêm các package NestJS cho phép tích hợp rất nhiều thư viện như `TypeORM`, `GraphQL`,

`Swagger`, v.v. Ví dụ để cài `TypeORM` và kết nối với cơ sở dữ liệu MySQL: npm install --save @nestjs/typeorm

typeorm mysql2

1. Tạo một module mới Để tạo thêm module mới, bạn có thể sử dụng CLI: nest generate module module-name Ví dụ: nest generate module users

Kết luận

NestJS là một framework mạnh mẽ giúp bạn phát triển backend nhanh chóng và dễ dàng với các tính năng hiện đại. Với việc hỗ trợ TypeScript, module hóa và tích hợp

dễ dàng với các dịch vụ khác, NestJS là lựa chọn tốt cho việc xây dựng các ứng dụng backend quy mô lớn.

# ReactJs

ReactJS là gì?

ReactJS là thư viện JavaScript phát triển bởi Facebook, dùng để xây dựng giao diện người dùng (UI), đặc biệt là các ứng dụng một trang (SPA). React cho phép tạo các thành phần (components) có thể tái sử dụng, giúp dễ quản lý và phát triển

Các tính năng chính:

* Component-based: Xây dựng UI bằng các thành phần có thể tái sử dụng.
* Virtual DOM: Cải thiện hiệu suất bằng cách cập nhật DOM hiệu quả.
* One-Way Data Binding: Dữ liệu được truyền theo một chiều, giúp dễ kiểm soát.

Cài đặt ReactJS:

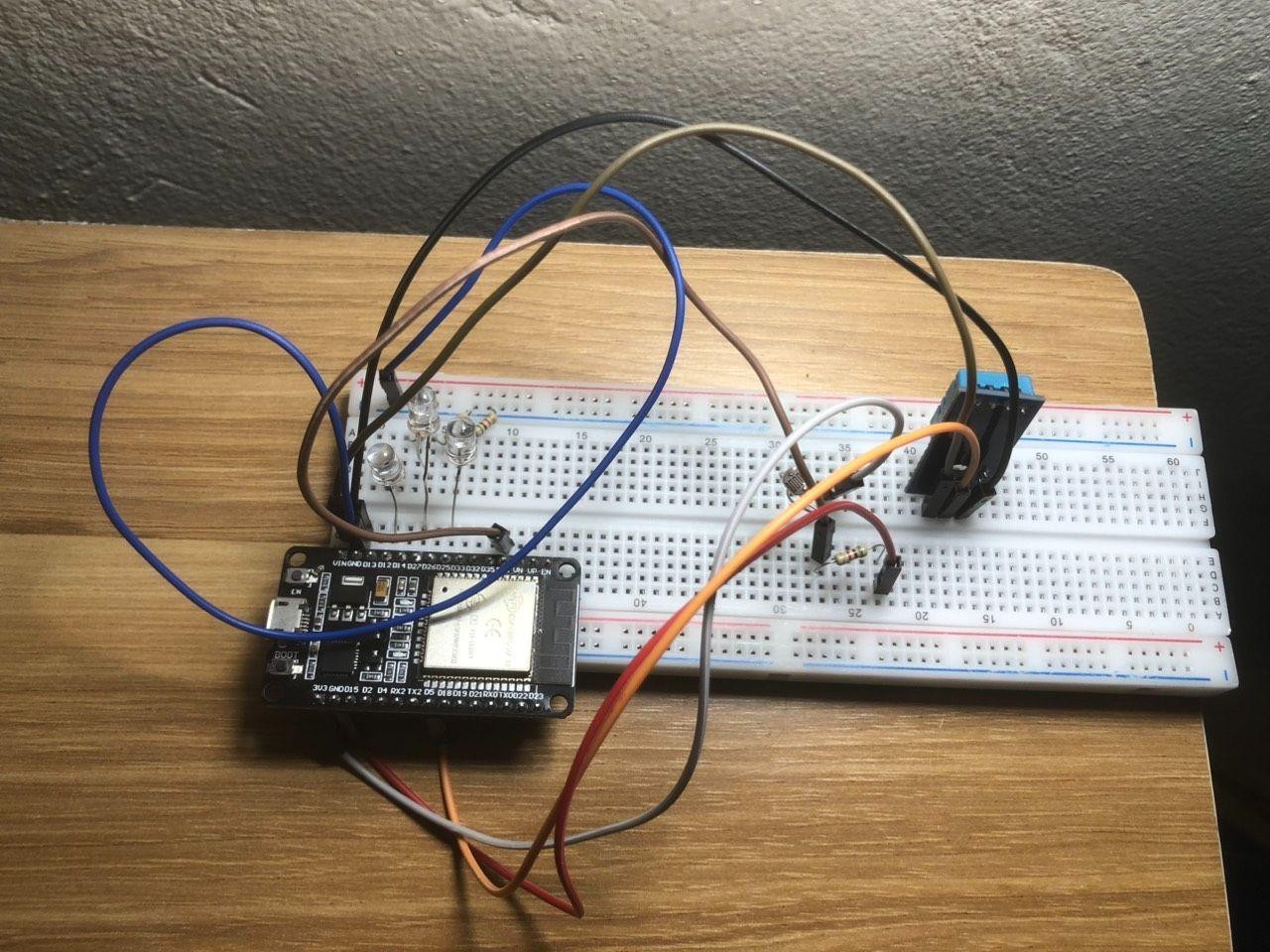
1. Cài đặt Node.js từ [Node.js](https://nodejs.org/en/).
2. Tạo ứng dụng React mới: npx create-react-app my-app
3. Chạy ứng dụng:

cd my-app npm start

Ứng dụng sẽ chạy tại `http://localhost:3000`.

# Giao diện

1. **Thiết kế tổng thể**



Giao diện web sẽ gồm 4 trang:

Giao diện web sẽ gồm 4 trang:

+ Dashboard: Sẽ chứa biểu đồ, các nút điều khiển thiết bị và 3 ô giá trị thời gian thực nhận được từ thiết bị

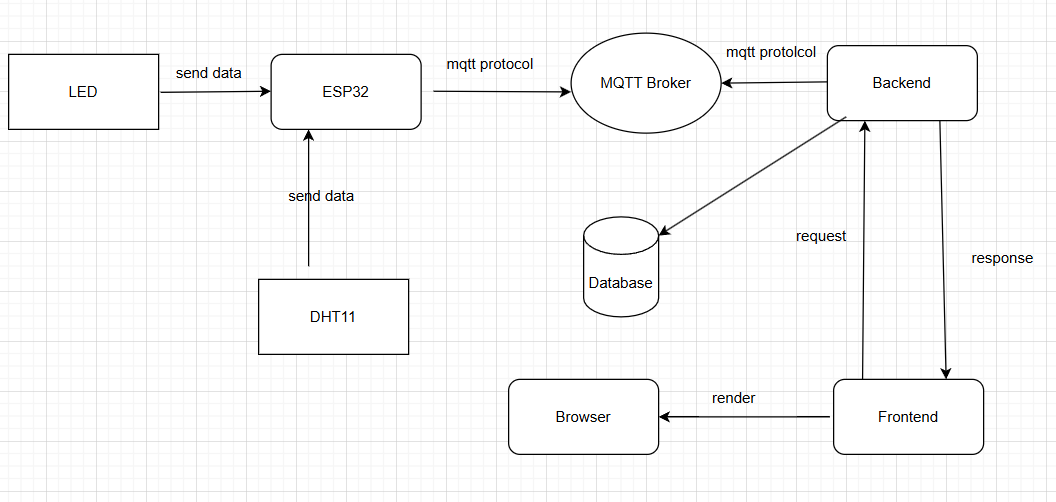
+ DataHistory: Một bảng gồm các cột thông tin về sensor và có thể tìm kiếm và sắp xếp

+ ActionHistory: Một bảng gồm các cột thông tin về lịch sử bật tắt thiết bị và có thể tìm kiếm.

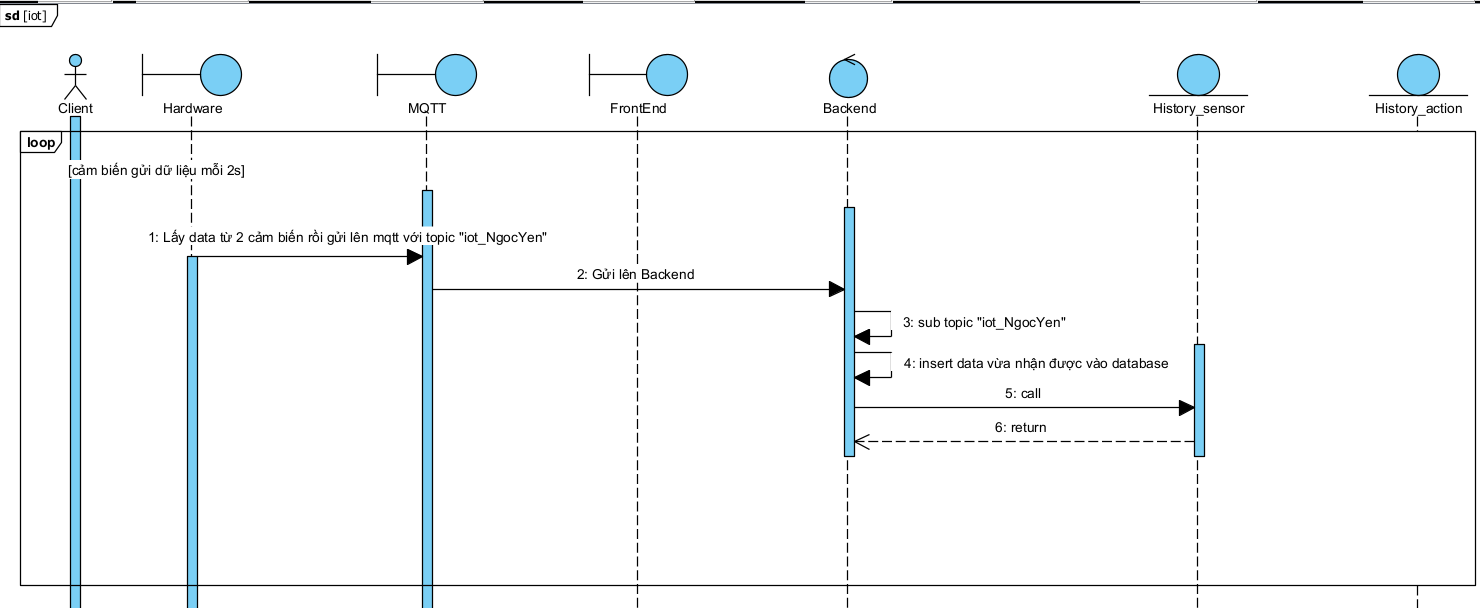
+ Profile: Chứa các thông tin các nhân và link src, api, báo cáo

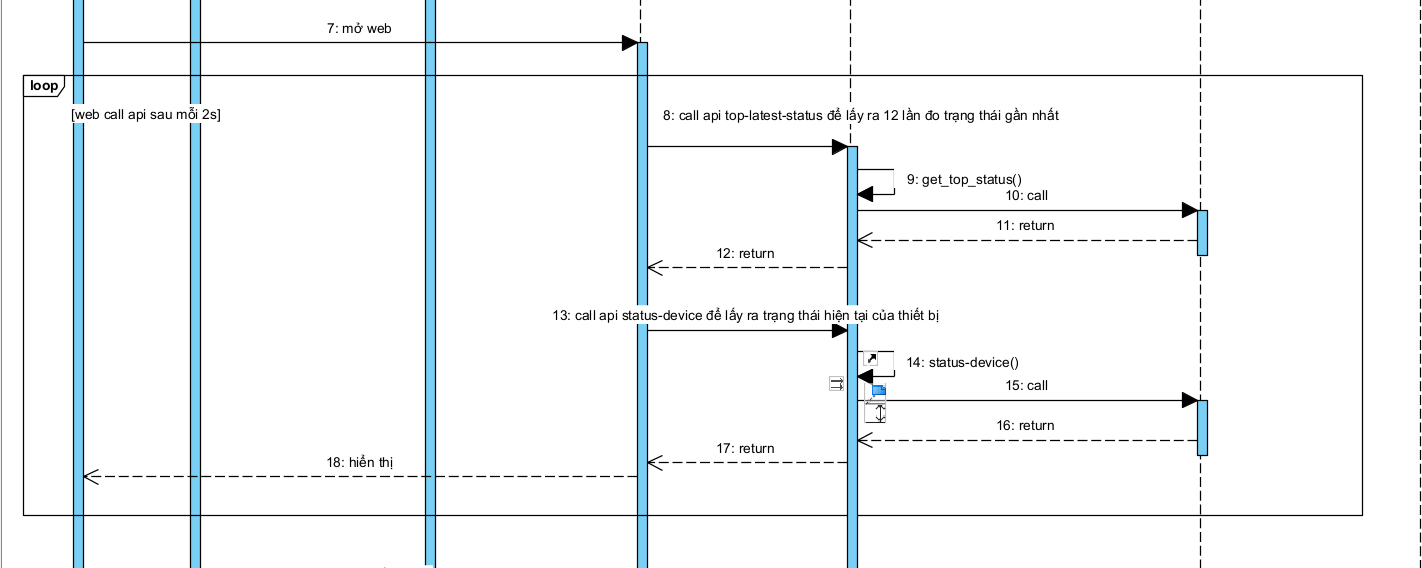
# Thiết kế chi tiết

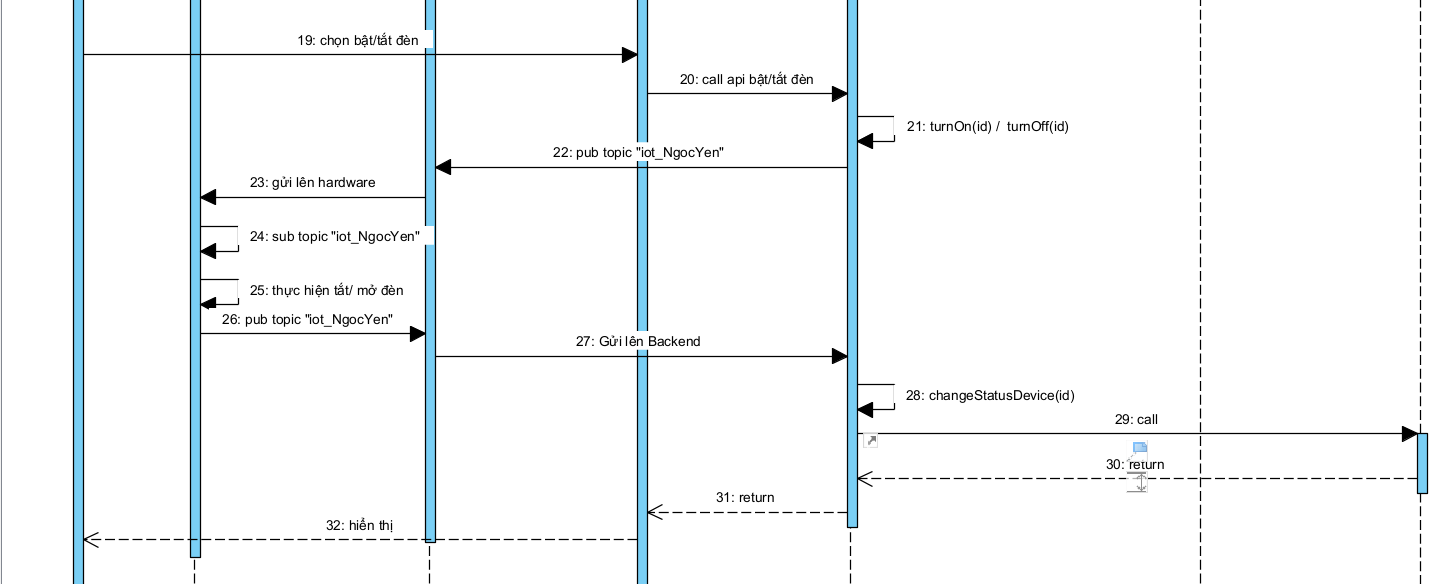
* 1. **Luồng dữ liệu**

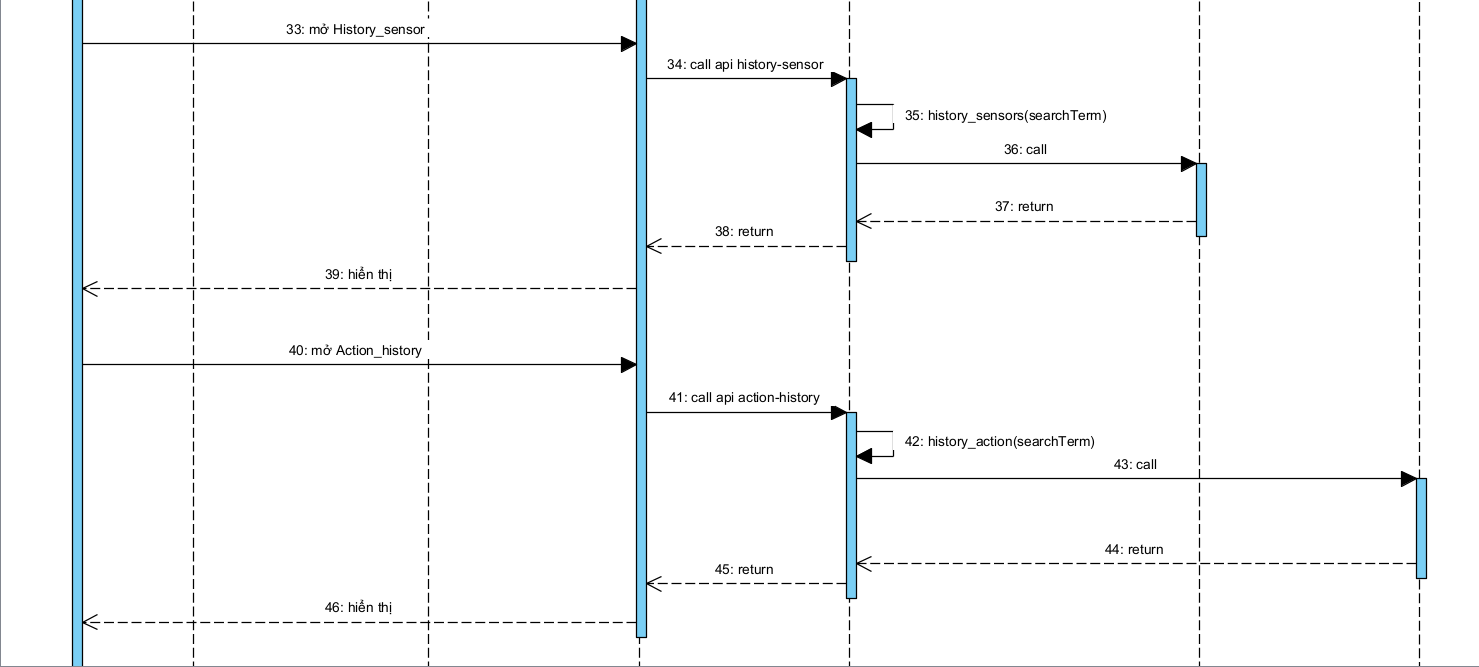


* 1. **Sơ đồ tuần tự**



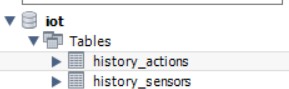






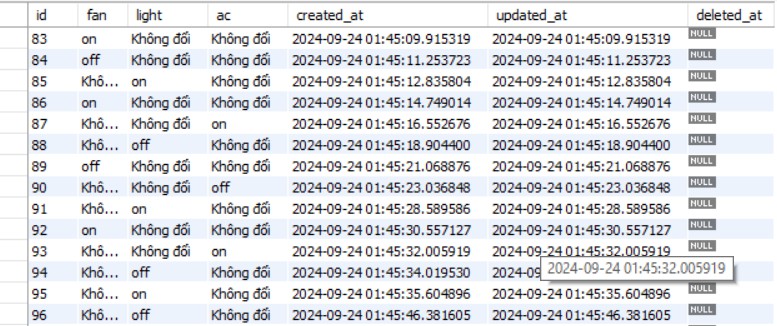
# Database

Gồm 2 bảng:

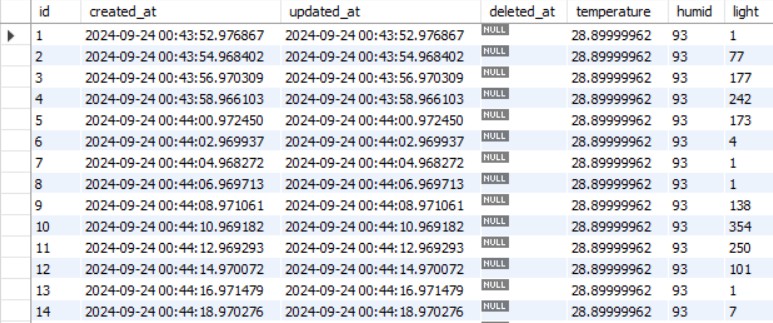


* + - history\_action: chứa data lịch sử bật tắt của thiết bị
    - history\_sensors : chứa data trạng thái của 3 thiết bị đèn

, quạt , điều hòa History\_actions :



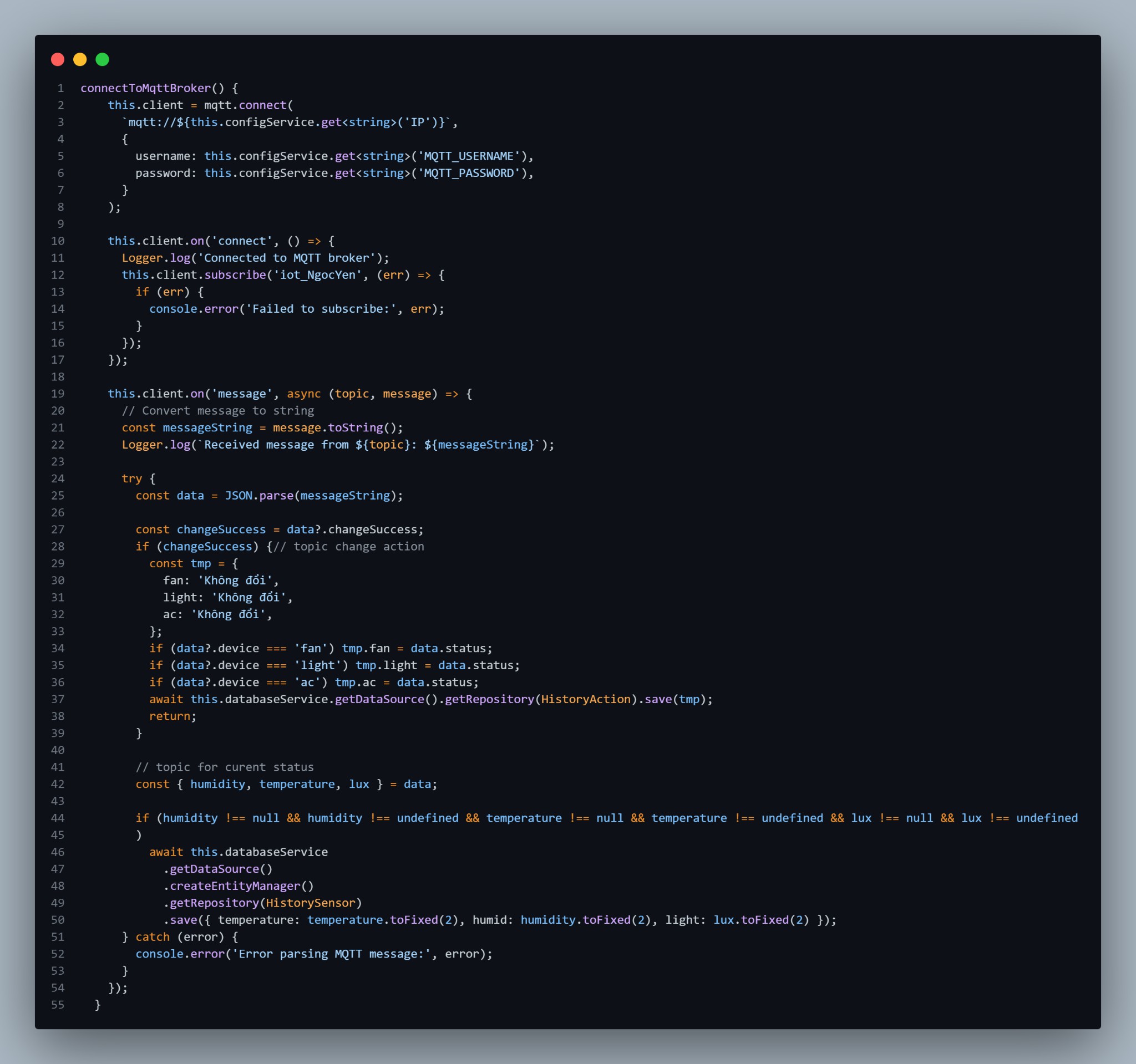
History\_sensors :



# Code

- Setup Mqtt:





- Module Controller :

+ mqtt.module.ts



+ mqtt.controller.ts





- mqtt.service.ts



Hình 1 : API tìm kiếm phân trang history\_sensors



Hình 2: API tìm kiếm và phân trang ActionHistory



Hình 3 : API lấy ra trạng thái của các thiết bị

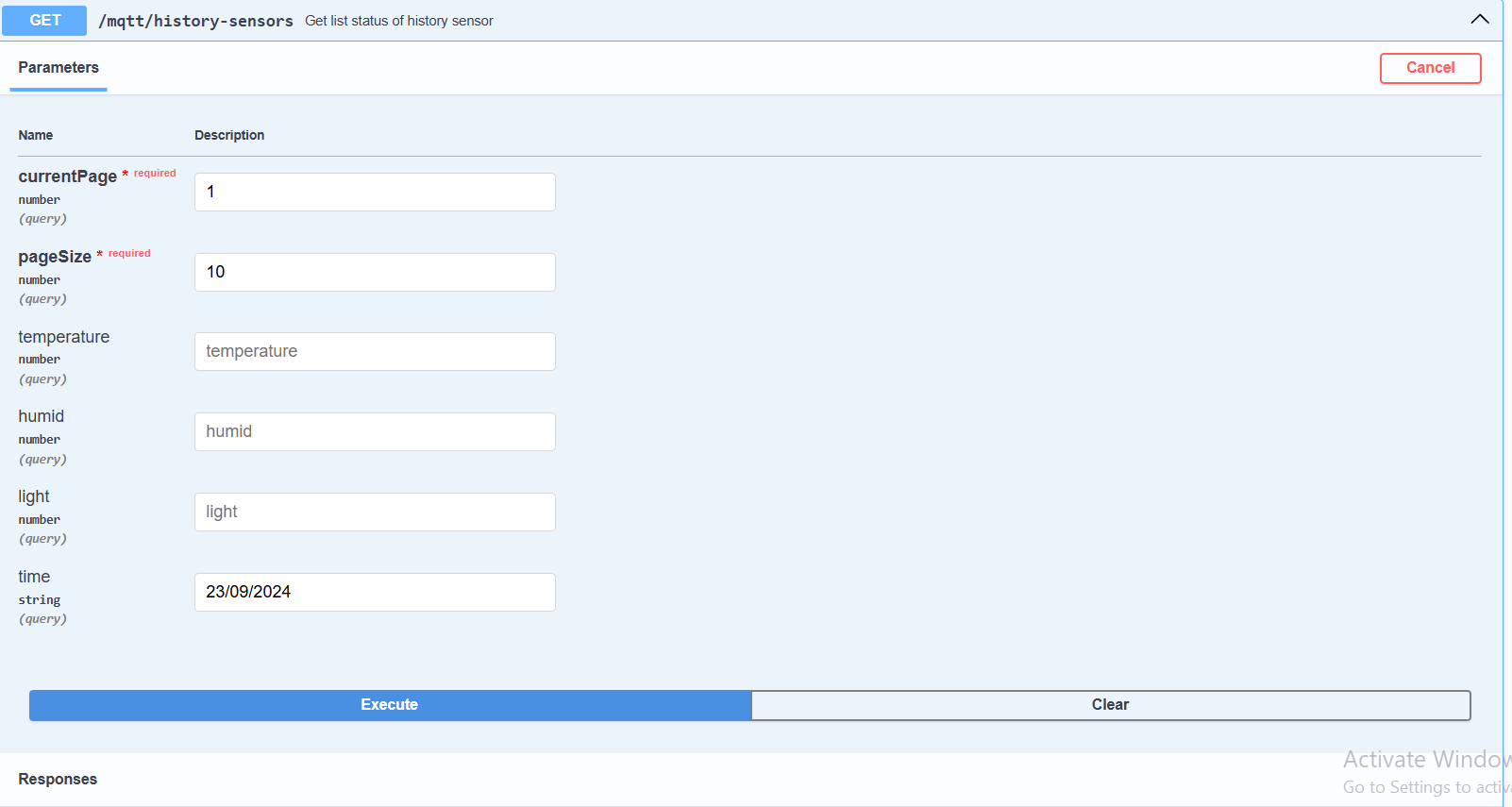


Hình 4: API thay đổi trạng thái thiết bị

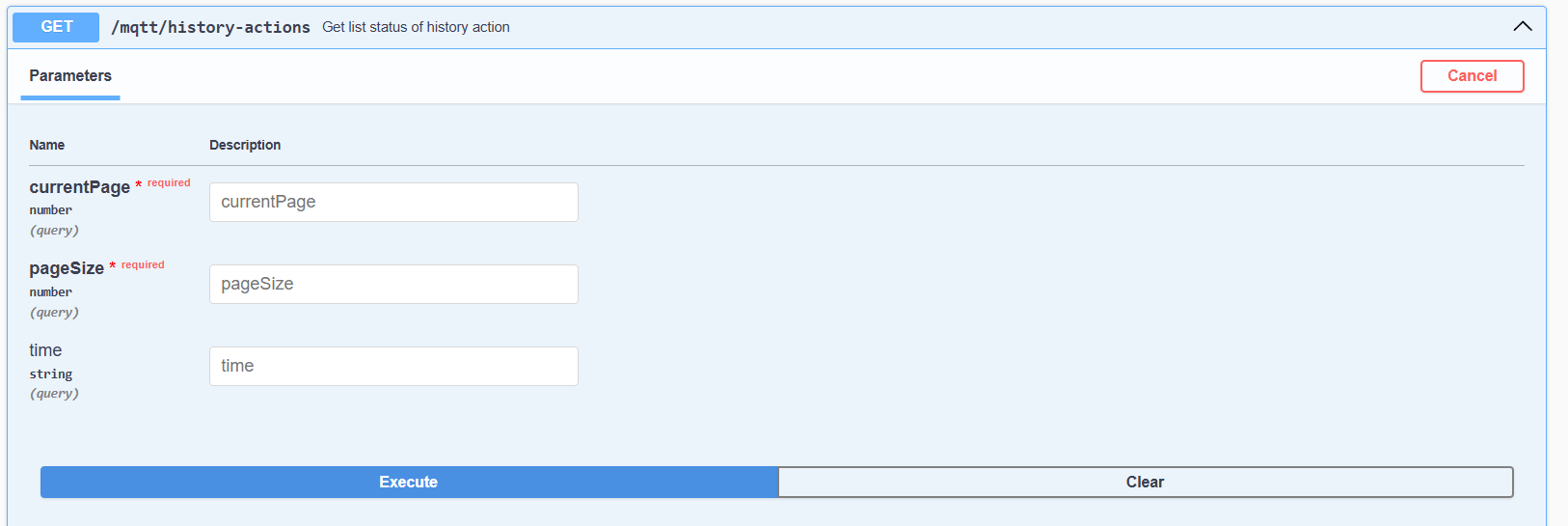


Hình 5: API lấy ra 12 trạng thái gần nhất của phòng

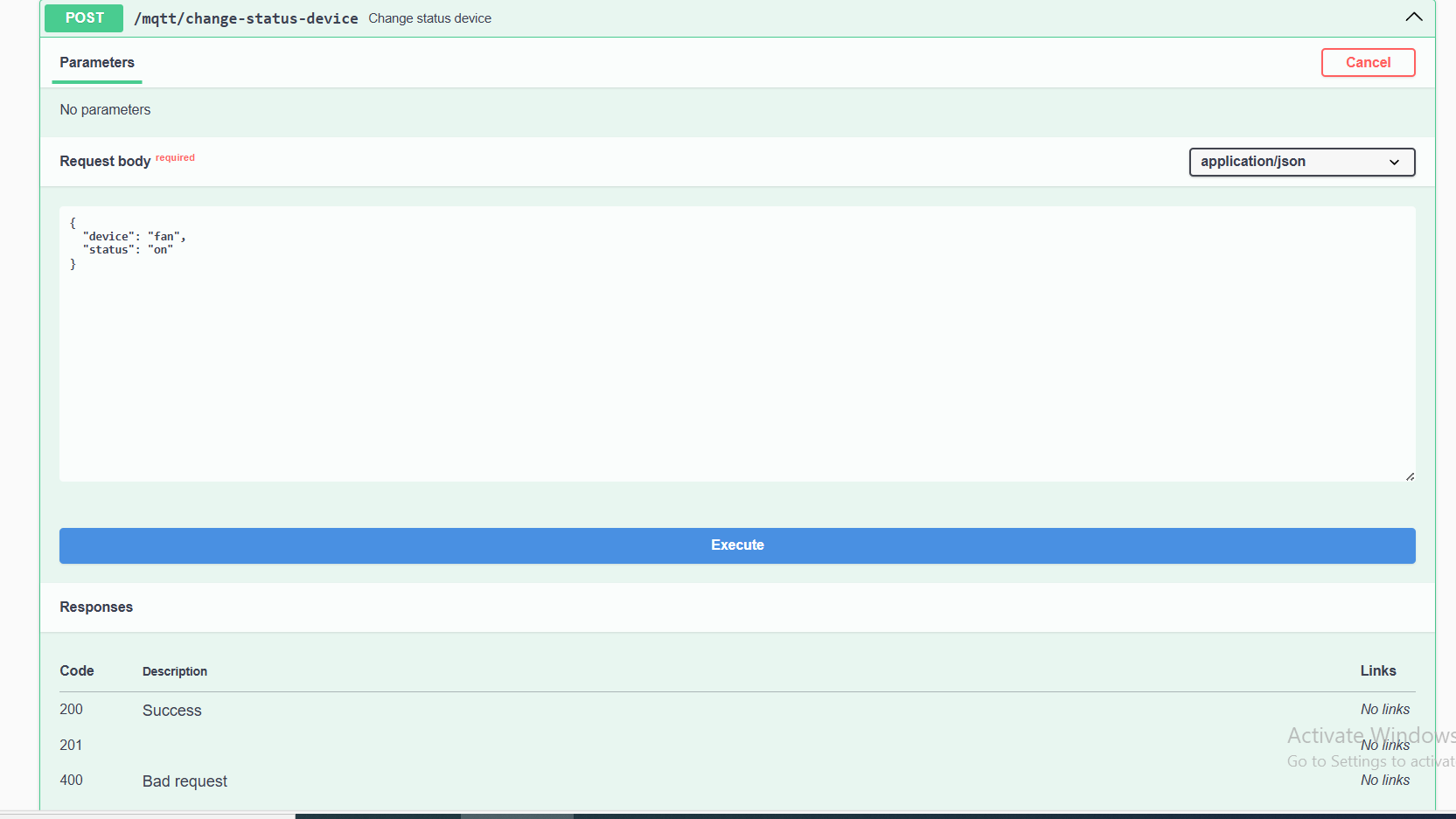
# Swagger



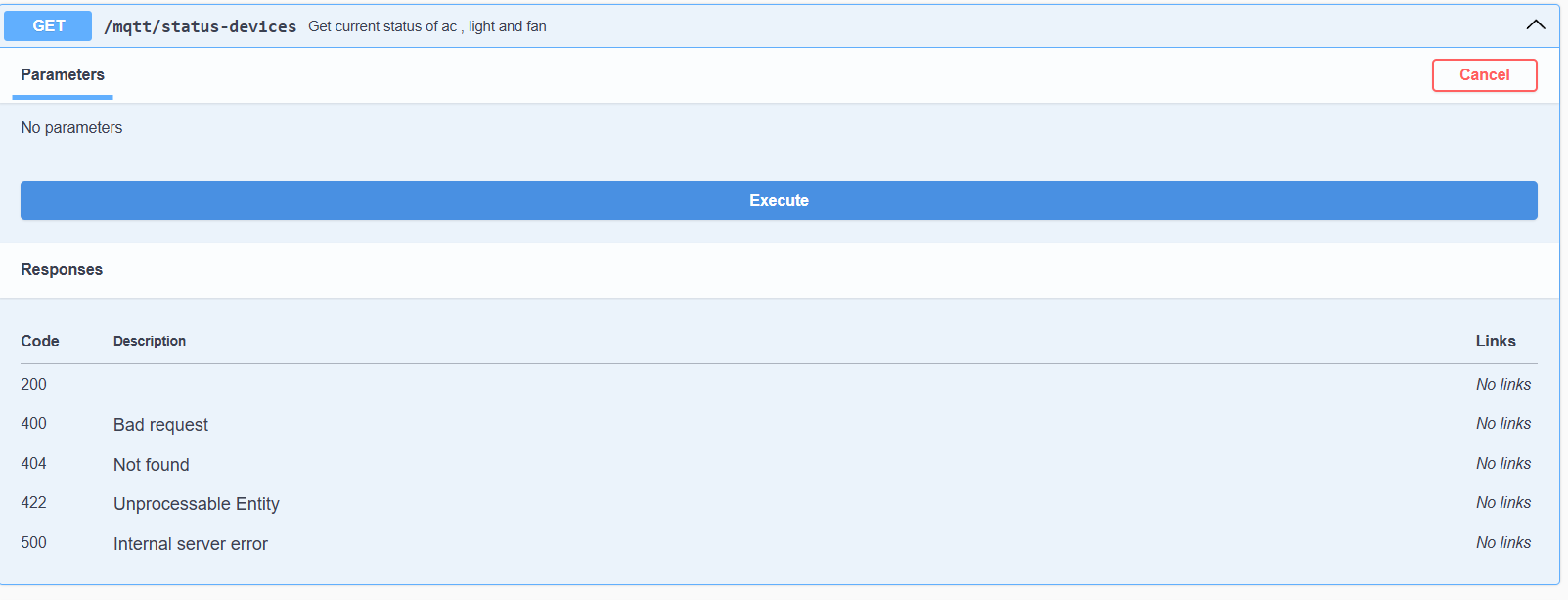
Hình 6 : API SWAGGER lấy danh sách kết quả đo cảm biến

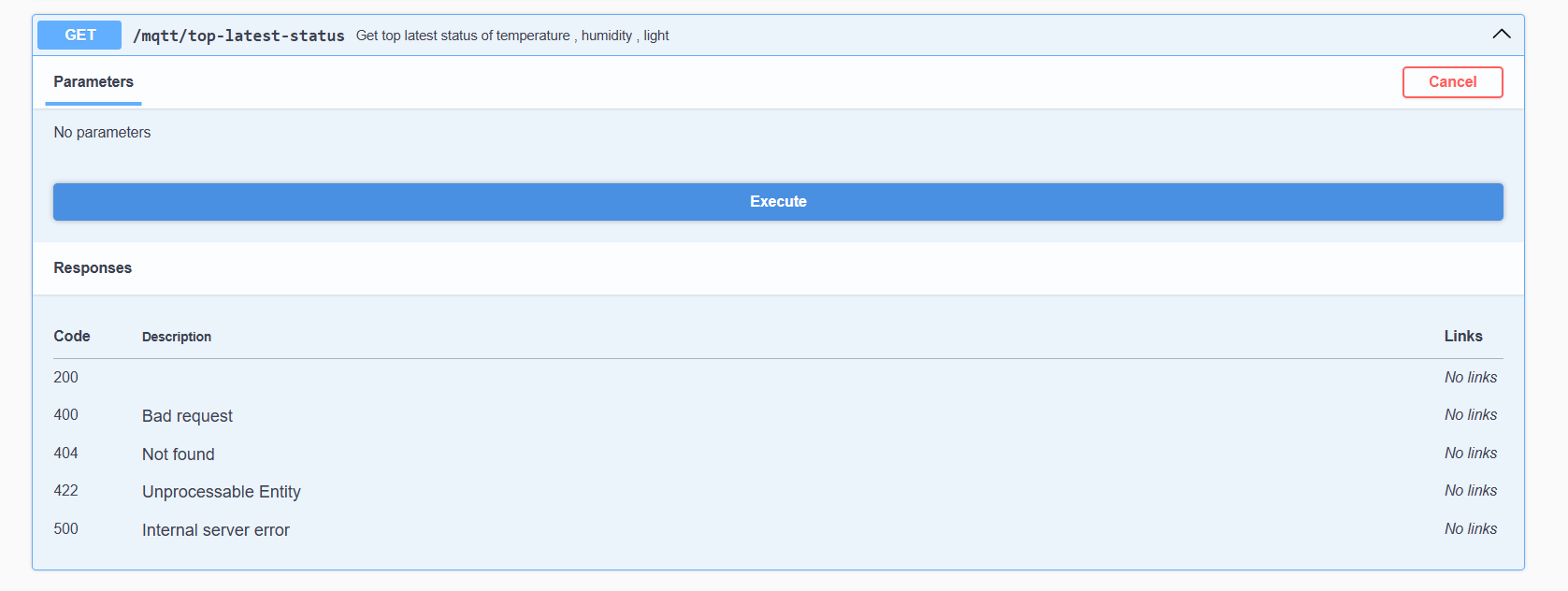


Hình 7 : API lấy danh sách lịch sử bật tắt thiết bị



Hình 8 : API SWAGGER bật tắt thiết bị



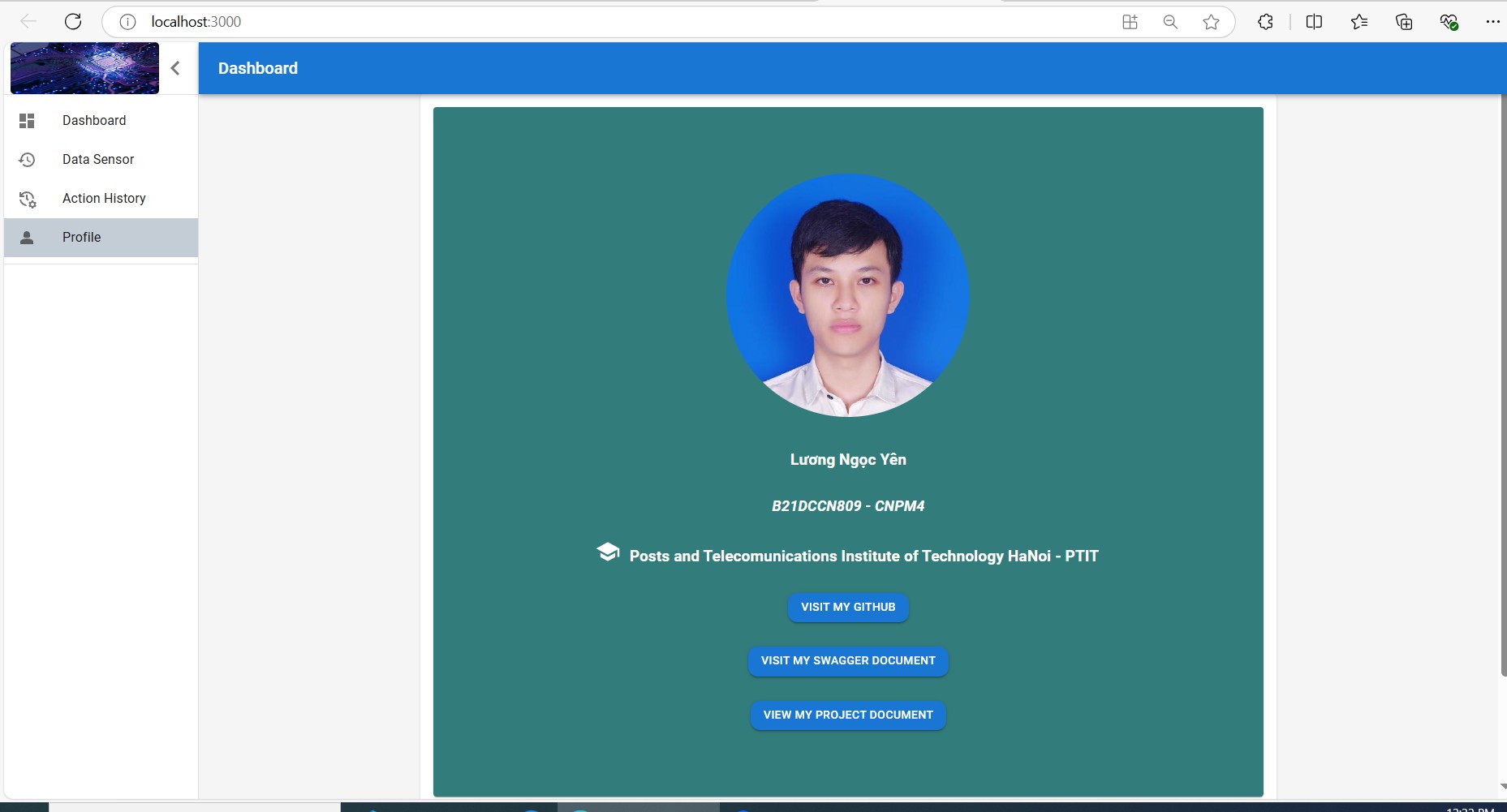
Hình 9 : API SWAGGER lấy ra trạng thái hiện tại của các thiết bị

Hình 10 : API SWAGGER lấy ra các trạng thái mới nhất của các giá trị kết quả cảm biến

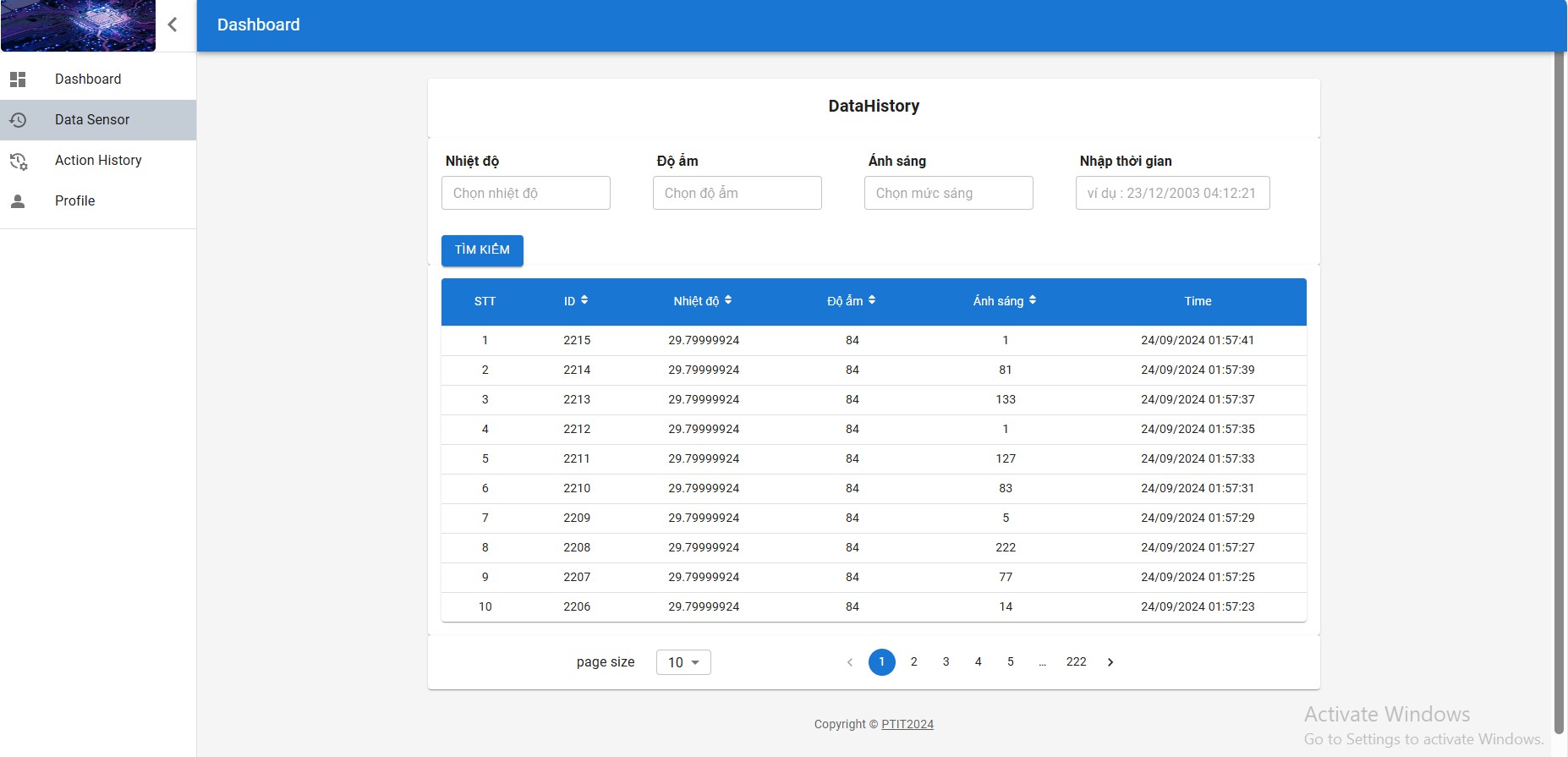
# Kết quả



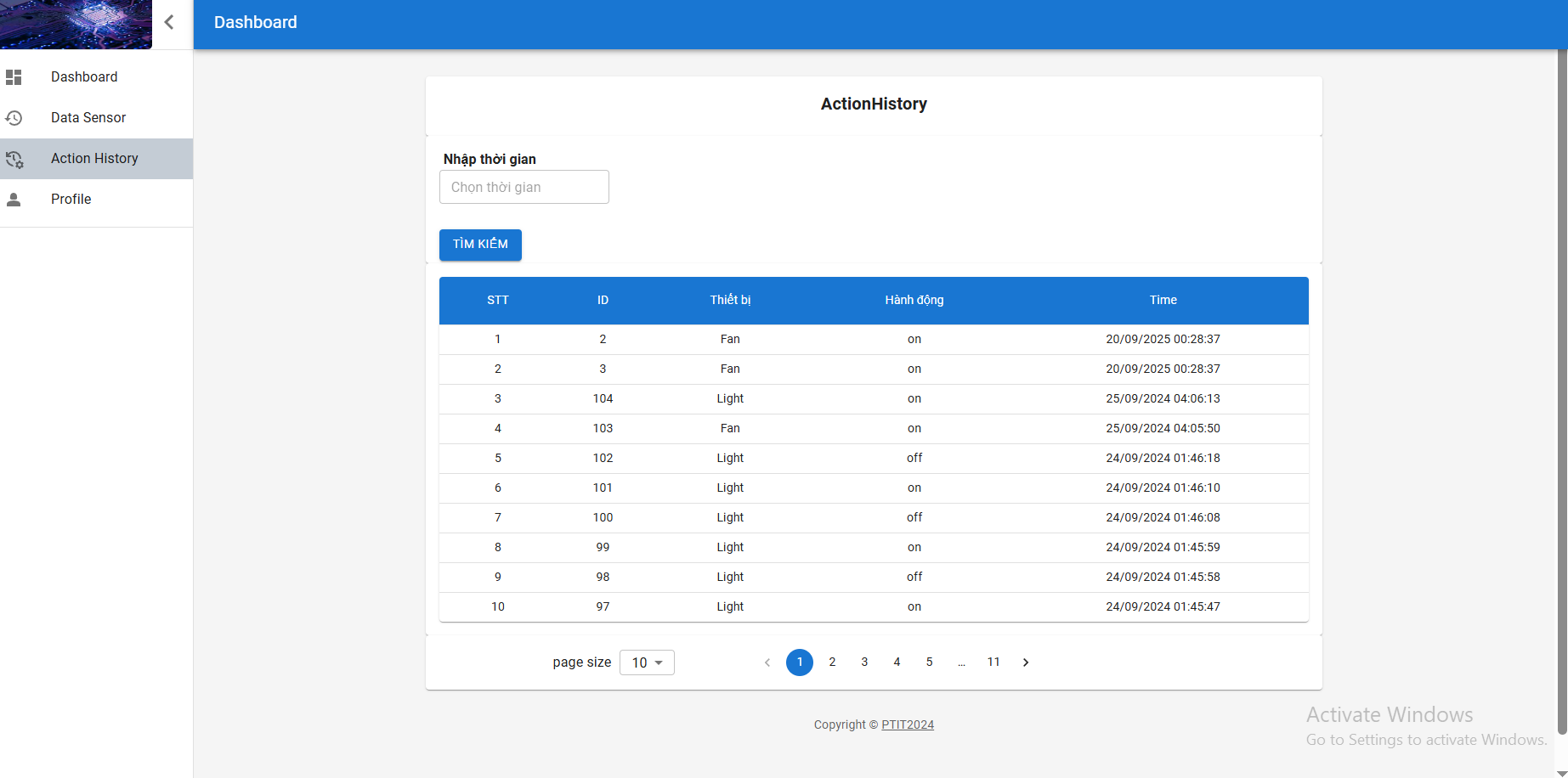
Hình 11: Giao diện dashboard với dữ liệu thời gian thực và các nút điều khiển thiết bị



Hình 12: Giao diện profile



Hình 13 : Giao diện DataHistory



Hình 14 : Giao diện ActionHistory

**Mục Lục**

1. Giới thiệu 2
2. [Đặt vấn đề 2](#_TOC_250003)
3. [Mục tiêu 2](#_TOC_250002)
4. [Các thiết bị được sử dụng 11](#_TOC_250001)
5. [Công nghệ sử dụng 17](#_TOC_250000)
6. Code 23
7. Swagger 36
8. Kết quả 40