HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

MÔN HỌC: IOT VÀ ỨNG DỤNG

ĐỀ TÀI: HỆ THỐNG IOT NHÀ THÔNG MINH MINI

Giảng viên hướng dẫn : Kim Ngọc Bách

Nhóm học phần : 06

Nhóm bài tập lớn : 12

Danh sách thành viên

B22DCCN349	Trần Minh Hoàng
B22DCCN686	Bùi Công Sơn
B22DCCN782	Nguyễn Trần Minh Thái
B22DCCN913	Lê Anh Vũ

Hà Nội – 2023

Mục Lục

I, Giới thiệu đề tài

1. Lý do chọn đề tài

Nhà thông minh là một trong những xu hướng công nghệ nổi bật trong thời đại hiện nay, khi mà IoT (Internet of Things) ngày càng được ứng dụng sâu rộng trong đời sống. Việc lựa chọn đề tài này xuất phát từ nhu cầu thực tế về sự tiện nghi, an toàn và tối ưu hóa năng lượng trong quản lý nhà ở. Trong bối cảnh hiện đại, con người ngày càng quan tâm đến việc cải thiện chất lượng cuộc sống, và nhà thông minh cung cấp các giải pháp tự động hóa, tiết kiệm thời gian, và tăng cường bảo mật.

Bên cạnh đó, với sự phát triển mạnh mẽ của các thiết bị IoT, mô hình nhà thông minh còn mở ra cơ hội học tập và nghiên cứu các công nghệ tiên tiến như cảm biến, vi điều khiển, giao thức truyền thông (như MQTT, Zigbee), và trí tuệ nhân tạo (AI). Đề tài cũng góp phần giải quyết những vấn đề xã hội quan tâm như tiêu thụ năng lượng hiệu quả, giảm thiểu rủi ro cháy nổ, hay bảo vệ môi trường thông qua việc sử dụng công nghệ tối ưu.

Ngoài ra, mô hình nhà thông minh có tiềm năng ứng dụng rộng rãi, không chỉ trong đời sống gia đình mà còn trong các lĩnh vực kinh doanh, giáo dục và y tế. Do đó, việc nghiên cứu và triển khai đề tài không chỉ có ý nghĩa trong việc học tập mà còn tạo ra giá trị thực tiễn cao, đồng thời cung cấp kiến thức nền tảng cho các dự án IoT trong tương lai.

2. Tổng quan về vấn đề nghiên cứu

Nghiên cứu về mô hình nhà thông minh tập trung vào việc giải quyết nhiều vấn đề quan trọng, từ thiết kế hệ thống đến ứng dụng thực tế. Một trong những vấn đề cốt lõi là kiến trúc hệ thống, bao gồm việc lựa chọn các thiết bị, cảm biến và giao thức truyền thông để đảm bảo sự linh hoạt và tương thích. Các giao thức như MQTT, Zigbee hay Bluetooth cần được nghiên cứu để đảm bảo kết nối ổn định giữa các thiết bị. Bên cạnh đó, tự động hóa và trí tuệ nhân tạo

đóng vai trò quan trọng trong việc tạo ra các kịch bản tự động hóa thông minh, tối ưu hóa năng lượng và cải thiện trải nghiệm người dùng. Một vấn đề khác cần được nghiên cứu là bảo mật và quyền riêng tư của người dùng, vì các thiết bị IoT dễ bị tấn công nếu không có các biện pháp bảo vệ thích hợp.

Ngoài ra, việc phát triển các ứng dụng di động thân thiện, cùng với các giao diện người dùng trực quan, là yếu tố không thể thiếu để điều khiển hệ thống nhà thông minh. Cuối cùng, việc đánh giá tính thực tiễn và khả năng mở rộng của hệ thống, đặc biệt là trong điều kiện hạ tầng mạng hạn chế, là vấn đề quan trọng để đảm bảo sự thành công và hiệu quả lâu dài của nhà thông minh.

3. Mục đích nghiên cứu

Mục đích chính của nghiên cứu này là phát triển và triển khai thành công một ứng dụng điều khiển từ xa trên android và thiết kế được một bảng mạch mô hình đơn giản về các cảm biến được sử dụng trong mô hình nhà thông minh Nghiên cứu tập trung vào việc tạo ra một ứng dụng dễ sử dụng, có thể tích hợp trong các môi trường làm việc từ xa đồng thời thiết kế nên một board mạch có tính ứng dụng cao trong thực tế

4. Đối tượng và phạm vi nghiên cứu

- Đối tượng nghiên cứu: Giao thức MQTT, module ESP32, các cảm biến thông dụng được sử dụng trong các mô hình nhà thông minh trong thực tế.
- Phạm vi nghiên cứu: Dự án tập trung vào việc phát triển ứng dụng bật tắt các thiết bị từ xa thông qua giao thức MQTT, đồng thời thiết kế và phát triển một board mạch sử dụng module ESP32 để kết nối cũng như tương tác giữa các cảm biến.

5. Sơ lược các thiết bị trong hệ thống (dự kiến)

Phần cứng:

- ESP32 WROOM 32
- Cảm biến DHT11
- Cảm biến LDR
- Cảm biến chuyển động HC-SR501 PIR
- Màn hình LCD 1602 để hiển thị thông số
- Module 2 relay để điều khiển bật tắt quạt, máy bơm
- Server để đóng mở cửa
- Kit để kết nối các linh kiện
- Các dây dẫn
- Nguồn điện
- …

Phần mềm:

- Arduino IDE : Arduino IDE để viết mã cho ESP32. Chương trình viết xong sẽ được nạp vào ESP32 thông qua giao tiếp USB. Arduino IDE có công cụ kiểm tra lỗi và nạp chương trình để giúp đảm bảo rằng chương trình hoạt động ổn định.
- Frontend(ReactJS): Xây dựng giao diện người dùng để tương tác với hệ thống cũng như theo dõi nồng độ khí gas cũng nhưng trạng thái của các thiết bị
- Backend(NodeJS): xây dựng một API để giao tiếp với Database và phục vụ các yêu cầu từ frontend ReactJS

II, Cơ sở lý thuyết

1. Khái niệm về IOT (Internet of things)

Internet of Things (IoT), hay Internet vạn vật, là mô hình công nghệ cho phép các thiết bị vật lý được kết nối với nhau qua mạng Internet để thu thập, trao đổi và xử lý dữ liệu một cách tự động. Các thiết bị này có thể là cảm biến, thiết

bị điện tử, máy móc, phương tiện hoặc các vật thể trong đời sống hàng ngày, được trang bị phần mềm và kết nối mạng nhằm tương tác với môi trường và con người.

Hệ thống IoT thường gồm ba lớp chính: (1) Lớp cảm nhận, nơi các cảm biến thu thập dữ liệu vật lý như nhiệt độ, độ ẩm, ánh sáng, chuyển động; (2) Lớp mạng, đảm nhiệm truyền dữ liệu qua các giao thức như Wi-Fi, Bluetooth, Zigbee hoặc mạng di động; và (3) Lớp ứng dụng, nơi dữ liệu được xử lý, lưu trữ và hiển thị cho người dùng qua các nền tảng web hoặc di động.

Công nghệ IoT hiện được ứng dụng rộng rãi trong nhiều lĩnh vực như nhà thông minh, nông nghiệp, y tế, công nghiệp và giao thông thông minh. Trong đó, mô hình nhà thông minh cho phép người dùng giám sát và điều khiển các thiết bị điện tử trong gia đình từ xa, góp phần nâng cao tiện nghi, tiết kiệm năng lượng và đảm bảo an toàn.

2. Tiện ích khi sử dụng hệ thống nhà thông minh

Hệ thống nhà thông minh mang lại nhiều tiện ích vượt trội trong đời sống hiện đại, giúp tối ưu hóa sự tiện nghi, an toàn và hiệu quả năng lượng cho người sử dụng. Trước hết, người dùng có thể dễ dàng điều khiển các thiết bị điện như đèn, quạt, điều hòa hay rèm cửa từ xa thông qua điện thoại hoặc máy tính có kết nối Internet, giúp tiết kiệm thời gian và nâng cao trải nghiệm sử dụng.

Bên cạnh đó, hệ thống còn cho phép tự động hóa các hoạt động trong nhà dựa trên cảm biến hoặc kịch bản được thiết lập sẵn, ví dụ: đèn tự bật khi phát hiện chuyển động, điều hòa tự điều chỉnh theo nhiệt độ, hay rèm cửa tự đóng khi ánh sáng quá mạnh. Nhờ đó, không chỉ mang lại sự thoải mái mà còn giảm lãng phí năng lượng.

Ngoài ra, nhà thông minh còn tăng cường khả năng giám sát và bảo mật thông qua camera, cảm biến chuyển động, cảm biến khí gas hoặc cảnh báo cháy

nổ. Người dùng có thể nhận thông báo tức thời khi có sự cố, đảm bảo an toàn cho ngôi nhà ngay cả khi vắng mặt. Với khả năng mở rộng linh hoạt và kết nối nhiều thiết bị, hệ thống nhà thông minh đang trở thành xu hướng tất yếu, góp phần xây dựng môi trường sống hiện đại, tiện nghi và bền vững.

3. Cấu trúc hệ thống nhà thông minh trong thực tế

Một hệ thống nhà thông minh trong thực tế thường được xây dựng theo mô hình nhiều lớp, nhằm đảm bảo khả năng kết nối, điều khiển và mở rộng linh hoạt giữa các thiết bị. Cấu trúc tổng quát bao gồm bốn thành phần chính: tầng cảm biến, tầng điều khiển trung tâm, tầng truyền thông và tầng ứng dụng.

- Tầng cảm biến (Perception Layer): là nơi thu thập dữ liệu từ môi trường xung quanh thông qua các cảm biến như nhiệt độ, độ ẩm, ánh sáng, chuyển động, khí gas, hay camera. Các thiết bị này có nhiệm vụ chuyển đổi tín hiệu vật lý thành dữ liệu số để gửi lên bộ xử lý trung tâm.
- Tầng điều khiển trung tâm (Control Layer): bao gồm các vi điều khiển như ESP32, Raspberry Pi hoặc các bộ gateway. Tầng này chịu trách nhiệm xử lý dữ liệu thu được, đưa ra quyết định và gửi lệnh điều khiển tới các thiết bị đầu ra như relay, đèn, quạt hoặc ổ cắm thông minh.
- Tầng truyền thông (Network Layer): là cầu nối giúp các thiết bị trao đổi dữ liệu thông qua các giao thức như Wi-Fi, Zigbee, Bluetooth, hoặc MQTT. Tầng này đảm bảo việc truyền dữ liệu ổn định giữa thiết bị, máy chủ và ứng dụng người dùng.
- Tầng ứng dụng (Application Layer): cung cấp giao diện để người dùng theo dõi và điều khiển hệ thống. Dữ liệu từ các thiết bị được lưu trữ và hiển thị trên web hoặc ứng dụng di động, đồng thời cho phép người dùng thiết lập kịch bản tự động hóa, quản lý người dùng và cập nhật phần mềm từ xa.

Nhờ cấu trúc này, hệ thống nhà thông minh có thể hoạt động một cách ổn định, dễ dàng mở rộng thêm thiết bị mới, đồng thời hỗ trợ tích hợp các công nghệ tiên tiến như trí tuệ nhân tạo (AI) hoặc điện toán đám mây (Cloud Computing) để tối ưu hiệu suất và trải nghiệm sử dụng.

4. Giao thức HTTP, WebSocket và MQTT

Trong các hệ thống IoT, đặc biệt là mô hình nhà thông minh, việc lựa chọn giao thức truyền thông đóng vai trò quan trọng để đảm bảo tốc độ, độ tin cậy và hiệu quả trong việc trao đổi dữ liệu giữa các thiết bị và máy chủ. Ba giao thức phổ biến nhất là HTTP, WebSocket và MQTT, mỗi giao thức có đặc điểm và ứng dụng riêng phù hợp với từng mục đích.

- Giao thức HTTP (HyperText Transfer Protocol): là giao thức truyền tải dữ liệu phổ biến trong các ứng dụng web. HTTP hoạt động theo mô hình client–server, trong đó client (ví dụ: ESP32 hoặc trình duyệt web) gửi yêu cầu (request) và server phản hồi (response). Ưu điểm của HTTP là đơn giản, dễ triển khai và tương thích với hầu hết các thiết bị có kết nối Internet. Tuy nhiên, HTTP hoạt động theo cơ chế truy vấn định kỳ (polling), không phù hợp cho các ứng dụng cần cập nhật dữ liệu liên tục do gây tốn băng thông và độ trễ cao.
- Giao thức WebSocket: là phần mở rộng của HTTP, cho phép thiết lập kênh truyền hai chiều (full-duplex) giữa client và server sau khi kết nối được thiết lập. Nhờ đó, dữ liệu có thể được gửi và nhận liên tục mà không cần thực hiện nhiều lần request–response như HTTP. WebSocket thích hợp cho các ứng dụng cần truyền dữ liệu thời gian thực, chẳng hạn như giám sát cảm biến, hiển thị trạng thái thiết bị hoặc điều khiển từ xa với độ trễ thấp.
- Giao thức MQTT (Message Queuing Telemetry Transport): là giao thức truyền thông nhẹ được thiết kế chuyên biệt cho các ứng dụng IoT. MQTT

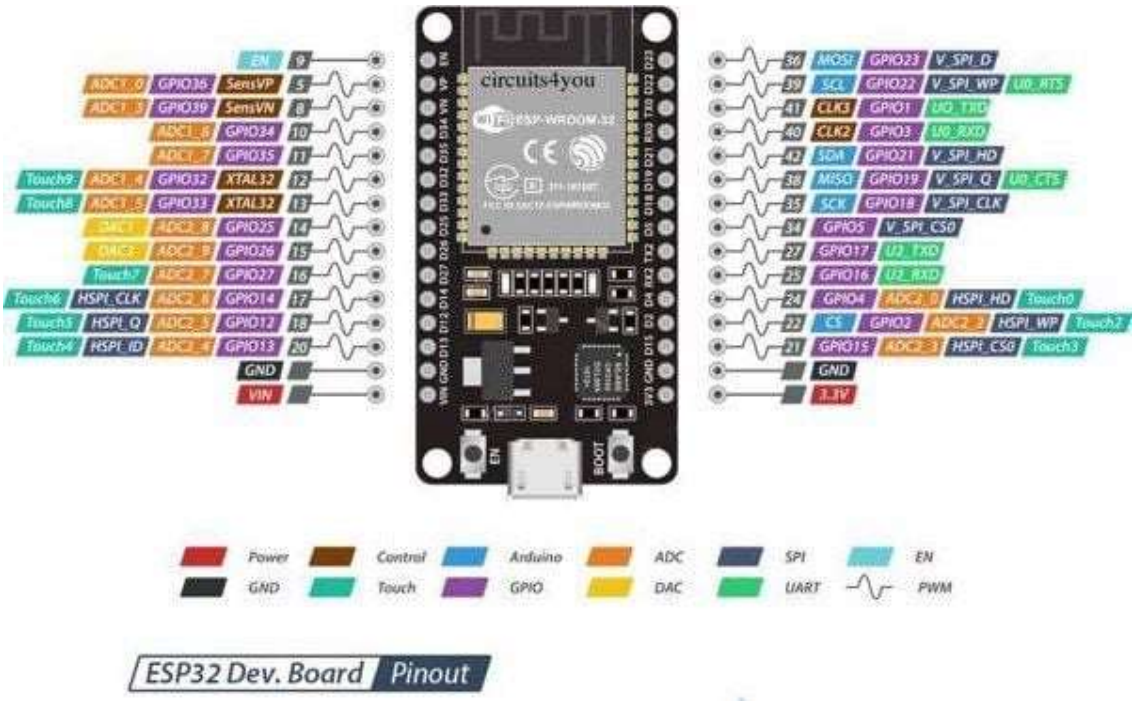
hoạt động theo mô hình publish–subscribe thông qua một máy chủ trung gian gọi là broker. Thiết bị gửi dữ liệu (publisher) và thiết bị nhận dữ liệu (subscriber) chỉ cần đăng ký vào cùng một chủ đề (topic) mà không cần biết địa chỉ cụ thể của nhau. Ưu điểm của MQTT là gọn nhẹ, tiết kiệm băng thông, hoạt động ổn định trong môi trường mạng yếu và hỗ trợ cơ chế xác nhận QoS (Quality of Service) để đảm bảo độ tin cậy khi truyền dữ liệu.

Tổng kết lại, trong hệ thống nhà thông minh, HTTP phù hợp cho việc cấu hình và gửi yêu cầu đơn giản, WebSocket thích hợp cho giao tiếp hai chiều liên tục giữa web và thiết bị, trong khi MQTT là lựa chọn tối ưu cho việc truyền dữ liệu cảm biến và điều khiển nhiều thiết bị IoT với hiệu suất cao và độ trễ thấp.

5. Các thiết bị trong hệ thống

5.1. ESP-WROOM-32:

Đây là bộ vi điều khiển mạnh mẽ với khả năng kết nối Wi-Fi và Bluetooth, rất phù hợp cho các ứng dụng IoT.

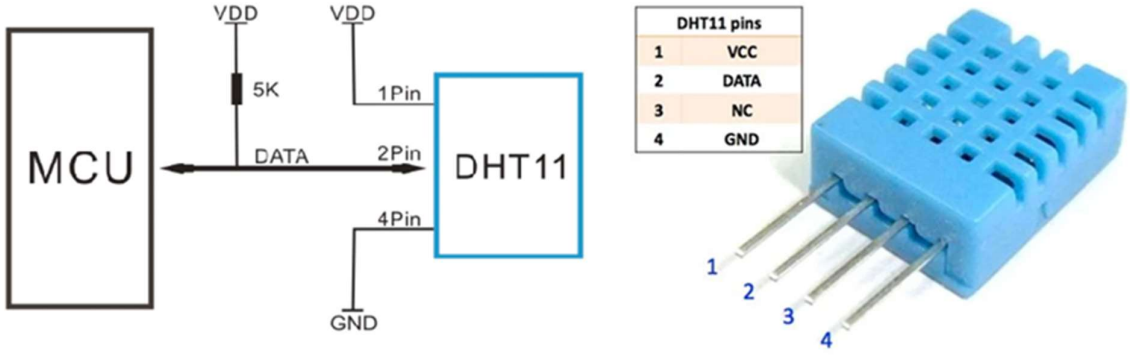


Phần cứng của ESP-WROOM-32:

- GPIO (General Purpose Input/Output): ESP32 có 34 chân GPIO có thể lập trình được, với mỗi chân có thể thực hiện nhiều chức năng khác nhau như đầu vào/đầu ra kỹ thuật số, ADC, UART, SPI, I2C, và PWM. Một số chân được chỉ định sẵn cho các chức năng đặc biệt như ADC hoặc DAC, nhưng các chân khác có thể được cấu hình tùy chỉnh qua chương trình.
- ADC (Analog to Digital Converter): ESP32 có 18 kênh ADC 12-bit chia thành hai khối (ADC1 và ADC2), cho phép chuyển đổi tín hiệu analog thành giá trị số từ 0 đến 4093. Độ phân giải cao của ADC giúp đo chính xác các tín hiệu từ cảm biến. DAC (Digital to Analog Converter): ESP32 tích hợp 2 kênh DAC 8-bit trên các chân GPIO25 và GPIO26, cho phép chuyển đổi tín hiệu số sang tín hiệu điện áp analog.
- PWM (Pulse Width Modulation): Có 16 kênh PWM độc lập, giúp điều khiển động cơ và đèn LED. Người dùng có thể điều chỉnh tần số, chu kỳ nhiệm vụ và gán kênh PWM cho bất kỳ chân GPIO nào.
- RTC GPIO (Real-Time Clock GPIO): ESP32 có 16 GPIO RTC, giúp đánh thức thiết bị khỏi chế độ ngủ sâu (Deep Sleep) nhờ các nguồn đánh thức bên ngoài.
- Cảm biến điện dung: Có 10 GPIO cảm ứng điện dung, sử dụng để phát hiện sự thay đổi điện dung khi có vật chạm vào chân cảm ứng, không cần phần cứng bổ sung.
- Giao diện UART: ESP32 có 3 giao diện UART cho truyền thông nối tiếp, thường sử dụng cho giao tiếp với máy tính hoặc các module khác.
- Giao diện SPI: ESP32 có 3 khối SPI (SPI, HSPI, VSPI), dùng cho truyền thông tốc độ cao với các thiết bị ngoại vi như bộ nhớ Flash.

- Giao diện I2C: 2 giao diện I2C cho phép giao tiếp với các cảm biến và thiết bị ngoại vi. Các chân I2C mặc định trong Arduino IDE là GPIO21 (SDA) và GPIO22 (SCL), nhưng có thể cấu hình lại.
- Chỉ đầu vào GPIO: Một số chân như GPIO34, GPIO35, GPIO36, và GPIO39 chỉ hỗ trợ đầu vào kỹ thuật số.

5.2. DHT 11

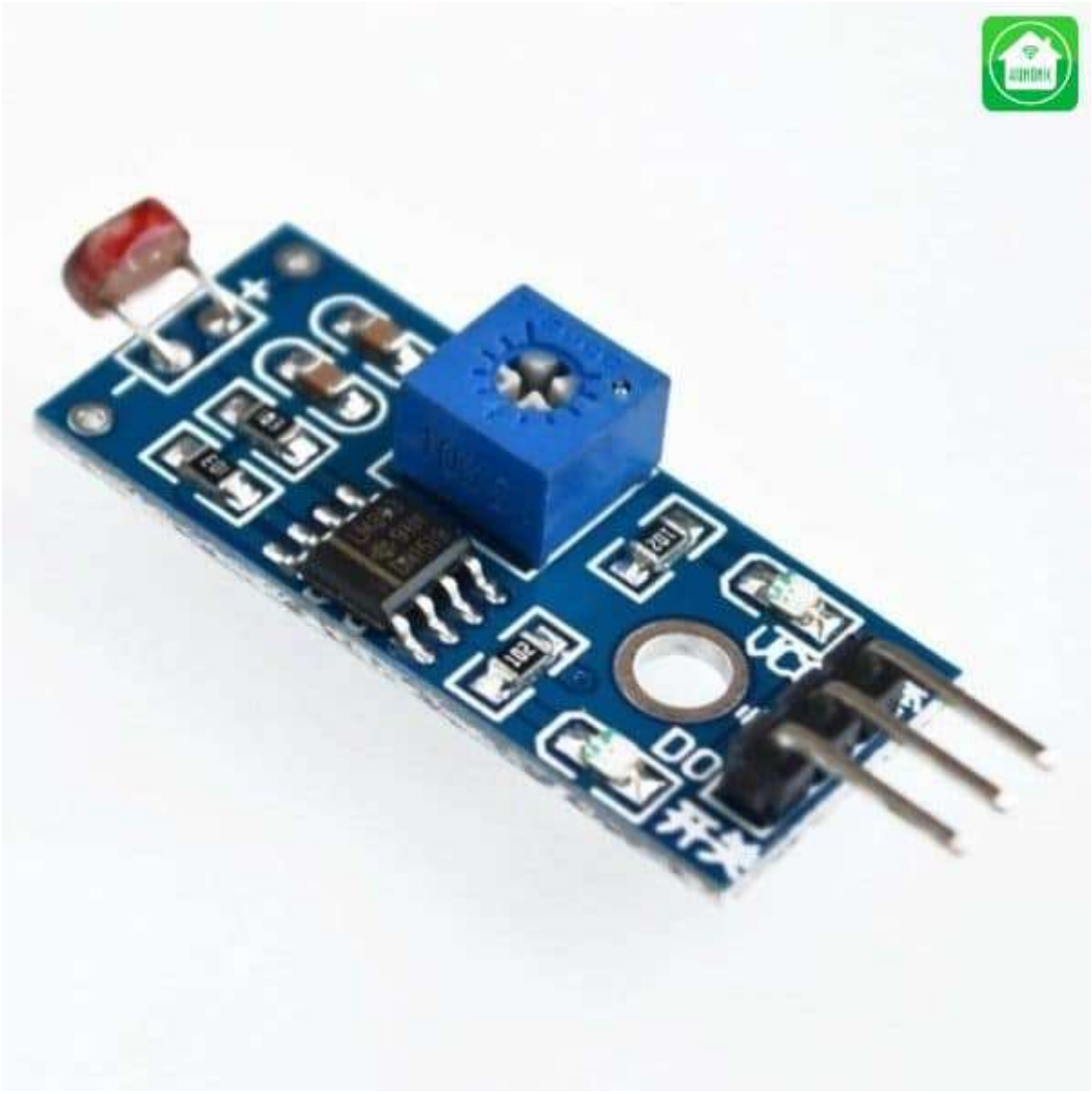


DHT-11 là module cảm biến nhiệt độ, độ ẩm có giao tiếp 1 dây (one wire). Cấu tạo cảm biến gồm 2 phần: một phần tử cảm biến độ ẩm bằng điện dung và một điện trở nhiệt để cảm nhận nhiệt độ. Tụ điện cảm biến độ ẩm có hai điện cực với chất nền giữ ẩm làm chất điện môi giữa chúng. Thay đổi giá trị điện dung xảy ra theo sự thay đổi của các mức độ ẩm. Ngoài ra bên trong module còn có các mạch chuyển đổi tương tự sang số. Cảm biến được tích hợp bộ tiền xử lý giúp cho dữ liệu nhận về được chính xác mà không cần phải qua bất kỳ khâu phân tích hay tính toán nào. Các thông số kỹ thuật của module cảm biến nhiệt độ, độ ẩm DHT-11 được cho như sau:

- Điện áp hoạt động 3,5÷5,5Vdc
- Dòng điện hoạt động cực đại là 2,5mA
- Dải độ ẩm hoạt động 20%÷95% RH, sai số ±5%RH (range humidity)
- Dải nhiệt độ hoạt động 0÷50^{0}C, sai số ±2^{0}C
- Nhiệt độ và độ ẩm đều có độ phân giải 16 bit.

- Tốc độ lấy mẫu không quá 1Hz (mỗi giây một lần).
- Tốc độ cảm nhận: trung bình 2s
- Khoảng cách truyền tối đa 20m.

5.3. LDR



Nguyên lý hoạt động: Cảm biến LDR (Light Dependent Resistor) là một loại điện trở có giá trị thay đổi theo cường độ ánh sáng chiếu vào. Khi ánh sáng mạnh, điện trở của LDR giảm; khi ánh sáng yếu hoặc tối, điện trở tăng cao.

Nguyên lý này cho phép LDR được sử dụng để phát hiện mức độ sáng trong môi trường và tự động điều khiển thiết bị như đèn, rèm cửa, hay hệ thống chiếu sáng.

Điện áp hoạt động: Cảm biến LDR thường hoạt động ở mức điện áp 3.3V – 5V DC, tương thích hoàn toàn với các vi điều khiển phổ biến như ESP32, Arduino hoặc Raspberry Pi.

Kết nối tín hiệu: LDR là cảm biến analog, nên cần được kết nối thông qua một mạch chia điện áp để tạo ra tín hiệu điện áp thay đổi theo cường độ sáng. Đầu ra của mạch này được nối vào chân ADC (Analog-to-Digital Converter) của ESP32 để đọc giá trị điện áp và quy đổi sang mức ánh sáng tương ứng. Với ESP32, có thể sử dụng các chân ADC như GPIO34, GPIO35 hoặc GPIO36 để đọc tín hiệu từ cảm biến.

Cấu tạo: Cảm biến LDR được chế tạo từ vật liệu bán dẫn Cadmium Sulfide (CdS), có đặc tính thay đổi điện trở khi ánh sáng chiếu vào. LDR thường được bọc trong vỏ nhựa có lỗ nhỏ để ánh sáng chiếu vào bề mặt cảm biến.

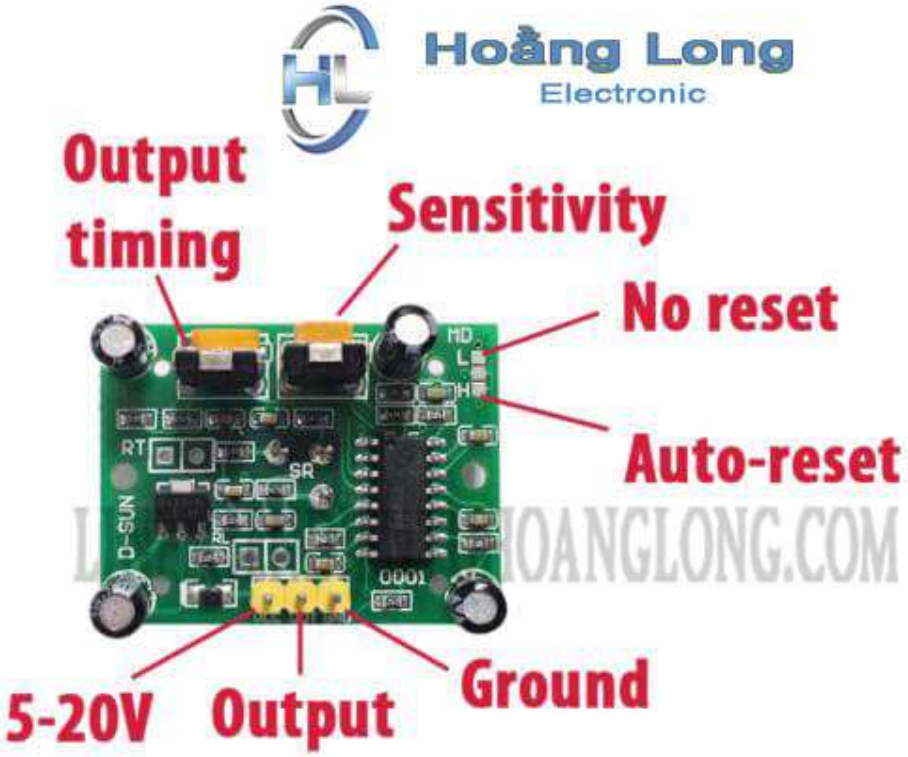
Thông số kỹ thuật cơ bản:

- Điện áp hoạt động: 3.3V – 5V DC
- Điện trở khi sáng: khoảng 1kΩ – 10kΩ
- Điện trở khi tối: khoảng 0.5MΩ – 2MΩ
- Thời gian đáp ứng: 20 – 30ms
- Giao tiếp: Analog (đọc qua ADC của ESP32)
- Nhiệt độ hoạt động: -30°C đến +70°C

Ứng dụng trong hệ thống nhà thông minh: Trong dự án này, LDR được sử dụng để đo độ sáng môi trường xung quanh. Khi ánh sáng giảm xuống dưới mức định sẵn, hệ thống sẽ tự động bật đèn; ngược lại, khi đủ sáng, đèn sẽ tự động tắt để tiết kiệm năng lượng. Ngoài ra, giá trị ánh sáng cũng được gửi lên máy chủ thông qua giao thức MQTT để hiển thị và lưu trữ, giúp người dùng có thể giám sát và điều chỉnh hệ thống từ xa.

LDR là loại cảm biến đơn giản, dễ sử dụng, giá thành rẻ và phù hợp cho các mô hình nhà thông minh mini hoặc các hệ thống tự động hóa cơ bản.

5.4. HC-SR501 PIR Sensor



Nguyên lý hoạt động: Cảm biến HC-SR501 là loại cảm biến thụ động hồng ngoại (PIR – Passive Infrared Sensor), có chức năng phát hiện chuyển động của con người hoặc động vật dựa trên sự thay đổi bức xạ hồng ngoại phát ra từ cơ thể sinh vật. Khi có người di chuyển trong phạm vi quét của cảm biến, tín hiệu hồng ngoại thay đổi và cảm biến sẽ gửi tín hiệu mức cao (HIGH) đến vi điều khiển để kích hoạt hành động tương ứng, ví dụ như bật đèn hoặc gửi cảnh báo.

Điện áp hoạt động: HC-SR501 hoạt động ổn định trong dải điện áp từ 4.5V đến 20V DC, phù hợp với cả các hệ thống 5V và 3.3V như ESP32. Tín hiệu đầu ra logic TTL (0V – 3.3V) nên có thể kết nối trực tiếp với chân GPIO của ESP32 mà không cần mạch chuyển mức điện áp.

Kết nối tín hiệu: Cảm biến HC-SR501 có 3 chân kết nối chính:

- VCC: cấp nguồn (4.5V – 20V)
- OUT: chân tín hiệu đầu ra (kết nối đến GPIO của ESP32, ví dụ GPIO27 hoặc GPIO14)
- GND: nối đất (ground)

Khi không phát hiện chuyển động, chân OUT ở mức thấp (LOW – 0V); khi phát hiện chuyển động, chân OUT ở mức cao (HIGH – 3.3V) trong khoảng thời gian được định trước.

Cấu tạo: HC-SR501 sử dụng phần tử cảm biến hồng ngoại Pyroelectric Infrared Sensor để phát hiện sự thay đổi năng lượng hồng ngoại. Mắt cảm biến được che phủ bởi thấu kính Fresnel, giúp mở rộng vùng quét và tập trung bức xạ hồng ngoại vào phần tử cảm biến. Ngoài ra, mạch còn tích hợp IC BISS0001 để khuếch đại tín hiệu và điều khiển đầu ra ổn định.

Thông số kỹ thuật cơ bản:

- Điện áp hoạt động: 4.5V – 20V DC
- Dòng tiêu thụ: < 65 µA
- Tín hiệu đầu ra: 0V (LOW) / 3.3V (HIGH)
- Phạm vi phát hiện: 3 – 7 mét (có thể điều chỉnh)
- Góc phát hiện: 100° – 120°
- Thời gian trễ đầu ra: 0.3s – 200s (điều chỉnh được)
- Kích thước module: 32mm x 24mm
- Giao tiếp: Digital (chân OUT nối trực tiếp với GPIO)

Ứng dụng trong hệ thống nhà thông minh: Trong mô hình nhà thông minh, cảm biến HC-SR501 được sử dụng để phát hiện sự hiện diện hoặc chuyển động của con người. Khi phát hiện có người, hệ thống sẽ tự động bật đèn, kích hoạt camera hoặc gửi tín hiệu cảnh báo qua MQTT. Cảm biến cũng có thể kết hợp với cảm biến ánh sáng LDR để đảm bảo đèn chỉ bật khi trời tối và có người di chuyển, giúp tiết kiệm năng lượng.

HC-SR501 là loại cảm biến có độ ổn định cao, dễ sử dụng và giá thành thấp, rất phù hợp cho các ứng dụng tự động hóa trong nhà thông minh, đặc biệt là trong các khu vực hành lang, sân, hoặc phòng khách.

III. Phân tích yêu cầu

1. Yêu cầu chức năng

1.1 Đo nhiệt độ, độ ẩm theo thời gian thực

- Hệ thống tích hợp cảm biến DHT11để đo nhiệt độ và độ ẩm của môi trường xung quanh theo thời gian thực
- Dữ liệu thu thập được sẽ được gửi về vi điều khiển ESP32 để xử lý và hiển thị lên màn hình LCD hoặc giao diện web/app
- Các giá trị đo được cũng được ghi lại vào bộ nhớ tạm thời hoặc cơ sở dữ liệu, phục vụ cho việc giám sát xu hướng biến đổi môi trường theo thời gian.

1.2. Đèn bật tắt tự động và điều khiển thủ công

- Hệ thống sử dụng cảm biến ánh sáng (LDR) để phát hiện cường độ ánh sáng môi trường
- Khi ánh sáng yếu (ban đêm hoặc trời tối), đèn sẽ tự động bật; khi đủ sáng, đèn tự động tắt

- Ngoài ra, người dùng có thể điều khiển thủ công việc bật/tắt đèn thông qua ứng dụng điện thoại hoặc công tắc ảo trên web server ESP32
- Trạng thái bật/tắt đèn được lưu lại vào hệ thống để đồng bộ giữa thiết bị thực tế và giao diện điều khiển.

1.3. Điều khiển quạt thủ công hoặc tự động

- Hệ thống tích hợp rơ-le điều khiển quạt để bật/tắt quạt dựa trên giá trị nhiệt độ đo được từ cảm biến SHT30
- Khi nhiệt độ vượt quá ngưỡng cho phép (ví dụ > 30°C), quạt tự động bật; khi nhiệt độ giảm xuống ngưỡng an toàn, quạt tự động tắt
- Người dùng vẫn có thể bật/tắt thủ công quạt từ xa thông qua ứng dụng điều khiển
- Dữ liệu về trạng thái quạt và ngưỡng nhiệt độ được lưu trữ trong bộ nhớ hoặc cơ sở dữ liệu, có thể thay đổi theo yêu cầu người dùng.

1.4. Điều khiển thiết bị qua điện thoại hoặc giao diện web

- Hệ thống cung cấp giao diện điều khiển qua điện thoại/ web, được xây dựng dựa trên giao tiếp Wi-Fi của ESP32
- Người dùng có thể thực hiện các thao tác như: bật/tắt đèn, quạt, xem nhiệt độ – độ ẩm, và theo dõi trạng thái hệ thống theo thời gian thực
- Dữ liệu từ điện thoại được gửi đến ESP32 thông qua HTTP hoặc MQTT, sau đó xử lý và phản hồi trực tiếp trên ứng dụng
- Thông tin điều khiển và phản hồi được ghi nhận tạm thời trong bộ nhớ hệ thống để đảm bảo đồng bộ.

1.5. Quản lý hệ thống từ xa (vòng đời, trạng thái)

- Người dùng có thể theo dõi và giám sát trạng thái của toàn hệ thống (thiết bị đang bật/tắt, nhiệt độ hiện tại, độ ẩm, kết nối mạng, v.v.) thông qua giao diện web hoặc app di động.
- Hệ thống hỗ trợ điều khiển và cập nhật thông tin từ xa nhờ vào kết nối Internet
- Các thông tin về vòng đời thiết bị (thời gian hoạt động, lỗi, log sự kiện) được lưu trữ trên cơ sở dữ liệu hoặc bộ nhớ trong ESP32, hỗ trợ việc quản lý và bảo trì sau này.

1.6. Gửi cảnh báo khi có sự kiện bất thường

- Khi phát hiện các điều kiện bất thường như nhiệt độ vượt ngưỡng, độ ẩm quá thấp/cao, hoặc thiết bị ngắt kết nối, hệ thống sẽ tự động gửi cảnh báo đến người dùng qua ứng dụng điện thoại hoặc email.
- Cảnh báo có thể được hiển thị dưới dạng thông báo đẩy (push notification) hoặc tin nhắn trên giao diện điều khiển.
- Mỗi cảnh báo được ghi lại trong nhật ký hệ thống để phục vụ việc kiểm tra và đánh giá hoạt động.

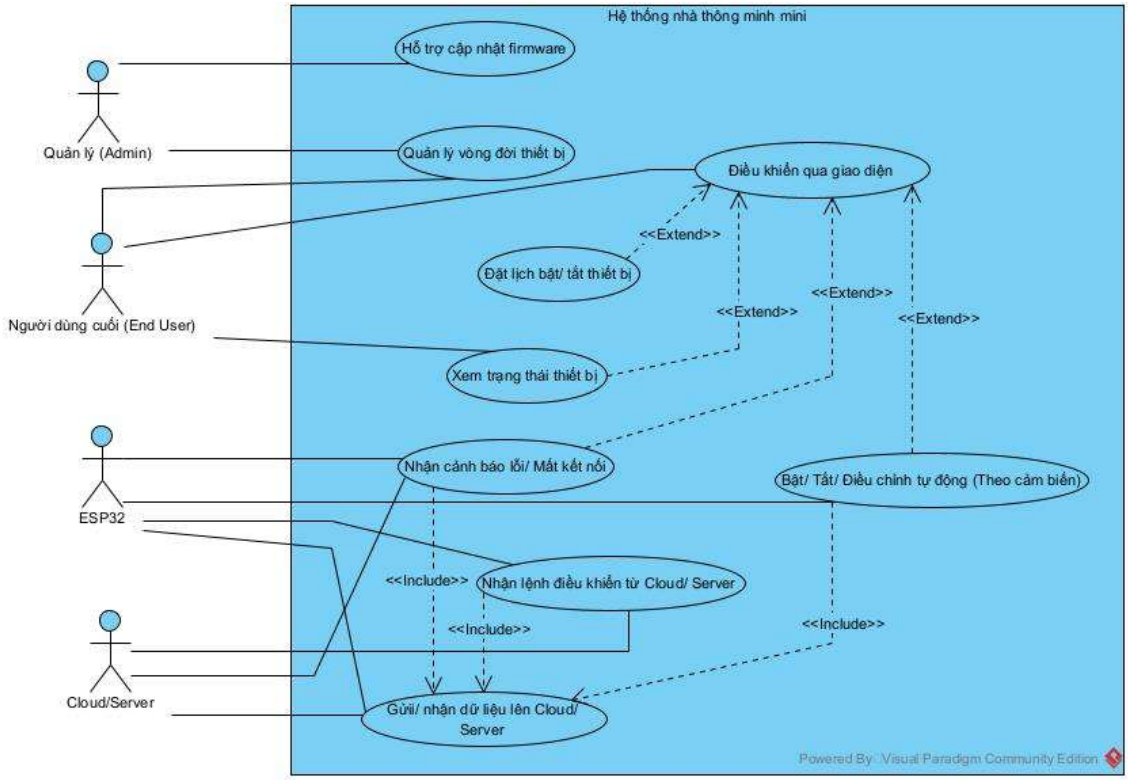
1.7. Hỗ trợ cập nhật firmware từ xa (OTA – Over-The-Air)

- Hệ thống hỗ trợ nâng cấp firmware từ xa cho ESP32 thông qua tính năng OTA.
- Khi có phiên bản mới, người dùng hoặc kỹ thuật viên có thể tải firmware lên hệ thống qua mạng Wi-Fi mà không cần kết nối trực tiếp qua cáp.
- Quá trình cập nhật được kiểm tra checksum và xác thực để đảm bảo an toàn.
- Thông tin về phiên bản hiện tại và lịch sử cập nhật được lưu lại trong bộ nhớ ESP32.

1.8. Lưu trữ thông tin hoạt động

- Tất cả các dữ liệu như nhiệt độ, độ ẩm, trạng thái đèn/quạt, thời điểm bật/tắt, và cảnh báo sẽ được ghi lại định kỳ.
- Dữ liệu có thể được lưu trữ cục bộ trên thẻ nhớ, bộ nhớ trong của ESP32 hoặc gửi lên server (Firebase, Thingspeak, v.v.).
- Việc lưu trữ giúp hệ thống phân tích xu hướng, theo dõi lịch sử hoạt động và hỗ trợ tính năng AI trong các phiên bản nâng cấp sau.

Sơ đồ Use-case tổng quan cho hệ thống



2.Yêu cầu phi chức năng

2.1. Hiệu năng (Performance)

- Khi cảm biến chuyển động (PIR) phát hiện có người, đèn phải phản hồi và bật sáng trong vòng ≤ 2s.

- Khi người dùng bấm nút trên ứng dụng web/di động để bật thiết bị điện đèn, quạt, phải nhận được tín hiệu và khởi động trong ≤ 2s trong môi trường mạng cục bộ
- Tần suất cập nhật dữ liệu: Cảm biến nhiệt độ, độ ẩm (DHT11) cần gửi dữ liệu mới lên máy chủ/ứng dụng ≤ 60s/lần, đảm bảo dữ liệu tương đối thời gian thực.

2.2. Độ tin cậy (Reliability)

- Uptime của các dịch vụ backend và môi trường cần đạt ≤ 99,9%
- Khả năng tự phục hồi:
    - Nếu bị mất kết nối Wi-Fi, vi điều khiển ESP32 phải có khả năng tự động kết nối lại trong vòng 60s khi mạng có trở lại.
    - Nếu mất điện, sau khi có điện trở lại, hệ thống phải tự khởi động và hoạt động với các cài đặt cuối cùng được lưu.
- Các lệnh điều khiển quan trọng (như bật/tắt thiết bị) không được phép bị mất trên đường truyền.

2.3. Bảo mật (Security)

- Ứng dụng web/di động có cơ chế đăng nhập (tên người dùng, mật khẩu) để chỉ người dùng đã đăng nhập mới có quyền xem và điều khiển thiết bị.
    - Backend sẽ triển khai API đăng nhập trả về một JSON Web Token (JWT). Mọi yêu cầu API sau đó từ frontend phải đính kèm JWT này trong header để xác thực
- Mật khẩu Wi-Fi, mật khẩu đăng nhập người dùng phải được lưu trữ an toàn trên ESP32 và không được lập trình ở dạng văn bản thuần (plaintext) trong code.
- Dữ liệu truyền giữa ESP32 và máy chủ phải được mã hóa. Khi hệ thống được truy cập từ bên ngoài, kết nối giữa client và backend phải sử dụng HTTPS.

2.4. Khả năng sử dụng (Usability)

- Lần đầu khởi động, ESP32 sẽ phát ra một mạng Wi-Fi riêng (Access Point mode). Người dùng kết nối vào mạng này, truy cập một trang web đơn giản trên ESP32 để nhập thông tin Wi-Fi của nhà. Thông tin này sau đó sẽ được lưu vào NVS.
- Khi người dùng thực hiện một hành động (ví dụ: đặt lịch hẹn giờ), hệ thống phải có phản hồi hệ thống: thông báo xác nhận hành động đó đã thành công hay thất bại.

2.5. Khả năng mở rộng

- Thêm thiết bị mới: Kiến trúc phần mềm (đặc biệt là backend và giao thức MQTT) phải được thiết kế để có thể dễ dàng thêm các cảm biến mới (cảm biến ánh sáng, cảm biến khí gas) hoặc các thiết bị điều khiển khác (ổ cắm thông minh) mà không cần phải thiết kế lại toàn bộ hệ thống.

2.6. Khả năng bảo trì (Maintainability)

- Cập nhật phần mềm (Firmware): Hệ thống nên hỗ trợ cập nhật firmware cho ESP32 từ xa qua mạng (OTA - Over-The-Air. Điều này giúp nâng cấp tính năng hoặc vá lỗi bảo mật mà không cần phải kết nối cáp vật lý.
- Tổ chức mã nguồn: Mã nguồn cho ESP32 và backend cần được viết một cách rõ ràng, có chú thích đầy đủ để người khác có thể đọc hiểu và chỉnh sửa.
- Ngoài việc ghi log qua cổng Serial, ESP32 sẽ gửi các thông tin trạng thái quan trọng (ví dụ: kết nối thành công, mất kết nối, lỗi cảm biến).

3. Yêu cầu về giao tiếp

Giao tiếp giữa người dùng và hệ thống:

- Người dùng tương tác với hệ thống qua giao diện web (ReactJS), và giao diện này phải dễ sử dụng, thân thiện, hiển thị rõ ràng các thông số cần thiết.
- Hệ thống phải đảm bảo phản hồi ngay lập tức khi người dùng thay đổi giá trị ngưỡng hoặc điều khiển thiết bị từ xa.

4. Yêu cầu về môi trường hoạt động

- Điều kiện hoạt động của cảm biến: Các cảm biến cần hoạt động ổn định trong các điều kiện môi trường khắc nghiệt như nhiệt độ cao hoặc độ ẩm lớn.
- Khả năng tích hợp với các thiết bị khác: Hệ thống cần có khả năng tích hợp với các thiết bị thông minh khác trong nhà thông minh, như hệ thống quạt thông gió tự động, cửa tự động và các thiết bị bảo vệ khác.

5. Yêu cầu về thử nghiệm

Thử nghiệm cảm biến:

- Kiểm tra độ chính xác của cảm biến DHT11, LDR, HC-SR501… trong các điều kiện môi trường khác nhau.
- Đảm bảo dữ liệu đo được phản hồi đúng và ổn định khi gửi lên hệ thống.

Thử nghiệm điều khiển thiết bị:

- Đánh giá khả năng bật/tắt đèn và quạt thủ công và tự động theo điều kiện cảm biến.
- Kiểm tra độ trễ giữa thao tác điều khiển trên web và phản hồi thực tế của thiết bị.

Thử nghiệm truyền thông và đồng bộ:

- Đảm bảo ESP32 truyền và nhận dữ liệu qua MQTT/Wifi ổn định, không mất gói hoặc trễ quá lâu.

- Kiểm tra khả năng đồng bộ dữ liệu giữa thiết bị và giao diện web theo thời gian thực.

Thử nghiệm OTA (Over-The-Air):

- Kiểm tra khả năng cập nhật firmware từ xa qua Wi-Fi, đảm bảo thiết bị khởi động lại và hoạt động bình thường sau khi cập nhật.

Thử nghiệm nhận diện AI (dự kiến):

- Đánh giá độ chính xác khi nhận diện người và bỏ qua vật nuôi.
- Kiểm tra phản hồi điều khiển tự động (ví dụ bật đèn khi phát hiện người) qua MQTT/Wifi.

Bảng phân chia công việc (dự kiến)

Họ Tên	Mã sinh viên	Công việc
Nguyễn Trần Minh Thái	B22DCCN782	Xây dựng web giao tiếp Lắp đặt thiết bị
Lê Anh Vũ	B22DCCN913	Báo cáo Slide Lắp đặt thiết bị Lập trình Arduino
Trần Minh Hoàng	B22DCCN349	Xây dựng web giao tiếp Lập trình Arduino Lắp đặt thiết bị

Bùi Công Sơn	B22DCCN686	Xây dựng web giao tiếp Lắp đặt thiết bị

IV. Phân tích thiết kế hệ thống

V. Đánh giá kết quả dự án

VI. Kết luận

Tài liệu tham khảo