**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**

**Lớp học phần: INT14149-20251-15**

**Họ và tên: Lê Quang Huy**

**Mã sinh viên: B22DCCN382**

***Hà Nội – 2025***

MỤC LỤC

[DANH MỤC HÌNH ẢNH 4](#_Toc211599286)

[DANH MỤC TỪ VIẾT TẮT 5](#_Toc211599287)

[CHƯƠNG I: GIỚI THIỆU CHUNG 6](#_Toc211599288)

[I. Thực trạng và mục đích hệ thống 6](#_Toc211599289)

[1. Thực trạng 6](#_Toc211599290)

[2. Mục tiêu dự án 6](#_Toc211599291)

[3. Yêu cầu hệ thống 6](#_Toc211599292)

[CHƯƠNG II: TÌM HIỂU CÔNG NGHỆ LIÊN QUAN 7](#_Toc211599293)

[I. NodeMCU 7](#_Toc211599294)

[1. Giới thiệu chung 7](#_Toc211599295)

[2. Thông tin cơ bản 7](#_Toc211599296)

[3. Ưu điểm 7](#_Toc211599297)

[4. Ứng dụng phổ biến 7](#_Toc211599298)

[II. Microsoft SQL Server 8](#_Toc211599299)

[1. SQL 8](#_Toc211599300)

[2. Microsoft SQL Server 8](#_Toc211599301)

[III. Javascript, HTML, CSS 8](#_Toc211599302)

[1. Javascript 8](#_Toc211599303)

[2. HTML 8](#_Toc211599304)

[3. CSS 9](#_Toc211599305)

[IV. Giao thức MQTT 9](#_Toc211599306)

[1. Giới thiệu chung 9](#_Toc211599307)

[2. Mosquitto 9](#_Toc211599308)

[CHƯƠNG III: THIẾT KẾ HỆ THỐNG 10](#_Toc211599309)

[I. Kiến trúc hệ thống 10](#_Toc211599310)

[II. Sơ đồ hoạt động 11](#_Toc211599311)

[1. Hiển thị dữ liệu 11](#_Toc211599312)

[2. Điều khiển LED 12](#_Toc211599313)

[3. Thiết kế cơ sở dữ liệu 13](#_Toc211599314)

[III. Thiết kế giao diện 13](#_Toc211599315)

[1. Trang profile 13](#_Toc211599316)

[2. Giao diện chính 14](#_Toc211599317)

[3. Giao diện dữ liệu cảm biến 14](#_Toc211599318)

[4. Giao diện lịch sử bật tắt thiết bị 15](#_Toc211599319)

[IV. Sequence diagram 15](#_Toc211599320)

[1. Hiển thị dữ liệu cảm biến 15](#_Toc211599321)

[2. Điều khiển LED 17](#_Toc211599322)

[CHƯƠNG IV: CÀI ĐẶT HỆ THỐNG 19](#_Toc211599323)

[I. Chuẩn bị phần cứng 19](#_Toc211599324)

[1. Thiết bị cần thiết 19](#_Toc211599325)

[2. Lắp đặt thiết bị 19](#_Toc211599326)

[II. Backend 19](#_Toc211599327)

[1. Tạo cơ chế xác thực token 19](#_Toc211599328)

[2. Cấu hình kết nối database 20](#_Toc211599329)

[3. Cấu hình kết nối MQTT Broker và đăng ký topic 21](#_Toc211599330)

[4. Xử lý dữ liệu nhận được từ Broker 21](#_Toc211599331)

[5. API Login 23](#_Toc211599332)

[6. API LED 23](#_Toc211599333)

[7. API Sensor 27](#_Toc211599334)

**DANH MỤC HÌNH ẢNH**

Hình 1 Kiến trúc hệ thống IoT 9

Hình 2 Activity Diagram của luồng dữ liệu 10

Hình 3 Activity Diagram của luồn điều khiển đèn 11

Hình 4 Entity Relationship Diagram 12

Hình 5 Giao diện trang Profile 12

Hình 6 Giao diện trang Home 13

Hình 7 Giao diện trang Sensor 13

Hình 8 Giao diện trang Device Log 14

Hình 9 Sequence Diagram cho luồng hiển thị dữ liệu 15

Hình 10 Sequence Diagram cho luông điều khiển đèn 16

Hình 11 Hàm verifyToken để xác thưc jtoken cho api 18

Hình 12 Cấu hình để kết nối database 19

Hình 13 Cấu hình để kết nối MQTT Broker 20

Hình 14 Code để xử lý dữ liệu cảm biến do Broker gửi về 20

Hình 15 Code để xử lý trạng thái đèn do Broker gửi 21

Hình 16 Mô tả api đăng nhập 22

Hình 17 Mô tả api điều khiển LED 23

Hình 18 Mô tả api hiển thị lịch sử bật/tắt LED 24

Hình 19 Mô tả api lấy trạng thái LED 25

Hình 20 Mô tả api lấy dữ liệu cẩm biến 26

Hình 21 Mô tả api lấy lịch sử dữ liệu cảm biến 27

# DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Từ đầy đủ | Giải nghĩa |
| IoT | Internet of Things | Internet kết nối vạn vật |
| MQTT | Message Queuing Telemetry Transport | Giao thức truyền tải thông điệp theo hàng đợi và đo lường từ xa |
| API | Application Programming Interface | Giao diện lập trình ứng dụng |
| RAM | Random Access Memory | Bộ nhớ truy cập ngẫu nhiên |
| USB | Universal Serial Bus | Tiêu chuẩn để truyền tải dữ liệu và cung cấp điện |
| SQL | Structured Query Language | Ngôn ngữ truy vấn có cấu trúc |
| HTML | Hyper Text Markup Language | Ngôn ngữ đánh dấu siêu văn bản |
| CSS | Cascading Style Sheets | Bảng định kiểu theo tầng |

**CHƯƠNG I: GIỚI THIỆU CHUNG**

1. **Thực trạng và mục đích hệ thống**
2. **Thực trạng**

Trong những năm gần đây, Internet of Things (IoT) phát triển mạnh mẽ và dần trở thành thành phần quan trọng trong nhiều lĩnh vực như nông nghiệp, nhà thông minh, công nghiệp tự động, y tế, và năng lượng. Tuy nhiên, trên thực tế, ở nhiều nơi — đặc biệt là hộ gia đình, trang trại nhỏ, hoặc các cơ sở sản xuất thủ công — việc theo dõi và điều khiển thiết bị vẫn còn mang tính thủ công, chưa được tự động hóa.

Các đặc điểm của thực trạng hiện nay:

* Giám sát thủ công: Người vận hành phải trực tiếp kiểm tra các thông số môi trường như nhiệt độ, độ ẩm, ánh sáng… thông qua thiết bị đo cục bộ hoặc cảm nhận thủ công. Điều này tốn nhiều thời gian và công sức.
* Không có khả năng điều khiển từ xa: Khi muốn bật/tắt thiết bị (ví dụ đèn, quạt, bơm nước…), người dùng phải có mặt tại hiện trường.
* Thiếu khả năng ghi log: Dữ liệu không được lưu trữ, khiến việc phân tích xu hướng, ra quyết định tối ưu (ví dụ điều chỉnh tưới tiêu, chiếu sáng) trở nên khó khăn.
* Phản ứng chậm: Khi có sự thay đổi bất thường (nhiệt độ quá cao, ánh sáng giảm mạnh), người vận hành không nhận được cảnh báo và không thể phản ứng kịp thời.

Hệ quả:

* Năng suất thấp do quá trình vận hành thiếu tính tự động.
* Khó mở rộng hệ thống vì phụ thuộc hoàn toàn vào con người.
* Dễ xảy ra sai sót, đặc biệt trong môi trường sản xuất quy mô lớn.

1. **Mục tiêu dự án**

* Kết nối cảm biến với máy chủ thông qua giao thức MQTT.
* Thiết kế backend API có xác thực.
* Hiển thị và điều khiển thiết bị theo thời gian thực.
* Ghi lịch sử dữ liệu để truy xuất và phân tích.

1. **Yêu cầu hệ thống**

Từ mục đích của hệ thống ta có thể đưa ra phạm vi và các yêu cầu trong hệ thống như sau:

Phạm vi: Hệ thống trang web hướng đến tất cả những người sử dụng internet có nhu cầu giám sát nhiệt độ, độ ẩm, điều kiện ánh sáng và điều khiển thiết bị trong nhà

Yêu cầu: Hệ thống IoT này cho phép người dung quan sát điều kiện nhiệt độ, độ ẩm, ánh sáng trong nhà, đồng thời thao tác với đèn được kết nối với hệ thống. Ngoài ra hệ thống cho phép người dung xem các giá trị của cảm biến trong quá khứ và sự thay đổi của nhiệt độ, độ ẩm, ánh sáng trong 1 khoảng thời gian nhất định

**CHƯƠNG II: TÌM HIỂU CÔNG NGHỆ LIÊN QUAN**

**I. NodeMCU**

**1. Giới thiệu chung**

NodeMCU là một bo mạch phát triển (development board) giá rẻ, nhỏ gọn, được sử dụng phổ biến trong các dự án Internet of Things (IoT). Nó tích hợp sẵn Wi-Fi giúp lập trình viên dễ dàng kết nối thiết bị với mạng Internet hoặc mạng nội bộ mà không cần thêm module rời.

### 2. Thông tin cơ bản

* Bộ vi điều khiển: ESP8266.
* Tốc độ xung nhịp: ~80–160 MHz.
* Bộ nhớ:
  + RAM: khoảng 50 KB khả dụng cho người dùng.
  + Flash: 4 MB (tùy phiên bản).
* Giao tiếp: Wi-Fi 802.11 b/g/n, UART, SPI, I2C, PWM, ADC.
* Điện áp hoạt động: 3.3V (cổng USB cấp 5V, bo mạch có mạch ổn áp).
* Cổng lập trình: Micro-USB.
* Ngôn ngữ lập trình: Lua hoặc C/C++ thông qua Arduino IDE.

### 3. Ưu điểm

* Tích hợp Wi-Fi: Không cần thêm module rời, tiết kiệm chi phí.
* Dễ lập trình: Có thể lập trình bằng Arduino IDE, rất quen thuộc với người mới học IoT.
* Giá rẻ: Thường chỉ vài chục nghìn đồng.
* Cộng đồng lớn: Nhiều tài liệu, ví dụ mẫu, hỗ trợ tốt.
* Kích thước nhỏ gọn: Dễ tích hợp vào các sản phẩm thực tế.

### 4. Ứng dụng phổ biến

* Hệ thống nhà thông minh (điều khiển đèn, quạt, cảm biến từ xa).
* Giám sát môi trường: nhiệt độ, độ ẩm, ánh sáng.
* Điều khiển thiết bị qua ứng dụng di động hoặc giao diện web.
* Các dự án robot tự động, thiết bị đo thông minh, hoặc hệ thống báo cháy.

**d. Một số phiên bản NodeMCU phổ biến**

* NodeMCU V1.0: bản đầu tiên, kích thước lớn.
* NodeMCU V2: nhỏ hơn, ổn định hơn.
* NodeMCU V3: bản thường thấy hiện nay — có cổng Micro-USB, dễ cấp nguồn và lập trình.

## II. Microsoft SQL Server

### 1. SQL

SQL là một loại ngôn ngữ máy tính phổ biến để tạo, sửa, và lấy dữ liệu từ một hệ quản trị cơ sở dữ liệu quan hệ. Ngôn ngữ này phát triển vượt xa so với mục đích ban đầu là để phục vụ các hệ quản trị cơ sở dữ liệu đối tượng-quan hệ. Nó là một tiêu chuẩn ANSI/ISO.

### 2. Microsoft SQL Server

Microsoft SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) do Microsoft phát triển, được sử dụng rộng rãi trong các hệ thống doanh nghiệp để lưu trữ, quản lý và truy xuất dữ liệu một cách an toàn và hiệu quả. Phần mềm hỗ trợ ngôn ngữ truy vấn SQL, có khả năng xử lý lượng dữ liệu lớn, tích hợp tốt với các ứng dụng khác của Microsoft, đồng thời cung cấp nhiều tính năng mạnh mẽ như sao lưu – phục hồi dữ liệu, bảo mật cao và khả năng mở rộng linh hoạt. Đây là một giải pháp lý tưởng cho các ứng dụng web, phần mềm quản lý và hệ thống phân tích dữ liệu.

## III. Javascript, HTML, CSS

### 1. Javascript

JavaScript là một ngôn ngữ lập trình linh hoạt, chủ yếu được sử dụng để phát triển các trang web tương tác và động. Đây là một trong ba công nghệ cốt lõi của web cùng với HTML và CSS, cho phép lập trình viên xử lý các sự kiện, thay đổi nội dung trang mà không cần tải lại, và giao tiếp với máy chủ để lấy dữ liệu. JavaScript có thể chạy trực tiếp trên trình duyệt hoặc phía máy chủ thông qua Node.js, giúp xây dựng từ các trang web đơn giản đến những ứng dụng web phức tạp, hiện đạ

### 2. HTML

HTML (HyperText Markup Language) là ngôn ngữ đánh dấu tiêu chuẩn ang để tạo cấu trúc và nội dung cho các trang web. HTML xác định các phần tử như tiêu đề, đoạn văn, hình ảnh, liên kết, bảng biểu và nhiều thành phần khác, giúp trình duyệt hiểu và hiển thị thông tin đúng định dạng. Đây là nền tảng cơ bản của mọi website, cho phép các nhà phát triển xây dựng khung sườn trang một cách rõ ang và có tổ chức.

### 3. CSS

CSS (Cascading Style Sheets) là ngôn ngữ được sử dụng để định dạng và trình bày giao diện của trang web. CSS cho phép thay đổi màu sắc, bố cục, kiểu chữ, hiệu ứng và nhiều yếu tố thiết kế khác, giúp trang web trở nên sinh động và bắt mắt hơn. Bằng cách tách phần nội dung (HTML) và phần trình bày (CSS), các lập trình viên có thể quản lý giao diện dễ dàng hơn, đồng thời tạo ra trải nghiệm người dùng trực quan và chuyên nghiệp.

## IV. Giao thức MQTT

**1. Giới thiệu chung**

MQTT (Message Queuing Telemetry Transport) là giao thức truyền thông điệp (message) theo mô hình publish/subscribe (cung cấp / thuê bao), được sử dụng cho các thiết bị IoT với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định. Nó dựa trên một Broker (tạm dịch là “Máy chủ môi giới”) “nhẹ” (khá ít xử lý) và được thiết kế có tính mở (tức là không đặc trưng cho ứng dụng cụ thể nào), đơn giản và dễ cài đặt.

MQTT là lựa chọn lý tưởng trong các môi trường như:

* Những nơi mà giá mạng viễn thông đắt đỏ hoặc băng thông thấp hay thiếu tin cậy.
* Khi chạy trên thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ.
* Bởi vì giao thức này sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng M2M (Machine to Machine).

**2. Mosquitto**

Mosquitto là một MQTT Broker mã nguồn mở cho phép thiết bị truyền nhận dữ liệu theo giao thức MQTT version 5.0, 3.1.1 và 3.1 – Một giao thức nhanh, nhẹ theo mô hình publish/subscribe được sử dụng rất nhiều trong lĩnh vực Internet of Things. Mosquitto cung cấp một thư viện viết bằng ngôn ngữ C để triển khai các MQTT Client và có thể dễ dàng sử dụng bằng dòng lệnh: “mosquitto\_pub” và “mosquitto\_sub”.

**CHƯƠNG III: THIẾT KẾ HỆ THỐNG**

## I. Kiến trúc hệ thống

A diagram of a computer system

AI-generated content may be incorrect.

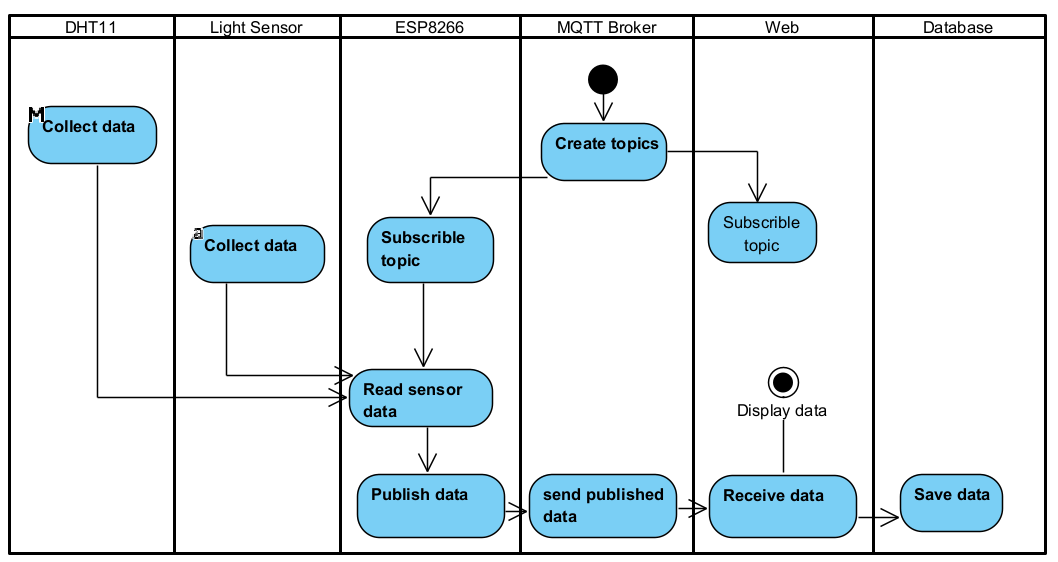
Hình 1 Kiến trúc hệ thống IoT

Mô tả hệ thống

* Phần cứng
  + Cảm biến: Thu thấp dữ liệu và gửi cho ESP8266
  + ESP8266: Nhận dữ liệu do cảm biến gửi, điều khiển LED, kết nối và giao tiếp với Broker thông qua wifi và mqtt
  + Thiết bị khác: LED và các thiết bị để kết nối các thiết bị phần cứng
* MQTT Broker
  + Giao tiếp giữa Backend và phần cứng
* Backend
  + MSSQL Server: Database để lưu dữ liệu cảm biến và trạng thái của đèn
  + NodeJS server: xử lý dữ liệu của cảm biến, gửi dữ liệu cho database, xử lý yêu cầu gọi api của Frontend
* Frontend: Giao diện để người dùng theo dõi và giao tiếp với hệ thống

## II. Sơ đồ hoạt động

### 1. Hiển thị dữ liệu

****

Hình 2 Activity Diagram của luồng dữ liệu

1. Cảm biến DHT11 và cảm biến ánh sáng thực hiện thu thập dữ liệu môi trường và gửi kết quả đến bộ xử lý trung gian ESP8266.
2. MQTT Broker khởi tạo các chủ đề (topics) để thiết lập kênh truyền dữ liệu. ESP8266 đăng ký chủ đề, đọc dữ liệu cảm biến và gửi dữ liệu lên MQTT Broker thông qua lệnh publish.
3. MQTT Broker phân phối dữ liệu đã nhận đến backend. Web subscribe chủ đề để hiển thị dữ liệu thời gian thực, đồng thời dữ liệu được lưu vào cơ sở dữ liệu để phục vụ phân tích và truy xuất sau này.

### 2. Điều khiển LED

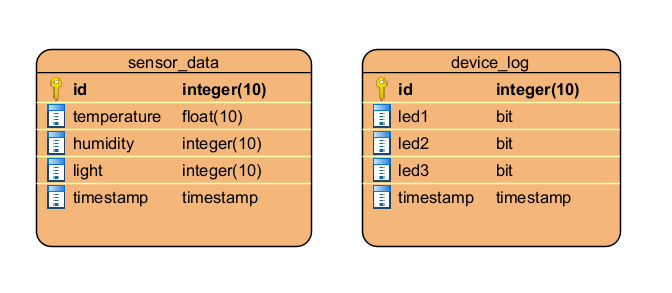
**A diagram of a chatbot

AI-generated content may be incorrect.**

Hình 3 Activity Diagram của luồn điều khiển đèn

1. Người dùng thực hiện thao tác bật/tắt đèn LED thông qua giao diện Web. Ứng dụng Web đăng ký chủ đề (subscribe topic) với MQTT Broker và gửi thông điệp điều khiển (publish message) khi người dùng thực hiện thao tác.
2. MQTT Broker khởi tạo các chủ đề và tiếp nhận thông điệp từ Web, sau đó chuyển tiếp thông điệp này đến thiết bị ESP8266 đã đăng ký chủ đề tương ứng.
3. ESP8266 nhận thông điệp, phân tích nội dung điều khiển (analyze payload) và thực hiện bật hoặc tắt đèn LED. Sau khi thay đổi trạng thái LED, ESP8266 gửi thông điệp phản hồi (publish message) để cập nhật trạng thái LED. MQTT Broker chuyển tiếp thông điệp này đến Web để giao diện hiển thị trạng thái mới, đồng thời dữ liệu được lưu vào cơ sở dữ liệu để theo dõi và quản lý lịch sử điều khiển.

### 3. Thiết kế cơ sở dữ liệu



Hình 4 Entity Relationship Diagram

## III. Thiết kế giao diện

### 1. Trang profile

A screenshot of a computer

AI-generated content may be incorrect.

Hình 5 Giao diện trang Profile

### 2. Giao diện chính

A screenshot of a computer

AI-generated content may be incorrect.

Hình 6 Giao diện trang Home

### 3. Giao diện dữ liệu cảm biến

A screenshot of a computer

AI-generated content may be incorrect.

Hình 7 Giao diện trang Sensor

### 4. Giao diện lịch sử bật tắt thiết bị

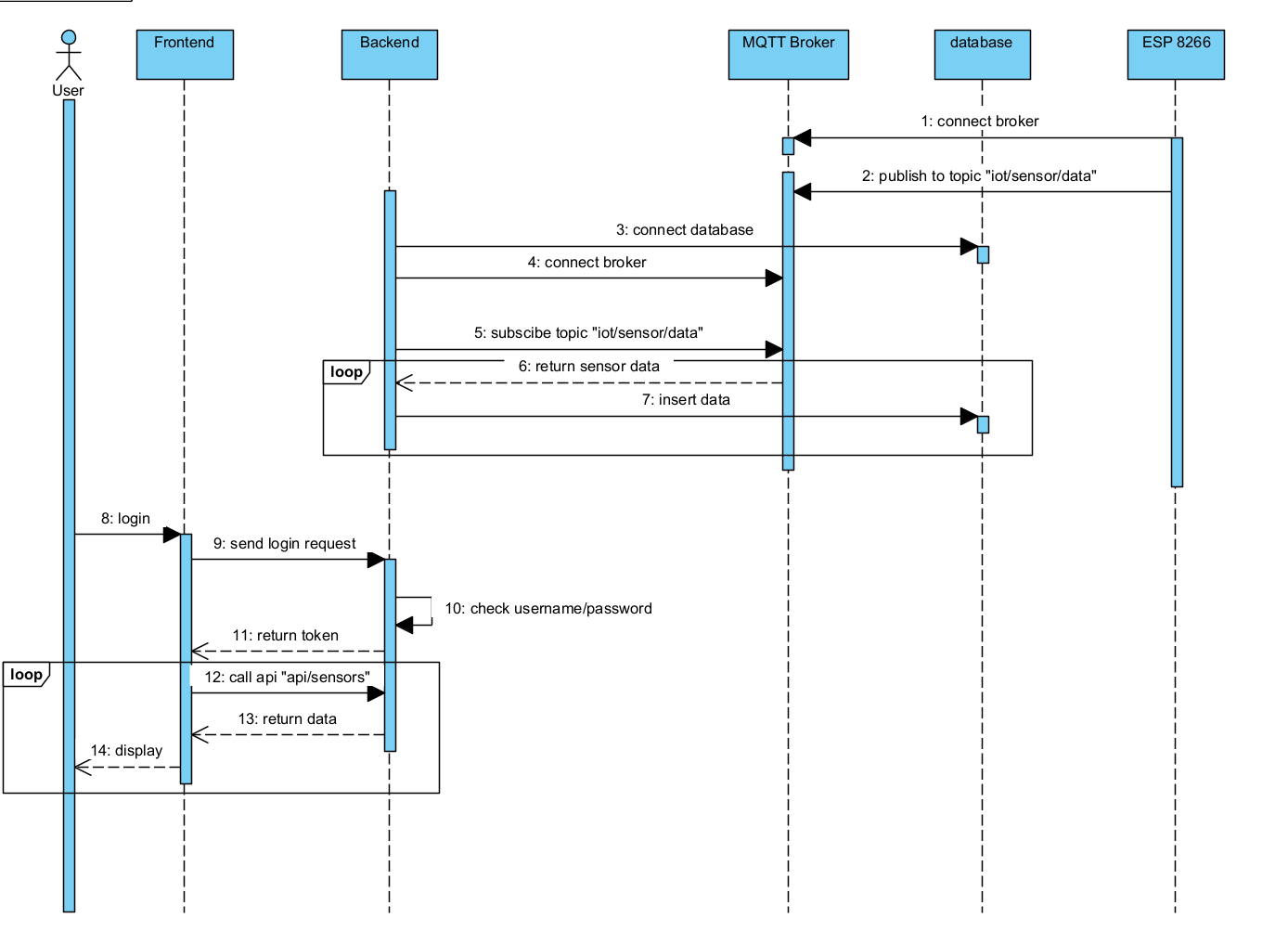
A screenshot of a computer

AI-generated content may be incorrect.

Hình 8 Giao diện trang Device Log

## IV. Sequence diagram

* + 1. **Hiển thị dữ liệu cảm biến**

****

Hình 9 Sequence Diagram cho luồng hiển thị dữ liệu

1. ESP8266 kết nối đến MQTT Broker.
2. ESP8266 gửi dữ liệu cảm biến bằng cách publish lên topic "iot/sensor/data".
3. Backend thiết lập kết nối với cơ sở dữ liệu.
4. Backend kết nối đến MQTT Broker.
5. Backend đăng ký topic "iot/sensor/data" để nhận dữ liệu cảm biến.
6. Backend nhận dữ liệu cảm biến từ Broker.
7. Backend ghi dữ liệu cảm biến vào cơ sở dữ liệu.
8. Người dùng thực hiện đăng nhập.
9. Frontend gửi yêu cầu đăng nhập đến Backend.
10. Backend kiểm tra thông tin đăng nhập (username/password) trong cơ sở dữ liệu.
11. Backend trả về token xác thực cho Frontend.
12. Frontend gọi API "api/sensor/data" để lấy dữ liệu cảm biến.
13. Backend trả về dữ liệu cảm biến.
14. Frontend hiển thị dữ liệu cảm biến cho người dùng.
    * 1. **Điều khiển LED**

**A diagram of a project

AI-generated content may be incorrect.**

Hình 10 Sequence Diagram cho luông điều khiển đèn

1. ESP8266 kết nối với Broker
2. ESP8266 đăng ký topic “iot/led/control” để nhận tin nhắn điều khiển đèn
3. ESP8266 gửi trạng thái đèn vào topic “iot/led/state”
4. Backend kết nối database
5. Backend kết nối Broker
6. Backend đăng ký topic “iot/led/state” để nhận trạng thái đèn
7. Broker gửi trạng thái đèn cho Backend
8. Người dung đăng nhập
9. Frontend gửi yêu cầu đăng nhập cho Backend
10. Backend kiểm tra username/password
11. Backend trả token cho Frontend
12. Frontend gọi api “api/led” để lấy trạng thái đèn
13. Backend trả trạng thái đèn cho Frontend
14. Frontend cập nhật trạng thái đèn
15. Người dung bật đèn
16. Frontend gửi yêu cầu bật đèn cho api “api/led”
17. Backend gửi tin nhắn vào topic “iot/led/control”
18. Broker gửi tin nhắn cho ESP8266
19. ESP8266 thực hiện bật đèn
20. ESP8266 gửi trạng thái đèn mới vào topic “iot/led/state”
21. MQTT chuyển trạng thái đèn mới cho Backend
22. Backend ghi trạng thái đèn mới vào Database
23. Backend chuyển trạng thái đèn mới cho Frontend
24. Frontend cập nhật trạng thái đèn cho người dùng

# CHƯƠNG IV: CÀI ĐẶT HỆ THỐNG

## I. Chuẩn bị phần cứng

### 1. Thiết bị cần thiết

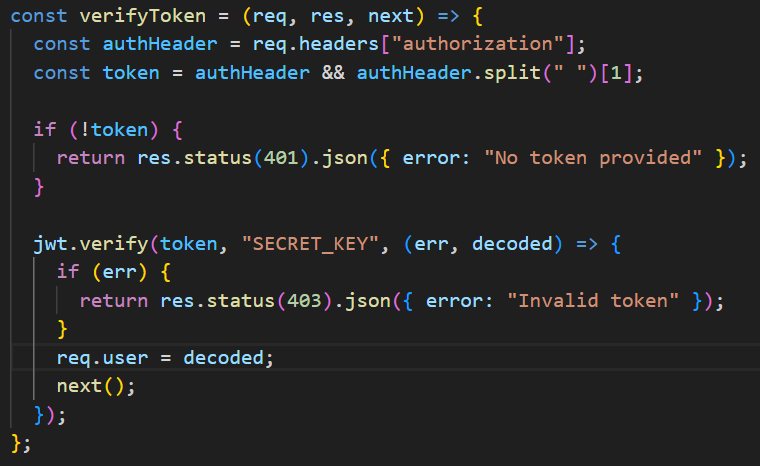
* Breadboard
* ESP8266
* Cảm biến nhiệt độ + độ ẩm DHT11
* Cảm biến ánh sáng quang trở LDR
* Đèn LED x3
* Dây jump đực-cái
* Điện trở x3

### 2. Lắp đặt thiết bị

1. Nối chân GND của ESP8266 với 1 hàng của Breadboard
2. Cắm chân (-) của 3 đèn LED cùng hàng với chân GND
3. Chân (+) của đèn LED được mắc nối tiếp với điện trở và cắm vào các chân D5 D6 D7 của ESP8266

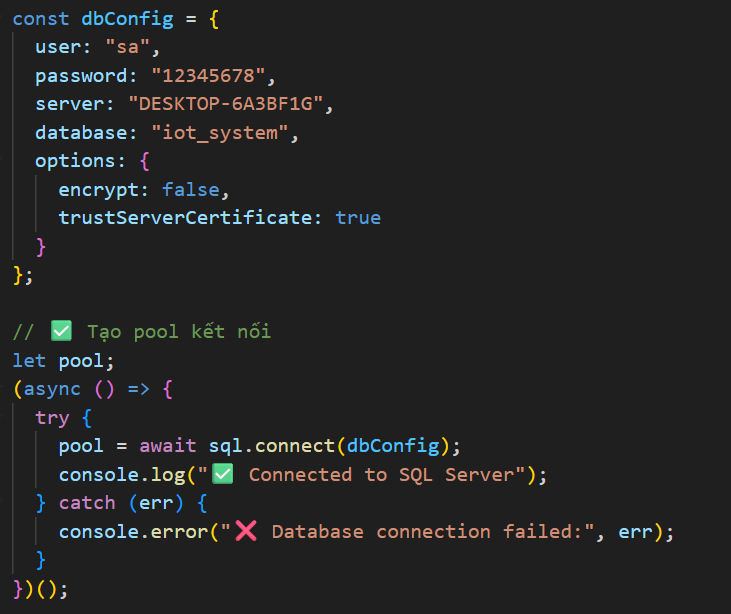
## II. Backend

### 1. Tạo cơ chế xác thực token

****

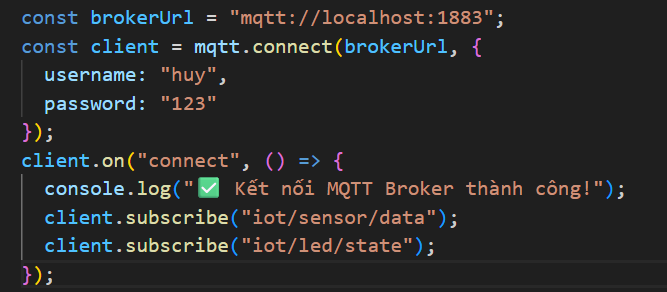
Hình 11 Hàm verifyToken để xác thưc jtoken cho api

### 2. Cấu hình kết nối database

****

Hình 12 Cấu hình để kết nối database

### 3. Cấu hình kết nối MQTT Broker và đăng ký topic



Hình 13 Cấu hình để kết nối MQTT Broker

### 4. Xử lý dữ liệu nhận được từ Broker

****

Hình 14 Code để xử lý dữ liệu cảm biến do Broker gửi về

****

Hình 15 Code để xử lý trạng thái đèn do Broker gửi

### 5. API Login

A screenshot of a computer

AI-generated content may be incorrect.

Hình 16 Mô tả api đăng nhập

### 6. API LED

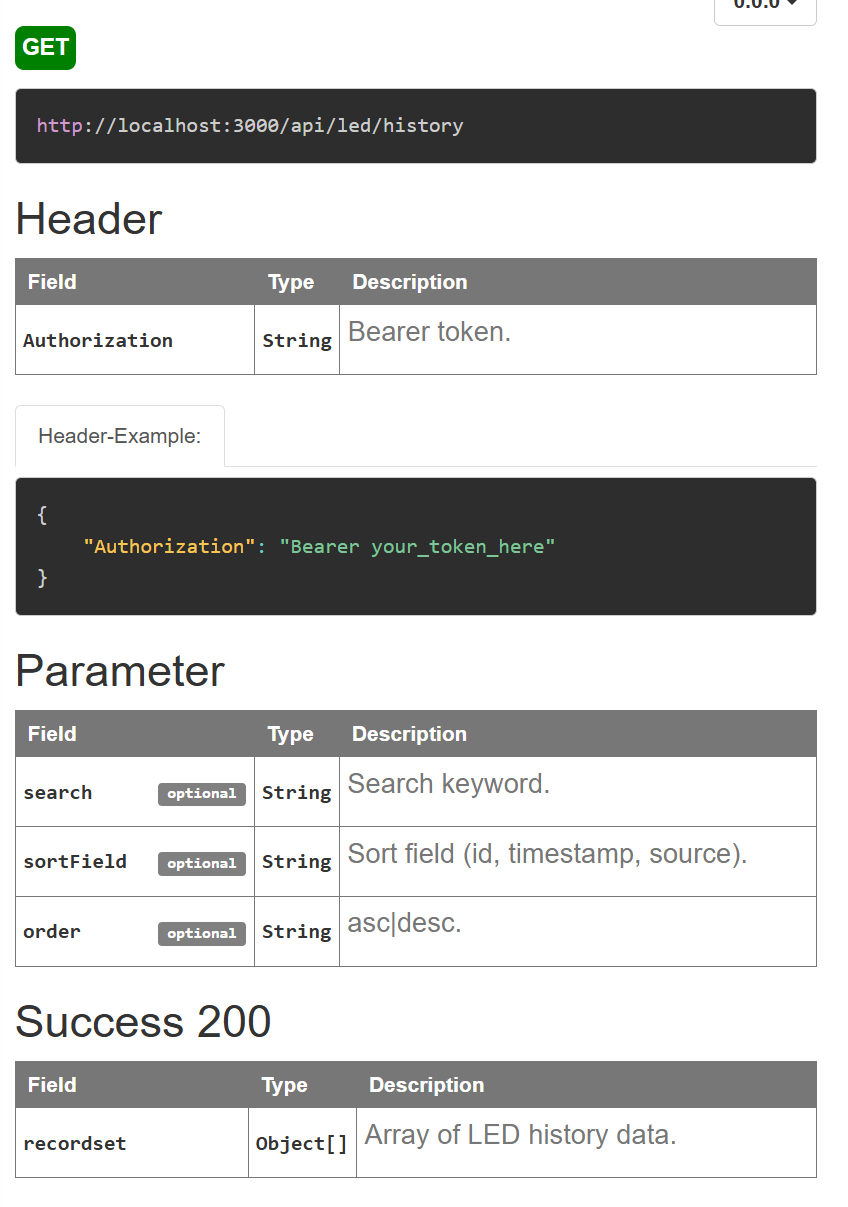
**a. Control LED**

A screenshot of a computer

AI-generated content may be incorrect.

Hình 17 Mô tả api điều khiển LED

**b. Get LED change history**

****

Hình 18 Mô tả api hiển thị lịch sử bật/tắt LED

**c. Get LED state**

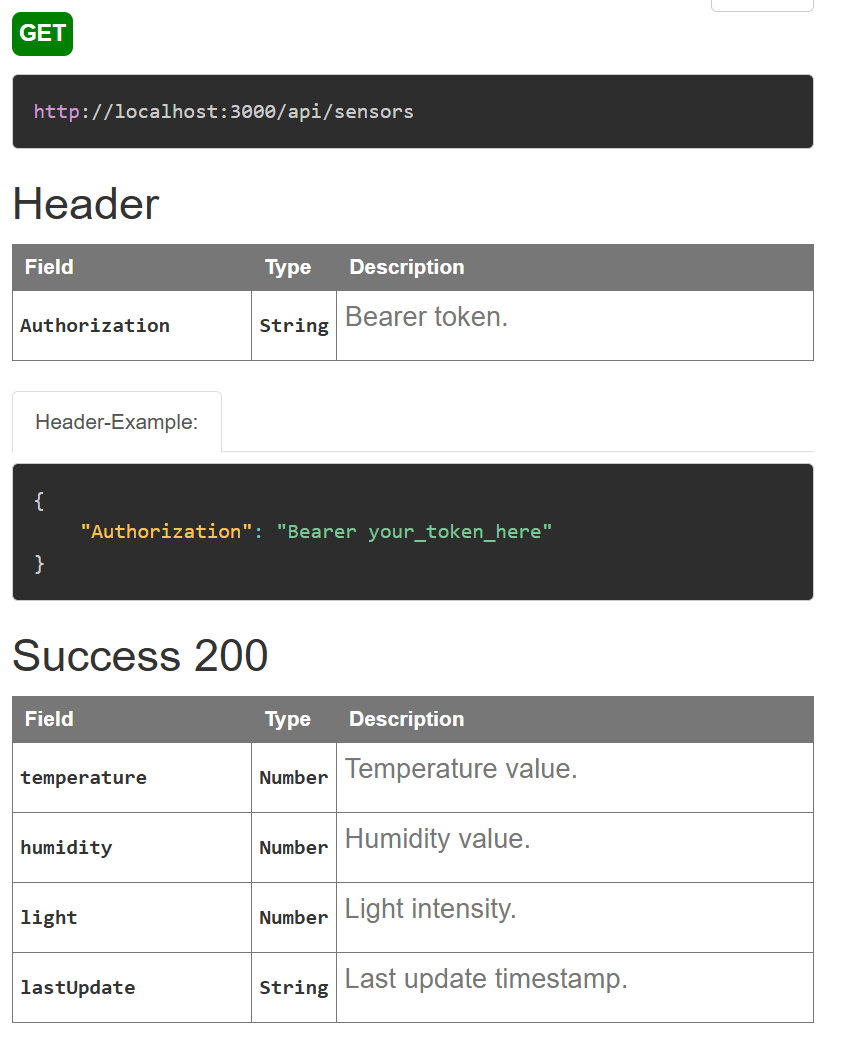
**A screenshot of a computer

AI-generated content may be incorrect.**

Hình 19 Mô tả api lấy trạng thái LED

### 7. API Sensor

**a. Get latest sensor data**

****

Hình 20 Mô tả api lấy dữ liệu cẩm biến

**b. Get sensor history**

**A screenshot of a computer

AI-generated content may be incorrect.**

Hình 21 Mô tả api lấy lịch sử dữ liệu cảm biến