TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

VÀ TRUYỀN THÔNG VIỆT – HÀN

**KHOA KỸ THUẬT MÁY TÍNH & ĐIỆN TỬ**



**BÁO CÁO**

**ĐỒ ÁN CHUYÊN NGÀNH I**

***Đề Tài*:** **XÂY DỰNG HỆ THỐNG ĐIỀU KHIỂN THIẾT BỊ ĐIỆN TỰ ĐỘNG**

**Sinh viên thực hiện     : Thái Văn Hòa**

**Lớp : 21IR**

**Niên khoá : 2021 - 2026**

**Giảng viên hướng dẫn : TS. Phan Thị Lan Anh**

***Đà Nẵng, ngày 01 tháng 05 năm 2024***

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

VÀ TRUYỀN THÔNG VIỆT – HÀN

**KHOA KỸ THUẬT MÁY TÍNH & ĐIỆN TỬ**



**BÁO CÁO**

**ĐỒ ÁN CHUYÊN NGÀNH I**

***Đề Tài*:** **XÂY DỰNG HỆ THỐNG ĐIỀU KHIỂN THIẾT BỊ ĐIỆN TỰ ĐỘNG**

**Sinh viên thực hiện     : Thái Văn Hòa**

**Lớp : 21IR**

**Niên khoá : 2021 - 2026**

**Giảng viên hướng dẫn : TS. Phan Thị Lan Anh**

***Đà Nẵng, ngày 01 tháng 05 năm 2024***

# NHẬN XÉT

**(Của Giảng viên hướng dẫn)**

....................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

Đà Nẵng,….. tháng … năm 2024

Giảng viên hướng dẫn

# LỜI CẢM ƠN

Lời đầu tiên cho phép em gửi lời cảm ơn tới các Thầy Cô giáo, các cán bộ công tác tại Trường ĐH Công Nghệ Thông Tin Và Truyền Thông Việt Hàn đã tạo mọi điều kiện giúp đỡ em trong thời gian xây dựng và hoàn thành báo cáo

Đặc biệt em xin bày tỏ lòng biết ơn sâu sắc tới cô **TS. Phan Thị Lan Anh**giảng viên hướng dẫn môn Đồ Án Chuyên Ngành I đã tận tình giúp đỡ, chỉ bảo về nghiệp vụ và trực tiếp hướng dẫn em trong suốt quá trình hoàn thành báo cáo này.

Tuy nhiên do thời gian có hạn và cùng với nhiều nguyên nhân khác, mặc dù em đã nỗ lực hết mình xong đồ án của em vẫn còn mắc phải những thiếu sót và hạn chế. Em rất mong nhận được sự thông cảm và chỉ bảo của các Thầy Cô cùng tất cả các bạn.

Em xin chân thành cảm ơn!

***Đà Nẵng, 01 tháng 05 năm 2024***

*Sinh viên thực hiện*

Thái Văn Hòa

**MỤC LỤC**

[NHẬN XÉT iii](#_Toc167341452)

[LỜI CẢM ƠN iv](#_Toc167341453)

[DANH MỤC HÌNH VẼ vii](#_Toc167341454)

[DANH MỤC CÁC TỪ VIẾT TẮT viii](#_Toc167341455)

[MỞ ĐẦU ix](#_Toc167341456)

[CHƯƠNG 1 - TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc167341457)

[1.1 Tổng quan 1](#_Toc167341458)

[1.1.1 Lý do chọn đề tài 1](#_Toc167341459)

[1.1.2 Mục tiêu và nội dung thực hiện đề tài 1](#_Toc167341460)

[1.1.3 Phương pháp nghiên cứu của đề tài 2](#_Toc167341461)

[1.2 Kết quả và ý nghĩa đề tài 2](#_Toc167341462)

[1.2.1. Kết quả 2](#_Toc167341463)

[1.2.2. Ý nghĩa đề tài 2](#_Toc167341464)

[1.3 Công cụ 3](#_Toc167341465)

[1.3.1. Phần mềm lấp trình 3](#_Toc167341466)

[1.3.2. Phần mềm vẽ mạch Altium 3](#_Toc167341467)

[1.3.3. Web Server 4](#_Toc167341468)

[CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT 5](#_Toc167341469)

[2.1 Giới thiệu chung về ESP32 5](#_Toc167341470)

[2.1.1 Cấu Trúc của ESP32 5](#_Toc167341471)

[2.1.2 Nguyên Lý Hoạt Động 6](#_Toc167341472)

[2.1.3 Ứng Dụng của ESP32 7](#_Toc167341473)

[2.1.4 Lợi Ích của Việc Sử Dụng ESP32 trong Hệ Thống Thông Minh 7](#_Toc167341474)

[2.2 Web server 8](#_Toc167341475)

[CHƯƠNG 3 – THIẾT KẾ & LẬP TRÌNH HỆ THỐNG 9](#_Toc167341476)

[3.1 Các Module và các linh kiện cần có 9](#_Toc167341477)

[3.1.1. ESP32 9](#_Toc167341478)

[3.1.2. Cảm biến nhiệt độ DS18B20 10](#_Toc167341479)

[3.1.3. Màn hình OLED I2C 0,96 inch 10](#_Toc167341480)

[3.1.4. Relay 5V 11](#_Toc167341481)

[3.2 Thiết kế phần cứng 13](#_Toc167341482)

[3.1.1. Vẽ schematic trên Altium 13](#_Toc167341483)

[3.1.2. Vẽ mạch PCB 14](#_Toc167341484)

[3.3 Các thư viện cần dùng 16](#_Toc167341485)

[3.1.1. WiFi.h 16](#_Toc167341486)

[3.1.2. AsyncTCP.h 16](#_Toc167341487)

[3.1.3. ESPAsyncWebServer.h 16](#_Toc167341488)

[3.1.4. ESPDash.h 16](#_Toc167341489)

[3.1.4. OneWire.h và DallasTemperature.h 16](#_Toc167341490)

[3.1.4. SPI.h và Wire.h 16](#_Toc167341491)

[3.1.4. Adafruit\_GFX.h và Adafruit\_SSD1306.h 16](#_Toc167341492)

[3.4 Mô phỏng thực tế hệ thống 17](#_Toc167341493)

[3.1.1. Sơ dồ mạch 17](#_Toc167341494)

[3.1.2. Mô phỏng testboard 17](#_Toc167341495)

[3.5 Sơ đồ và cách thức hoạt động của hệ thống 18](#_Toc167341496)

[3.1.1. Sơ đồ hoạt động của hệ thống 18](#_Toc167341497)

[3.1.2. Cách thức hoạt động của hệ thống 18](#_Toc167341498)

[3.6 Sản phẩm hoàn thiện 19](#_Toc167341499)

[3.1.1. Mạch PCB hoàn chỉnh 19](#_Toc167341500)

[3.1.2. Giao diện dashboard web server 20](#_Toc167341501)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 21](#_Toc167341502)

[TÀI LIỆU THAM KHẢO 22](#_Toc167341503)

[PHỤ LỤC 23](#_Toc167341504)

# DANH MỤC HÌNH VẼ

[Hình 1. 1: Phần mềm Arduno IDE 3](#_Toc167246005)

[Hình 1. 2: Phần mềm Altium Design 4](#_Toc167246006)

[Hình 2. 1: Datasheet ESP32 5](#_Toc167245351)

[Hình 2. 2: Các loại ESP32 7](#_Toc167245352)

[Hình 2. 3: Web Server 8](#_Toc167245353)

[Hình 3. 1: ESP32 9](#_Toc167247149)

[Hình 3. 2: Cảm biến nhiệt độ DS18B20 10](#_Toc167247150)

[Hình 3. 3: Màn hình OLED I2C 11](#_Toc167247151)

[Hình 3. 4: Relay 12](#_Toc167247152)

[Hình 3. 5: Schematic 13](#_Toc167247153)

[Hình 3. 6: Mạch PCB ở lớp Top layer 14](#_Toc167247154)

[Hình 3. 7: Mạch PCB ở lớp Bottom layer 15](#_Toc167247155)

[Hình 3. 8: Mạch PCB 3D 15](#_Toc167247156)

[Hình 3. 9: Sơ đồ mạch 17](#_Toc167247157)

[Hình 3. 10: Mô phỏng hệ thống trên testboard 17](#_Toc167247158)

[Hình 3. 11: Sơ đồ hoạt động hệ thống 18](#_Toc167247159)

[Hình 3. 12: Sản phẩm hoàn thiện 19](#_Toc167247160)

[Hình 3. 13: Giao diện web server trên laptop 20](#_Toc167247161)

[Hình 3. 14: Giao diện web server trên diện thoại 20](#_Toc167247162)

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Cụm từ | Viết tắt |
| 1 | Espressif System Platform | ESP |
| 2 | Internet of Think | IoT |

# MỞ ĐẦU

Trong thời đại số hóa ngày nay, việc sử dụng công nghệ để cải thiện chất lượng cuộc sống đã trở thành một xu hướng không thể tránh khỏi. Đặc biệt, việc tạo ra các hệ thống thông minh, nhờ vào sự phát triển của các công nghệ như ESP32 và Web Server, đã mở ra những cơ hội mới cho chúng ta để tạo ra những giải pháp tối ưu cho cuộc sống hàng ngày.

ESP32 là một chip có khả năng kết nối không dây và nhiều tính năng khác, đã trở thành một công cụ quan trọng trong việc phát triển các hệ thống thông minh. Bên cạnh đó, Web Server, một nền tảng cho phép người dùng tạo ra các ứng dụng IoT một cách dễ dàng và nhanh chóng, cũng đã đóng góp một phần quan trọng trong việc tạo ra những giải pháp IoT hiệu quả.

Đề tài sẽ tập trung vào việc khám phá sức mạnh của ESP32 và Web Server, cũng như cách chúng có thể được sử dụng để tạo ra các hệ thống thông minh. Chúng ta sẽ đi sâu vào việc tìm hiểu về cách hoạt động của ESP32 và Web Server, cũng như cách chúng có thể được lập trình để tạo ra các hệ thống thông minh.

Kết quả , chúng ta sẽ áp dụng những kiến thức đã học để thiết kế và lập trình một hệ thống thông minh cụ thể, minh họa cho khả năng và tiềm năng của ESP32 và Web Server trong việc tạo ra các giải pháp IoT hiệu quả. Thông qua đề tài này, ta sẽ có thể mở rộng hiểu biết về công nghệ và tạo ra những giải pháp tốt nhất cho cuộc sống của chúng ta.

# CHƯƠNG 1 - TỔNG QUAN VỀ ĐỀ TÀI

## Tổng quan

### Lý do chọn đề tài

Hệ thống thông minh là một xu hướng phát triển mạnh mẽ trong nhiều lĩnh vực của đời sống xã hội. Hệ thống thông minh có khả năng tự động hóa các nhiệm vụ, xử lý thông tin và đưa ra quyết định một cách thông minh.

ESP32 và Web Server là hai công nghệ hiện đại có thể được sử dụng để tạo ra các hệ thống thông minh với chi phí thấp và dễ triển khai. ESP32 là một vi điều khiển mạnh mẽ, có khả năng kết nối Wi-Fi và Bluetooth.

Đề tài này có tính ứng dụng cao. Các hệ thống thông minh sử dụng ESP32 và Web Server có thể được ứng dụng trong nhiều lĩnh vực khác nhau, bao gồm:

- Quản lý nhà thông minh

- An ninh thông minh

- Nông nghiệp thông minh

- Công nghiệp thông minh

Với những lý do trên, đề tài " Xây dựng hệ thống điều khiển thiết bị tự động " là một đề tài có ý nghĩa khoa học và thực tiễn. Đề tài này sẽ giúp sinh viên hiểu rõ hơn về hệ thống thông minh, ESP32 và Web Server. Ngoài ra, đề tài này cũng sẽ giúp sinh viên có cơ hội thực hành thiết kế và lập trình các hệ thống thông minh sử dụng ESP32 và Web Server.

### Mục tiêu và nội dung thực hiện đề tài

* Hiểu sâu hơn về cơ sở và hoạt động của hệ thống thông minh, ESP32 và Web Server.
* Thiết kế một hệ thống thông minh sử dụng ESP32 và Web Server.
* Lập trình và triển khai hệ thống thông minh này.
* Tiến hành thực nghiệm và đánh giá hiệu suất của hệ thống sử dụng ESP32 và Web Server.

### Phương pháp nghiên cứu của đề tài

* Nghiên cứu tài liệu: Nghiên cứu các tài liệu liên quan đến hệ thống thông minh, ESP32 và Web Server. Các tài liệu này có thể bao gồm sách, bài báo, tài liệu kỹ thuật,...
* Thực hành: Thực hành thiết kế và lập trình các hệ thống thông minh sử dụng ESP32 và Web Server. Sinh viên có thể sử dụng các phần mềm mô phỏng hoặc các thiết bị thực tế để thực hành.
* Áp dụng kiến thức đã học: Áp dụng các kiến thức đã học về hệ thống thông minh, ESP32 và Web Server để thiết kế và lập trình hệ thống thông minh.

## Kết quả và ý nghĩa đề tài

### 1.2.1. Kết quả

* Hiểu rõ về hệ thống thông minh, ESP32 và Web Server.
* Thiết kế được một hệ thống thông minh sử dụng ESP32 và Web Server.
* Lập trình được hệ thống thông minh sử dụng ESP32 và Web Server.
* Thực nghiệm và đánh giá được hệ thống thông minh sử dụng ESP32 và Web Server.

### 1.2.2. Ý nghĩa đề tài

* Giúp sinh viên hiểu rõ hơn về hệ thống thông minh, ESP32 và Web Server.
* Giúp sinh viên có cơ hội thực hành thiết kế và lập trình các hệ thống thông minh sử dụng ESP32 và Web Server.
* Đề tài có tính ứng dụng cao, có thể được sử dụng trong nhiều lĩnh vực khác nhau

## Công cụ

### 1.3.1. Phần mềm lấp trình

Arduino IDE: là một phần mềm với một mã nguồn mở, được sử dụng chủ yếu để viết và biên dịch mã vào module Arduino. Nó bao gồm phần cứng và phần mềm. Phần cứng chứa đến 300,000 board mạch được thiết kế sẵn với các cảm biến, linh kiện. Phần mềm giúp bạn có thể sử dụng các cảm biến, linh kiện ấy của Arduino một cách linh hoạt phù hợp với mục đích sử dụng.



Hình 1. 1: Phần mềm Arduno IDE

### 1.3.2. Phần mềm vẽ mạch Altium

Altium Designer: là một phần mềm thiết kế mạch điện tử tích hợp được phát triển bởi công ty Altium Limited có trụ sở tại Úc. Hiện nay, Altium là một trong những phần mềm vẽ mạch điện tử và thiết kế PCB được ưa chuộng ở Việt Nam.

Các tính năng của Altium Designer

- Giao diện thiết kế, quản lý và chỉnh sửa thân thiện, dễ dàng biên dịch, quản lý file, quản lý phiên bản cho các tài liệu thiết kế.

- Hỗ trợ mạnh mẽ cho việc thiết kế tự động, đi dây tự động theo thuật toán tối ưu, phân tích lắp ráp linh kiện. Hỗ trợ việc tìm các giải pháp thiết kế hoặc chỉnh sửa mạch, linh kiện, netlist có sẵn từ trước theo các tham số mới.

- Mở, xem và in các file thiết kế mạch dễ dàng với đầy đủ các thông tin linh kiện, netlist, dữ liệu bản vẽ, kích thước, số lượng…

- Hệ thống các thư viện linh kiện phong phú, chi tiết và hoàn chỉnh bao gồm tất cả các linh kiện nhúng, số, tương tự…

- Đặt và sửa đối tượng trên các lớp cơ khí, định nghĩa các luật thiết kế, tùy chỉnh các lớp mạch in, chuyển từ schematic sang PCB, đặt vị trí linh kiện trên PCB.

- Mô phỏng mạch PCB 3D, đem lại hình ảnh mạch điện trung thực trong không gian 3 chiều, hỗ trợ MCAD-ECAD, liên kết trực tiếp với mô hình STEP, kiểm tra khoảng cách cách điện, cấu hình cho cả 2D và 3D

- Hỗ trợ thiết kế PCB sang FPGA và ngược lại.

- Từ đó, chúng ta thấy Altium designer có nhiều điểm mạnh so với các phần mềm khác như đặt luật thiết kế, quản lý đề tài mô phỏng dễ dàng, giao diện thân thiện,…



Hình 1. 2: Phần mềm Altium Design

### 1.3.3. Web Server

Web Server là một phần mềm hoặc một thiết bị máy tính chạy phần mềm có nhiệm vụ phục vụ các trang web cho các máy tính khác qua mạng Internet hoặc mạng nội bộ. Nó hoạt động bằng cách nhận các yêu cầu HTTP từ các máy tính khác và gửi lại các trang web hoặc dữ liệu tương ứng. Điều này cho phép người dùng truy cập vào các trang web, ứng dụng web và các dịch vụ khác thông qua trình duyệt web của họ.

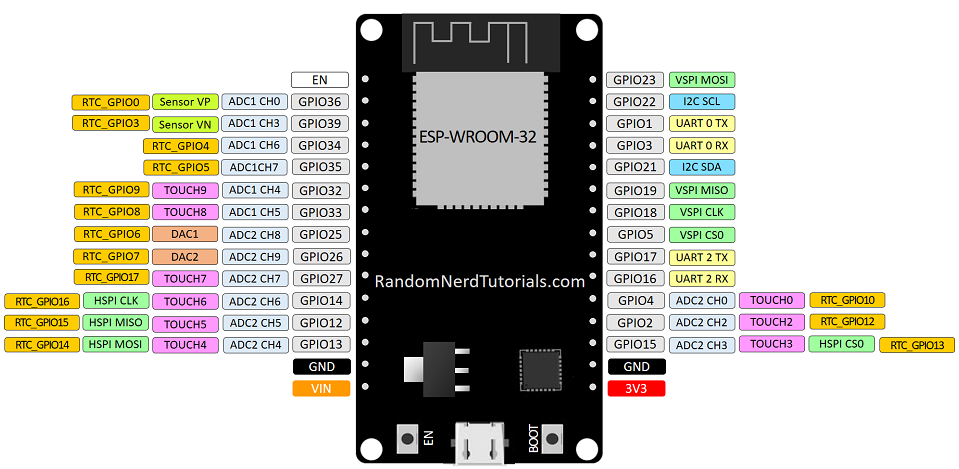
# CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT

## Giới thiệu chung về ESP32

### 2.1.1 Cấu Trúc của ESP32

ESP32 bao gồm nhiều thành phần chính, bao gồm:

* SoC (System on a Chip): Nền tảng tích hợp nhiều chức năng trên một chip, bao gồm vi xử lý, bộ nhớ, module WiFi và Bluetooth, GPIO (General Purpose Input/Output), các cổng giao tiếp khác như UART, SPI, I2C, v.v.
* Bộ xử lý kép (Dual-core): ESP32 có hai nhân xử lý, giúp chia công việc và tối ưu hiệu suất.
* WiFi và Bluetooth: Cung cấp khả năng kết nối không dây với mạng WiFi và các thiết bị Bluetooth khác



Hình 2. 1: Datasheet ESP32

Mô tả các chân ESP32

-Chân Input Only:

GPIO từ 34 đến 39 là GPI – chân chỉ đầu vào. Các chân này không có điện trở kéo lên hoặc kéo xuống bên trong. Chúng không thể được sử dụng làm đầu ra

-Chân tích hợp Flash trên ESP32:

GPIO 6 đến GPIO 11 dùng để kết nối Flash SPI

-Chân cảm biến điện dung

Các chân ESP32 này có chức năng như 1 nút nhấn cảm ứng, có thể phát hiện sự thay đổi về điện áp cảm ứng trên chân.

-Analog to Digital Converter (ADC)

ESP32 có các kênh đầu vào ADC 18 x 12 bit (trong khi ESP8266 chỉ có ADC 1x 10 bit

-Digital to Analog Converter (DAC)

Các kênh DAC 2 x 8 bit trên ESP32 để chuyển đổi tín hiệu kỹ thuật số thành đầu ra tín hiệu điện áp tương tự. Các kênh này chỉ có độ phân giải 8 bit, nghĩa là có giá trị từ 0 – 255 tương ứng với 0 – 3.3V

-Các chân thời gian thực RTC

Các chân có tác dụng đánh thức ESP32 khi trong chế độ Low Power Mode. Sử dụng như 1 chân ngắt ngoài.

-Chân PWM

ESP32 LED PWM có 16 kênh độc lập có thể được định cấu hình để tạo tín hiệu PWM với các thuộc tính khác nhau. Tất cả các chân có thể hoạt động như đầu ra đều có thể được sử dụng làm chân PWM (GPIO từ 34 đến 39 không thể tạo PWM).

-Chân I2C

ESP32 có hai kênh I2C và bất kỳ chân nào cũng có thể được đặt làm SDA hoặc SCL. Khi sử dụng ESP32 với Arduino IDE.

### 2.1.2 Nguyên Lý Hoạt Động

* WiFi: ESP32 sử dụng giao thức WiFi để kết nối với mạng không dây, cho phép truyền dữ liệu giữa thiết bị và các điểm truy cập WiFi.
* Bluetooth: ESP32 hỗ trợ Bluetooth Classic và Bluetooth Low Energy (BLE), mở rộng khả năng kết nối với các thiết bị khác như điện thoại di động, cảm biến, và thiết bị thông minh khác.
* GPIO và Cổng Giao Tiếp: GPIO cho phép ESP32 tương tác với nhiều loại thiết bị ngoại vi thông qua các giao tiếp như UART, SPI, I2C

### 2.1.3 Ứng Dụng của ESP32

* IoT (Internet of Things): ESP32 thường được sử dụng trong các ứng dụng IoT như đọc dữ liệu từ cảm biến và truyền tải chúng qua WiFi hoặc Bluetooth.
* Điều Khiển Thiết Bị: ESP32 có thể được sử dụng để điều khiển và giám sát thiết bị qua mạng không dây.
* Dự Án Mô Hình: Được ứng dụng rộng rãi trong các dự án mô hình và prototype nhờ tính linh hoạt và hiệu suất.



Hình 2. 2: Các loại ESP32

### 2.1.4 Lợi Ích của Việc Sử Dụng ESP32 trong Hệ Thống Thông Minh

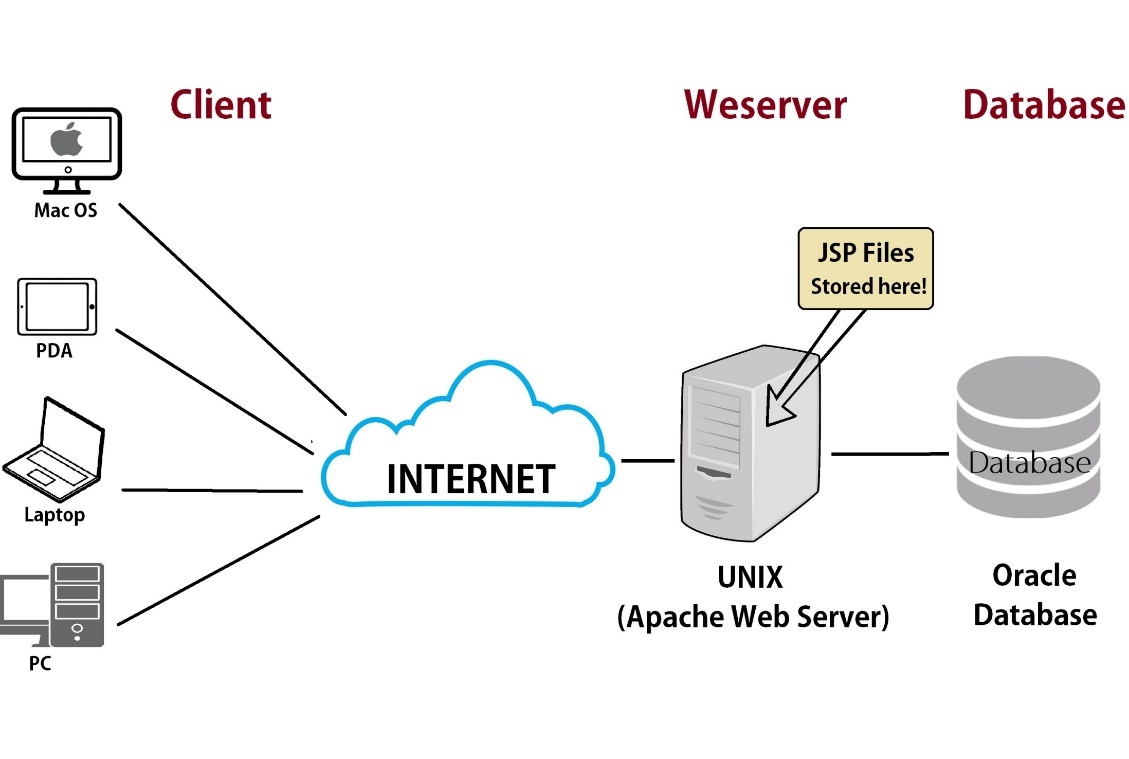
* Tiết Kiệm Năng Lượng: ESP32 được thiết kế để tiêu thụ ít năng lượng, phù hợp cho các ứng dụng di động và pin-kéo dự án.
* Kết Nối Đa Dạng: Với khả năng kết nối WiFi và Bluetooth, ESP32 có thể tương tác với nhiều loại thiết bị khác nhau.
* Hiệu Suất Cao: Bộ xử lý kép giúp tối ưu hiệu suất và xử lý đa nhiệm một cách hiệu quả.
* Phát Triển Nhanh Chóng: Sự hỗ trợ mạnh mẽ từ cộng đồng và sự linh hoạt của ESP32 giúp phát triển nhanh chóng các ứng dụng và dự án.

## Web server

Web server là một phần mềm hoặc một hệ thống máy tính được sử dụng để lưu trữ, quản lý và cung cấp các tệp và ứng dụng web cho các máy khách (clients) thông qua giao thức HTTP (Hypertext Transfer Protocol). Nhiệm vụ chính của một máy chủ web là xử lý các yêu cầu từ các trình duyệt web hoặc ứng dụng khác và trả về các tài nguyên web tương ứng, chẳng hạn như các trang HTML, hình ảnh, tệp tin CSS, JavaScript, và các tài nguyên đa phương tiện khác.

Khi một trình duyệt web như Chrome, Firefox hoặc Safari gửi yêu cầu đến một địa chỉ web, yêu cầu đó sẽ được gửi đến máy chủ web của trang đó. Máy chủ web sẽ xử lý yêu cầu, tìm kiếm các tệp và tài nguyên tương ứng, sau đó gửi lại các dữ liệu đó đến trình duyệt để hiển thị cho người dùng.

Có nhiều phần mềm máy chủ web phổ biến, bao gồm Apache HTTP Server, Nginx, Microsoft IIS (Internet Information Services), và LiteSpeed. Những máy chủ web này đều có tính năng và cấu hình riêng, nhưng đều thực hiện các chức năng cơ bản của máy chủ web: lưu trữ và phục vụ các tệp và ứng dụng web.



Hình 2. 3: Web Server

# CHƯƠNG 3 – THIẾT KẾ & LẬP TRÌNH HỆ THỐNG

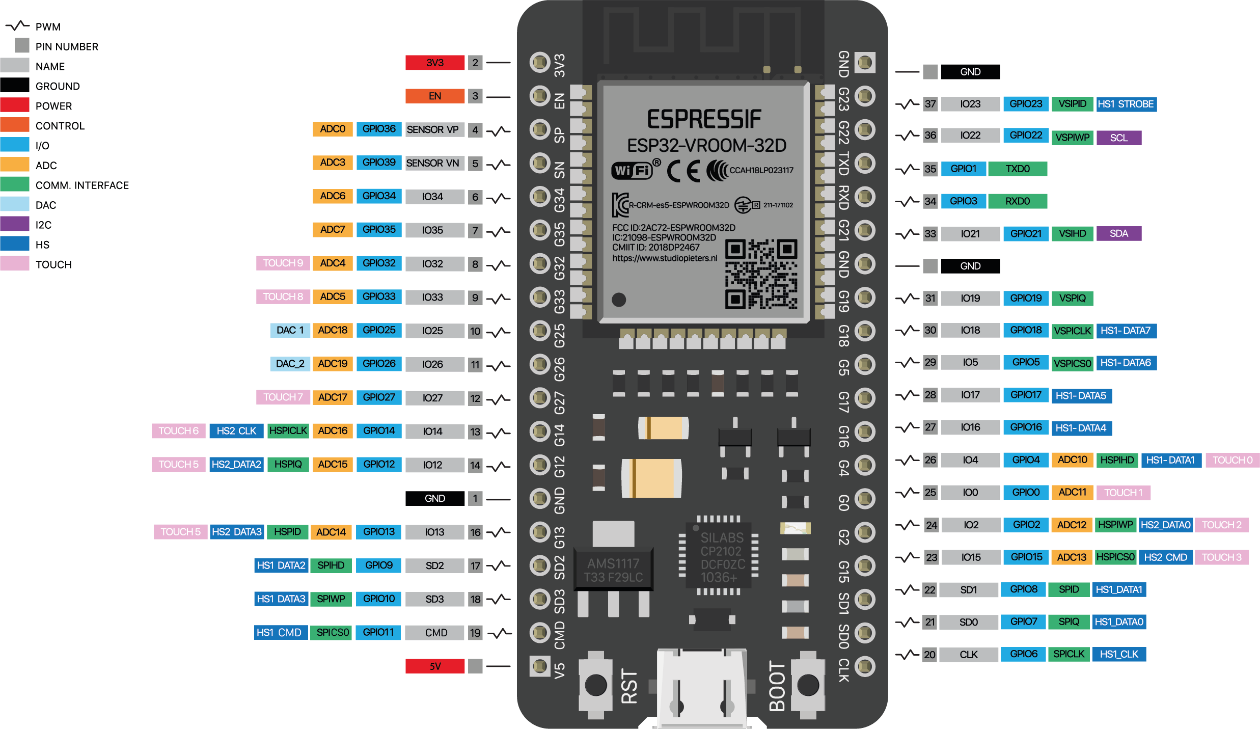
* 1. Các Module và các linh kiện cần có

### 3.1.1. ESP32

ESP32 là một bộ vi điều khiển thuộc danh mục vi điều khiển trên chip công suất thấp và tiết kiệm chi phí. Hầu hết tất cả các biến thể ESP32 đều tích hợp Bluetooth và Wi-Fi chế độ kép, làm cho nó có tính linh hoạt cao, mạnh mẽ và đáng tin cậy cho nhiều ứng dụng.

Nó là sự kế thừa của vi điều khiển NodeMCU ESP8266 phổ biến và cung cấp hiệu suất và tính năng tốt hơn. Bộ vi điều khiển ESP32 được sản xuất bởi Espressif Systems và được sử dụng rộng rãi trong nhiều ứng dụng khác nhau như IoT, robot và tự động hóa.

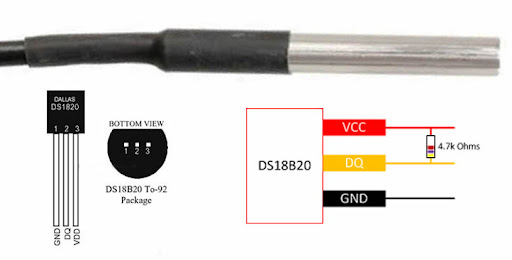
ESP32 cũng được thiết kế để tiêu thụ điện năng thấp, lý tưởng cho các ứng dụng chạy bằng pin. Nó có hệ thống quản lý năng lượng cho phép nó hoạt động ở chế độ ngủ và chỉ thức dậy khi cần thiết, điều này có thể kéo dài tuổi thọ pin rất nhiều.



Hình 3. 1: ESP32

### 3.1.2. Cảm biến nhiệt độ DS18B20

DS18B20 là cảm biến nhiệt độ có giao thức 1-wire được tích hợp tối đa. Sử dụng rộng rãi để đo nhiệt độ trong các môi trường như trong dung dịch hóa chất, hầm mỏ hoặc đất, v.v.



Hình 3. 2: Cảm biến nhiệt độ DS18B20

### 3.1.3. Màn hình OLED I2C 0,96 inch

màn hình Oled 0.96inch sử dụng giao tiếp I2C cho chất lượng đường truyền ổn định và rất dễ giao tiếp chỉ với 2 chân GPIO.

Sơ đồ chân:

- VCC: là nguồn cấp cho màn hình có thể là 3.3V hoặc 5V

- GND: chân nối đất

- SCL: chân xung clock

- SDA: chân dữ liệu nối tiếp

Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm

Mô tả được tạo tự động

Hình 3. 3: Màn hình OLED I2C

### 3.1.4. Relay 5V

Rơle 5v là một công tắc tự động thường được sử dụng trong mạch điều khiển tự động và để điều khiển dòng điện cao bằng tín hiệu dòng điện thấp. Điện áp đầu vào của tín hiệu rơle nằm trong khoảng từ 0 đến 5V.

**Cấu hình chân rơle 5V**

Cấu hình chân của rơle 5V được hiển thị bên dưới. Rơle này bao gồm 5 chân trong đó mỗi chân và chức năng của nó được hiển thị bên dưới.

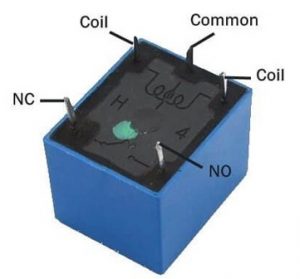
Pin1 (Cuối 1): Dùng để kích hoạt rơle; thông thường, một đầu chân này được kết nối với 5Volts trong khi đầu còn lại được kết nối với mặt đất.

Chân 2 (Cuối 2): Chân này dùng để kích hoạt Rơle.

Chân 3 (Chung (COM)): Chân này được kết nối với cực chính của Tải để làm cho nó hoạt động.

Chân 4 (Thường đóng (NC)): Đầu cuối thứ hai của tải được kết nối với các chân NC/NO. Nếu chân này được nối với tải thì nó sẽ BẬT trước công tắc.

Chân 5 (Thường mở (NO)): Nếu cực thứ hai của tải được liên kết với chân NO thì tải sẽ bị tắt trước khi chuyển mạch.



Hình 3. 4: Relay

**Đặc trưng**

Các tính năng của rơle 5V bao gồm:

• Điện áp bình thường là 5V DC

• Dòng điện bình thường là 70mA

• Dòng tải AC tối đa là 10A ở 250VAC hoặc 125V AC

• Dòng tải DC Max là 10A ở 30V DC hoặc 28V DC

• Nó bao gồm 5 chân & được thiết kế bằng chất liệu nhựa

• Thời gian hoạt động là 10msec

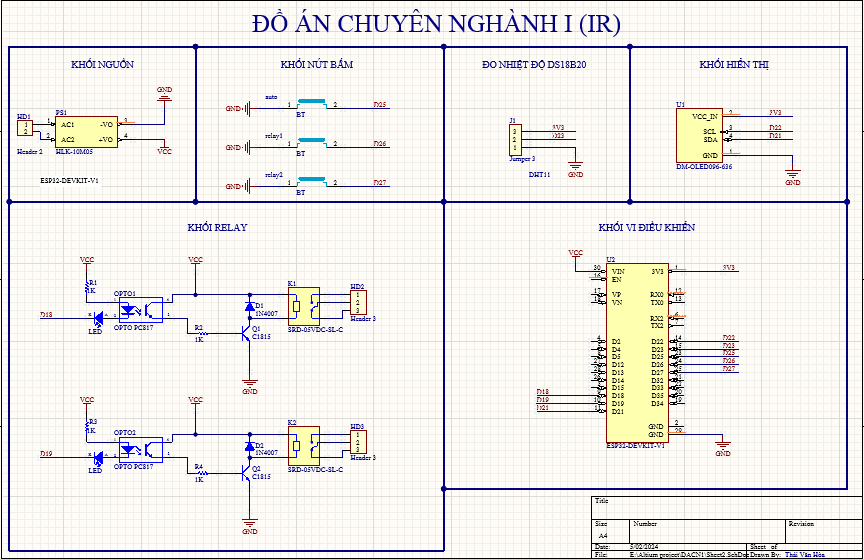
• Thời gian phát hành là 5msec

• Chuyển đổi tối đa là 300 hoạt động mỗi phút

* 1. Thiết kế phần cứng

### 3.2.1. Vẽ schematic trên Altium

* Khối điều khiển :Điều khiển toàn bộ hệ thống, xử lý lệnh và tín hiệu.
* Khối relay:Đóng/mở mạch điện theo tín hiệu điều khiển, cách ly tín hiệu điều khiển khỏi mạch điện cao áp.
* Khối hiển thị :Hiển thị thông tin cho người dùng (LCD, LED,).
* Khối nguồn :Cung cấp điện áp ổn định cho toàn hệ thống.
* Nút bấm:Tương tác và điều khiển hệ thống qua các nút nhấn.
* Khối đo nhiệt độ:Đo và giám sát nhiệt độ, gửi tín hiệu nhiệt độ đến khối điều khiển.



Hình 3. 5: Schematic

### 3.2.2. Vẽ mạch PCB

A red circuit board with blue and yellow text

Description automatically generated

Hình 3. 6: Mạch PCB ở lớp Top layer

* Lớp Top Layer được sử dụng để định vị và đặt các linh kiện trên bề mặt của mạch PCB. Các linh kiện bao gồm vi điều khiển, resistor, capacitor, ICs và các linh kiện khác.
* Đặt và Định Vị Linh Kiện: Lớp Top Layer được sử dụng để đặt và định vị các linh kiện trên bề mặt của mạch PCB. Điều này bao gồm việc xác định vị trí và hướng của các linh kiện, bố trí chúng sao cho phù hợp với yêu cầu kỹ thuật và thiết kế.
* Tạo Vias và Pads: Lớp Top Layer cũng chứa các pads (đế linh kiện) và vias (lỗ thông) để tạo kết nối giữa các linh kiện trên cùng một lớp và giữa các lớp khác nhau của mạch PCB.
* Tạo Vùng Mạch và Trở Kháng: Trên lớp này, vùng mạch (copper pour) được tạo ra để cung cấp đường dẫn điện hoặc để kiểm soát trở kháng của dây dẫn. Các vùng mạch cũng có thể được sử dụng để tạo ra một môi trường đất tốt hơn cho mạch.

A blue circuit board with red and white text

Description automatically generated

Hình 3. 7: Mạch PCB ở lớp Bottom layer

* Lớp bottom layer dùng để hàn các linh kiện điện tử vào mạch PCB.

A green circuit board with blue and black objects

Description automatically generated

Hình 3. 8: Mạch PCB 3D

* Dùng để hiển thị các bộ phận bố cục của mạch và các linh kiện cần có để dễ hình dung và lắp ráp mạch .
  1. Các thư viện cần dùng

### 3.3.1. WiFi.h

Thư viện này cho phép ESP32 kết nối vào mạng Wi-Fi, cho phép truy cập Internet hoặc giao tiếp với các thiết bị khác trong mạng.

### 3.3.2. AsyncTCP.h

Thư viện này cung cấp các công cụ cho việc giao tiếp TCP không đồng bộ (asynchronous) trên ESP32, cho phép thiết lập và quản lý các kết nối TCP không đồng bộ.

### 3.3.3. ESPAsyncWebServer.h

Thư viện này giúp tạo và quản lý một máy chủ web trên ESP32, cho phép xử lý các yêu cầu HTTP từ các trình duyệt web và phản hồi lại các trang web hoặc dữ liệu tương ứng.

### 3.3.4. ESPDash.h

Là một thư viện đồ họa để tạo ra giao diện người dùng (UI) dựa trên web cho các dự án IoT sử dụng ESP32. Nó giúp tạo ra các trang web tương tác mà người dùng có thể điều khiển các thiết bị hoặc hiển thị dữ liệu.

### 3.3.5. OneWire.h và DallasTemperature.h

Thư viện này được sử dụng cùng nhau để đọc dữ liệu từ cảm biến nhiệt độ DS18B20, một cảm biến nhiệt độ kỹ thuật số.

### 3.3.6. SPI.h và Wire.h

Thư viện giao tiếp với các thiết bị ngoại vi thông qua giao thức SPI (Serial Peripheral Interface) và I2C (Inter-Integrated Circuit).

### 3.3.7. Adafruit\_GFX.h và Adafruit\_SSD1306.h

Thư viện này được sử dụng để điều khiển màn hình OLED sử dụng chip điều khiển SSD1306, cho phép hiển thị đồ họa và văn bản trên màn hình.

* 1. Mô phỏng thực tế hệ thống

### 3.4.1. Sơ dồ mạch

A circuit board with wires

Description automatically generated

Hình 3. 9: Sơ đồ mạch

### 3.4.2. Mô phỏng testboard

**A circuit board with wires and a wire

Description automatically generated**

Hình 3. 10: Mô phỏng hệ thống trên testboard

* 1. Sơ đồ và cách thức hoạt động của hệ thống

### 3.5.1. Sơ đồ hoạt động của hệ thống

A diagram of a process

Description automatically generated

Hình 3. 11: Sơ đồ hoạt động hệ thống

### 3.5.2. Cách thức hoạt động của hệ thống

Hệ thống dùng ESP32 đọc nhiệt độ từ cảm biến nhiệt độ DS18B20 và hiển thị trên màn hình OLED . sử dụng nhiệt độ đọc từ cảm biến để điều khiển relay máy sưởi và máy lạnh ở chế độ tự động hoặc thủ công. Các giá trị điểm đặt và độ trễ có thể được điều chỉnh từ xa thông qua giao diện web.

* Ở chế độ tự động , chương trình sẽ bật rơle sưởi nếu nhiệt độ thấp hơn điểm đặt trừ đi giá trị trễ hoặc rơle làm mát nếu nhiệt độ cao hơn điểm đặt cộng với giá trị trễ .
* Ở chế độ thủ công , chương trình sẽ bật rơle sưởi hoặc làm mát dựa trên các nút trên bảng điều khiển .
  1. Sản phẩm hoàn thiện

### 3.6.1. Mạch PCB hoàn chỉnh

**A circuit board with wires and a wire

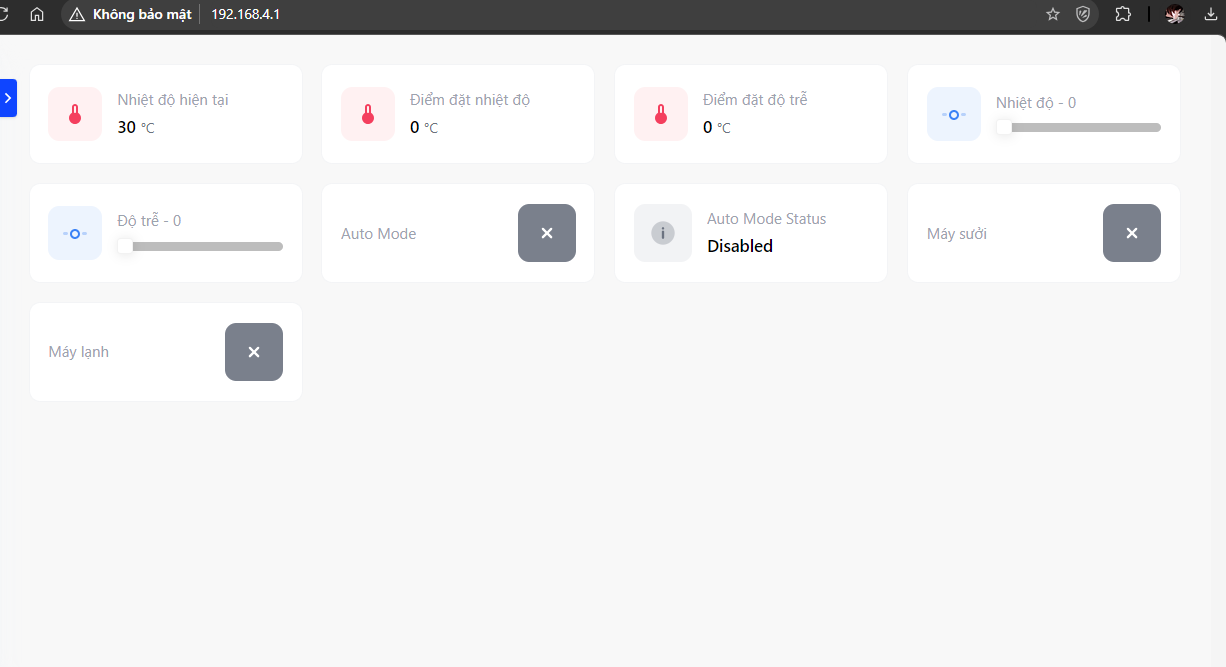
Description automatically generated**

A circuit board with wires and a black screen

Description automatically generated

Hình 3. 12: Sản phẩm hoàn thiện

### 3.6.2. Giao diện dashboard web server



Hình 3. 13: Giao diện web server trên laptop

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

Hình 3. 14: Giao diện web server trên diện thoại

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

**Kết quả đạt được**

* Dự án đã đạt được mục tiêu ban đầu của việc xây dựng một hệ thống thông minh sử dụng ESP32 và Web Server. Điều này bao gồm việc kiểm soát nhiệt độ tự động, tích hợp công tắc và màn hình OLED để theo dõi thông tin.
* Quá trình lập trình và triển khai cho dự án đã là cơ hội tuyệt vời để học và áp dụng kiến thức về lập trình ESP32, sử dụng thư viện Web Server, và tương tác với các linh kiện IoT khác nhau.
* Mã nguồn : https://github.com/ThaiVanHoa/Arduno\_VKU/tree/main/ESP32webserver

**Hạn chế**

* Gặp phải một số thách thức như ổn định kết nối Web Server và xử lý sự cố của cảm biến. Đã giải quyết bằng cách thực hiện các kiểm tra liên tục và xử lý lỗi thông qua việc sử dụng các hàm chuyển đổi trạng thái.

**Hướng phát triển**

* Tích Hợp Nhiều Cảm Biến:
* Mở rộng chức năng cảm biến bằng cách tích hợp các cảm biến khác như cảm biến khí CO2, ánh sáng, hoặc cảm biến chất lượng không khí để có cái nhìn toàn diện về môi trường.
* Tối Ưu Hóa Giao Diện Người Dùng:
* Nâng cao trải nghiệm người dùng bằng cách phát triển một giao diện người dùng trực quan và dễ sử dụng trên ứng dụng Web Server, bao gồm biểu đồ, thông báo và chức năng điều khiển nhanh.
* Bảo Mật Tăng Cường:
* Nâng cấp bảo mật thông qua mã hóa dữ liệu truyền và lưu trữ, xác thực hai yếu tố, và giải pháp bảo vệ khác để ngăn chặn truy cập trái phép.
* Tương Tác Đa Thiết Bị:

# TÀI LIỆU THAM KHẢO

[1] https://docs.espressif.com/

[2] https://docs.wokwi.com/

[3] https://www.arduino.cc/reference/en/libraries/

# PHỤ LỤC

Code hệ thống

#include <Arduino.h>

#include <WiFi.h>

#include <AsyncTCP.h>

#include <ESPAsyncWebServer.h>

#include <ESPDash.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include <SPI.h>

#include <Wire.h>

#include <Adafruit\_GFX.h>

#include <Adafruit\_SSD1306.h>

// Định nghĩa kích thước của màn hình OLED

#define SCREEN\_WIDTH 128 // Chiều rộng màn hình OLED, tính bằng pixel

#define SCREEN\_HEIGHT 64 // Chiều cao màn hình OLED, tính bằng pixel

#define OLED\_RESET -1 // Chân reset (hoặc -1 nếu chia sẻ chân reset của Arduino)

#define SCREEN\_ADDRESS 0x3C // Địa chỉ I2C của màn hình OLED

Adafruit\_SSD1306 display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET);

// Thông tin Wi-Fi

const char\* ssid = "DACN1"; // Tên Wi-Fi (SSID)

const char\* password = "11111111"; // Mật khẩu Wi-Fi

// Giá trị đặt và độ trễ (tính bằng độ C)

int setpoint = 0;

int hysteresis = 0;

// Số chân cho relay điều khiển sưởi và làm mát

const int HEATING\_PIN = 18; // Relay1

const int COOLING\_PIN = 19; // Relay2

#define ONE\_WIRE\_BUS 23 // Chân cho cảm biến nhiệt độ OneWire

OneWire oneWire(ONE\_WIRE\_BUS);

DallasTemperature sensors(&oneWire);

// Chế độ điều khiển (tự động hoặc thủ công)

bool automaticMode = 0;

bool heaterbtn = 0;

bool coolerbtn = 0;

// Khởi tạo web server

AsyncWebServer server(80);

// Kết nối ESP-DASH với AsyncWebServer

ESPDash dashboard(&server);

// Các thẻ trên Dashboard

Card temperature(&dashboard, TEMPERATURE\_CARD, "Nhiệt độ hiện tại", "°C");

Card setTemp(&dashboard, SLIDER\_CARD, "Nhiệt độ", "", 0, 50);

Card setHyss(&dashboard, SLIDER\_CARD, "Độ trễ", "", 0, 5);

Card autoMode(&dashboard, BUTTON\_CARD, "Auto Mode");

Card ModeStatus(&dashboard, STATUS\_CARD, "Trạng thái Auto Mode");

Card Heater(&dashboard, BUTTON\_CARD, "Máy sưởi");

Card Cooler(&dashboard, BUTTON\_CARD, "Máy lạnh");

void setup() {

  Serial.begin(115200);

  // Khởi tạo relay và cảm biến nhiệt độ

  pinMode(HEATING\_PIN, OUTPUT);

  pinMode(COOLING\_PIN, OUTPUT);

  // Khởi tạo màn hình OLED

  if (!display.begin(SSD1306\_SWITCHCAPVCC, SCREEN\_ADDRESS)) {

    Serial.println(F("Khởi tạo SSD1306 thất bại"));

    for (;;); // Dừng lại, lặp vô hạn

  }

  display.clearDisplay();

  // Thiết lập Access Point

  WiFi.mode(WIFI\_AP);

  WiFi.softAPConfig(IPAddress(192, 168, 4, 1), IPAddress(192, 168, 4, 1), IPAddress(255, 255, 255, 0));

  WiFi.softAP(ssid, password);

  Serial.print("Địa chỉ IP: ");

  Serial.println(WiFi.softAPIP());

  display.setTextSize(1);

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(2, 2);

  display.println("IP: ");

  display.setCursor(22, 2);

  display.println(WiFi.softAPIP());

  display.display();

  // Tắt các relay ban đầu

  digitalWrite(HEATING\_PIN, HIGH);

  Serial.println("Máy sưởi TẮT");

  digitalWrite(COOLING\_PIN, HIGH);

  Serial.println("Máy lạnh TẮT");

  // Khởi động WebServer

  server.begin();

  // Khởi động thư viện cảm biến nhiệt độ

  sensors.begin();

}

void loop() {

  // Gửi lệnh để lấy nhiệt độ

  sensors.requestTemperatures();

  float currentTemp = sensors.getTempCByIndex(0);

  // Thiết lập xác thực cho Dashboard

  dashboard.setAuthentication("admin", "");

  // Hiển thị thông tin trên màn hình OLED

  display.clearDisplay();

  display.setTextSize(1);

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(2, 2);

  display.println("IP: ");

  display.setCursor(22, 2);

  display.println(WiFi.softAPIP());

  display.display();

  display.setTextSize(3);

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(0, 17);

  display.println(currentTemp);

  display.println(" ");

  display.drawRect(90, 17, 5, 5, WHITE); // Ký hiệu độ (°)

  display.setCursor(97, 17);

  display.println("C");

  display.display();

  // Cập nhật giá trị thẻ trên Dashboard

  temperature.update((int)(currentTemp));

  // Callback cho thẻ nhiệt độ

  setTemp.attachCallback([&](int value) {

    Serial.println("[setTemp] Slider Callback Triggered: " + String(value));

    setpoint = value;

    setTemp.update(value);

    dashboard.sendUpdates();

  });

  // Callback cho thẻ độ trễ

  setHyss.attachCallback([&](int value) {

    Serial.println("[setHyss] Slider Callback Triggered: " + String(value));

    hysteresis = value;

    setHyss.update(value);

    dashboard.sendUpdates();

  });

  Serial.println(hysteresis);

  display.setTextSize(1);

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(0, 55);

  display.print("Hys: ");

  display.print(hysteresis);

  display.display();

  Serial.println(setpoint);

  display.setTextSize(1);

  display.setTextColor(SSD1306\_WHITE);

  display.setCursor(70, 55);

  display.print("Temp: ");

  display.print(setpoint);

  display.display();

  // Callback cho chế độ tự động

  autoMode.attachCallback([&](int value) {

    Serial.println("[autoMode] Button Callback Triggered: " + String((value == 1) ? "true" : "false"));

    automaticMode = value;

    autoMode.update(value);

    dashboard.sendUpdates();

  });

  // Hiển thị trạng thái chế độ tự động trên Dashboard

  if (automaticMode == 1) {

    Serial.println("Chế độ tự động bật");

    ModeStatus.update("Enabled", "success");

    dashboard.sendUpdates();

  } else {

    Serial.println("Chế độ tự động tắt");

    ModeStatus.update("Disabled", "danger");

    dashboard.sendUpdates();

  }

  // Callback cho nút máy sưởi

  Heater.attachCallback([&](int value) {

    Serial.println("[Heater] Button Callback Triggered: " + String((value == 1) ? "true" : "false"));

    heaterbtn = value;

    if (automaticMode == 0 && heaterbtn == 1) {

      digitalWrite(HEATING\_PIN, LOW);

      Serial.println("Máy sưởi BẬT");

    } else if (automaticMode == 0 && heaterbtn == 0) {

      digitalWrite(HEATING\_PIN, HIGH);

      Serial.println("Máy sưởi TẮT");

    }

    Heater.update(value);

    dashboard.sendUpdates();

  });

  // Callback cho nút máy lạnh

  Cooler.attachCallback([&](int value) {

    Serial.println("[Cooler] Button Callback Triggered: " + String((value == 1) ? "true" : "false"));

    coolerbtn = value;

    if (automaticMode == 0 && coolerbtn == 1) {

      digitalWrite(COOLING\_PIN, LOW);

      Serial.println("Máy lạnh BẬT");

    } else if (automaticMode == 0 && coolerbtn == 0) {

      digitalWrite(COOLING\_PIN, HIGH);

      Serial.println("Máy lạnh TẮT");

    }

    Cooler.update(value);

    dashboard.sendUpdates();

  });

  // Điều khiển relay sưởi và làm mát trong chế độ tự động

  if (automaticMode == 1) {

    if (currentTemp > setpoint + hysteresis) {

      digitalWrite(HEATING\_PIN, HIGH);

      Serial.println("Máy sưởi TẮT");

      digitalWrite(COOLING\_PIN, LOW);

      Serial.println("Máy lạnh BẬT");

    } else if (currentTemp < setpoint - hysteresis) {

      digitalWrite(HEATING\_PIN, LOW);

      Serial.println("Máy sưởi BẬT");

      digitalWrite(COOLING\_PIN, HIGH);

      Serial.println("Máy lạnh TẮT");

    } else {

      digitalWrite(HEATING\_PIN, HIGH);

      Serial.println("Máy sưởi TẮT");

      digitalWrite(COOLING\_PIN, HIGH);

      Serial.println("Máy lạnh TẮT");

    }

  } else if (automaticMode == 0) {

    if (heaterbtn == 1) {

      digitalWrite(HEATING\_PIN, LOW);

      Serial.println("Máy sưởi BẬT");

    } else {

      digitalWrite(HEATING\_PIN, HIGH);

      Serial.println("Máy sưởi TẮT");

    }

    if (coolerbtn == 1) {

      digitalWrite(COOLING\_PIN, LOW);

      Serial.println("Máy lạnh BẬT");

    } else {

      digitalWrite(COOLING\_PIN, HIGH);

      Serial.println("Máy lạnh TẮT");

    }

  }

  // Gửi cập nhật đến Dashboard (theo thời gian thực)

  dashboard.sendUpdates();

  // Dùng delay chỉ để minh họa, trong dự án thực tế nên thay bằng 'millis'

  delay(3000);

}