HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG KHOA

AN TOÀN THÔNG TIN

Môn học: IOT và ứng dụng

Báo cáo bài tập lớn

Họ và tên: Trần Trọng Mạnh

Mã sinh viên: B21DCAT127

Nhóm môn học: 05

Giảng viên: Nguyễn Quốc Uy

Hà Nội, 4/2024

MỤC LỤC

I. Phần mở đầu

a. Tính cấp thiết của vấn đề nghiên cứu

Trong bối cảnh hiện nay, công nghệ Internet of Things (IoT) đang phát triển mạnh mẽ và được ứng dụng trong nhiều lĩnh vực khác nhau, từ nông nghiệp, y tế, đến quản lý môi trường và nhà thông minh. Một trong những ứng dụng quan trọng của IoT là giám sát các điều kiện môi trường, bao gồm nhiệt độ, độ ẩm và ánh sáng. Các yếu tố này có ảnh hưởng lớn đến nhiều khía cạnh của đời sống và sản xuất, như kiểm soát chất lượng không khí trong nhà, duy trì điều kiện tối ưu cho cây trồng trong nông nghiệp, hay quản lý năng lượng trong các hệ thống tự động hóa nhà ở.

Việc giám sát thời gian thực các thông số nhiệt độ, độ ẩm và ánh sáng là vô cùng cần thiết, vì nó không chỉ giúp phát hiện sớm các thay đổi bất thường trong môi trường mà còn hỗ trợ đưa ra những quyết định kịp thời nhằm tối ưu hóa quá trình vận hành và tiết kiệm năng lượng. Bên cạnh đó, hệ thống giám sát này cũng có thể cung cấp dữ liệu chính xác cho việc phân tích xu hướng môi trường, giúp dự đoán và quản lý rủi ro trong tương lai.

Sự cần thiết của việc nghiên cứu và triển khai hệ thống IoT đo lường các yếu tố môi trường theo thời gian thực xuất phát từ yêu cầu ngày càng cao về tính tự động hóa và tối ưu hóa trong mọi lĩnh vực. Điều này thúc đẩy sự phát triển của các giải pháp công nghệ tiên tiến, giúp cải thiện chất lượng cuộc sống và tăng cường hiệu quả sản xuất.

b. Mục đích của đối tượng nghiên cứu

Mục đích chính của việc nghiên cứu và triển khai hệ thống IoT đo nhiệt độ, độ ẩm và ánh sáng theo thời gian thực là xây dựng một giải pháp công nghệ hiệu quả để giám sát và quản lý môi trường một cách liên tục, chính xác và tức thời. Cụ thể, hệ thống được thiết kế nhằm:

Giám sát các thông số môi trường trong thời gian thực: Cung cấp các dữ liệu chính xác về nhiệt độ, độ ẩm và ánh sáng tại thời điểm hiện tại, từ đó hỗ trợ người dùng theo dõi và nắm bắt tình trạng môi trường nhanh chóng.

Phân tích và dự đoán xu hướng môi trường: Thu thập và lưu trữ dữ liệu trong một khoảng thời gian dài, giúp phân tích xu hướng biến đổi của các yếu tố môi trường, từ đó đưa ra các dự đoán và giải pháp quản lý hiệu quả.

Tối ưu hóa quá trình vận hành thiết bị: Dựa trên dữ liệu thu thập được, hệ thống có thể tự động điều khiển các thiết bị (như quạt, máy lạnh, hệ thống chiếu sáng) để duy trì các điều kiện môi trường lý tưởng, đồng thời tiết kiệm năng lượng và giảm thiểu chi phí.

Cảnh báo và xử lý kịp thời: Hệ thống giúp phát hiện sớm các thay đổi bất thường trong điều kiện môi trường, từ đó đưa ra các cảnh báo và đề xuất biện pháp xử lý kịp thời nhằm giảm thiểu thiệt hại hoặc rủi ro.

Ứng dụng đa lĩnh vực: Hệ thống không chỉ giới hạn trong quản lý môi trường trong nhà mà còn có thể mở rộng ứng dụng trong các lĩnh vực khác như nông nghiệp, công nghiệp, và quản lý năng lượng, góp phần nâng cao hiệu quả sản xuất và bảo vệ môi trường.

c. Phạm vi nghiên cứu

Phạm vi nghiên cứu của hệ thống IoT đo nhiệt độ, độ ẩm, và ánh sáng theo thời gian thực tập trung vào các khía cạnh sau:

- **Phạm vi về công nghệ:** Nghiên cứu và phát triển hệ thống dựa trên công nghệ Internet of Things (IoT), sử dụng các cảm biến để đo nhiệt độ, độ ẩm, và ánh sáng. Hệ thống sẽ thu thập dữ liệu thông qua các thiết bị cảm biến, sau đó truyền dữ liệu này qua giao thức MQTT hoặc WebSocket đến máy chủ để xử lý và hiển thị trên giao diện web.
- **Phạm vi về đối tượng giám sát:** Hệ thống được triển khai để giám sát các yếu tố môi trường như nhiệt độ, độ ẩm, ánh sáng trong không gian nội thất (như phòng học, văn phòng, nhà ở), hoặc trong các khu vực nông nghiệp nhỏ lẻ.
- **Phạm vi thời gian:** Nghiên cứu tập trung vào giám sát các thông số môi trường theo thời gian thực, với tần suất cập nhật liên tục và lưu trữ dữ liệu dài hạn để phục vụ việc phân tích và dự đoán xu hướng.
- **Phạm vi lưu trữ và xử lý dữ liệu:** Dữ liệu được lưu trữ trong cơ sở dữ liệu MongoDB với khả năng truy xuất và hiển thị theo thời gian cụ thể (giờ, ngày, tháng). Ngoài ra, hệ thống cũng bao gồm khả năng sắp xếp, tìm kiếm, và phân tích dữ liệu đã lưu trữ.
- **Phạm vi ứng dụng:** Hệ thống được thiết kế để dễ dàng mở rộng, ứng dụng trong nhiều lĩnh vực khác nhau như quản lý môi trường, nông nghiệp thông minh, và nhà thông minh, nhưng nghiên cứu chủ yếu tập trung vào không gian nhỏ như hộ gia đình hoặc văn phòng.

d. Phương pháp nghiên cứu

Trong quá trình nghiên cứu và phát triển hệ thống IoT đo nhiệt độ, độ ẩm, ánh sáng theo thời gian thực, các phương pháp sau đây đã được áp dụng:

- **Phương pháp thu thập dữ liệu:** Sử dụng các cảm biến IoT như DHT11, DHT22, và LDR để đo nhiệt độ, độ ẩm và ánh sáng. Các cảm biến này sẽ truyền dữ liệu qua các mô-đun vi điều khiển (như ESP8266) để thu thập và gửi dữ liệu đến hệ thống xử lý.
- **Phương pháp truyền thông:** Dữ liệu được truyền tải qua giao thức MQTT, một giao thức nhẹ và hiệu quả dành cho các thiết bị IoT, đảm bảo việc gửi và nhận dữ liệu liên tục giữa cảm biến và máy chủ. Ngoài ra, WebSocket cũng được sử dụng để tạo ra kết nối hai chiều thời gian thực giữa máy chủ và giao diện người dùng.
- **Phương pháp lưu trữ và xử lý dữ liệu:** Sử dụng cơ sở dữ liệu MongoDB để lưu trữ dữ liệu về nhiệt độ, độ ẩm, ánh sáng và thời gian. Dữ liệu được lưu trữ theo định dạng JSON, giúp truy vấn và phân tích dễ dàng. Hệ thống còn sử dụng các kỹ thuật tối ưu hoá để xử lý và hiển thị dữ liệu theo thời gian thực.
- **Phương pháp thiết kế và phát triển hệ thống:** Áp dụng phương pháp phát triển phần mềm theo mô hình Agile, chia quá trình phát triển thành các giai đoạn nhỏ để liên tục kiểm tra, cải tiến và hoàn thiện hệ thống. Giao diện người dùng được phát triển dựa trên HTML, CSS và JavaScript kết hợp với thư viện biểu đồ để trực quan hoá dữ liệu thời gian thực.
- **Phương pháp thử nghiệm và đánh giá:** Tiến hành các thử nghiệm thực tế trong các điều kiện môi trường khác nhau để kiểm tra tính chính xác và ổn định của hệ thống. Dữ liệu sau khi thu thập sẽ được so sánh với các thiết bị đo lường chuẩn để đánh giá độ chính xác. Đồng thời, đánh giá hiệu suất hệ thống trong việc xử lý và hiển thị dữ liệu thời gian thực.

II. Phần nội dung

1. Giới thiệu IOT

a. IOT là gì

IoT từ là viết tắt của Internet of Things có nghĩa là **Internet vạn vật**. Hệ thống các thiết bị, máy móc hoặc con người cơ học và máy tính kỹ thuật số được kết nối với nhau và khả năng truyền dữ liệu qua mạng mà không cần sự tương tác giữa con người với máy tính.

Ý tưởng về một mạng lưới các thiết bị thông minh đã được tranh luận từ năm 1982, với thiết bị đầu tiên là thiết bị rót nước **Coca-Cola** tùy chỉnh trong Carnegie Mellon. Đầu tiên được kết nối với **internet** và có thể báo cáo tình trạng tồn kho và độ lạnh của nước đóng chai mới và bỏ vào máy. Năm 1999, Kevin Ashton đặt ra thuật ngữ Internet of Things để mô tả một hệ thống trong đó Internet được kết nối với thế giới vật chất thông qua các cảm biến.[CITATION wik22 \l 1033]

b. Cấu trúc một hệ thống IOT

Trong một hệ thống IoT, chúng bao gồm 4 thành phần chính: thiết bị (Things), cổng kết nối (Gateways), hạ tầng mạng (Network and Cloud) và bộ xử lý và phân tích dữ liệu (Service Creation and Solution Layers). Cảm biến sẽ có nhiệm vụ thu nhận các tín hiệu từ môi trường như nhiệt độ, áp suất, ánh sáng,… và chuyển thành dạng dữ liệu trên môi trường internet. Sau đó, tín hiệu được xử lý. Quản lý và thực hiện các thay đổi theo quyết định của người tiêu dùng. Ngày nay chúng thường được áp dụng thông qua ứng dụng trên điện thoại hoặc máy tính.

c. Ưu nhược điểm IOT

- Ưu điểm :
    - Truy cập thông tin từ mọi nơi mọi lúc trên mọi thiết bị.
    - Cải thiện giao tiếp giữa các thiết bị được kết nối.
    - Truyền dữ liệu qua Internet thiết kiệm thời gian và tiền bạc.
- Nhược điểm :

Khi nhiều thiết bị được kết nối hơn và nhiều thông tin được trao đổi giữa các thiết bị, thì khả năng tin tặc đánh cắp thông tin nhạy cảm cũng tăng lên thiết bị trở thành một thách thức.

- Trong trường hợp hệ thống bị lỗi, tất cả thiết bị được kết nối có khả năng bị lỗi.
- Do không có tiêu chuẩn tương thích quốc tế cho IoT nên các thiết bị từ các

nhà sản xuất khác nhau rất khó giao tiếp với nhau.

d. Ứng dụng IOT

Internet of Things (IoT) đang mở ra nhiều cơ hội mới trong việc giám sát, quản lý và điều khiển các hệ thống từ xa thông qua việc kết nối các thiết bị với nhau qua internet. Đặc biệt, trong lĩnh vực giám sát môi trường, IoT có thể được áp dụng vào nhiều trường hợp sử dụng cụ thể nhằm nâng cao hiệu quả quản lý và tiết kiệm tài nguyên. Các ứng dụng của IoT trong hệ thống đo nhiệt độ, độ ẩm, ánh sáng theo thời gian thực bao gồm:

- **Nhà thông minh (Smart Home):** Hệ thống IoT có thể được triển khai để theo dõi nhiệt độ, độ ẩm và ánh sáng trong nhà, từ đó tự động điều chỉnh các thiết bị như máy lạnh, quạt, đèn chiếu sáng, đảm bảo môi trường sống thoải mái và tiết kiệm năng lượng. Người dùng có thể giám sát và điều khiển từ xa qua điện thoại hoặc máy tính.
- **Nông nghiệp thông minh (Smart Agriculture):** IoT có vai trò quan trọng trong việc theo dõi điều kiện môi trường cho cây trồng. Hệ thống có thể giám sát nhiệt độ và độ ẩm đất, ánh sáng trong vườn, từ đó cung cấp dữ liệu để điều chỉnh hệ thống tưới tiêu hoặc ánh sáng nhân tạo. Điều này giúp tối ưu hóa điều kiện canh tác, cải thiện năng suất cây trồng và giảm thiểu lãng phí tài nguyên nước và năng lượng.
- **Giám sát môi trường (Environmental Monitoring):** Trong các khu vực công cộng hoặc không gian rộng như nhà kính, nhà máy hoặc công viên, hệ thống IoT có thể được dùng để giám sát và phân tích các yếu tố môi trường nhằm duy trì điều kiện lý tưởng hoặc cảnh báo khi có các biến động bất thường. Điều này đặc biệt quan trọng trong việc bảo vệ sức khỏe và an toàn của con người trong các môi trường khắc nghiệt.
- **Quản lý năng lượng (Energy Management):** IoT giúp quản lý các hệ thống chiếu sáng, quạt thông gió và máy lạnh dựa trên dữ liệu nhiệt độ, độ ẩm và ánh sáng thực tế, từ đó tối ưu hóa việc sử dụng năng lượng, giảm chi phí vận hành và bảo vệ môi trường. Hệ thống có thể tự động bật/tắt thiết bị dựa trên điều kiện môi trường hiện tại, giúp tiết kiệm điện năng một cách hiệu quả.
- **Ứng dụng trong công nghiệp (Industrial IoT):** Trong các nhà máy, IoT có thể giúp giám sát các điều kiện làm việc của thiết bị và môi trường xung quanh, đảm bảo nhiệt độ, độ ẩm và ánh sáng được duy trì ở mức tối ưu để bảo vệ thiết bị và nhân công. Ngoài ra, việc thu thập và phân tích dữ liệu thời gian thực cũng giúp doanh nghiệp đưa ra các quyết định kịp thời và hiệu quả.

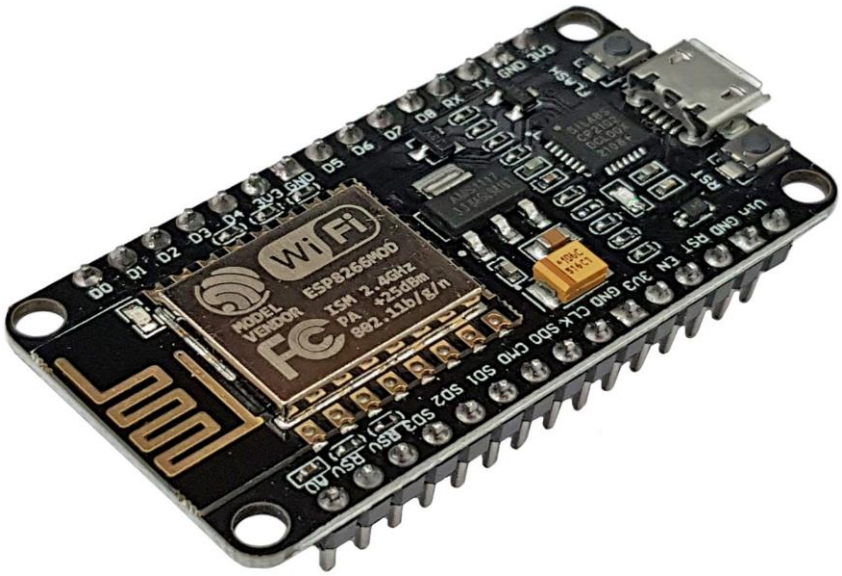
2. Giới thiệu phần cứng

a. Node MCU Esp8266

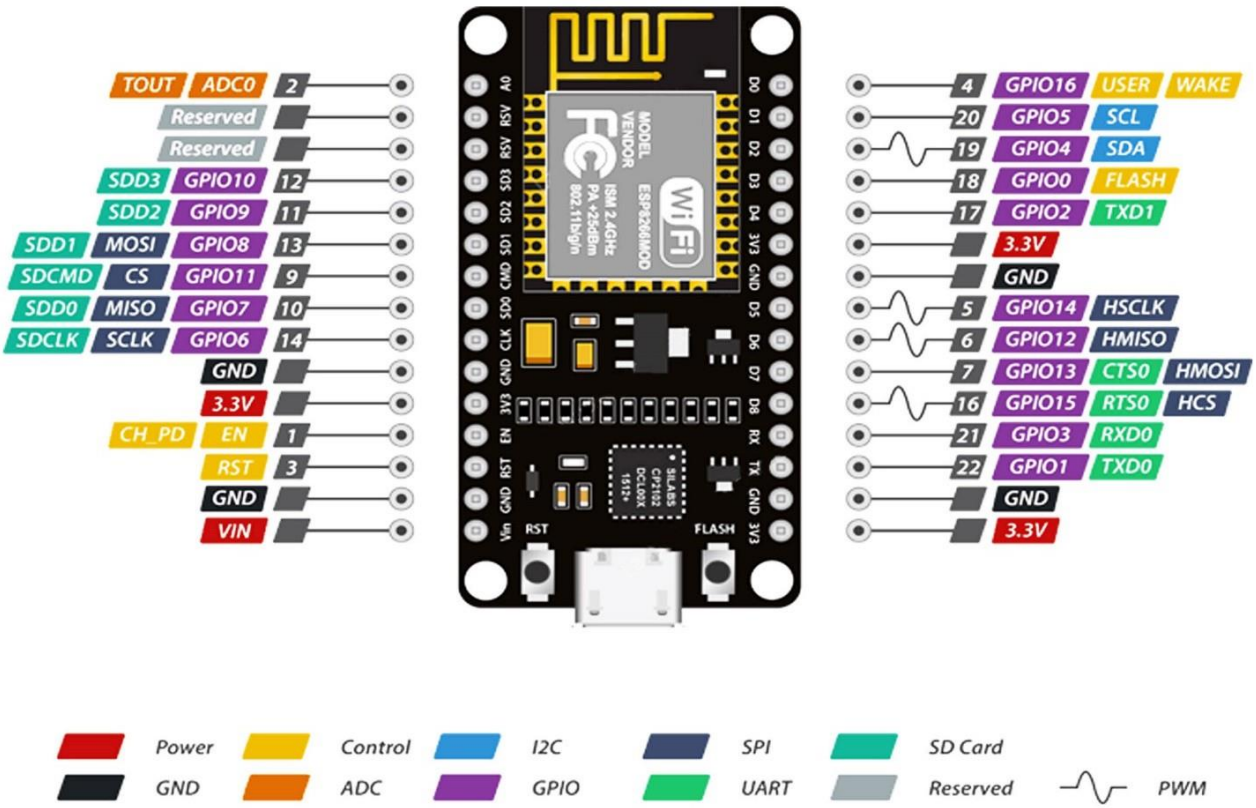
NodeMCU ESP8266 là một nền tảng mã nguồn mở dựa trên Wi-Fi, được sử dụng rộng rãi trong các dự án IoT nhờ vào tính năng mạnh mẽ và giá thành phải chăng. Được tích hợp với module ESP8266, NodeMCU cung cấp khả năng kết nối mạng Wi-Fi và thực hiện các tác vụ mạng một cách dễ dàng. Một số đặc điểm nổi bật của NodeMCU ESP8266 bao gồm:

- **Kết nối Wi-Fi:** ESP8266 được tích hợp module Wi-Fi cho phép nó kết nối với mạng không dây, giúp truyền và nhận dữ liệu qua internet. Điều này cực kỳ quan trọng trong các ứng dụng IoT, nơi việc kết nối với các thiết bị từ xa qua mạng là yếu tố cốt lõi.

- **Bộ vi điều khiển 32-bit:** NodeMCU ESP8266 sử dụng vi điều khiển 32-bit với tốc độ xử lý nhanh, phù hợp với các ứng dụng yêu cầu xử lý dữ liệu thời gian thực.
- **Hỗ trợ ngôn ngữ lập trình Lua và Arduino:** NodeMCU có thể lập trình bằng ngôn ngữ Lua, nhưng thường được sử dụng rộng rãi với Arduino IDE, giúp người dùng dễ dàng lập trình và phát triển các dự án.
- **I/O đa dạng:** NodeMCU có nhiều cổng I/O cho phép kết nối với các cảm biến và thiết bị ngoại vi khác nhau như cảm biến nhiệt độ, độ ẩm, ánh sáng và các thiết bị điều khiển như đèn LED, quạt, máy điều hòa.
- **Nguồn cung cấp linh hoạt:** Module có thể hoạt động với nguồn cung cấp từ 3.3V đến 5V, dễ dàng kết nối với các nguồn năng lượng khác nhau, giúp mở rộng khả năng ứng dụng.
- **Thư viện phong phú:** NodeMCU có một cộng đồng phát triển mạnh mẽ với nhiều thư viện và tài liệu hỗ trợ, giúp đơn giản hóa việc tích hợp các cảm biến và thiết bị ngoại vi.



Hình ảnh Module ESP8266



Sơ đồ chân Module thu phát Wifi ESP8266 NodeMCU

- Thông số kỹ thuật :

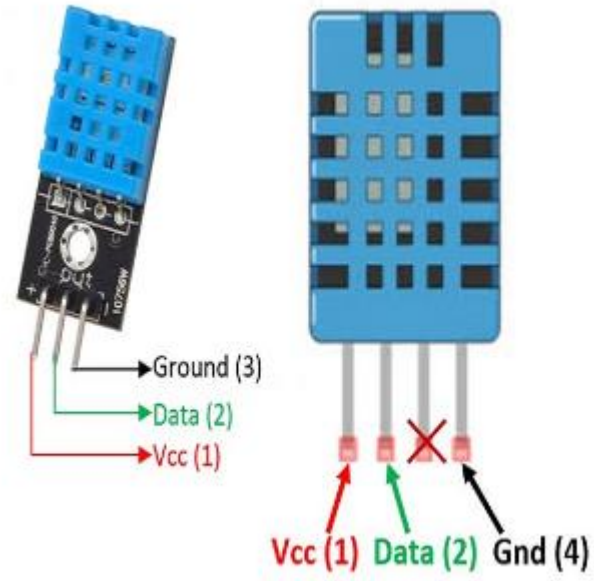
  - Tương thích các chuẩn wifi : 802.11 b/g/n
  - Hỗ trợ: Wi-Fi Direct (P2P), soft-AP
  - Tích hợp TCP/IP protocol stack
  - Tích hợp TR switch, balun, LNA, power amplifier and matching network
  - Tích hợp bộ nhân tần số, ổn áp, DCXO and power management units
  - +25.dBm output power in 802.11b mode
  - Power down leakage current of <10uA
  - Integrated low power 32-bit CPU could be used as application processor
  - SDIO 1.1/2.0, SPI, UART
  - STBC, 1×1 MIMO, 2×1 MIMO
  - A-MPDU & A-MSDU aggregation & 0.4ms guard interval

- Wake up and transmit packets in < 2ms
- Dòng tiêu thụ ở Standby Mode < 1.0mW (DTIM3)

b. Module cảm biến nhiệt độ độ ẩm

Cảm biến độ ẩm nhiệt độ DHT11 ra chân được tích hợp sẵn điện trở 5,1k giúp người dùng dễ dàng kết nối và sử dụng hơn so với cảm biến DHT11 chưa ra chân, module lấy dữ liệu thông qua giao tiếp 1 wire (giao tiếp 1 dây). Bộ tiền xử lý tín hiệu tích hợp

Trong cảm biến giúp bạn có được dữ liệu chính xác mà không cần phải qua bất kỳ tính toán nào. Module được thiết kế hoạt động ở mức điện áp 5VDC. [CITATION hsh22 \l 1033]



Hình ảnh module DHT11

Thông số kỹ thuật DHT11
- Điện áp hoạt động: 3V - 5V DC
- Dòng điện tiêu thụ: 2.5mA
- Phạm vi cảm biến độ ẩm: 20% - 90% RH, sai số ±5%RH
- Phạm vi cảm biến nhiệt độ: 0°C ~ 50°C, sai số ±2°C
- Tần số lấy mẫu tối đa: 1Hz (1 giây 1 lần)
- Kích thước: 23 * 12 * 5 mm

c. Module cảm biến ánh sáng

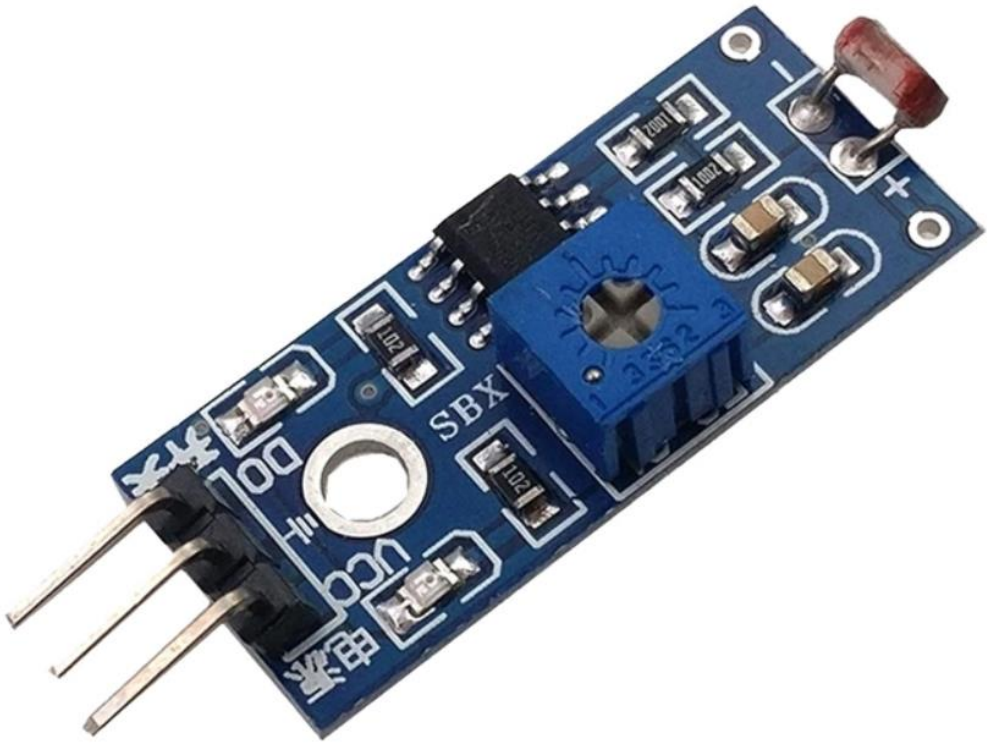
Cảm biến ánh sáng 3 Pin, Module cảm biến cảm quang

Cảm biến cường độ ánh sáng quang trở rất nhạy cảm với cường độ ánh sáng môi trường thường được sử dụng để phát hiện độ sáng môi trường xung quanh và cường độ ánh sáng. Khi cường độ ánh sáng môi trường xung quanh bên ngoài vượt quá một ngưỡng quy định, ngõ ra

của module D0 là mức logic thấp. Ngoài ra còn có ngõ ra Analog ở chân A0 để xử lí mức độ ánh sáng.

Cảm biến cường độ ánh sáng phát hiện cường độ ánh sáng, sử dụng bộ cảm biến photoresistor loại nhạy cảm, cho tín hiệu ổn định, rõ ràng và chính xác hơn so với quang trở độ nhạy có thể tùy chỉnh. Thiết kế đơn giản nhưng hiệu quả và độ tin cậy cao, độ nhiễu thấp do được thiết kế mạch lọc tín hiệu trước khi so sánh với ngưỡng.

Thân thiện với người dùng hơn khi hỗ trợ cả 2 dạng tín hiệu ngõ ra dạng số (tín hiệu 0 1) và dạng analog.



Hình ảnh cảm biến ánh sáng 3 Pin

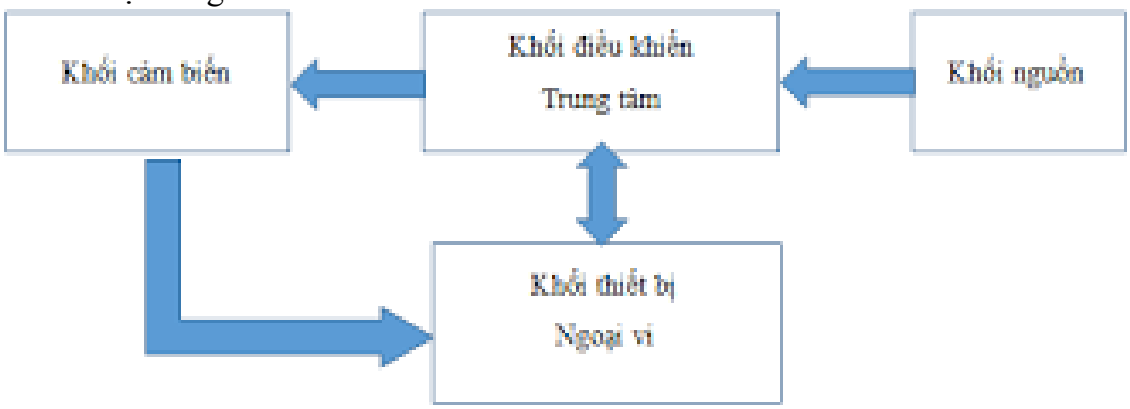
THÔNG SỐ KỸ THUẬT

- Điện áp hoạt động 3.3 – 5 V
- Kết nối 4 chân với 2 chân cấp nguồn (VCC và GND) và 2 chân tín hiệu ngõ ra (AO và DO).
- Hỗ trợ cả 2 dạng tín hiệu ra Analog và TTL. Ngõ ra Analog 0 – 5V tỷ lệ thuận với cường độ ánh sáng, ngõ TTL tích cực mức thấp.
- Độ nhạy cao với ánh sáng được tùy chỉnh bằng biến trở .

- Kích thước 32 x 14mm

3. Thiết kế hệ thống

a. Sơ đồ hệ thống



Khối điều khiển trung tâm : Sử dụng Node MCU esp8266 để điều khiển các thiết bị khác trong hệ thống :

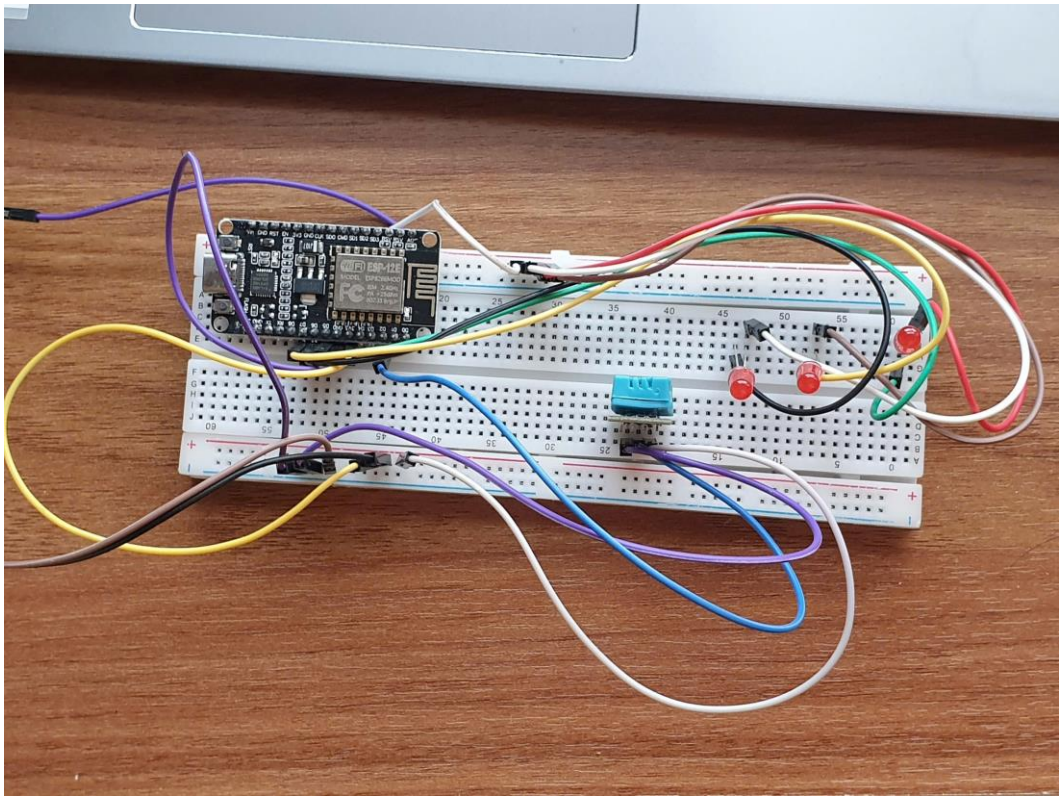
- Điều khiển đọc dữ liệu từ cảm biến
- Truyền dữ liệu lên web qua thiết bị trung gian

Khối cảm biến : bao gồm các cảm biến nhiệt độ, độ ẩm môi trường, độ ẩm đất, dùng để thu thập dữ liệu về nhiệt độ, độ ẩm, độ sáng xung quanh đối tượng canh tác, từ đó đưa tín hiệu về khối điều khiển trung tâm, so sánh với giá trị đặt trước, rồi sau đó khối điều khiển trung tâm sẽ sử lý để phù hợp với các giá trị đặt trước.

Khối nguồn : Cung cấp điện cho toàn hệ thống. Cần tính toán hợp lí để khối nguồn có thể cung cấp để dòng và áp để mạch hoạt động tốt.

Khối thiết bị ngoại vi : nhận tín hiệu từ khối điều khiển để hoạt động (hoạt động dựa trên tín hiệu cảm biến, được lập trình bởi khối điều khiển).

b. Thiết kế mạch



Hình ảnh bảng mạch ESP8266

4. Ngôn ngữ lập trình
   a. Phần mềm lập trình Arduino IDE 2.3.3

Arduino IDE (Integrated Development Environment) là môi trường phát triển tích hợp dành cho việc lập trình và nạp mã nguồn vào các vi điều khiển như Arduino, ESP8266, ESP32 và nhiều loại bo mạch khác. Phiên bản **Arduino IDE 2.3.3** mang đến nhiều cải tiến về giao diện và tính năng nhằm nâng cao trải nghiệm lập trình, đặc biệt trong các dự án IoT. Một số đặc điểm nổi bật của phiên bản này bao gồm:

- **Giao diện đồ họa hiện đại:** So với các phiên bản trước, Arduino IDE 2.3.3 có giao diện đồ họa hiện đại hơn, dễ sử dụng và trực quan hơn, với các thanh công cụ được bố trí hợp lý, giúp người dùng dễ dàng truy cập các chức năng cần thiết.
- **Trình soạn thảo mã thông minh:** IDE cung cấp tính năng gợi ý mã (code auto-completion) giúp tăng tốc độ viết code và giảm thiểu lỗi cú pháp. Ngoài ra, tính năng này còn giúp lập trình viên dễ dàng tìm hiểu các lệnh và hàm hỗ trợ.
- **Hỗ trợ nhiều loại vi điều khiển:** Arduino IDE 2.3.3 hỗ trợ không chỉ các bo mạch Arduino mà còn các bo mạch khác như ESP8266, ESP32, STM32, và các nền tảng IoT khác. Người dùng có thể cài đặt thêm các board hỗ trợ thông qua Arduino Board Manager.
- **Debugging tích hợp:** Một tính năng mới được tích hợp trong phiên bản 2.x là khả năng debug code trực tiếp, cho phép lập trình viên theo dõi các biến, dòng lệnh và trạng thái của bo mạch trong thời gian thực, giúp việc tìm và sửa lỗi nhanh chóng và hiệu quả hơn.

- **Trình quản lý thư viện (Library Manager):** Arduino IDE 2.3.3 có tích hợp trình quản lý thư viện, cho phép người dùng dễ dàng cài đặt và cập nhật các thư viện từ cộng đồng Arduino. Thư viện phong phú này giúp hỗ trợ nhiều cảm biến, module, và các thiết bị ngoại vi, giúp quá trình lập trình nhanh chóng hơn.
- **Khả năng tùy chỉnh mạnh mẽ:** IDE cho phép người dùng tùy chỉnh giao diện, màu sắc mã nguồn, và các thiết lập bàn phím theo sở thích cá nhân, tạo ra một môi trường làm việc thuận tiện hơn.
- **Tích hợp công cụ nạp mã:** Người dùng có thể dễ dàng nạp mã nguồn trực tiếp lên bo mạch từ IDE qua kết nối USB, và theo dõi quá trình nạp mã với các thông báo chi tiết về lỗi hoặc trạng thái thành công.
- **Hỗ trợ đa nền tảng:** Arduino IDE 2.3.3 có thể chạy trên nhiều hệ điều hành phổ biến như Windows, macOS và Linux, tạo điều kiện thuận lợi cho nhiều người dùng.

Code hệ thống:

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <DHT.h>

// Định nghĩa thông tin WiFi
// const char* ssid = "602";
// const char* password = "0985150059";
const char* ssid = "Mạng";
const char* password = "12345678";
// const char* ssid = "Duy5";
// const char* password = "88888888";
// const char* ssid = "PTIT_WIFI";
// const char* password = "88888888";

// Định nghĩa thông tin MQTT Broker
// const char* mqtt_server = "192.168.1.4";  // Địa chỉ IP của MQTT Broker
const char* mqtt_server = "192.168.43.88";  // Địa chỉ IP của MQTT Broker
const int mqtt_port = 1883;                 // Cổng của MQTT Broker
const char* mqtt_user = "ttm";    // Username của MQTT Broker
const char* mqtt_pass = "ttm";    // Password của MQTT Broker

// Định nghĩa các chân cho cảm biến và thiết bị
#define DHTPIN D2          // Chân cảm biến DHT11 nối với D2
#define DHTTYPE DHT11      // Loại cảm biến DHT11
#define LDRPIN A0          // Chân cảm biến ánh sáng nối với A0
#define LED1 D5            // Chân LED1 nối với D5
#define FANPIN D6          // Chân điều khiển quạt nối với D6
#define ACIN D7            // Chân điều khiển điều hòa nối với D7

DHT dht(DHTPIN, DHTTYPE);  // Khởi tạo cảm biến DHT11
WiFiClient espClient;
PubSubClient mqttClient(espClient);
```

```
unsigned long lastMsg = 0;  // Biến lưu thời gian gửi dữ liệu cuối cùng
int ledState1 = HIGH;
int fanState = HIGH;
int acState = HIGH;

// Khai báo biến để lưu trạng thái trước đó của thiết bị
int prevLedState = -1;
int prevFanState = -1;
int prevAcState = -1;

// Khai báo biến để quản lý thời gian không sử dụng delay
unsigned long previousMillis = 0;
const long interval = 2000;  // 2 giây

// Kết nối WiFi
void setup_wifi() {
  delay(10);
  Serial.begin(115200);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
}

// Callback nhận dữ liệu từ MQTT Broker
void callback(char* topic, byte* payload, unsigned int length) {
  String message;
  for (int i = 0; i < length; i++) {
    message += (char)payload[i];
  }
  Serial.print("Message received on topic: ");
  Serial.print(topic);
  Serial.print(". Message: ");
  Serial.println(message);

  // Kiểm tra và điều khiển thiết bị theo topic và payload
  if (String(topic) == "home/led1") {
```

```
    if (message == "ON") {
      ledState1 = HIGH;
    } else if (message == "OFF") {
      ledState1 = LOW;
    }
    digitalWrite(LED1, ledState1);
  } else if (String(topic) == "home/fan") {
    if (message == "ON") {
      fanState = HIGH;
    } else if (message == "OFF") {
      fanState = LOW;
    }
    digitalWrite(FANPIN, fanState);
  } else if (String(topic) == "home/ac") {
    if (message == "ON") {
      acState = HIGH;
    } else if (message == "OFF") {
      acState = LOW;
    }
    digitalWrite(ACIN, acState);
  } else if (String(topic) == "home/all") {
    if (message == "ON") {
      ledState1 = HIGH;
      fanState = HIGH;
      acState = HIGH;
    } else if (message == "OFF") {
      ledState1 = LOW;
      acState = LOW;
      fanState = LOW;
    }
    digitalWrite(LED1, ledState1);
    digitalWrite(FANPIN, fanState);
    digitalWrite(ACIN, acState);
  }
}

// Kết nối đến MQTT Broker và đăng ký các topic
void reconnect() {
  while (!mqttClient.connected()) {
    Serial.print("Attempting MQTT connection...");

    // Thêm username và password khi kết nối đến MQTT Broker
    if (mqttClient.connect("ESP8266Client", mqtt_user, mqtt_pass)) {
      Serial.println("connected");
      // Đăng ký các topic điều khiển
      mqttClient.subscribe("home/led1");
      mqttClient.subscribe("home/fan");
```

```
      mqttClient.subscribe("home/ac");
      mqttClient.subscribe("home/all");
    } else {
      Serial.print("failed, rc=");
      Serial.print(mqttClient.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}

void setup() {
  setup_wifi();
  Serial.println("NodeMCU connected!");  // Thông điệp xác nhận.

  mqttClient.setServer(mqtt_server, mqtt_port);
  mqttClient.setCallback(callback);

  dht.begin();  // Khởi động cảm biến DHT11

  // Cài đặt các chân output cho thiết bị
  pinMode(LED1, OUTPUT);
  pinMode(FANPIN, OUTPUT);
  pinMode(ACIN, OUTPUT);

  // Tắt các thiết bị ban đầu
  digitalWrite(LED1, LOW);
  digitalWrite(FANPIN, LOW);
  digitalWrite(ACIN, LOW);
}

void loop() {
  unsigned long currentMillis = millis();

  // Kiểm tra kết nối MQTT
  if (!mqttClient.connected()) {
    reconnect();
  }
  mqttClient.loop();

  // Đọc trạng thái hiện tại của led1, fan, ac
  int currentLedState = digitalRead(LED1);
  int currentFanState = digitalRead(FANPIN);
  int currentAcState = digitalRead(ACIN);

  // Kiểm tra xem có thay đổi trạng thái nào không
```

```
  if (currentLedState != prevLedState || currentFanState != prevFanState ||
currentAcState != prevAcState) {
    // Nếu có thay đổi, gửi dữ liệu MQTT

    // Chuyển đổi trạng thái sang ON/OFF
    String ledStatus = (currentLedState == HIGH) ? "ON" : "OFF";
    String fanStatus = (currentFanState == HIGH) ? "ON" : "OFF";
    String acStatus = (currentAcState == HIGH) ? "ON" : "OFF";

    // Đóng gói JSON và gửi trạng thái thiết bị qua MQTT
    char statusData[150];
    snprintf(statusData, sizeof(statusData),
      "{\"led1\": \"%s\", \"fan\": \"%s\", \"ac\": \"%s\"}",
      ledStatus.c_str(), fanStatus.c_str(), acStatus.c_str());
    mqttClient.publish("home/status", statusData);  // Gửi lên topic "home/status"

    // Cập nhật trạng thái trước đó thành trạng thái hiện tại
    prevLedState = currentLedState;
    prevFanState = currentFanState;
    prevAcState = currentAcState;

    // Debug thông tin trạng thái đã gửi
    Serial.print("LED1: ");
    Serial.print(ledStatus);
    Serial.print(" | Fan: ");
    Serial.print(fanStatus);
    Serial.print(" | AC: ");
    Serial.println(acStatus);
  }

  // Kiểm tra thời gian để gửi dữ liệu cảm biến
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;

    // Đọc dữ liệu từ cảm biến DHT11
    float temperature = dht.readTemperature();
    float humidity = dht.readHumidity();
    int lightLevel = analogRead(LDRPIN);  // Đọc giá trị ánh sáng từ LDR

    // Kiểm tra lỗi cảm biến
    if (isnan(temperature) || isnan(humidity)) {
      Serial.println("Failed to read from DHT sensor!");
      // Không sử dụng 'return' để tránh thoát khỏi loop
    } else {
      // Gửi dữ liệu nhiệt độ, độ ẩm và ánh sáng qua MQTT (topic cũ)
      char sensorData[200];
      snprintf(sensorData, sizeof(sensorData),
```

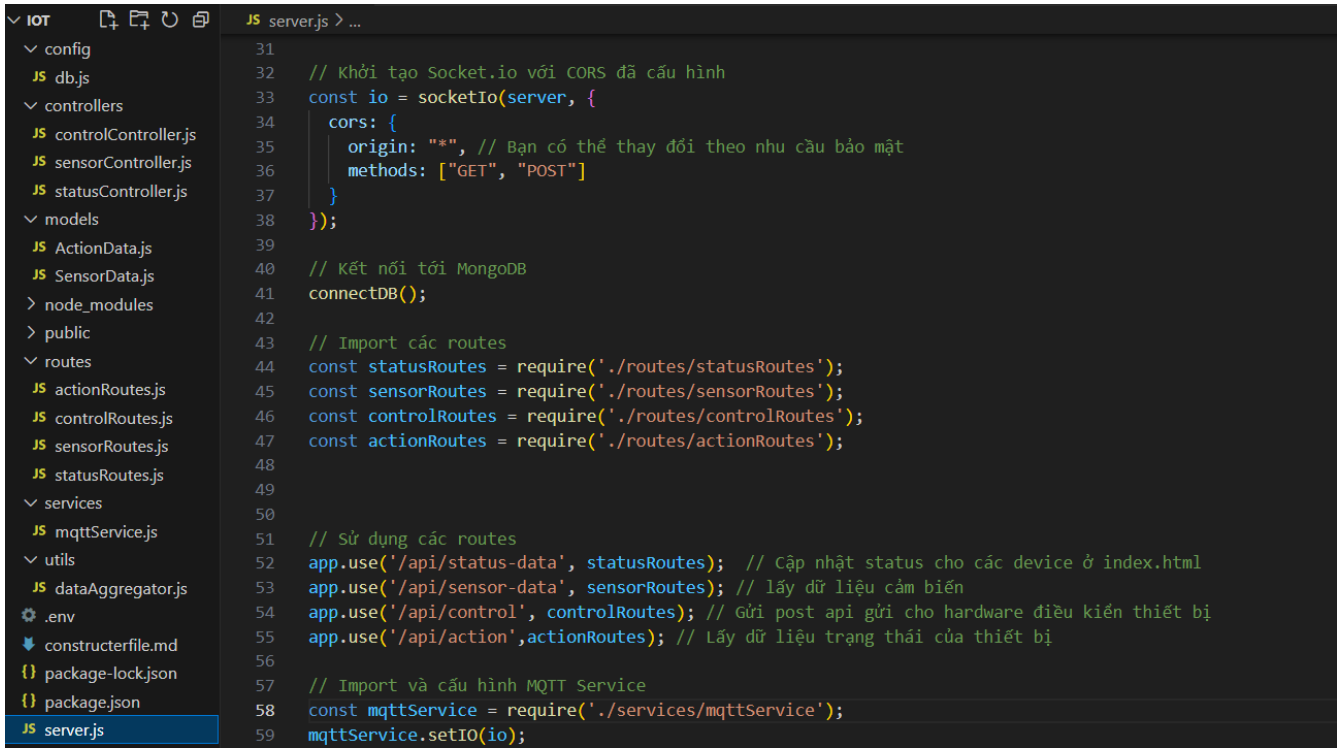
```
        "{\"temperature\": %.2f, \"humidity\": %.2f, \"lighting\": %d}",
        temperature, humidity, (1030-lightLevel));
      mqttClient.publish("home/sensor", sensorData);  // Cập nhật với topic mong muốn
của bạn

      // Debugging
      Serial.print("Temperature: ");
      Serial.print(temperature);
      Serial.print(" °C, Humidity: ");
      Serial.print(humidity);
      Serial.print(" %, Light Level: ");
      Serial.println(lightLevel);
    }
  }

  // Không sử dụng delay để tránh chặn loop
}
```

b. Backend project

Công nghệ: NodeJS

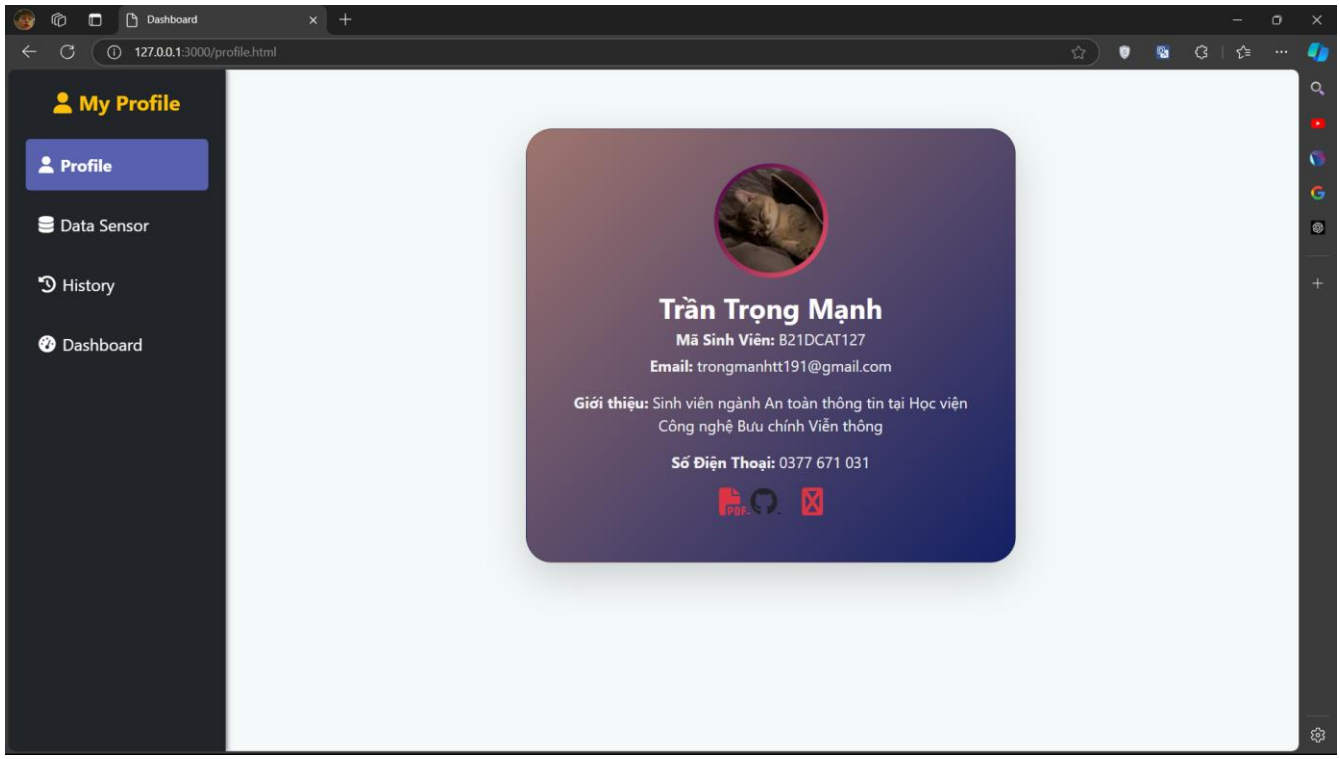
Database: MongoDB

Cấu trúc thư mục:

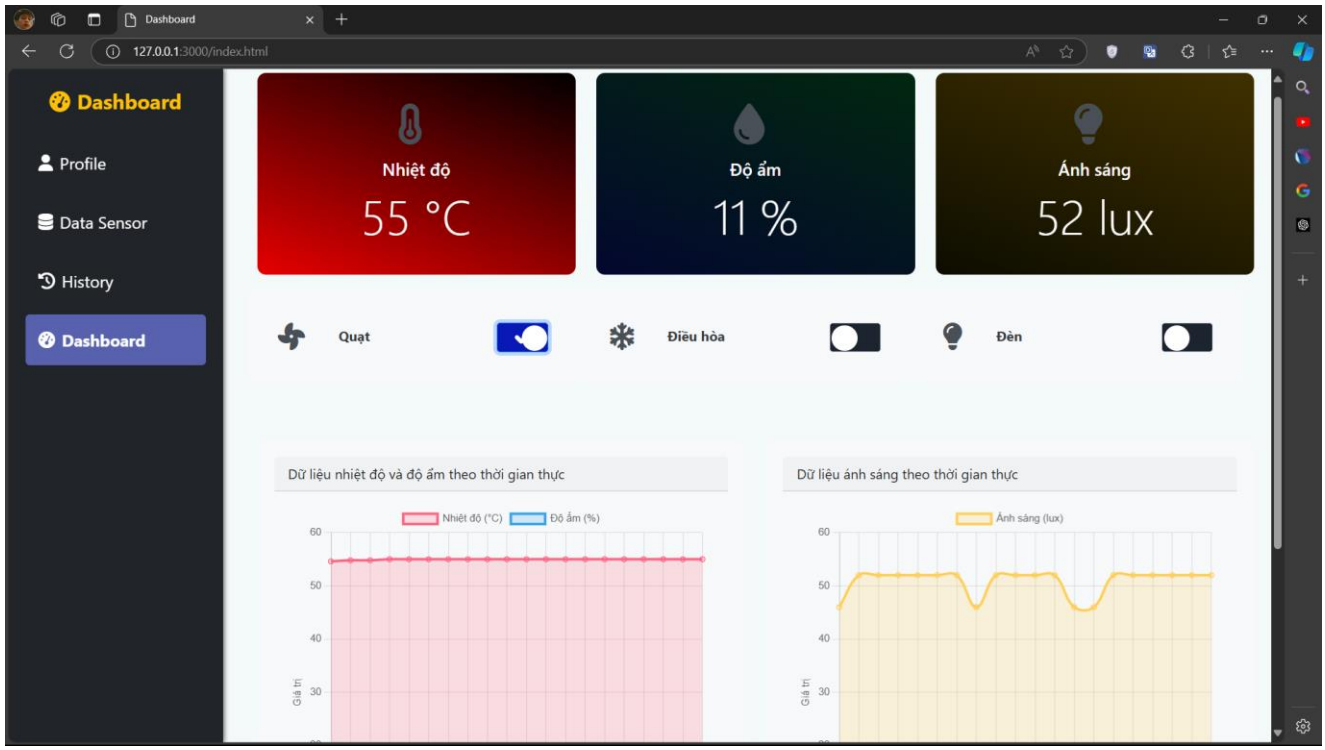


Hình ảnh backend NodeJS

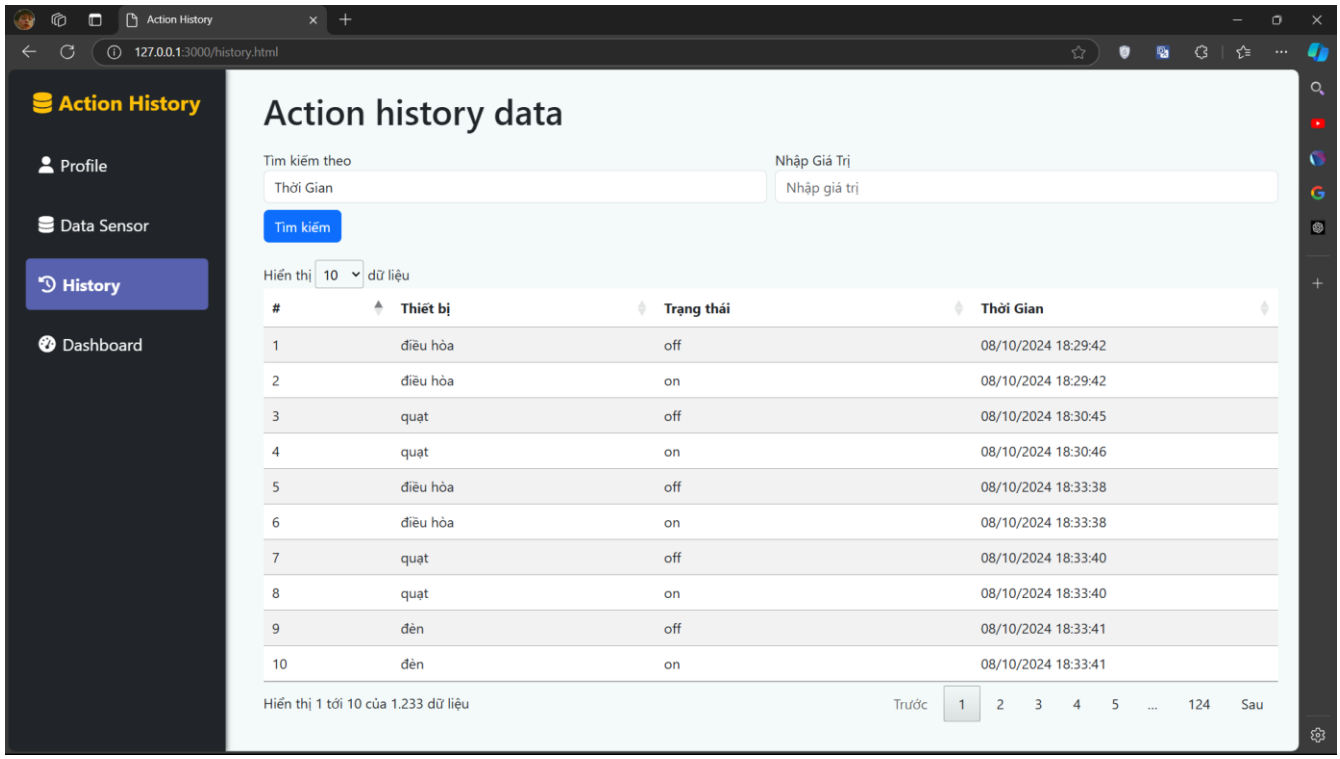
c. Font-end project



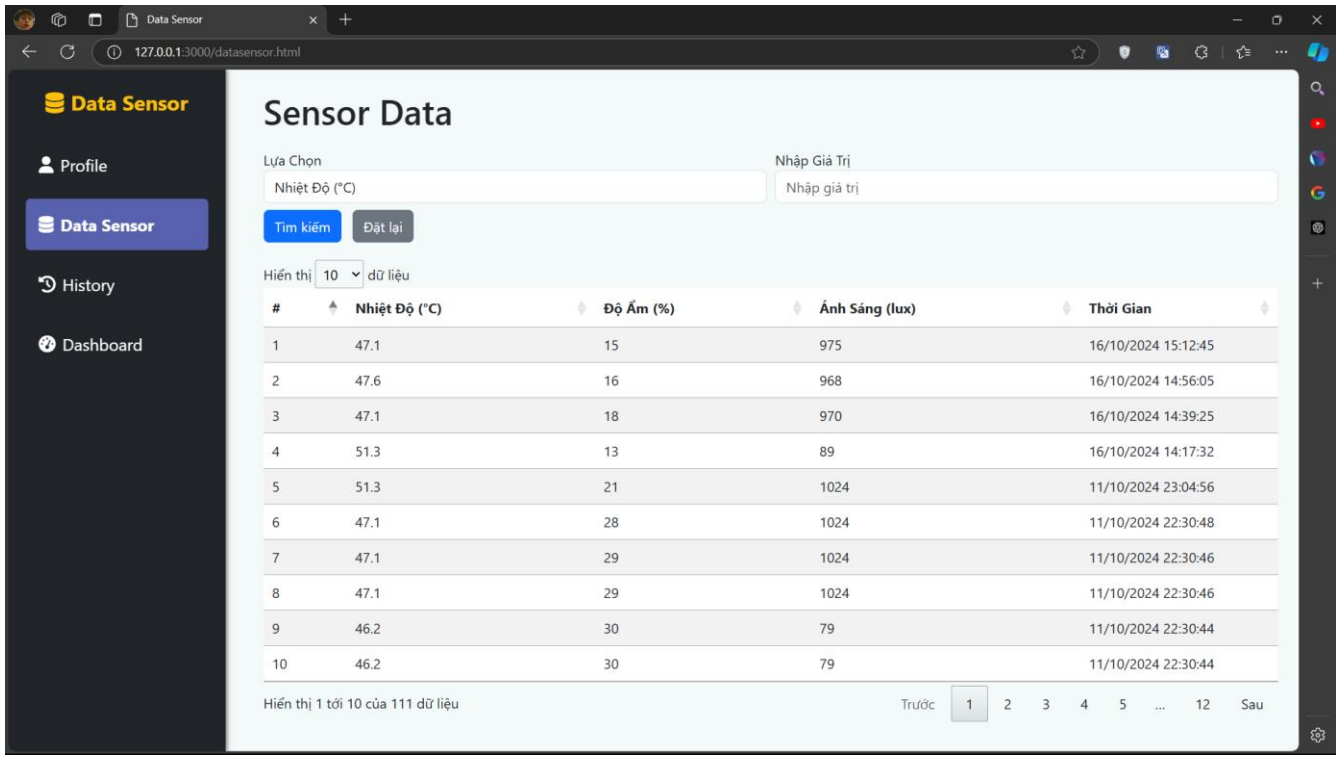
Hình ản trang profile



Hình ảnh trang dashboard



Hình ảnh trang Action history data



Hình ảnh Sensor Data

KẾT LUẬN

1. Đánh gía hiệu quả

Hệ thống IoT đo nhiệt độ, độ ẩm và ánh sáng theo thời gian thực mang lại nhiều lợi ích và hiệu quả đáng kể trong việc giám sát và điều khiển môi trường. Dưới đây là một số điểm chính trong đánh giá hiệu quả của hệ thống:

- **Theo dõi liên tục:** Hệ thống cho phép giám sát liên tục các thông số môi trường như nhiệt độ, độ ẩm và ánh sáng, từ đó cung cấp dữ liệu thời gian thực. Điều này giúp người dùng có cái nhìn tổng quan về điều kiện môi trường trong thời gian thực và kịp thời phát hiện các bất thường.
- **Cải thiện chất lượng sống:** Với khả năng theo dõi và điều chỉnh các thông số môi trường, hệ thống góp phần nâng cao chất lượng sống, đặc biệt trong các ứng dụng như nhà thông minh, nông nghiệp thông minh, và quản lý môi trường trong nhà kính.
- **Tăng cường hiệu quả năng lượng:** Hệ thống có khả năng tự động điều chỉnh các thiết bị như đèn, quạt, hay máy điều hòa dựa trên dữ liệu cảm biến, giúp tiết kiệm năng lượng và giảm thiểu lãng phí.
- **Giảm thiểu rủi ro:** Việc theo dõi liên tục giúp người dùng phát hiện sớm các vấn đề như nhiệt độ quá cao hay độ ẩm không đạt yêu cầu, từ đó có thể thực hiện các biện pháp khắc phục kịp thời, giảm thiểu rủi ro cho thiết bị và môi trường.
- **Dữ liệu phân tích:** Hệ thống lưu trữ và phân tích dữ liệu theo thời gian, cung cấp thông tin quý giá cho người dùng trong việc ra quyết định. Những dữ liệu này có thể được sử dụng để tối ưu hóa hoạt động của thiết bị và cải thiện điều kiện sống hoặc sản xuất.
- **Tính linh hoạt và mở rộng:** Hệ thống được thiết kế với tính năng mở rộng, cho phép người dùng dễ dàng thêm các cảm biến mới hoặc điều khiển thêm các thiết bị khác trong tương lai mà không cần thay đổi cấu trúc cơ bản của hệ thống.
- **Giao diện người dùng thân thiện:** Với sự tích hợp của các ứng dụng di động như Blynk, người dùng có thể dễ dàng quản lý và điều khiển các thiết bị từ xa thông qua giao diện thân thiện, không cần phải có kỹ năng lập trình cao.
- **Chi phí hiệu quả:** So với các hệ thống giám sát truyền thống, hệ thống IoT thường có chi phí thấp hơn nhờ vào việc sử dụng các cảm biến giá rẻ và công nghệ không dây, đồng thời dễ dàng bảo trì và nâng cấp.

2. Hướng phát triển

Hệ thống IoT đo nhiệt độ và độ ẩm theo thời gian thực cần tập trung phát triển theo các hướng sau để đạt hiệu quả tối ưu. Trước hết, tính năng thời gian thực phải được cải tiến để dữ liệu nhiệt độ và độ ẩm được thu thập, cập nhật liên tục thông qua các giao thức truyền thông nhanh chóng và đáng tin cậy như MQTT hoặc WebSocket. Dữ liệu cần được lưu trữ trong cơ sở dữ liệu như MongoDB để truy xuất và phân tích dễ dàng. Việc trực quan hóa dữ liệu cũng cần chú trọng, với các biểu đồ và bảng dữ liệu có tính năng lọc, tìm kiếm, sắp xếp để hỗ trợ người dùng theo dõi và phân tích xu hướng dễ dàng. Đặc biệt, hệ thống phải đảm bảo tính ổn định và bảo mật cao, sử dụng các cơ chế mã hóa và xác thực để bảo vệ dữ liệu trong quá trình truyền tải và lưu trữ.

Ngoài ra, khả năng mở rộng và tương thích cũng là yếu tố quan trọng, giúp hệ thống dễ dàng tích hợp thêm cảm biến mới và hỗ trợ cập nhật phần mềm từ xa (OTA) khi cần thiết. Tính

năng điều khiển và quản lý thiết bị từ xa thông qua giao diện web hoặc ứng dụng di động cũng cần được tích hợp, cho phép người dùng không chỉ theo dõi dữ liệu mà còn điều khiển các thiết bị như quạt và điều hòa từ xa. Với những định hướng phát triển này, hệ thống IoT sẽ không chỉ cung cấp dữ liệu thời gian thực chính xác mà còn mang lại sự tiện lợi, linh hoạt cho người dùng, đồng thời đáp ứng được các yêu cầu về bảo mật và khả năng mở rộng trong tương lai.