|  |
| --- |
| TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN  **KHOA CÔNG NGHỆ THÔNG TIN**  A logo with a black background  Description automatically generated  **ĐỒ ÁN 3**  **Thiết kế hệ thống giám sát, điều khiển dùng trong ngôi nhà thông minh**  Giảng viên hướng dẫn : **Chu Bá Thành**  Sinh viên thực hiện : **Trần Văn Đông**  Mã sinh viên : **10121623**  Lớp : **101213**  **Tháng 5/2024** |

**MỤC LỤC**

[DANH SÁCH HÌNH VẼ 3](#_Toc187730026)

[DANH SÁCH BẢNG BIỂU 4](#_Toc187730027)

[DANH SÁCH TỪ VIẾT TẮT 5](#_Toc187730028)

[1.1 Lý do chọn đề tài 6](#_Toc187730029)

[1.2 Mục tiêu của đề tài 7](#_Toc187730030)

[1.3 Đối tượng và phạm vi của đề tài 7](#_Toc187730031)

[1.4 Nội dung thực hiện 7](#_Toc187730032)

[1.5 Phương pháp tiếp cận 8](#_Toc187730033)

[2.1 Tổng quan về ngoại vi 9](#_Toc187730034)

[2.1.1 Module vi điều khiển ESP32-DEV KIT 9](#_Toc187730035)

[2.1.2 Cảm biến nhiệt độ DHT11 11](#_Toc187730036)

[2.1.3 Cảm biến mưa 12](#_Toc187730037)

[2.1.4 Cảm biến khí gas MQ2 14](#_Toc187730038)

[2.2 Phần mềm cho hệ thống nhúng 15](#_Toc187730039)

[3.1 Xây dựng kiến trúc của hệ thống 17](#_Toc187730040)

[3.1.1 Kiến trúc tổng quan của hệ thống 17](#_Toc187730041)

[3.1.2 Đặc tả yêu cầu hệ thống 17](#_Toc187730042)

[3.2 Thiết kế hệ thống. 17](#_Toc187730043)

[3.2.1 Thiết kế phần cứng hệ thống 17](#_Toc187730044)

[3.2.2 Thiết kế phần mềm hệ thống 18](#_Toc187730045)

[4.1 Kết quả đạt được của đề tài 19](#_Toc187730046)

[4.2 Hạn chế của đề tài 20](#_Toc187730047)

[4.3 Hướng phát triển của đề tài 20](#_Toc187730048)

[TÀI LIỆU THAM KHẢO 21](#_Toc187730049)

[PHỤ LỤC 22](#_Toc187730050)

DANH SÁCH HÌNH VẼ

[Hình 2.2. 1 Module ESP32 dev kit 9](#_Toc187730681)

[Hình 2.2. 2 Sơ đồ chân ESP32 10](#_Toc187730682)

[Hình 2.2. 3 Cảm biến nhiệt độ, độ ẩm 11](#_Toc187730683)

[Hình 2.2. 4 cảm biến mưa 13](#_Toc187730684)

[Hình 2.2. 5 Phần mềm arduino IDE 16](#_Toc187730685)

[Hình 3.2.2. 1 Tạo kết nối wifi và firebase 19](#_Toc187730687)

DANH SÁCH BẢNG BIỂU

[Bảng 2.1.2 1 12](#_Toc187730877)

[Bảng 2.1.2 2 12](#_Toc187730878)

DANH SÁCH TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| Từ viết tắt | Từ đầy đủ | Giải thích |
| CMS | Content Management System | Hệ quản trị nội dung |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# TỔNG QUAN VỀ ĐỀ TÀI

* 1. Lý do chọn đề tài

Trong những năm gần đây, **Arduino IDE** trở thành công cụ phổ biến và tiện lợi cho các dự án phát triển hệ thống nhúng và IoT. Các giải pháp dựa trên Arduino như ESP32, ESP8266, … đã được nhiều cá nhân và doanh nghiệp lựa chọn để triển khai hệ thống điều khiển nhà thông minh. Tuy nhiên, các dự án sử dụng Arduino IDE cũng đối mặt với một số hạn chế như: thiết kế phần mềm thường thiếu tính cấu trúc, khó bảo trì khi dự án mở rộng, yêu cầu người phát triển phải nắm vững các kiến thức về lập trình nhúng, và khả năng tối ưu hiệu suất không cao. Đặc biệt, việc tích hợp hệ thống với các giao thức IoT phổ biến như MQTT, HTTP, hoặc WebSocket cần sự can thiệp thủ công, dẫn đến tăng thời gian và chi phí triển khai.

Tại Việt Nam, số lượng đơn vị triển khai các dự án IoT và Smart Home dựa trên Arduino IDE đang tăng lên, nhưng đa số các dự án này mang tính cá nhân hoặc phục vụ mục đích thương mại nhỏ lẻ. Trong thực tế, việc phát triển hệ thống điều khiển nhà thông minh cần một quy trình tối ưu để đảm bảo tính linh hoạt, dễ triển khai và khả năng mở rộng, thay vì phải viết lại toàn bộ mã nguồn từ đầu cho mỗi dự án.

Hệ thống điều khiển nhà thông minh dựa trên nền tảng **Arduino IDE** tận dụng sức mạnh của các vi điều khiển như ESP32 để kết nối và điều khiển các thiết bị thông minh thông qua giao thức không dây. Điều này giúp giảm chi phí phần cứng và đáp ứng nhu cầu về hiệu suất. Tuy nhiên, để hệ thống thực sự hiệu quả, cần xây dựng một framework phần mềm dựa trên Arduino IDE, hỗ trợ sẵn các module xử lý như: điều khiển thiết bị qua MQTT, lập lịch điều khiển, giám sát trạng thái thiết bị, và tích hợp giao tiếp thời gian thực.

Bằng cách phát triển giải pháp phần mềm tối ưu, người dùng có thể dễ dàng tạo và quản lý hệ thống nhà thông minh một cách nhanh chóng mà không yêu cầu kiến thức lập trình chuyên sâu, đồng thời giảm thiểu chi phí và thời gian triển khai hệ thống.

* 1. Mục tiêu của đề tài

Giải pháp thiết kế hệ thống điều khiển nhà thông minh được xây dựng dựa trên Arduino IDE, mang lại sự tiện lợi và hiệu quả cao trong việc phát triển các hệ thống IoT. Arduino IDE, một công cụ lập trình nhúng phổ biến, cho phép kết nối và kiểm soát các thiết bị thông minh như cảm biến, công tắc, và các module điều khiển một cách dễ dàng và linh hoạt. Hệ thống được thiết kế nhằm hỗ trợ người dùng cấu hình và quản lý các thiết bị trong nhà mà không cần nhiều kiến thức lập trình chuyên sâu, đồng thời cung cấp giao diện đơn giản để điều khiển từ xa qua ứng dụng di động hoặc web.

Việc sử dụng Firebase làm nền tảng lưu trữ và xử lý dữ liệu thời gian thực giúp đảm bảo khả năng giám sát liên tục và đồng bộ hóa trạng thái thiết bị nhanh chóng. Giải pháp này tập trung vào việc giảm thiểu chi phí phát triển, tối ưu thời gian triển khai và nâng cao khả năng mở rộng, mang đến một hệ thống nhà thông minh hiệu quả, linh hoạt và dễ sử dụng cho mọi người dùng.

* 1. Đối tượng và phạm vi của đề tài

Đề tài tập trung vào việc phát triển hệ thống điều khiển nhà thông minh bằng cách kết hợp nền tảng Firebase và Android Studio, sử dụng ngôn ngữ lập trình Java. Firebase, một nền tảng đám mây của Google, được sử dụng để quản lý cơ sở dữ liệu thời gian thực, đồng bộ hóa trạng thái thiết bị và hỗ trợ giao tiếp giữa ứng dụng và hệ thống IoT. Android Studio là công cụ chính để phát triển ứng dụng di động, cung cấp giao diện người dùng thân thiện giúp người dùng dễ dàng điều khiển các thiết bị thông minh trong nhà. Mục tiêu chính là tạo ra một giải pháp tối ưu, dễ sử dụng, và tiết kiệm chi phí cho các cá nhân hoặc hộ gia đình muốn áp dụng công nghệ nhà thông minh.

* 1. Nội dung thực hiện

Hiện nay, một số giải pháp giám sát và điều khiển từ xa trong ngôi nhà thông minh vẫn tồn tại các hạn chế như : chỉ cố định ở một vị trí, thiết kế phức tạp, khó khăn trong việc phát triển và bảo trì, đồng thời yêu cầu người dùng phải có trình độ chuyên môn cao.

Vì vậy, việc nghiên cứu và phát triển hệ thống giám sát, điều khiển đơn giản, linh hoạt và dễ sử dụng là cần thiết. Đưa ra cấu trúc, tính năng và mô hình hoạt động của hệ thống nhằm đáp ứng nhu cầu giám sát và điều khiển từ xa trong ngôi nhà thông minh.

Lập trình và cài đặt chức năng : Phát triển phần mềm để thực hiện các chức năng giám sát và điều khiển, đảm bảo khả năng tương thích với các thiết bị trong hệ thống nhà thông minh.

Thiết kế giao diện điều khiển trên app mobile : Tạo một nền tảng trực quan, thân thiện với người dùng, cho phép điều khiển các thiết bị và theo dõi thông tin từ xa qua trình duyệt web.

Kiểm thử hệ thống : Thực hiện các bước kiểm tra, đánh giá để đảm bảo hệ thống hoạt động ổn định, chính xác và an toàn trước khi đưa vào sử dụng.

Triển khai thực tế : Đưa hệ thống vào thử nghiệm trong môi trường thực tế, đánh giá khả năng hoạt động và tiến hành tối ưu hóa dựa trên phản hồi.

* 1. Phương pháp tiếp cận

Để xây dựng một hệ thống giám sát và điều khiển trong ngôi nhà thông minh, cần áp dụng các phương pháp nghiên cứu và tìm hiểu sâu về các công nghệ liên quan đến IoT, robot học, truyền thông và hệ thống điều khiển. Cụ thể, quy trình thực hiện như sau :

* Nghiên cứu các công nghệ hiện đại trên Internet liên quan đến IoT, truyền thông và điều khiển thông minh.
* Theo dõi tiến bộ mới nhất trong lĩnh vực nhà thông minh để áp dụng vào thiết kế và triển khai hệ thống.

Tiếp thu đóng góp từ giáo viên hướng dẫn: Luôn lắng nghe và tiếp thu các đóng góp, góp ý từ giáo viên hướng dẫn của bạn. Họ có thể cung cấp những chỉ dẫn quý báu và phản hồi giúp bạn hoàn thiện hơn hệ thống và phương pháp nghiên cứu của mình.

# CƠ SỞ LÝ THUYẾT

* 1. Tổng quan về ngoại vi
     1. Module vi điều khiển ESP32-DEV KIT

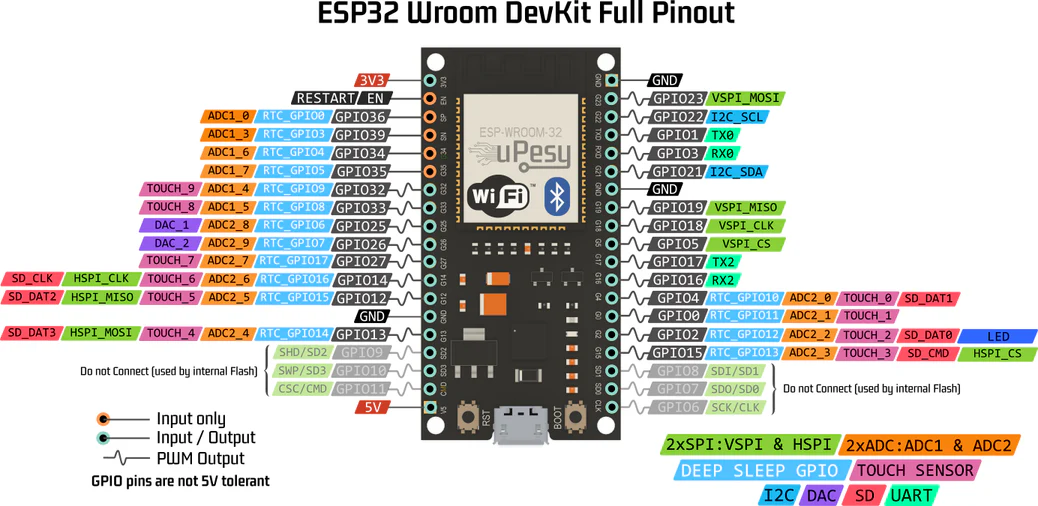


Hình 2.2. Module ESP32 dev kit

ESP32 Dev Kit là một bo mạch phát triển mạnh mẽ với kích thước nhỏ gọn và tính năng vượt trội, lý tưởng cho các ứng dụng IoT. Với khả năng xử lý độc lập và hỗ trợ đa dạng các giao thức kết nối, ESP32 Dev Kit mang đến sự linh hoạt và hiệu quả cao cho nhiều dự án.

ESP32 Dev Kit nổi bật với thiết kế nhỏ gọn, kích thước chỉ 40 x 27 x 12 mm, giúp dễ dàng tích hợp vào các hệ thống IoT. Với khả năng tiêu thụ năng lượng thấp, dòng nghỉ chỉ 6mA, bo mạch này phù hợp cho các ứng dụng yêu cầu hoạt động liên tục mà vẫn tiết kiệm năng lượng. Tích hợp cả WiFi và Bluetooth BLE, ESP32 Dev Kit mang đến khả năng kết nối mạnh mẽ, lý tưởng cho các hệ thống giám sát và điều khiển không dây. Được trang bị bộ vi điều khiển ESP32 hiệu suất cao, bo mạch hỗ trợ xử lý đa nhiệm và đảm bảo kết nối mạng ổn định. Bên cạnh đó, khả năng tương thích với Arduino IDE cùng sự hỗ trợ mạnh mẽ từ cộng đồng giúp việc lập trình và phát triển dự án trở nên dễ dàng, giảm thiểu các khó khăn trong triển khai và bảo trì.

1. **Sơ đồ chân**

****

Hình 2.2. Sơ đồ chân ESP32

ESP32 Dev Kit là một bo mạch phát triển mạnh mẽ, cung cấp nhiều chân GPIO với khả năng cấu hình linh hoạt, hỗ trợ WiFi và Bluetooth BLE, thích hợp cho các dự án IoT. Dưới đây là phần mô tả chi tiết về các chân và chức năng của chúng.

Chân nguồn:

* GND: Chân nối đất (Ground), cần thiết cho mạch điện.
* Vin: Chân cấp nguồn đầu vào (5V từ nguồn ngoài hoặc USB).
* 3.3V: Đầu ra nguồn 3.3V, cung cấp điện cho các cảm biến hoặc module nhỏ.

Các chân GPIO của ESP32 Dev Kit được thiết kế đa năng, hỗ trợ nhiều chức năng như đọc tín hiệu digital, tín hiệu analog, điều khiển PWM, giao tiếp UART, SPI, I2C... Dưới đây là một số chân nổi bật:

Chân GPIO:

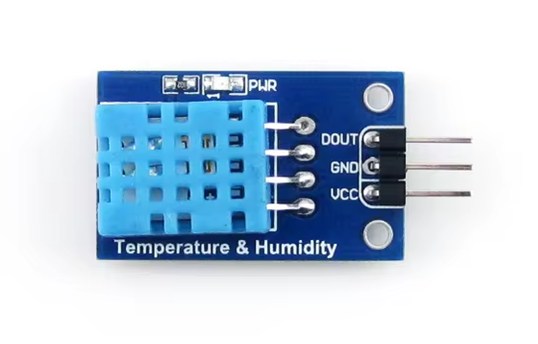
* GPIO2: Tích hợp đèn LED on-board, sử dụng làm chỉ báo trạng thái.
* GPIO0: Chân chọn chế độ flash. Để lập trình, nối GPIO0 với GND.
* GPIO12, GPIO13, GPIO14, GPIO15: Thường dùng cho giao tiếp SPI.
* GPIO21, GPIO22: Mặc định cấu hình cho giao tiếp I2C (SDA và SCL).
* GPIO34 - GPIO39: Chỉ có thể sử dụng làm đầu vào digital hoặc analog.

Chân giao tiếp UART:

* + TX (GPIO1): Chân truyền dữ liệu UART.
  + RX (GPIO3): Chân nhận dữ liệu UART.

Chân ADC/DAC:

* **ADC (0–3.3V)**: Đo tín hiệu analog trên các chân GPIO34–GPIO39.
* **DAC**: Có 2 chân DAC (GPIO25 và GPIO26) cho phép xuất tín hiệu analog.
  + 1. Cảm biến nhiệt độ DHT11



Hình 2.2. Cảm biến nhiệt độ, độ ẩm

1. ***Giới thiệu***

Cảm biến DHT11 là loại cảm biến kỹ thuật số tích hợp cho phép đo nhiệt độ và độ ẩm, thường được sử dụng trong các ứng dụng IoT và hệ thống nhúng. Cảm biến này hoạt động dựa trên cảm biến nhiệt điện trở để đo nhiệt độ và cảm biến độ ẩm điện dung để đo độ ẩm không khí. DHT11 có khả năng cung cấp dữ liệu chính xác với chi phí thấp và dễ dàng kết nối với các vi điều khiển thông dụng như Raspberry Pi, Arduino, và ESP32. Cảm biến DHT11 được ứng dụng phổ biến trong hệ thống giám sát môi trường, hệ thống nhà thông minh, và các dự án nông nghiệp để giám sát điều kiện thời tiết.

1. ***Thông số kỹ thuật***

|  |  |
| --- | --- |
| **Thông số** | **Giá trị** |
| Điện áp hoạt động | 3.3V - 5.5V |
| Dải đo nhiệt độ | 0°C - 50°C với độ chính xác ±2°C |
| Dải đo độ ẩm | 20% - 90% RH với độ chính xác ±5% |
| Thời gian đáp ứng | ≤5 giây |
| Chu kỳ đo | Tối thiểu 2 giây |
| Kích thước | 15.5mm x 12mm x 5.5mm |

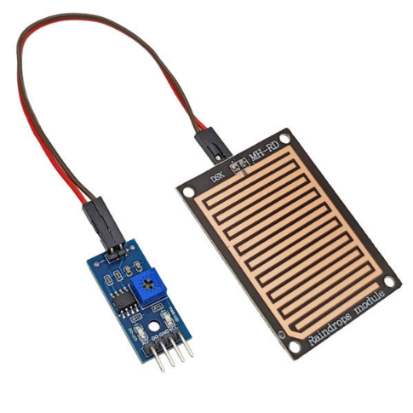
Bảng 2.1.2

1. ***Chức năng các chân***

| **Ký hiệu pin** | **Mô tả** |
| --- | --- |
| VCC | Pin nguồn, hoạt động ở 3.3V/5V |
| GND | Nối đất, điện áp 0V |
| Out | Chân dữ liệu số, truyền thông tin về nhiệt độ và độ ẩm |

Bảng 2.1.2

* + 1. Cảm biến mưa



Hình 2.2. cảm biến mưa

1. ***Giới thiệu***

Cảm biến mưa là loại cảm biến kỹ thuật số và analog được sử dụng để phát hiện sự hiện diện của nước mưa. Nó hoạt động dựa trên nguyên tắc thay đổi điện trở trên bề mặt mạch khi có nước rơi vào. Cảm biến mưa thường được ứng dụng trong các dự án IoT, hệ thống tự động hóa và nhà thông minh, như điều khiển mái che tự động, cảnh báo thời tiết hoặc hệ thống tưới cây. Với chi phí thấp, dễ sử dụng và khả năng tương thích cao với các vi điều khiển như Arduino và ESP32, cảm biến mưa là lựa chọn phổ biến trong nhiều dự án nhúng và nghiên cứu.

1. ***Thông số kỹ thuật***

|  |  |
| --- | --- |
| Thông số | Giá trị |
| Điện áp hoạt động | 3.3V - 5V |
| Dòng tiêu thụ | 0.1mA |
| Loại tín hiệu đầu ra | Digital và Analog |
| Kích thước module | 3cm x 5cm (bảng cảm biến) |
| Điện áp tín hiệu | TTL (0 - 5V) |
| Thời gian đáp ứng | ≤2 giây |
| Chất liệu bảng cảm biến | FR4 chống ăn mòn |

1. Chức năng các chân

|  |  |
| --- | --- |
| Ký hiệu pin | Mô tả |
| VCC | Pin nguồn, hoạt động ở 3.3V/5V |
| GND | Nối đất, điện áp 0V |
| D0 | Chân dữ liệu số, báo trạng thái mưa (1: không mưa, 0: có mưa) |
| A0 | Chân dữ liệu analog, cho giá trị điện áp tỷ lệ với lượng nước mưa |

* + 1. Cảm biến khí gas MQ2

a) Giới thiệu:

Cảm biến khí gas MQ2 là loại cảm biến chuyên dụng để phát hiện các loại khí dễ cháy. Nó hoạt động dựa trên nguyên lý thay đổi điện trở của phần tử bán dẫn khi tiếp xúc với khí mục tiêu. Với độ nhạy cao và thời gian đáp ứng nhanh, cảm biến MQ2 được ứng dụng rộng rãi trong các hệ thống phát hiện rò rỉ khí gas, báo cháy và các dự án IoT. Cảm biến này tương thích với các vi điều khiển như Arduino và ESP32.

1. Thông số kỹ thuật

|  |  |
| --- | --- |
| **Thông số** | **Giá trị** |
| Điện áp hoạt động | 3.3V - 5V |
| Dòng tiêu thụ | ~150mA |
| Loại tín hiệu đầu ra | Digital và Analog |
| Khả năng phát hiện khí | LPG, Methane, Butane, Hydrogen, Smoke |
| Thời gian đáp ứng | ≤20 giây |
| Nhiệt độ hoạt động | -20°C đến 50°C |
| Kích thước module | 32mm x 20mm |
| Thông số | Giá trị |
| Điện áp hoạt động | 3.3V - 5V |

1. Chức năng các chân

|  |  |
| --- | --- |
| Ký hiệu pin | Mô tả |
| VCC | Pin nguồn, hoạt động ở 3.3V/5V |
| GND | Nối đất, điện áp 0V |
| A0 | Chân dữ liệu analog, cung cấp giá trị tỷ lệ với nồng độ khí phát hiện |

* 1. Phần mềm cho hệ thống nhúng

a) Giới thiệu

Intergrated Development Environment (IDE) là công cụ được sử dụng để lập trình cho các board Arduino. Arduino IDE là một phần mềm mã nguồn mở chủ yếu được sử dụng để viết và biên dịch mã vào module Arduino.

A white circle with a plus and minus sign

Description automatically generated

Hình 2.2. Phần mềm arduino IDE

Đây là một phần mềm Arduino chính thức, giúp cho việc biên dịch mã trở nên dễ dàng mà ngay cả một người bình thường không có kiến thức kỹ thuật cũng có thể làm được.

Nó có các phiên bản cho các hệ điều hành như MAC, Windows, Linux và chạy trên nền tảng Java đi kèm với các chức năng và lệnh có sẵn đóng vai trò quan trọng để gỡ lỗi, chỉnh sửa và biên dịch mã trong môi trường.

Có rất nhiều các module Arduino như Arduino Uno, Arduino Mega, Arduino Leonardo, Arduino Micro và nhiều module khác. Mỗi module chứa một bộ vi điều khiển trên bo mạch được lập trình và chấp nhận thông tin dưới dạng mã.

Mã chính, còn được gọi là sketch, được tạo trên nền tảng IDE sẽ tạo ra một file Hex, sau đó được chuyển và tải lên trong bộ điều khiển trên bo.

Môi trường IDE chủ yếu chứa hai phần cơ bản: Trình chỉnh sửa và Trình biên dịch, phần đầu sử dụng để viết mã được yêu cầu và phần sau được sử dụng để biên dịch và tải mã lên module Arduino. Môi trường này hỗ trợ cả ngôn ngữ C và C ++.

b) Cách thức hoạt động

Khi người dùng viết mã và biên dịch, IDE sẽ tạo file Hex cho mã. File Hex là các file thập phân Hexa được Arduino hiểu và gửi đến bo mạch bằng cáp USB. Mỗi bo Arduino đều được tích hợp một bộ vi điều khiển, bộ vi điều khiển sẽ nhận file Hex và chạy theo mã được viết.

c) Cách tải

Bạn có thể tải phần mềm từ trang web chính thức của Arduino. Như đã nói trước đó, phần mềm có các phiên bản cho các hệ điều hành phổ biến như Linux, Windows và MAC, vì vậy hãy đảm bảo tải xuống đúng phiên bản phần mềm tương thích với hệ điều hành của bạn.

Nếu bạn muốn tải xuống phiên bản ứng dụng Windows, bạn phải có Windows 8.1 hoặc Windows 10, vì phiên bản ứng dụng không tương thích với Windows 7 hoặc phiên bản cũ hơn của hệ điều hành này.….

# NỘI DUNG THỰC HIỆN

* 1. Xây dựng kiến trúc của hệ thống
     1. Kiến trúc tổng quan của hệ thống

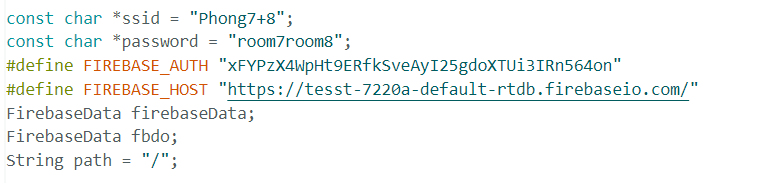
Kiến trúc của hệ thống là nền tảng để tích hợp và triển khai các thành phần phần cứng và phần mềm, đảm bảo hệ thống hoạt động hiệu quả và ổn định. Trong dự án này, hệ thống giám sát và điều khiển được thiết kế dựa trên các cảm biến thông minh (DHT11, cảm biến mưa, cảm biến khí gas MQ2) và module ESP32 Dev Kit làm trung tâm điều khiển. Dữ liệu thu thập sẽ được xử lý và truyền đến người dùng qua mạng không dây.

* + 1. Đặc tả yêu cầu hệ thống
       1. Các yêu cầu chức năng
* Người dùng có thể điều khiển các thiết bị trong nhà (đèn, quạt, rèm cửa, bơm nước, v.v.) thông qua giao diện trên website hoặc ứng dụng di động.
* Các lệnh điều khiển (bật/tắt, điều chỉnh cường độ, v.v.) phải được thực hiện trong thời gian thực.
  + - 1. Các yêu cầu phi chức năng
* Hệ thống phải hoạt động liên tục, không bị gián đoạn.
* Các thiết bị trong nhà thông minh phải luôn phản hồi đúng lệnh điều khiển.
  1. Thiết kế hệ thống.
     1. Thiết kế phần cứng hệ thống

Trong quá trình thiết kế phần cứng cho nhà thông minh điều khiển từ xa, bạn sẽ cần chuẩn bị và lắp ráp một số linh kiện cơ bản để tạo thành một hệ thống hoạt động. Dưới đây là một hướng dẫn chi tiết về việc mua linh kiện và lắp ráp chúng:

**Chuẩn bị Linh Kiện:**

* Board ESP32 dev kit module: Chọn một board ESP32 phổ biến, dễ dàng sử dụng.
* Module Servo: Servo SG90 để điều khiển hướng quay của cửa
* Led bật tắt ánh sáng
* Các cảm biến
* Cảm biến và Linh Kiện Khác (tùy chọn): Tùy thuộc vào yêu cầu cụ thể.
  + 1. Thiết kế phần mềm hệ thống
       1. Cài đặt chế độ wifi và firebase

Khu cấp nguồn điện cho ESP32 sẽ khởi động theo chế độ Station và khởi tạo firebase.

Hình 3.2.2. Tạo kết nối wifi và firebase

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

* 1. Kết quả đạt được của đề tài

Kết quả đạt được từ đề tài "Thiết kế hệ thống nhà thông minh" không chỉ đơn thuần là việc hoàn thành một nhiệm vụ học thuật mà còn là một hành trình sáng tạo và bổ ích. Qua quá trình thực hiện dự án, em đã thu nhận được nhiều kiến thức và kỹ năng quan trọng, cụ thể như sau:

* Làm việc với vi điều khiển ESP32:

Em đã học cách sử dụng và lập trình vi điều khiển ESP32 để làm trung tâm điều khiển trong hệ thống nhà thông minh. Bao gồm việc thiết lập môi trường phát triển (Arduino IDE) và viết code để ESP32 tương tác với các cảm biến và thiết bị đầu ra. Cách giao tiếp giữa vi điều khiển với các module chức năng.

* DHT11: Đo nhiệt độ và độ ẩm.
* Cảm biến mưa: Phát hiện và đo cường độ mưa.
* Cảm biến khí gas MQ2: Phát hiện rò rỉ khí gas hoặc khói.
* Việc này bao gồm việc đọc dữ liệu từ các cảm biến, xử lý và truyền dữ liệu lên giao diện điện thoại. Cách giao tiếp với Động cơ servo SG90

Thiết lập giao tiếp WiFi để kết nối ESP32 với mạng và máy chủ, truyền dữ liệu cảm biến và nhận lệnh điều khiển từ người dùng.

Phát triển giao thức truyền thông như HTTP hoặc MQTT để đảm bảo dữ liệu được gửi và nhận nhanh chóng, an toàn.

Dự án đã giúp em xây dựng nền tảng vững chắc cho sự phát triển trong lĩnh vực IoT và hệ thống tự động hóa. Đây là bước đầu để em tiến xa hơn trong việc nghiên cứu và ứng dụng các công nghệ tiên tiến vào cuộc sống thực tiễn.

* 1. Hạn chế của đề tài
* Hệ thống chỉ tập trung vào một số thiết bị và cảm biến cụ thể (DHT11, cảm biến mưa, MQ2, relay module). Điều này giới hạn khả năng mở rộng đối với các ứng dụng nhà thông minh phức tạp hơn, như điều khiển hệ thống âm thanh, chiếu sáng đa vùng, hay các thiết bị hiện đại khác.
* Mặc dù hệ thống sử dụng WiFi để truyền dữ liệu, vẫn có thể xảy ra độ trễ trong quá trình gửi và nhận tín hiệu, đặc biệt khi mạng WiFi không ổn định hoặc bị gián đoạn.
* Hệ thống chưa được thiết kế để hoạt động liên tục khi mất điện. Điều này làm giảm tính khả dụng và có thể gây gián đoạn trong các trường hợp khẩn cấp.
  1. Hướng phát triển của đề tài
* Tích hợp thêm cảm biến và thiết bị hiện đại để mở rộng phạm vi ứng dụng.
* Tích hợp nguồn dự phòng (UPS) để đảm bảo hệ thống hoạt động liên tục ngay cả khi mất điện.
* Sử dụng cảm biến có độ chính xác cao hơn để nâng cao chất lượng dữ liệu.
* Phát triển giao diện người dùng nâng cao, bao gồm hiển thị dữ liệu lịch sử, đồ thị, và thông báo thời gian thực.
* Tối ưu hóa kiến trúc phần mềm để hỗ trợ khả năng mở rộng linh hoạt hơn mà không ảnh hưởng đến hiệu suất hệ thống.

TÀI LIỆU THAM KHẢO

x

|  |  |
| --- | --- |
| [1] | Phạm Hữu Khang, *Lập trình ASP.NET 2.0*. Hà Nội, Việt Nam: Nhà xuất bản Lao Động Xã Hội, 2005. |
| [2] | Nguyễn Văn Quyết, *Giáo trình Công nghệ Web và ứng dụng*. Hưng Yên, Việt Nam: Khoa Công nghệ Thông tin - UTEHY, 2010. |

x

PHỤ LỤC

void initFirebase() {

  Firebase.begin(FIREBASE\_HOST, FIREBASE\_AUTH);

  Firebase.reconnectWiFi(true);

  if (!Firebase.beginStream(firebaseData, path)) {

    Serial.println("REASON: " + firebaseData.errorReason());

    Serial.println();

  }

  Serial.println("Kết nối Firebase thành công");

}

void setup() {

  Serial.begin(115200);

  Serial.println();

  Serial.print("Connecting to wifi: ");

  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL\_CONNECTED) {

    Serial.print(".");

  }

  Serial.println("");

  Serial.print("Connected, IP address: ");

  Serial.println(WiFi.localIP());  // In ra địa chỉ IP của ESP32

  initFirebase();                  // Kết nối Firebase

  initRFID();

  dht.begin();

  pinMode(den1, OUTPUT);

  pinMode(den2, OUTPUT);

  pinMode(den3, OUTPUT);

  pinMode(GAS\_SENSOR\_PIN, INPUT);

  myServo.attach(servoPin);

  myServo.write(0);

}

void sendGasData(){

  float gasValue = analogRead(GAS\_SENSOR\_PIN);

  if(Firebase.setFloat(firebaseData, "GAS", gasValue)){

    Serial.println(gasValue);

    Serial.println("Gas value data sent successfully");

  }

  else{

    Serial.println("Failed to send GAS data");

  }

}

// Hàm đọc và gửi dữ liệu nhiệt độ lên Firebase

void sendTemperatureData() {

  float temperature = dht.readTemperature();

  if (Firebase.setFloat(firebaseData, "TEMP", temperature)) {

    Serial.println(temperature);

    Serial.println("Temperature data sent successfully");

  } else {

    Serial.println("Failed to send temperature data");

  }

}

// Hàm đọc và gửi dữ liệu độ ẩm lên Firebase

void sendHumidityData() {

  float humidity = dht.readHumidity();

  if (Firebase.setFloat(firebaseData, "HUMINITY", humidity)) {

    Serial.println(humidity);

    Serial.println("Humidity data sent successfully");

  } else {

    Serial.println("Failed to send humidity data");

  }

}