

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN-ĐIỆN TỬ - BỘ MÔN VIỄN THÔNG



LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

**XÂY DỰNG MÔ HÌNH
ĐIỀU KHIỂN THIẾT BỊ
TRONG NHÀ THÔNG MINH**

HỘI ĐỒNG: KHOA ĐIỆN – ĐIỆN TỬ

SVTH: VÕ HÀ THÀNH (1413596)

GVHD: VƯƠNG PHÁT

TP.HCM 2020

Số: _____/BKĐT
Khoa: **Điện – Điện tử**
Bộ Môn: **Viễn Thông**

NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

- Họ và tên: Võ Hà Thành MSSV: 1413596
- Ngành: Điện – Điện tử Chuyên ngành: Kỹ thuật Điện tử - Truyền thông
- Đề tài: Xây dựng mô hình điều khiển thiết bị trong nhà thông minh.
- Nhiệm vụ:
 - Tìm hiểu về các sản phẩm đã có trên thị trường liên quan đến điều khiển thiết bị được ứng dụng trong nhà thông minh.
 - Tìm hiểu và lựa chọn các linh kiện phù hợp cho đề tài và phần mềm thiết kế mạch Altium Designer.
 - Tìm hiểu về các giao tiếp mạng, ngôn ngữ lập trình C++, ngôn ngữ lập trình và thiết kế giao diện website HTML, CSS, Javascript.
 - Thiết kế và thi công mô hình điều khiển thiết bị qua mạng internet hoàn chỉnh.
 - Thiết kế và xây dựng mạch phần cứng nhỏ gọn đáp ứng được như cầu sử dụng.
 - Thiết kế và xây dựng firmware cho vi điều khiển chính trong mạch điện.
 - Thiết kế và xây dựng giao diện người dùng để điều khiển thiết bị từ xa trên máy tính và điện thoại.
- Ngày giao nhiệm vụ luận văn: 30/03/2020
- Ngày hoàn thành nhiệm vụ: 19/07/2020
- Họ và tên người hướng dẫn: Vương Phát Phân hướng dẫn: Toàn bộ

Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

TP.HCM, ngày tháng năm 2020

CHỦ NHIỆM BỘ MÔN

NGƯỜI HƯỚNG DẪN CHÍNH

PHẦN DÀNH CHO KHOA, BỘ MÔN:

Người duyệt (chấm sơ bộ):.....
Đơn vị:.....
Ngày bảo vệ:
Điểm tổng kết:
Nơi lưu trữ luận văn:

LỜI CẢM ƠN

Lời đầu tiên, em xin gửi lời cảm ơn chân thành đến thầy Vương Phát, người thầy hướng dẫn em xuyên suốt luận văn này. Trong quá trình thực hiện luận văn, em đã nhận được sự giúp đỡ của thầy, được thầy hướng dẫn rất nhiệt tình và giúp em có cái nhìn đa chiều và định hướng sản phẩm tốt nhất, giúp em hoàn thiện được tư duy và lối suy nghĩ đúng đắn khi giải quyết vấn đề. Em cũng xin gửi lên cảm ơn đến ba mẹ cũng như trường Đại học Bách Khoa TP HCM đã động viên và hỗ trợ em trong quá trình hình học tập và phấn đấu có phần lâu dài này. Ngoài ra em cũng xin cảm ơn đến các bạn đã và đang làm việc ở Hshop và những bạn khác đã hỗ trợ hết mình về kiến thức cũng như tạo các điều kiện thuận lợi để em có thể hoàn thành được luận văn này.

Do kiến thức có hạn nên em rất mong nhận được được sự góp ý của các thầy và các bạn để giúp em hạn chế các thiếu sót của mình và để khoá luận văn được hoàn thiện hơn.

LỜI CAM ĐOAN

Em tên Võ Hà Thành, là sinh viên chuyên ngành Điện tử Viễn thông, khoa Điện - Điện tử, trường Đại học Bách Khoa TP HCM. Em xin cam đoan các nội dung sau đều là sự thật:

- Đề tài luận văn này hoàn toàn do em thực hiện.
- Các tài liệu được sử dụng trong luận văn được trích dẫn nguồn gốc và thông tin đầy đủ.
- Các thông tin được sử dụng đều do chính em nghiên cứu cũng như phát triển sản phẩm, không sao chép trực tiếp bất cứ bài luận hay nghiên cứu nào hay sao chép mà chưa có trích dẫn.
- Các nghiên cứu trong đề tài là hoàn toàn không sao chép từ các nguồn tài liệu nào.

TÓM TẮT LUẬN VĂN

Trong cuộc sống hiện đại ngày nay, công nghệ đóng một vai trò vô cùng quan trọng, nó thúc đẩy nền kinh tế xã hội của một quốc gia hay khu vực và mang tầm quốc tế. Công nghệ càng hiện đại thì con người càng có thêm nhu cầu về tiện nghi cho cuộc sống của mình, do vậy IoT trở thành một lĩnh vực hàng đầu hiện nay, cùng với đó là sự phát triển của lĩnh vực nhà thông minh. Đề tài **“Xây dựng mô hình điều khiển thiết bị trong nhà thông minh”**, có tên tiếng Anh là **“Build the model of devices controller in smart home”**, là một phần trong lĩnh vực thiết kế và xây dựng những ngôi nhà thông minh đáp ứng các nhu cầu thiết yếu về tiện ích công nghệ không ngừng tăng lên của con người. Bài luận gồm các chương chính như sau:

Chương 1: Tổng quan

Chương 2: Một số vấn đề lý thuyết liên quan

Chương 3: Thiết kế hệ thống

Chương 4: Thiết kế phần cứng

Chương 5: Thiết kế phần mềm

Chương 6: Kết quả và hướng phát triển

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN	1
1.1 Giới thiệu.....	1
1.2 Tình hình nghiên cứu trong và ngoài nước.....	2
1.1.1 Sonoff	2
1.1.2 Smart Life	5
1.1.3 Shelly Smart Switch.....	6
1.3 Mục tiêu nghiên cứu	7
CHƯƠNG 2: MỘT SỐ VẤN ĐỀ LÝ THUYẾT LIÊN QUAN	9
2.1 Firebase Realtime-Database	9
2.1.1 Firebase là gì?	9
2.1.2 Firebase Realtime-Database là gì?	9
2.1.3 Tại sao chọn Firebase Realtime-Database?	10
2.2 HTTP và HTTPS	12
2.2.1 HTTP là gì?	12
2.2.2 HTTPS là gì?	14
2.3 Firebase REST API.....	14
2.3.1 API là gì?	15
2.3.2 REST là gì?	15
2.3.3 RESTful API là gì?	15
2.3.4 Firebase REST API	16
2.4 ESP8266.....	17
2.4.1 Tổng quan về ESP8266	17
2.4.2 Wifi SoC ESP8266 ESP-12F	19
2.5 Arduino IDE.....	20
2.5.1 Arduino là gì?	20
2.5.2 Arduino IDE	21
2.6 HTML, CSS và Javascript.....	22
2.6.1 Giới thiệu về HTML	22
2.6.2 Giới thiệu về CSS	22
2.6.3 Giới thiệu về Javascript.....	23
2.7 Ngôn ngữ lập trình C/C++.....	24

2.7.1	Ngôn ngữ lập trình C	24
2.7.2	Ngôn ngữ lập trình C++	25
CHƯƠNG 3: THIẾT KẾ HỆ THỐNG		27
3.1	Tổng quan	27
3.2	Sơ đồ hệ thống.....	27
3.3	Phân tích và lựa chọn phương án.....	28
3.3.1	Lựa chọn nguồn	28
3.3.2	Lựa chọn vi điều khiển	29
3.3.3	Lựa chọn nút nhấn.....	30
3.3.4	Lựa chọn relay	30
CHƯƠNG 4: THIẾT KẾ PHẦN CỨNG		31
4.1	Yêu cầu thiết kế.....	31
4.2	Sơ đồ khối của mạch	32
4.2.1	Sơ đồ khối chi tiết	32
4.2.2	Khối nguồn Hi-link 5VDC 3W	32
4.2.3	Khối nguồn ASM1117 3.3V	33
4.2.4	Khối MCU ESP8266-12F	34
4.2.5	Khối nút nhấn cảm ứng điện dung	34
4.2.6	Khối relay và LED trạng thái relay	35
4.2.7	Khối hỗ trợ cảm biến giao tiếp I2C qua jack 3.5mm	36
4.2.8	Khối LED nguồn và trạng thái mạng	36
4.3	Sơ đồ nguyên lý mạch và thiết kế PCB	37
4.3.1	Giới thiệu về Altium Designer	37
4.3.2	Sơ đồ nguyên lý mạch.....	38
4.3.2.1	Khối nguồn	38
4.3.2.2	Khối vi điều khiển ESP8266 ESP-12F	39
4.3.2.3	Khối nút nhấn và giao tiếp.....	41
4.3.2.4	Khối relay và opto	42
4.3.3	Thiết kế và vẽ PCB	43
CHƯƠNG 5: THIẾT KẾ PHẦN MỀM		45
5.1	Yêu cầu kỹ thuật	45
5.2	Thiết kế và lập trình firmware cho ESP8266	46
5.2.1	Sơ đồ khối tổng quan	46
5.2.2	Thư viện Firebase-ESP8266	46

5.2.3 Xây dựng thư viện quản lý và điều khiển thiết bị	49
5.3 Thiết kế và lập trình giao diện người dùng trên website.....	51
CHƯƠNG 6: KẾT QUẢ VÀ HƯỚNG PHÁT TRIỂN.....	54
6.1 Kết quả thiết kế phần cứng	54
6.2 Kết quả thiết kế phần mềm	55
6.3 Kết luận và phương hướng phát triển	57
TÀI LIỆU THAM KHẢO	57

DANH SÁCH HÌNH

Hình 1.1 - Sonoff Basic.	2
Hình 1.2 - Sản phẩm Sonoff Wall Switch.	3
Hình 1.3 - Sản phẩm Sonoff TH16.	3
Hình 1.4 - Sản phẩm Sonoff Pow R2.	4
Hình 1.5 - Một sản phẩm của Smart Life.	5
Hình 1.6 - Sản phẩm Shelly 1.	6
Hình 1.7 - Sản phẩm Shelly 2.5.	6
Hình 2.1 - Cấu trúc của một Firebase Realtime-Database.	10
Hình 2.2 - Các nhà cung cấp tính năng tài khoản xác thực cho Firebase.	11
Hình 2.3 - Cấu trúc của một yêu cầu HTTP.	12
Hình 2.4 - Cấu trúc thẻ lệnh của một yêu cầu HTTP.	13
Hình 2.5 - Cấu trúc thẻ lệnh của một phản hồi HTTP.	13
Hình 2.6 - Sơ đồ chân của IC ESP8266.	18
Hình 2.7 - Hình ảnh thực tế của IC ESP8266.	18
Hình 2.8 - Sơ đồ chân của module ESP8266 ESP-12F.	19
Hình 2.9 - Logo thương hiệu của Arduino.	20
Hình 2.10 - Giao diện cơ bản của phần mềm Arduino IDE.	21
Hình 2.11 - Một đoạn mã HTML.	22
Hình 2.12 - Một đoạn mã CSS.	23
Hình 2.13 - Một đoạn mã Javascript.	23
Hình 2.14 - Một đoạn chương trình C.	24
Hình 2.15 - Một đoạn chương trình C++.	25
Hình 3.1 - Sơ đồ hệ thống điều khiển.	27
Hình 4.1 - Sơ đồ khối chi tiết của mạch.	32
Hình 4.2 - Module nguồn AC-DC Hi-link HLK-PM01 5VDC 3W.	33
Hình 4.3 - IC ổn áp ASM1117 3.3V.	33
Hình 4.4 - Module thu phát Wifi SoC ESP8266-12F.	34
Hình 4.5 - Module cảm ứng chạm điện dung.	35
Hình 4.6 - Relay HF46F-G 5VDC.	35
Hình 4.7 - Cổng Audio 3.5mm 4 Cực 6 Chân Dán SMD.	36
Hình 4.9 - Giao diện của phần mềm Altium Designer 18.0.	37
Hình 4.10 - Giao diện thiết kế PCB của Altium Designer.	38
Hình 4.11 - Sơ đồ nguyên lý khối nguồn.	38
Hình 4.12 - Sơ đồ nguyên lý khối vi điều khiển.	40
Hình 4.13 - Sơ đồ nguyên lý khối nút nhấn và cảm biến.	41
Hình 4.14 - Sơ đồ nguyên lý khối relay và opto.	42
Hình 4.15 - PCB 2 lớp sau khi được thiết kế.	44
Hình 4.16 - Mô hình mạch 3D xây dựng trên Altium Designer.	44
Hình 5.1 - Sơ đồ khối tổng quan chương trình trên ESP8266.	46
Hình 5.2 - Sơ đồ giao diện trang web điều khiển.	51

Hình 6.1 - Mạch sau khi được thi công.	54
Hình 6.2 - Mạch in sau khi thi công.	55
Hình 6.3 - Giao diện điều khiển trên điện thoại.....	56
Hình 6.4 - Giao diện điều khiển trên máy tính.....	56

DANH SÁCH BẢNG

Bảng 4.1 - Danh sách linh kiện khối nguồn.	39
Bảng 4.2 - Danh sách linh kiện khối vi điều khiển.....	40
Bảng 4.3 - Danh sách linh kiện khối nút nhấn và cảm biến.....	42
Bảng 4.4 - Danh sách linh kiện trong khối relay và opto.	43

DANH SÁCH TỪ VIẾT TẮT

Chữ viết tắt	Tên đầy đủ	Định nghĩa Tiếng Việt
ADC	Analog-to-Digital Converter	Chuyển đổi một tín hiệu analog sang digital.
API	Application Programming Interface	Giao diện lập trình ứng dụng.
CSS	Cascading Style Sheets	Ngôn ngữ tạo phong cách cho trang web.
GPIO	General Purpose Input/Output	Cửa ngõ giao tiếp vào/ra của vi điều khiển.
HTML	Hypertext Markup Language	Ngôn ngữ đánh dấu siêu văn bản.
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản.
HTTPS	Hyper Text Transfer Protocol Secure	Giao thức truyền tải siêu văn bản an toàn.
I2C	Inter-IC (multi-master bus)	Giao thức truyền tin nối tiếp đồng bộ sử dụng 2 dây.
IC	Integrated Circuit	Mạch tích hợp.
IDE	Integrated Development Environment	Môi trường tích hợp dùng để viết code để phát triển ứng dụng.
IoT	Internet of Things	Kết nối vạn vật.
IP	Internet Protocol	Giao tiếp mạng.
JS	Javascript	Ngôn ngữ lập trình thông dịch.
JSON	JavaScript Object Notation	Cấu trúc đối tượng Javascript.
LED	Light Emitting Diode	Điốt phát sáng.
NoSQL	not only Structured Query Language	Ngôn ngữ truy vấn có cấu trúc.
PCB	Printed Circuit Board	Mạch in.
REST	Representational State Transfer	Chuyển đổi cấu trúc dữ liệu.
SPI	Serial Peripheral Interface	Giao tiếp ngoại vi nối tiếp.
SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc.
SSL	Secure Sockets Layer	Lớp kết nối an toàn.
TCP	Transmission Control Protocol	Giao thức điều khiển truyền vận.
UART	Universal asynchronous receiver-transmitter	Giao tiếp thu phát không đồng bộ.
URL	Uniform Resource Locator	Đường dẫn tham chiếu tài nguyên.
XML	Extensible Markup Language	Ngôn ngữ đánh dấu mở rộng.

Chương 1: Tổng Quan

1.1 Giới thiệu

Hiện nay, khoa học và kỹ thuật ngày càng phát triển vượt trội, các tiện ích hay ứng dụng được phát triển và ra đời ngày càng nhiều, tăng theo những nhu cầu thiết yếu của con người nhằm giúp cuộc sống trở nên dễ dàng và thuận tiện hơn.

Trong cuộc sống ngày nay, con người luôn đặt ra những câu hỏi, những nhu cầu thiết yếu để giải quyết các vấn đề xoay quanh cuộc sống hằng ngày. Ai trong số chúng ta cũng đều từng nghĩ tới một cuộc sống tiện nghi hơn, nơi mà các thiết bị và máy móc đóng một vai trò cực kỳ quan trọng và có thể trở thành những người bạn tốt nhất với con người, chính những nhu cầu đó đôi khi lại đem lại những ý tưởng tuyệt vời mà các nhà khoa học có thể chưa nghĩ tới.

Vì lý do đó, những nhà khoa học luôn cảm thấy thích thú và tò mò với những điều mới mẻ mà người xung quanh mang lại cho họ. Các nhà khoa học đã và đang nghiên cứu và cho ra các sản phẩm để đáp ứng lại những nhu cầu mà thị trường đang quan tâm đến. IoT (Internet of Things – kết nối vạn vật) là một trong những lĩnh vực đang được quan tâm nhất hiện nay, trở thành một chủ đề nổi bật trong giới khoa học, các ngành công nghiệp đua nhau cạnh tranh để chiếm lĩnh thị phần, do đó ngành công nghiệp IoT trở nên rất nóng nực và cạnh tranh hơn bao giờ hết.

Trong lĩnh vực IoT hiện nay, Smart home (ngôi nhà thông minh) có thể nói là ứng dụng gây được tiếng vang lớn trên toàn cầu và được tìm kiếm nhiều nhất trên Internet với ứng dụng của IoT. Lĩnh vực này có thể giúp bạn có thể điều khiển các thiết bị trong nhà, theo dõi thông tin từ xa mà thậm chí không cần phải có mặt ở nhà ngay lúc đó. Ngoài ra Smart home còn có thể tự điều chỉnh để đáp ứng lại các nhu cầu của con người mà không cần sự tác động trực tiếp hay điều khiển từ con người, giúp con người cảm thấy thoải mái và tiện nghi hơn. Những ngôi nhà thông minh sẽ giúp chúng ta tiết kiệm được thời gian và luôn tạo ra một sự yên tâm về độ an toàn.

Trong bài luận này, chúng ta sẽ nghiên cứu về lĩnh vực điều khiển từ xa, lĩnh vực thiết yếu và quan trọng nhất của Smart home, mà cụ thể là công tắc âm tường điều khiển bật tắt thiết bị từ xa bằng điện thoại hay máy tính.

1.2 Tình hình nghiên cứu trong và ngoài nước

1.1.1 Sonoff

Sonoff đang là thương hiệu dẫn đầu về thiết bị điều khiển thông qua wifi và internet, thương hiệu này có nhiều sản phẩm đa dạng nhằm phục vụ cho nhu cầu của người dùng ở thị trường thế giới, các sản phẩm nổi tiếng như:

- **Sonoff Basic** [1]: Cho phép người dùng điều khiển 1 thiết bị sử dụng wifi có kết nối internet và mobile app.



Hình 1.1 - Sonoff Basic.

- **Sonoff Wall Switch** [2]: Đây là loạt thiết bị thay thế cho công tắc âm tường thường bằng công tắc âm tường có thể điều khiển qua phím cảm ứng hoặc điều khiển từ xa thông qua mạng wifi có kết nối internet.



Hình 1.2 - Sản phẩm Sonoff Wall Switch.

- **Sonoff TH16** [3]: Cho phép người dùng điều khiển thiết bị có giám sát của cảm biến nhiệt độ-độ ẩm, cho phép người dùng có thể thiết lập các chế độ đóng ngắt thiết bị theo giá trị của cảm biến, đồng thời giúp người dùng giám sát giá trị thông qua mobile app.



Hình 1.3 - Sản phẩm Sonoff TH16.

- **Sonoff Pow R2** [4]: Cho phép người dùng điều khiển đóng ngắt thiết bị và giám sát công suất tiêu thụ của tải thông qua mobile app.



Hình 1.4 - Sản phẩm Sonoff Pow R2.

Ngoài ra hãng này còn cho ra các sản phẩm khác liên quan đến IoT.

❖ **Nhận xét:**

- **Ưu điểm:** Sonoff có một hệ sinh thái về IoT lớn với nhiều loại sản phẩm khác nhau được kết nối với một ứng dụng, cho phép điều khiển các thiết bị xung quanh một cách dễ dàng, các sản phẩm được hoàn thiện kỹ càng với các nhiều chức năng như điều khiển từ xa, hẹn giờ bật tắt thiết bị, dimmer từ xa...
- **Nhược điểm:** Các sản phẩm của Sonoff khá đa dạng, các chức năng khác nhau được thiết kế riêng biệt cho từng loại sản phẩm, làm cho một sản phẩm bị thiếu đi sự đa dạng về chức năng, điều này khiến người sử dụng khó có thể lựa chọn được thiết bị đáp ứng mọi nhu cầu cần thiết, ngoài ra Sonoff không có giao diện điều khiển trên trình duyệt web, mà phải điều khiển thông qua ứng dụng trên điện thoại.

1.1.2 Smart Life [5]

Smart Life cũng là một thương hiệu có tiếng trên thị trường, các sản phẩm của Smart Life có phần tương đồng với các sản phẩm của Sonoff. Đây cũng là một thương hiệu có sự đa dạng về mẫu mã các thiết bị IoT.



Hình 1.5 - Một sản phẩm của Smart Life.

❖ Nhận xét:

- **Ưu điểm:** Smart Life là một thương hiệu đi sau nhưng vẫn có thị trường rộng lớn. Cũng như Sonoff, Smart Life cũng có sự đa dạng về sản phẩm IoT, cho phép người dùng có thể kết nối các thiết bị cần được điều khiển vào chung một ứng dụng để có thể điều khiển dễ dàng.
- **Nhược điểm:** Smart Life cũng có các mặt hạn chế tương tự như Sonoff, đó là thiếu đi sự đa dạng chức năng trong từng loại sản phẩm nhất định, khiến người dùng khó có thể lựa chọn sản phẩm có đầy đủ chức năng mình mong muốn, Smart Life cũng không có trang web để cho phép người dùng điều khiển các thiết bị từ xa trực tiếp trên trình duyệt web.

1.1.3 Shelly Smart Switch

Shelly cũng là một thương hiệu về IoT, tuy không được nổi tiếng như hai hãng bên trên, song Shelly cũng thường được cộng đồng nghiên cứu đưa ra so sánh cùng với Sonoff và cũng được đánh giá cao về độ ổn định và sự dễ dàng truy cập. Không như Sonoff cần có mobile app để điều khiển, Shelly là thiết bị có thể cấu hình và điều khiển thông qua Web app, điều này giúp Shelly được đánh giá cao hơn về sự tiện lợi khi có thể truy cập từ bất cứ thiết bị máy tính hay điện thoại.



Hình 1.6 - Sản phẩm Shelly 1 [6].



Hình 1.7 - Sản phẩm Shelly 2.5 [7].

❖ **Nhận xét:**

- ***Ưu điểm:*** Shelly là sản phẩm nhỏ gọn, có giao diện điều khiển trên web và ứng dụng di động, dễ dàng sử dụng để điều khiển ở mọi thiết bị như điện thoại hay máy tính, ngoài ra Shelly còn là mô hình mã nguồn mở để mọi người cùng tham gia nghiên cứu và học tập.
- ***Nhược điểm:*** Đây là sản phẩm khá cơ bản về ngoại hình lẫn chức năng.

1.3 Mục tiêu nghiên cứu

Mục tiêu của đề tài là thiết kế mô hình của một hộp công tắc âm tường có thể điều khiển qua mạng Internet thay cho các hộp công tắc vật lý thông thường có ngoài thị trường.

Yêu cầu kỹ thuật:

❖ ***Thiết bị vật lý:***

- Thiết bị có thể cấp nguồn 220VAC.
- Thiết bị có thể điều khiển tắt mở được 2, 3 hoặc 4 tải thông qua relay có cách ly bằng opto.
- Thiết bị sử dụng nút nhấn cảm ứng điện dung thay cho các công tắc bấm bênh.
- Thiết bị có thể kết nối được wifi và kết nối internet để cấu hình và điều khiển.
- Thiết bị có các đèn báo trạng thái.
- Thiết bị cho phép cấu hình wifi và tài khoản người dùng thông qua Access Point và Web App.

❖ Phần mềm và giao tiếp qua internet:

- Hệ thống có thể tạo tài khoản người dùng để cấu hình và điều khiển thiết bị, các thông tin người dùng và thiết bị sẽ được lưu lại thông qua một Database (Google Firebase), đảm bảo các hình thức bảo mật cần thiết để bảo mật thông tin người dùng và các thiết bị được liên kết.
- Người dùng có thể liên kết nhiều thiết bị vật lý vào một tài khoản để dễ dàng điều khiển thông qua trang điều khiển.
- Trang điều khiển là một Web app cho phép người dùng có thể tạo tài khoản đăng nhập và liên kết các thiết bị cùng lúc để dễ dàng điều khiển từ xa, có đầy đủ các chức năng cài đặt thiết bị như đổi tên, tạo nhóm các thiết bị.

Chương 2: Một số vấn đề lý thuyết liên quan

2.1 Firebase Realtime-Database [8]

2.1.1 *Firebase là gì?*

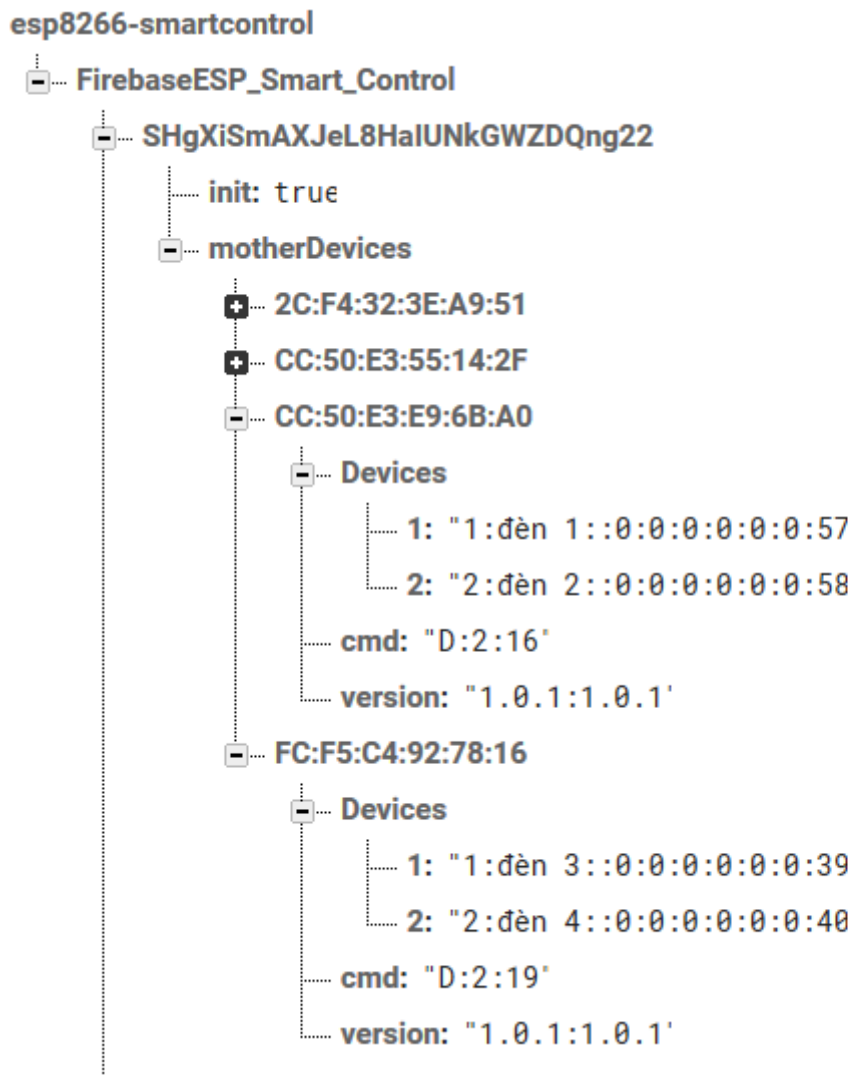
Firebase có tiền thân là một ứng dụng khởi nghiệp có tên là Envolv. Envolv cung cấp cho những nhà phát triển một API (Application Programming Interface – giao diện lập trình ứng dụng) giúp họ có thể chèn tính năng chat (trò chuyện trực tuyến) vào website của mình.

Điều khác biệt đó là các nhà phát triển đã sử dụng Envolv để chứa các dữ liệu cho các ứng dụng của mình thay vì các đoạn chat như ban đầu. Điều này làm cho những người sáng lập nên Envolv là James Tamplin và Andrew Lee đã tách rời tính năng chat và hệ thống dữ liệu thời gian thực ra. Sau đó hệ thống này đã được mua lại bởi Google vào năm 2014, ngay sau đó Firebase đã được nâng cấp nhanh chóng để trở thành một nền tảng ứng dụng đồ sộ với nhiều chức năng như hiện nay.

Firebase là một nền tảng phát triển ứng dụng di động và website, cung cấp cho những nhà phát triển rất nhiều công cụ và dịch vụ để giúp họ phát triển nhiều ứng dụng có chất lượng cao, tăng cao khả năng quản lý người dùng và mang lại nhiều lợi nhuận cho việc phát triển.

2.1.2 *Firebase Realtime-Database là gì?*












Firebase Realtime-Database là một nền tảng lưu trữ đám mây với cấu trúc NoSQL, điều này giúp các nhà phát triển luôn có thể dễ dàng lưu trữ và đồng bộ dữ liệu với người dùng ngay lập tức.



Hình 2.1 - Cấu trúc của một Firebase Realtime-Database.

2.1.3 Tại sao chọn Firebase Realtime-Database?

Firebase Realtime-Database thực chất chỉ là một đối tượng JSON mà các nhà phát triển có thể quản lý trong thời gian thực. Chỉ bằng cách sử dụng một API đơn giản, Firebase Realtime-Database sẽ trả về cho các ứng dụng những thông tin dữ liệu cần thiết và cập nhật những dữ liệu đó theo thời gian thực.

Sign-in providers	
Provider	Status
 Email/Password	Enabled
 Phone	Disabled
 Google	Disabled
 Play Games	Disabled
 Game Center Beta	Disabled
 Facebook	Disabled
 Twitter	Disabled
 GitHub	Disabled
 Yahoo	Disabled
 Microsoft	Disabled
 Apple	Disabled

Hình 2.2 - Các nhà cung cấp tính năng tài khoản xác thực cho Firebase.

Việc đồng bộ thời gian thực sẽ giúp người dùng dễ dàng truy cập đến dữ liệu của họ bằng các thiết bị khác nhau, có thể là website hoặc một ứng dụng di động nào đó, đây cũng là điểm mạnh mà Firebase Realtime-Database cung cấp cho nhà phát triển, tính năng này giúp các dữ liệu có thể đồng bộ kịp thời với nhau mà không cần một máy chủ vật lý nào cả. Ngoài ra, hệ thống này còn cung cấp các tính năng bảo mật như Authentication (Xác thực) giúp người dùng có thể bảo mật dữ liệu của mình bằng email và mật khẩu hay số điện thoại, thậm chí người dùng vẫn có thể bảo mật dữ liệu của mình bằng các tài khoản mạng xã hội như Google, Facebook, Twitter...

Firebase Realtime-Database cung cấp cho những nhà phát triển các API nhằm giúp họ có thể chủ động trong việc truy cập và quản lý các dữ liệu của mình, các API này được xây dựng dựa trên các gói tin HTTPS được người dùng gửi đến máy chủ của hệ thống cơ sở dữ liệu nhằm yêu cầu thực hiện một chức năng nào đó vào cơ sở dữ liệu được hướng đến.

2.2 HTTP và HTTPS^[9] [9]

2.2.1 HTTP là gì?

HTTP là viết tắt của Hypertext Transfer Protocol, có nghĩa là “giao thức truyền tải siêu văn bản”.

HTTP bao gồm các tiêu chuẩn để giúp các thông tin có thể được truyền tải trong mạng internet, ngoài ra HTTP cung cấp các luật và chuẩn giao tiếp để kết nối giữa các trình duyệt và máy chủ lại với nhau.

HTTP là một giao thức mạng nằm trong lớp ứng dụng và được xây dựng dựa trên giao thức TCP (Transmission Control Protocol - Giao thức điều khiển truyền vận). HTTP sử dụng cấu trúc siêu văn bản để mô tả cách thức liên kết giữa các nút chứa văn bản lại với nhau. Giao thức này còn được gọi là “giao thức không trạng thái” vì các câu lệnh được xử lý riêng lẻ và không liên quan đến các lệnh đã được xử lý trước đó.

```
Request URL: https://en.wikipedia.org/w/load.php?lang=en&modules=startup&only=scripts&raw=1&skin=vector
Request Method: GET
Remote Address: 103.102.166.224:443
Status Code: 200 OK ⓘ
Version: HTTP/2
Referrer Policy: no-referrer-when-downgrade
```

Hình 2.3 - Cấu trúc của một yêu cầu HTTP.

```
Host: en.wikipedia.org
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:77.0) Gecko/20100101 Firefox/77.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: https://en.wikipedia.org/wiki/Firebase
Cookie: WMF-Last-Access-Global=01-Jul-2020; WMF-Last-Access=01-Jul-2020; GeoIP=VN:SG:Ho_Chi_Minh_City:10.81:106.64:v4; enwikimuser-sessi
TE: Trailers
If-None-Match: W/"18k1y"
```

Hình 2.4 - Cấu trúc thẻ lệnh của một yêu cầu HTTP.

```
HTTP/2 200 OK
date: Wed, 01 Jul 2020 14:10:11 GMT
etag: W/"dtstj"
expires: Wed, 01 Jul 2020 14:15:03 GMT
cache-control: public, max-age=300, s-maxage=300
vary: Accept-Encoding
server: ATS/8.0.8
x-content-type-options: nosniff
content-type: text/javascript; charset=utf-8
content-encoding: gzip
x-cache: cp5008 hit, cp5010 hit/2249
x-cache-status: hit-front
server-timing: cache;desc="hit-front"
strict-transport-security: max-age=106384710; includeSubDomains; preload
x-client-ip: 14.169.252.197
age: 0
accept-ranges: bytes
content-length: 22217
X-Firefox-Spdy: h2
```

Hình 2.5 - Cấu trúc thẻ lệnh của một phản hồi HTTP.

HTTP quy định ra một tập hợp các phương thức để thực thi yêu cầu của người lập trình tới dữ liệu mong muốn, bao gồm GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE, PATCH. Mỗi phương thức thực hiện một nhiệm vụ khác nhau:

- **GET**: Phương thức **GET** yêu cầu trả về một dữ liệu được chỉ định, phương thức này thường chỉ sử dụng để truy xuất dữ liệu.
- **HEAD**: Phương thức **HEAD** yêu cầu một phản hồi giống hệt như phương thức **GET** nhưng không trả về dữ liệu.
- **POST**: Phương thức **POST** được sử dụng để gửi một đối tượng dữ liệu đến dữ liệu được chỉ định, thường dùng để thay đổi trạng thái hay trả về dữ liệu thêm cho máy chủ.
- **PUT**: Phương thức **PUT** được sử dụng để thay thế toàn bộ dữ liệu được chỉ định bằng dữ liệu được gửi kèm theo.

- **DELETE**: Phương thức **DELETE** dùng để xoá dữ liệu được chỉ định.
- **CONNECT**: Phương thức **CONNECT** dùng để tạo ra một liên kết đến máy chủ được chỉ định bởi tài nguyên được nhắm tới.
- **OPTIONS**: Phương thức **OPTIONS** dùng để mô tả các tùy chọn của liên kết đến tài nguyên hay dữ liệu được nhắm tới.
- **TRACE**: Phương thức **TRACE** thực hiện một thông điệp kiểm tra vòng lặp đến tài nguyên được chỉ dẫn.
- **PATCH**: Phương thức **PATCH** được dùng để áp dụng sửa đổi một phần của tài nguyên hay dữ liệu.

2.2.2 HTTPS là gì?

HTTPS là viết của Hyper Text Transfer Protocol Secure, có nghĩa là “giao thức truyền tải siêu văn bản an toàn”.

Đây là một phiên bản cao cấp và an toàn hơn của HTTP. Giao thức này sử dụng Port 443 cho quá trình truyền tải dữ liệu và cho phép bảo mật các gói tin bằng cách mã hoá toàn bộ quá trình trao đổi dữ liệu bằng SSL (Secure Sockets Layer – một tiêu chuẩn an ninh công nghệ). HTTPS là một sự kết hợp giữa giao thức SSL/TLS và HTTP. Giao thức cung cấp trình mã hoá và nhận diện an toàn đối với các máy chủ mạng.

Ngoài ra, HTTPS còn cho phép người dùng tạo ra một kết nối mã hoá an toàn giữa trình duyệt và máy chủ. Nó giúp cho dữ liệu được giữ an toàn trong cả hai chiều trong quá trình kết nối và truyền tải nhằm giúp người dùng bảo vệ được các thông tin nhạy cảm có khả năng bị đánh cắp.

2.3 Firebase REST API [10]

Nguyên lý REST và cấu trúc dữ liệu RESTful được biết đến rộng rãi trong giới lập trình web nói chung và lập trình ứng dụng nói riêng. REST không phải là

một loại công nghệ, nó là phương thức tạo API với nguyên lý tổ chức nhất định. Những nguyên lý này nhằm hướng dẫn người lập trình viên tạo ra môi trường xử lý API request được toàn diện. Để hiểu rõ về Firebase REST API là gì, tiếp theo chúng ta cần tìm hiểu được các định nghĩa bên trong nó.

2.3.1 API là gì? [11]

API là viết tắt của Application Programming Interface, đó là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một số ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà người lập trình cần cho ứng dụng của họ ở những kiểu dữ liệu phổ biến như JSON hay XML.

2.3.2 REST là gì? [11]

REST là viết tắt của REpresentational State Transfer, đó là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP hay HTTPS để tạo ra giao tiếp giữa các thiết bị. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, PUT, DELETE... đến một URL để xử lý dữ liệu.

2.3.3 RESTful API là gì? [11]

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web hay các ứng dụng liên quan tới mạng để quản lý các tài nguyên hay dữ liệu. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến hiện nay để cho các ứng dụng giao tiếp với nhau.

Chức năng quan trọng nhất của REST là quy định cách sử dụng giao thức HTTP và cách định dạng các URL cho các ứng dụng để quản lý các tài nguyên và dữ liệu. RESTful không quy định cách lập trình và không giới hạn bởi ngôn ngữ lập trình, bất kỳ ngôn ngữ nào cũng có thể sử dụng để thiết kế một RESTful API.

2.3.4 *Firebase REST API* [10]

Firebase được thiết kế sao cho mỗi URL đều là một điểm cuối của REST, người lập trình chỉ cần thêm đuôi “.json” vào cuối mỗi URL của Firebase Realtime-Database để gửi một yêu cầu HTTPS từ thiết bị hay ứng dụng bất kỳ.

Firebase hiện tại chỉ cho phép các yêu cầu HTTPS và chỉ phản hồi cho các yêu cầu đã được mã hoá để đảm bảo an toàn thông tin của người dùng.

Các phương thức cơ bản được Firebase Realtime-Database hỗ trợ và mục đích sử dụng như sau:

- **GET:** Yêu cầu **GET** dùng để đọc dữ liệu từ cơ sở dữ liệu. Người lập trình có thể gửi một yêu cầu **GET** đến URL để lấy dữ liệu, ví dụ:

```
curl 'https://[PROJECT_ID].firebaseio.com/users/jack/name.json'
```

Phản hồi trả về là chuỗi JSON chứa dữ liệu ở nút được chỉ định nếu yêu cầu được xử lý thành công, ví dụ:

```
{ "first": "Jack", "last": "Sparrow" }
```

- **PUT:** Yêu cầu **PUT** được dùng để viết dữ liệu. Người lập trình gửi yêu cầu **PUT** đến URL để viết dữ liệu vào nút được chỉ định kèm theo dữ liệu muốn ghi vào theo định dạng JSON, ví dụ:

```
curl -X PUT -d '{ "first": "Jack", "last": "Sparrow" }' \  
'https://[PROJECT_ID].firebaseio.com/users/jack/name.json'
```

Phản hồi trả về là chuỗi JSON chứa dữ liệu ở nút được chỉ định nếu yêu cầu được xử lý thành công, ví dụ:

```
{ "first": "Jack", "last": "Sparrow" }
```

- **POST:** Yêu cầu **POST** dùng để đẩy dữ liệu hay thêm dữ liệu. Người lập trình gửi yêu cầu **POST** đến URL và kèm theo dữ liệu muốn thêm, ví dụ:

```
curl -X POST -d '{"user_id": "jack", "text": "Ahoy!"}' \  
'https://[PROJECT_ID].firebaseio.com/message_list.json'
```

Phản hồi trả về là chuỗi JSON chứa dữ liệu của nốt con vừa được thêm vào nốt được chỉ định nếu yêu cầu được xử lý thành công, ví dụ:

```
{ "name": "-INOQPH-aV_psbk3ZXEX" }
```

- **PATCH:** Yêu cầu **PATCH** được dùng để cập nhật giá trị của dữ liệu ở nốt con được chỉ định mà không cần phải ghi đè lên nốt cấp trên. Người dùng gửi yêu cầu **PATCH** đến URL và kèm theo dữ liệu ở nốt con muốn thay đổi, ví dụ:

```
curl -X PATCH -d '{"last":"Jones"}'\n'https://[PROJECT_ID].firebaseio.com/users/jack/name/.json'
```

Phản hồi trả về là chuỗi JSON chứa dữ liệu của nốt con được chỉ định nếu yêu cầu được xử lý thành công, ví dụ:

```
{ "last": "Jones" }
```

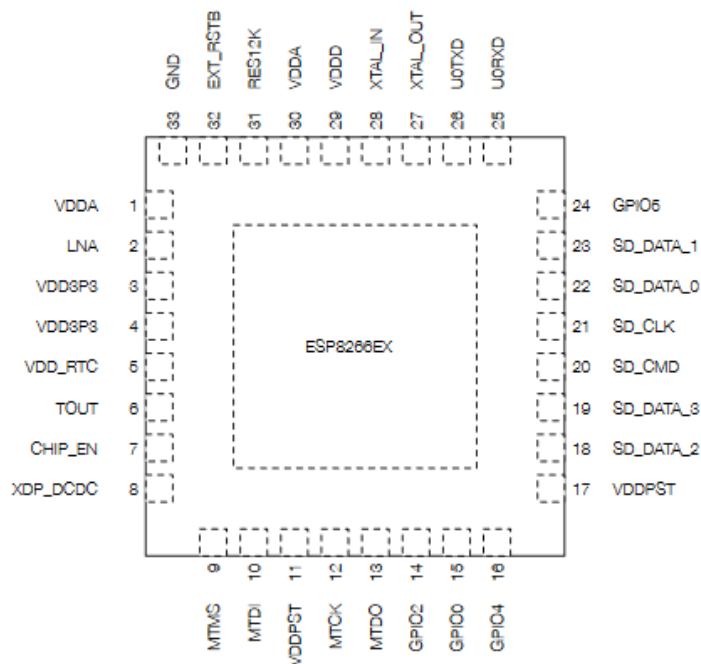
- **DELETE:** Yêu cầu **DELETE** được sử dụng để xóa dữ liệu ở nốt được chỉ định. Người dùng có thể gửi yêu cầu này đến URL của nốt cần xóa, ví dụ:

```
curl -X DELETE\n'https://[PROJECT_ID].firebaseio.com/users/jack/name/last.json'
```

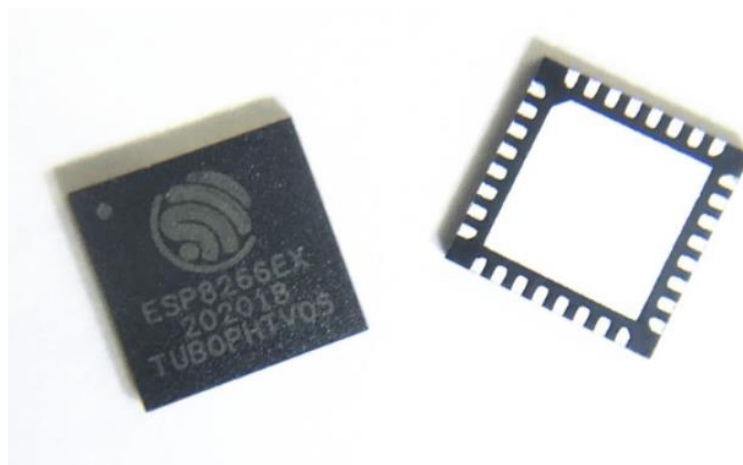
2.4 ESP8266

2.4.1 Tổng quan về ESP8266 [12]

ESP8266 là một vi điều khiển tích hợp Wifi của hãng Espressif Systems có giá thành tương đối rẻ và hỗ trợ giao tiếp TCP/IP. Vi điều khiển này sử dụng điện áp trong khoảng từ 3VDC đến 3.6VDC và thường được biết đến với tên gọi ESP8266 Wireless Transceiver.



Hình 2.6 - Sơ đồ chân của IC ESP8266 [13].



Hình 2.7 - Hình ảnh thực tế của IC ESP8266 [14].

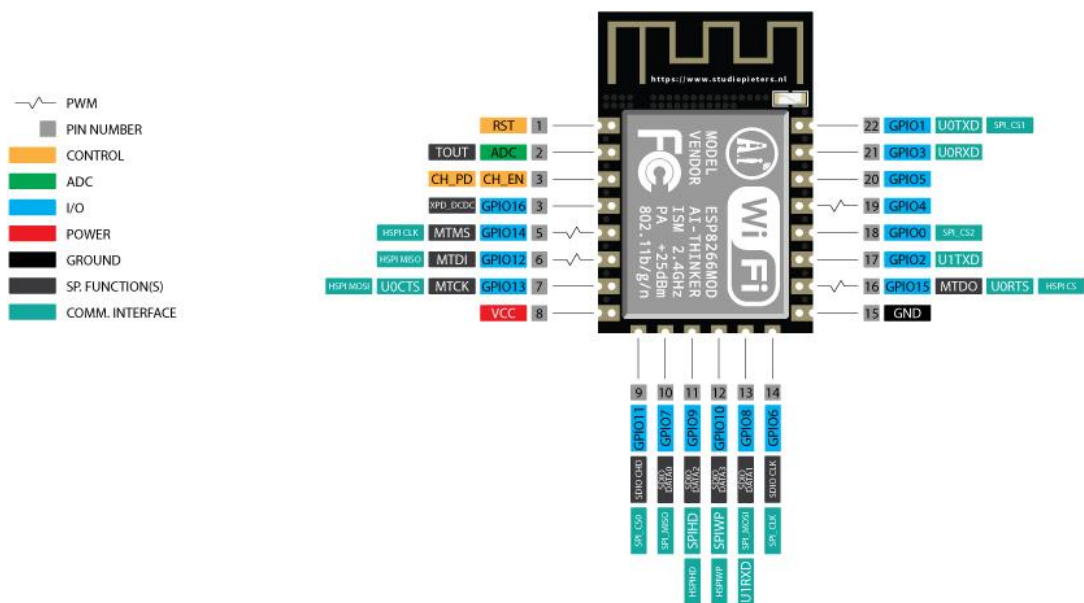
ESP8266 hỗ trợ chuẩn Wifi 802.11 b/g/n ở tần số 2.4 GHz và các tính năng khác bao gồm 16 chân GPIO dùng để giao tiếp ngoại vi, trong đó có hỗ trợ các chuẩn giao tiếp như I2C, SPI, I2S và một kênh ADC có độ phân giải 10 bits.

Vi điều khiển này thân thiện với người sử dụng bởi vì họ có thể tạo ra kết nối TCP/IP chỉ bằng cách cho ESP8266 kết nối với Wifi. Ngoài ra vi điều khiển này còn có đầy đủ các chức năng của Wifi hơn so với các loại tương tự, các chức năng nổi bật như làm HTTP Server, Web Server, Wifi Client, Wifi Repeater...

2.4.2 Wifi SoC ESP8266 ESP-12F

ESP8266 ESP-12F vượt trội hơn các thế hệ trước về tốc độ xử lý và khả năng lưu trữ. Module này còn có thể giao tiếp với các cảm biến hay các thiết bị ngoại vi khác và cần rất ít sự biến đổi để có thể tích hợp với các thiết bị khác. Các linh kiện và các chân giao tiếp được tích hợp sẵn trong module làm cho module có khả năng tương thích tốt với các vị trí nhỏ gọn khó có thể chạm tới được, ngoài ra module cũng không cần những mạch khác để có thể sử dụng được.

Ngoài ra module ESP8266 ESP-12F còn có tích hợp sẵn bộ nhớ flash ngoài với dung lượng 4MB thông qua giao tiếp SPI để tăng dung lượng bộ nhớ chương trình, giúp cải thiện tính năng và độ phức tạp mà vi điều khiển ESP8266 có thể điều khiển được.



Hình 2.8 - Sơ đồ chân của module ESP8266 ESP-12F [15].

Đây được coi là một thiết bị kết nối internet hữu ích, tuy nhiên còn một số mặt hạn chế như cần phải có mạch chuyển mức tín hiệu khi giao tiếp với các ngoại vi

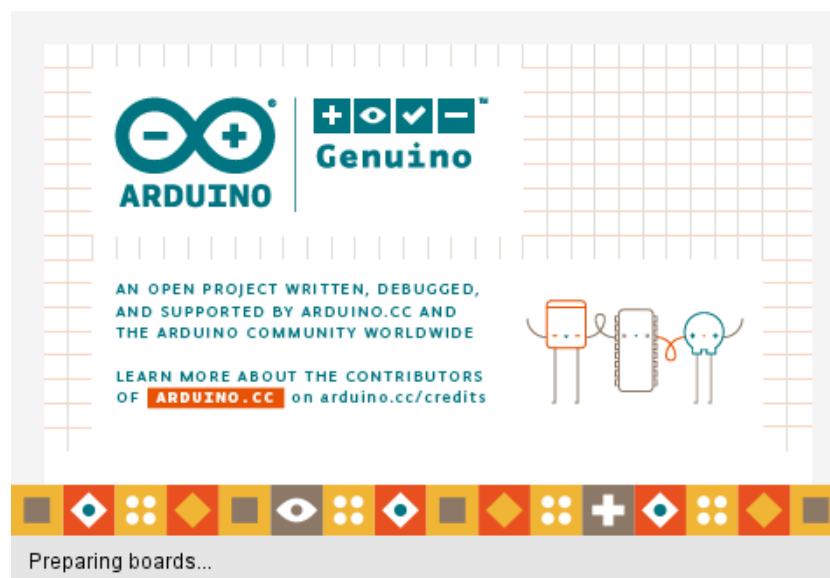
chỉ sử dụng điện áp giao tiếp là 5VDC, vì ESP8266 chỉ hoạt động ở mức logic là 3.3VDC và không hỗ trợ tích hợp dây điện áp từ 3-5VDC.

Hiện nay ESP8266 nói chung và ESP8266 ESP-12F nói riêng đang được phát triển mạnh trong cộng đồng Arduino với sự đa dạng và phong phú về tài liệu cũng như hướng dẫn sử dụng. Người dùng thường sử dụng ngôn ngữ C/C++ là ngôn ngữ lập trình chính cho vi điều khiển này, ngoài ra người dùng còn sử dụng phần mềm lập trình có sẵn của Arduino là Arduino IDE để lập trình cho vi điều khiển ESP8266 cũng bởi vì cộng đồng rộng lớn và lượng tài liệu cùng hướng dẫn lập trình phong phú này.

2.5 Arduino IDE

2.5.1 Arduino là gì? [16]

Arduino là một dự án mã nguồn mở, bao gồm cả phần cứng và phần mềm, được thiết kế lại để giúp việc lập trình vi điều khiển trở nên dễ dàng và đơn giản hơn cho những người không chuyên hay vừa mới bắt đầu, giúp họ có thể tiếp cận và làm quen với lĩnh vực lập trình nhúng dễ dàng hơn



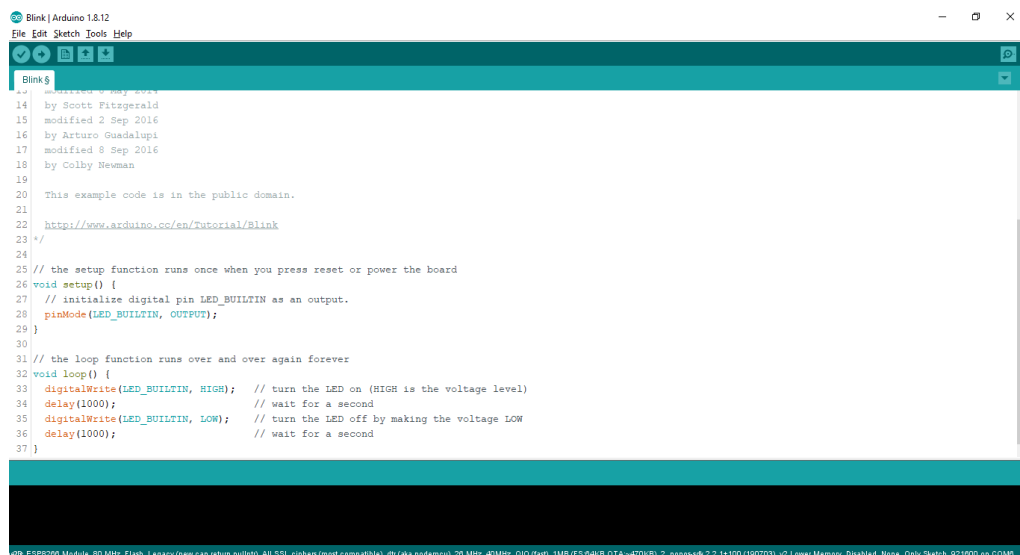
Hình 2.9 - Logo thương hiệu của Arduino.

Điểm hấp dẫn của Arduino với những người lập trình đó là Arduino sử dụng C/C++ là ngôn ngữ lập trình chính, đây cũng là một ngôn ngữ lập trình cơ bản của lập trình nhúng hay lập trình ứng dụng.

2.5.2 *Arduino IDE*

IDE (Integrated Development Environment) là môi trường tích hợp để lập trình phát triển ứng dụng. IDE có vùng soạn thảo lập trình và thường tích hợp các công cụ hỗ trợ như trình biên dịch (Compiler), trình thông dịch (Interpreter), kiểm tra lỗi (Debugger) tương ứng với các ngôn ngữ lập trình được IDE hỗ trợ.

Arduino IDE là một phần mềm IDE mã nguồn mở của Arduino hỗ trợ người dùng có thể lập trình và tải ứng dụng vào các board mạch cần được lập trình. Điểm mạnh của phần mềm Arduino IDE được xây dựng dựa trên nền Java khiến IDE này có thể sử dụng đa nền tảng, phần mềm có thể cài đặt lên các hệ điều hành khác nhau như Windows, Mac OS X hay Linux.



Hình 2.10 - Giao diện cơ bản của phần mềm Arduino IDE.

Ngoài ra phần mềm còn được cộng đồng rộng lớn của Arduino phát triển các tiện ích và board mạch khiến IDE này hỗ trợ rất nhiều board mạch như các dòng

board của Arduino hay ESP8266 và ESP32, ngoài ra Arduino IDE còn cho phép lập trình viên có thể tạo ra các bộ thư viện hỗ trợ và chia sẻ với nhau thông qua các tính năng cài đặt và tải thư viện trực tiếp ngay trong phần mềm.

2.6 HTML, CSS và Javascript

2.6.1 Giới thiệu về HTML [17]

HTML là viết tắt của Hypertext Markup Language (ngôn ngữ đánh dấu siêu văn bản), đây một ngôn ngữ đánh dấu được dùng để tạo ra các trang siêu văn bản được sử dụng rộng rãi trên toàn thế giới dưới dạng của một website. HTML được gọi là ngôn ngữ đánh dấu là vì HTML sử dụng các thẻ để đánh dấu các thành phần khác nhau trên website để tạo ra một trang web với độ phức tạp cao, độ tùy chỉnh được nâng tới mức tối đa giúp người thiết kế có thể thiết kế ra các sản phẩm mà mình mong muốn.

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Hình 2.11 - Một đoạn mã HTML.

2.6.2 Giới thiệu về CSS [18]

CSS là viết tắt của Cascading Style Sheets, đây là ngôn ngữ được định nghĩa theo tầng được dùng để tạo bố cục và trang trí cho HTML. Cùng với sự kết hợp giữa CSS và HTML, người thiết kế có thể dễ dàng bố trí và quản lý hiển thị HTML tốt hơn và dễ dàng hơn. Ngoài ra CSS còn có tính thừa hưởng, điều này giúp cho độ

phức tạp của các thiết kế giao diện được nâng cao hơn và dễ dàng hơn, giảm được thời gian thiết kế cũng như dễ quản lý các phần tử hơn.

```
body {  
    background-color: lightblue;  
}  
  
h1 {  
    color: white;  
    text-align: center;  
}  
  
p {  
    font-family: verdana;  
    font-size: 20px;  
}
```

Hình 2.12 - Một đoạn mã CSS.

2.6.3 Giới thiệu về Javascript [19]

Javascript hay còn được gọi là JS, là một ngôn ngữ lập trình thông dịch (được dịch trong lúc chạy), ngôn ngữ lập trình này được ứng dụng rộng rãi trên nhiều môi trường khác nhau và được coi là một ngôn ngữ lập trình đa nền tảng. Ngày nay Javascript đều được sử dụng hầu hết trong các lĩnh vực lập trình, Javascript đóng vai trò chủ chốt và không thể thiếu trong lĩnh vực lập trình và phát triển website hiện nay.

```
selectFilter = function() {  
    var filter = document.getElementById("filter");  
    var rule = filter.options[filter.selectedIndex].value;  
    var devices = document.getElementsByClassName("devicebutton");  
  
    for(var i=0; i < devices.length; i++) {  
        if(rule == "All" || rule == devices[i].getAttribute("room")) {  
            devices[i].style.display = "block";  
        }else {  
            devices[i].style.display = "none";  
        }  
    }  
}
```

Hình 2.13 - Một đoạn mã Javascript.

2.7 Ngôn ngữ lập trình C/C++

2.7.1 Ngôn ngữ lập trình C [20]

Ngôn ngữ lập trình C là một ngôn ngữ mệnh lệnh được phát triển từ đầu thập niên 1970 bởi Dennis Ritchie để dùng trong hệ điều hành UNIX. Từ đó, ngôn ngữ này đã lan rộng ra nhiều hệ điều hành khác và trở thành một những ngôn ngữ phổ dụng nhất. C là ngôn ngữ rất hiệu quả và được ưa chuộng nhất để viết các phần mềm hệ thống, ngoài ra nó cũng được dùng cho việc viết các ứng dụng. Ngôn ngữ lập trình C cũng thường được dùng làm phương tiện giảng dạy trong khoa học máy tính mặc dù ngôn ngữ này không được thiết kế dành cho người nhập môn.

```
// C program to illustrate for loop
#include <stdio.h>

int main()
{
    int i=0;

    for (i = 1; i <= 10; i++)
    {
        printf( "Hello World\n");
    }

    return 0;
}
```

Hình 2.14 - Một đoạn chương trình C.

C là một ngôn ngữ lập trình tương đối nhỏ gọn vận hành gần với phần cứng và nó giống với ngôn ngữ Assembler hơn hầu hết các ngôn ngữ bậc cao. Hơn thế, C đôi khi được đánh giá như là "có khả năng di động", cho thấy sự khác nhau quan trọng giữa nó với ngôn ngữ bậc thấp như là Assembler, đó là việc mã C có thể được dịch và thi hành trong hầu hết các máy tính, hơn hẳn các ngôn ngữ hiện tại trong khi đó thì Assembler chỉ có thể chạy trong một số máy tính đặc biệt. Vì lý do này C được xem là ngôn ngữ bậc trung.

C đã được tạo ra với một mục tiêu là làm cho việc viết các chương trình lớn trở nên thuận tiện hơn, với số lỗi ít hơn so với lập trình bằng các ngôn ngữ bậc thấp, giúp người lập trình dễ dàng quản lý chương trình cũng như sửa lỗi và phát triển.

2.7.2 *Ngôn ngữ lập trình C++* [21]

Ngôn ngữ lập trình C++, một trong những ngôn ngữ lập trình phổ biến trên thế giới, được Bjarne Stroustrup phát triển từ ngôn ngữ C từ cuối thập niên 1970. C++ là một phiên bản mở rộng của ngôn ngữ C, kết hợp tất cả các tính năng đã có của C và được coi như là ngôn ngữ bậc trung (middle-level), kết hợp các đặc điểm và tính năng của ngôn ngữ bậc cao và bậc thấp. Ngoài ra, C++ có thể dùng để lập trình nhúng, lập trình hệ thống, hoặc những ứng dụng, game...

```
// C++ program to illustrate for loop
#include <iostream>
using namespace std;

int main()
{
    for (int i = 1; i <= 10; i++)
    {
        cout << "Hello World\n";
    }

    return 0;
}
```

Hình 2.15 - Một đoạn chương trình C++.

Ngôn ngữ lập trình C++ là ngôn ngữ "đa hướng". Nghĩa là nó hướng cấu trúc giống C và có thêm một tính năng cực kỳ quan trọng đó là tính năng hướng đối tượng.

❖ Một số đặc trưng của ngôn ngữ C++:

- C++ là một ngôn ngữ lập trình bậc trung, có thể sử dụng C++ để phát triển những ứng dụng bậc cao và cả những chương trình bậc thấp hoạt động tốt trên phần cứng.
- C++ là một ngôn ngữ lập trình hướng đối tượng. Khác với ngôn ngữ lập trình C - một ngôn ngữ lập trình hướng thủ tục, chương trình được tổ chức theo thuật ngữ “chức năng”, một chức năng gồm có những hành động muốn thực hiện. C++ được thiết kế với một cách tiếp cận hoàn toàn mới được gọi là lập trình hướng đối tượng, nơi mà chúng ta sử dụng những đối tượng, các lớp và sử dụng các khái niệm như: thừa kế, đa hình, tính đóng gói, tính trừu tượng ...
- C++ là một ngôn ngữ lập trình hướng cấu trúc giống ngôn ngữ C, nó có nghĩa là chúng ta có thể tổ chức chương trình trên khái niệm functions.
- C++ có thể chạy trên nhiều nền tảng khác nhau như Windows, Mac OS, một số biến thể của UNIX...

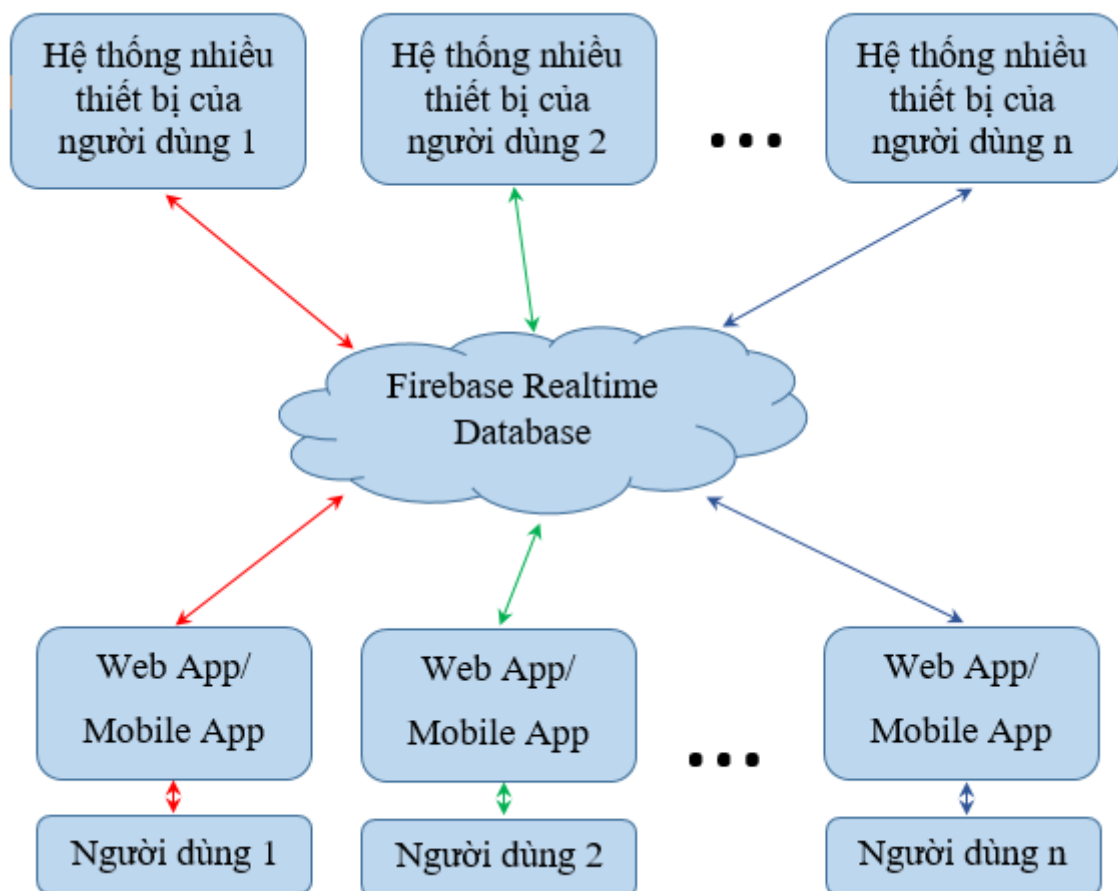
Chương 3: Thiết kế hệ thống

3.1 Tổng quan

Hệ thống bao gồm mô hình công tắc âm tường điều khiển 02 thiết bị có thể kết nối được internet thông qua Wifi, cơ sở dữ liệu chứa các thông tin về người dùng và trạng thái các thiết bị, giao diện ứng dụng dùng để điều khiển thông qua website và ứng dụng di động.

Các giao tiếp của người dùng tới thiết bị ngoại vi và ngược lại đều được thực hiện thông qua giao tiếp với cơ sở dữ liệu của người dùng, mà ở đây là Firebase Realtime Database.

3.2 Sơ đồ hệ thống



Hình 3.1 - Sơ đồ hệ thống điều khiển.

❖ Giải thích sơ đồ:

Hệ thống cho phép nhiều người sử dụng khác nhau cùng một lúc. Mỗi người dùng cần có một tài khoản để đăng nhập vào hệ thống thông qua Web App hoặc Mobile App. Sau đó tiến hành kết nối các thiết bị công tắc điều khiển từ xa vào tài khoản của mình, sau đó mỗi người sử dụng có thể điều khiển các thiết bị của mình một cách an toàn.

Hệ thống sử dụng tính năng quản lý tài khoản sử dụng email và mật khẩu được cung cấp bởi Firebase, cho phép việc quản lý dễ dàng hơn và không bị hạn chế về số lượng người dùng.

3.3 Phân tích và lựa chọn phương án

3.3.1 Lựa chọn nguồn

Mục tiêu đề tài là bộ công tắc âm tường, vậy nên thiết bị phải được cấp nguồn từ nguồn điện xoay chiều AC, được thiết kế để phù hợp điện áp xoay chiều phổ biến ở nhiều quốc gia là 110VAC và 220VAC, ngoài ra đầu ra cần phải được giảm áp xuống 5VDC và 3.3VDC để cung cấp nguồn nuôi vi điều khiển và các ngoại vi khác.

❖ Các phương án được đặt ra:

- *Nguồn xung giảm áp 220VAC/5VDC loại thông thường:* Đây là các loại nguồn xung có giá thành rất rẻ có trong thị trường, được thiết kế là một module mạch không có vỏ bảo vệ, nguồn xung loại này có chất lượng kém dẫn đến tuổi thọ thiết bị không cao. Ngoài ra nguồn chỉ hỗ trợ cho điện áp cố định là 110VAC hoặc 220VAC.
- *Thiết kế nguồn xung giảm áp trên mạch:* Đây là phương án thiết kế các linh kiện của nguồn xung ngay trên mạch điều khiển, làm giảm giá thành và tự chủ được các giá trị linh kiện cũng như thiết kế làm cho mạch hoạt động ổn định hơn. Điểm yếu của phương án này là hầu như

nguồn cần các linh kiện cắm, làm tăng thời gian gia công mạch, đồng thời khó bảo trì sửa chữa.

- *Module nguồn AC-DC Hi-Link 5VDC 3W*: Đây là module nguồn có chân cắm có thể thiết kế ngay trên mạch, ngoài ra module này còn được tích hợp sẵn các linh kiện bên trong vỏ bảo vệ, làm cho mạch nhỏ gọn, độ thẩm mỹ cao và chuyên nghiệp hơn. Ngoài ra module này còn hỗ trợ dây điện áp AC từ 100-240VAC / 50-60Hz.

➤ Trong các phương án trên, lựa chọn nguồn Hi-Link 5VDC 3W là lựa chọn tối ưu nhất cho người thiết kế cũng như người vận hành sửa chữa và thay thế.

3.3.2 Lựa chọn vi điều khiển

Mô hình được hướng đến là bộ công tắc được điều khiển bằng Wifi, vậy nên việc sử dụng các loại vi điều khiển có tích hợp Wifi làm trung tâm là tối ưu nhất. Trong đó thì ESP8266 là lựa chọn tốt do giá thành rẻ, nhiều phiên bản để lựa chọn, ngoài ra còn cung cấp đủ ngoại vi và bộ nhớ cho các tính năng cần thiết.

❖ Các phương án được đặt ra:

- *ESP8266 NodeMCU*: Đây là mạch tích hợp vi điều khiển ESP8266 cùng với mạch nguồn ổn áp DC cùng với cổng giao tiếp qua USB, là một mạch được ưa chuộng trong việc phát triển ứng dụng cho ESP8266. Tuy nhiên giá thành tương đối cao và có dư các chức năng không dùng tới làm cho mạch này không được nhỏ gọn.
- *ESP8266 ESP-01S*: Đây là module nhỏ gọn nhất của ESP8266, tuy nhiên module này chỉ hỗ trợ bộ nhớ flash ngoài là 1MB, làm cho việc phát triển bị giới hạn so với các dòng module khác. Ngoài ra điểm yếu của module này đó là không ra đầy đủ chân GPIO để có thể thiết kế mạch.

- *ESP8266 ESP-12F*: Đây cũng là một module nhỏ gọn của ESP8266, có anten được thiết kế ngay trên mạch. Điểm mạnh của module này chính là hỗ trợ bộ nhớ flash ngoài lên đến 4MB và ra chân đầy đủ với hình thức là một linh kiện dán, giúp việc thiết kế và gia công dễ dàng hơn.

➤ **Như vậy, rõ ràng ESP8266 ESP-12F là sự lựa chọn tốt nhất và khả quan nhất so với các dòng tương đương. Đây cũng là lựa chọn được sử dụng cho đề tài này.**

3.3.3 Lựa chọn nút nhấn

Mô hình là bộ công tắc âm tường nên việc có nút nhấn trực tiếp ngay trên mô hình dùng để điều khiển bật tắt là điều quan trọng. Trong đó có 2 phương án được đặt ra là sử dụng nút nhấn vật lý thông thường và sử dụng nút nhấn cảm biến điện dung. Nút nhấn cảm biến điện dung được sử dụng sẽ làm cho thiết bị trở nên hiện đại và tiện nghi hơn so với các loại nút nhấn vật lý cũ thông thường.

➤ **Vì các lý do đã nêu, nút nhấn cảm ứng điện dung sẽ được chọn làm phương án cho thiết kế nút nhấn ngay trên mô hình.**

3.3.4 Lựa chọn relay

Trong các loại relay có trên thị trường, relay HF46 có thiết kế nhỏ gọn, phù hợp với cho việc thiết kế các mạch điện bị hạn chế về kích thước. Ngoài ra relay HF46 đáp ứng được nhu cầu sử dụng điện sinh hoạt như đèn, quạt với công suất thực có thể lên đến 200W.

➤ **Vì vậy việc lựa chọn relay HF46 cho thiết kế mô hình của đề tài là tối ưu về kích thước cũng như giá thành sản phẩm.**

Chương 4: Thiết kế phần cứng

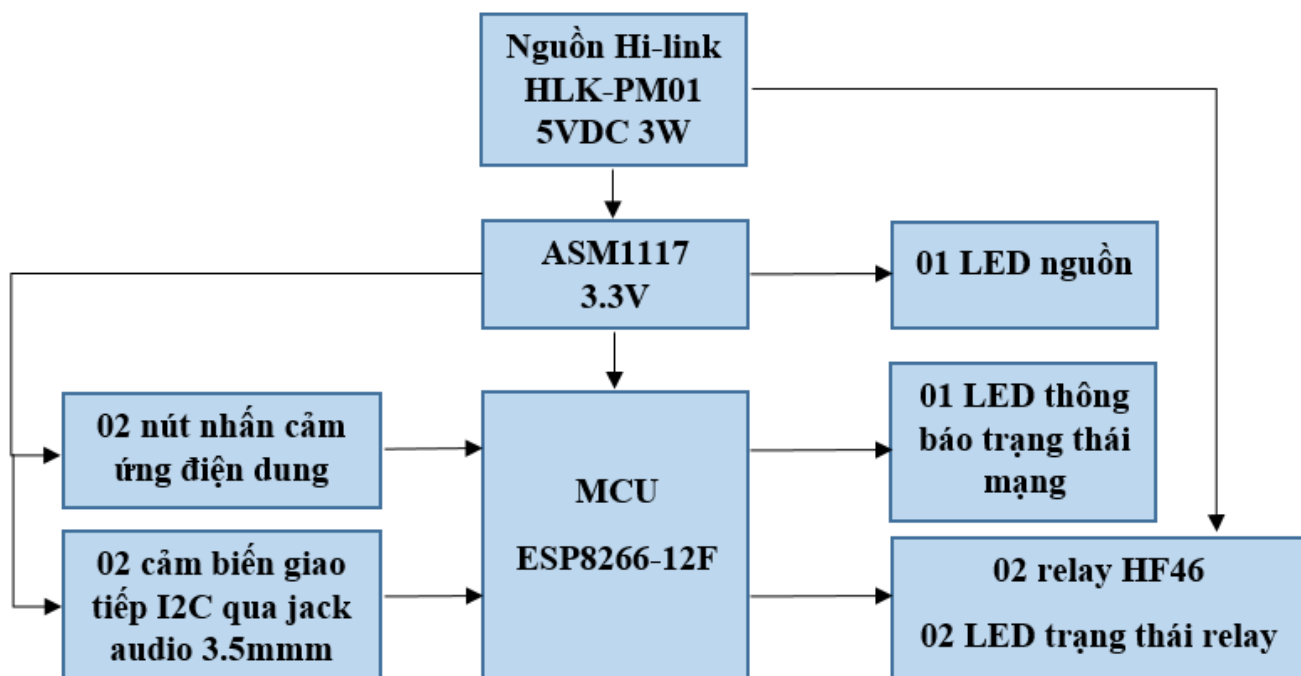
4.1 Yêu cầu thiết kế

Sau khi phân tích và lựa chọn phương án, hệ thống có yêu cầu kỹ thuật phần cứng như sau:

- Thiết bị có thể cấp nguồn 220VAC, sử dụng 01 module Hi-link HLK-PM01 5VDC 3W để chuyển điện áp từ 220VAC từ điện lưới thành 5VDC để cấp nguồn cho mạch điện với công suất là 3W.
- Thiết bị có 02 relay HF46 điện áp kích 5VDC, được điều khiển tắt mở thông qua Opto cách ly quang học PC817.
- Thiết bị có 02 nút nhấn điện dung dùng để điều khiển bật tắt relay trực tiếp ngay trên hộp.
- Thiết bị có 01 Module Wifi SoC ESP8266-12F để kết nối Wifi, cho phép bật tắt relay thông qua Internet.
- Sử dụng IC giảm áp ASM1117 3.3V để hạ áp từ 5VDC xuống thành 3.3VDC cấp nguồn cho Module ESP8266-12F.
- Thiết bị có 02 cổng Audio 3.5mm giao tiếp I2C dùng để hỗ trợ tích hợp cảm biến cho việc phát triển sau này.
- Thiết bị có 04 đèn LED, trong đó có 02 đèn LED thông báo trạng thái bật tắt của relay, 01 đèn LED thông báo trạng thái mạng, 01 đèn LED nguồn.
- Ra hàng rào có các chân TX, RX, GND của module ESP8266-12F dùng để nạp chương trình, có 01 Connector 6 chân để cấp nguồn AC và ra chân relay.
- Ngoài ra thiết bị còn có các linh kiện như điện trở, tụ điện giúp mạch hoạt động ổn định.

4.2 Sơ đồ khối của mạch

4.2.1 Sơ đồ khối chi tiết



Hình 4.1 - Sơ đồ khối chi tiết của mạch.

4.2.2 Khối nguồn Hi-link 5VDC 3W

❖ *Vai trò: Module nguồn AC-DC Hi-link HLK-PM01 5VDC 3W có thiết kế nhỏ gọn với vỏ bọc nhựa an toàn, chuyên nghiệp, được sử dụng để chuyển nguồn xoay chiều AC sang 5VDC với công suất lên đến 3W để cấp nguồn cho mạch điện.*

❖ *Thông số kỹ thuật:*

- Điện áp ngõ vào: 100V – 240VAC /50-60Hz.
- Điện áp ngõ ra: 5VDC.
- Công suất trung bình: 3W.

❖ *Hình ảnh thực tế:*



Hình 4.2 - Module nguồn AC-DC Hi-link HLK-PM01 5VDC 3W [22].

4.2.3 Khối nguồn ASM1117 3.3V

❖ *Vai trò:* IC ổn áp 3.3VDC ASM1117 được sử dụng để tạo điện áp 3.3VDC/800mA để cấp nguồn cho mạch, được sử dụng để cấp điện áp 3.3VDC giúp cho module MCU ESP8266-12F hoạt động.

❖ *Thông số kỹ thuật:*

- Điện áp ngõ vào: 4.5-9VDC.
- Điện áp ngõ ra: 3.3VDC, 800mA.

❖ *Hình ảnh thực tế:*



Hình 4.3 - IC ổn áp ASM1117 3.3V [23].

4.2.4 *Khởi MCU ESP8266-12F*

❖ *Vai trò: Module thu phát Wifi SoC ESP8266-12F có nhân xử lý bên trong là IC ESP8266, được sử dụng để kết nối Wifi và là trung tâm điều khiển chính của mạch.*

❖ *Thông số kỹ thuật cơ bản:*

- Điện áp sử dụng: 3.0-3.6VDC (thường được sử dụng ở 3.3VDC).
- Dòng điện tiêu thụ: có thể lên đến 170mA.
- SRAM size: 36KB.
- ROM size: 4MB (sử dụng IC bộ nhớ flash ngoài thông qua giao tiếp I2C).
- Hỗ trợ Wifi tần số 2.4GHz, chuẩn kết nối 802.11 b/g/n.
- 10 chân GPIO có thể sử dụng, hỗ trợ các giao tiếp cơ bản như SPI, UART, I2C.

❖ *Hình ảnh thực tế:*



Hình 4.4 - Module thu phát Wifi SoC ESP8266-12F [24].

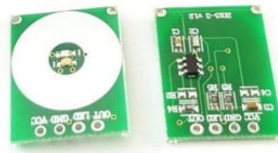
4.2.5 *Khởi nút nhấn cảm ứng điện dung*

❖ *Vai trò: Module cảm ứng một chạm điện dung được sử dụng trong thiết kế công tắc cảm ứng điện dung nhằm cho phép bật tắt thiết bị ngay tại bộ công tắc điều khiển, thay cho các công tắc vật lý thông thường, làm cho sản phẩm hiện đại và tiện nghi hơn.*

❖ **Thông số kỹ thuật:**

- Điện áp làm việc: 3-5VDC.
- Dòng điện tiêu thụ: 0.025mA.
- Cảm ứng xuyên qua các bề mặt phi kim có độ dày từ 3-8mm tùy chất liệu.

❖ **Hình ảnh thực tế:**



Hình 4.5 - Module cảm ứng chạm điện dung [25].

4.2.6 Khôi relay và LED trạng thái relay

❖ **Vai trò:** Relay đóng vai trò quan trọng trong việc bật tắt các thiết bị, mạch sử dụng relay HF46 5VDC 7A với kích thước nhỏ gọn hơn so với các relay thông thường, dùng để bật tắt cho các tải AC với công suất khoảng 200W.

❖ **Thông số kỹ thuật:**

- Điện áp kích: 5VDC.
- Thông số tải: 7A 250VAC hoặc 7A 30VDC.

❖ **Hình ảnh thực tế:**

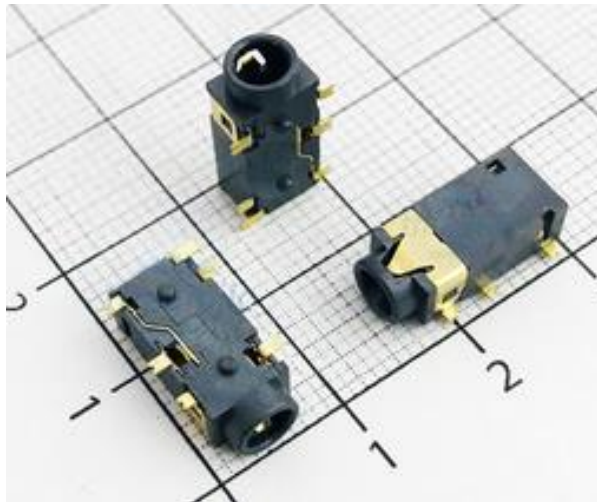


Hình 4.6 - Relay HF46F-G 5VDC [26].

4.2.7 Khối hỗ trợ cảm biến giao tiếp I2C qua jack 3.5mm

❖ *Vai trò: Khối cảm biến giao tiếp I2C giúp hỗ trợ người dùng có thể kiểm soát được các giá trị mà cảm biến mang lại như nhiệt độ, độ ẩm, ánh sáng... thông qua jack Audio 3.5mm thuận tiện để người dùng có thể tháo lắp.*

❖ *Hình ảnh thực tế:*



Hình 4.7 - Cổng Audio 3.5mm 4 Cực 6 Chân Dán SMD [27].

4.2.8 Khối LED nguồn và trạng thái mạng

❖ *Vai trò: Mạch có LED nguồn để thông báo trạng thái hoạt động của mạch, ngoài ra còn có LED báo trạng thái mạng, khi sáng lên có nghĩa là mạch đang tiến hành giao tiếp mạng.*

❖ *Ngoài ra còn có một số chức năng để kiểm tra tình trạng của mạch điện như sau:*

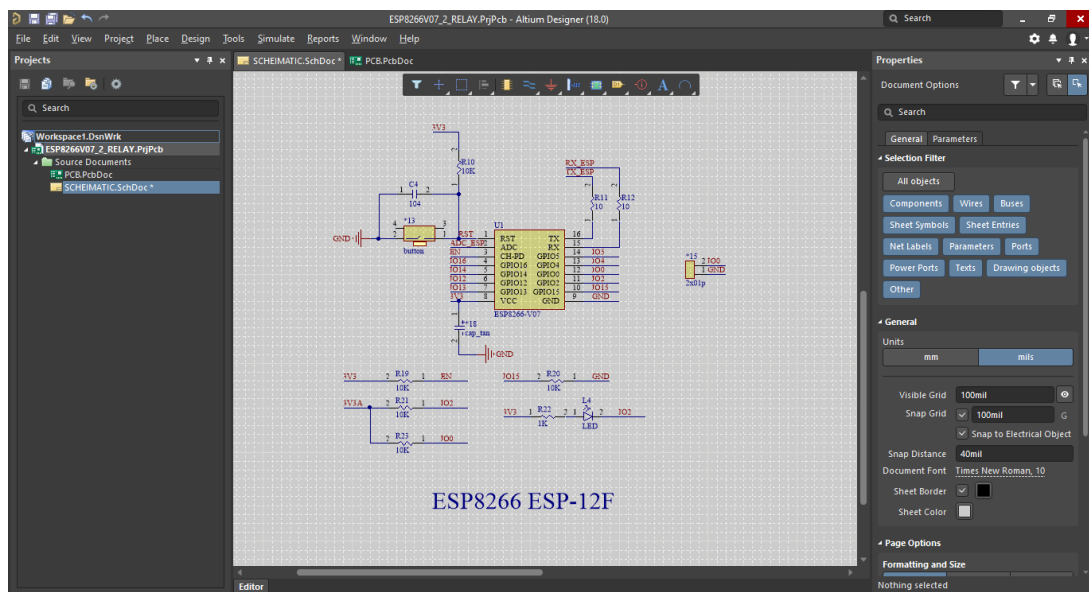
- LED mạng nhấp theo chu kỳ 1 giây: mạch đang ở trạng thái tìm và kết nối với Wifi được cài đặt.
- LED mạng nhấp theo chu kỳ 2 giây: mạch chưa tìm thấy được dữ liệu người dùng, có thể xảy ra khi người dùng không tồn tại.

- LED mạng nháy theo chu kỳ 3 giây: mạch đã kết nối được với Firebase nhưng vẫn chưa thiết lập được đường truyền, có thể do trạng thái mạng yếu.
- LED mạng bật tắt theo chu kỳ 1 giây: mạch đang ở chế độ cấu hình.
- LED mạng tắt: mạch hoạt động kết nối bình thường.

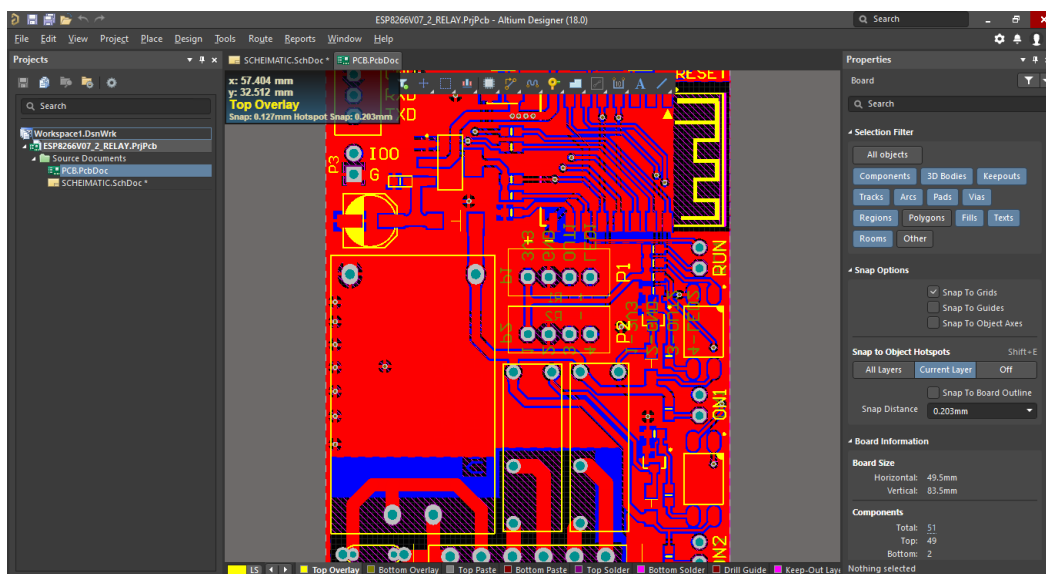
4.3 Sơ đồ nguyên lý mạch và thiết kế PCB

4.3.1 Giới thiệu về Altium Designer

Altium Designer, hay còn được gọi tắt là Altium, là một trong những phần mềm thiết kế mạch điện tử mạnh nhất hiện nay. Được phát triển bởi hãng Altium Limited, đây là một phần mềm chuyên ngành được sử dụng trong thiết kế nguyên lý mạch điện tử cũng như vẽ layout PCB cho thiết kế nguyên lý.



Hình 4.8 - Giao diện của phần mềm Altium Designer 18.0.

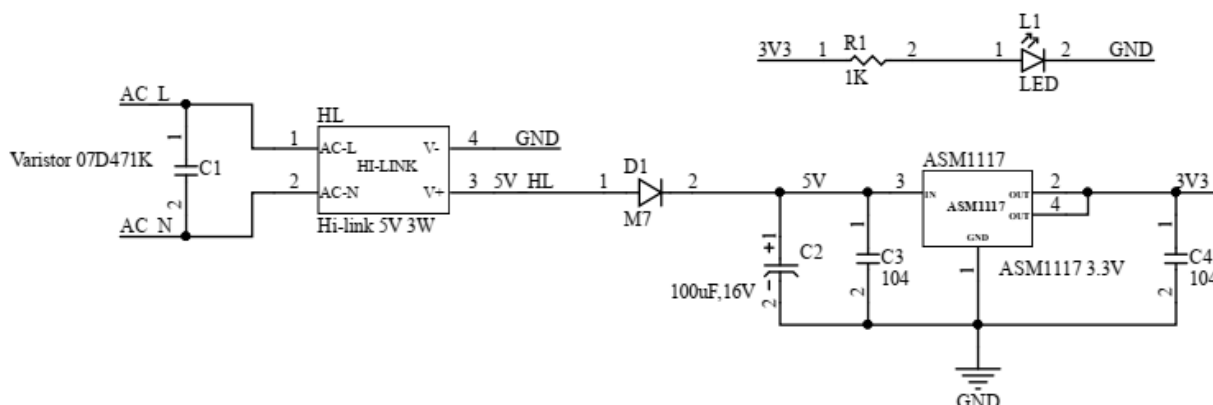


Hình 4.9 - Giao diện thiết kế PCB của Altium Designer.

4.3.2 Sơ đồ nguyên lý mạch

4.3.2.1 Khối nguồn

Khối nguồn được thiết kế để cấp ngõ vào là điện xoay chiều 220VAC, sau đó được biến đổi thành 5VDC thông qua module Hi-link 220VAC/5VDC để cấp nguồn DC 5VDC cho khối relay và opto, ngoài ra để cấp nguồn cho vi điều khiển hoạt động cần phải ổn áp từ 5VDC xuống thành 3.3VDC thông qua IC ổn áp tuyến tính ASM1117 3.3VDC.



Hình 4.10 - Sơ đồ nguyên lý khối nguồn.

Các linh kiện được thiết kế và tính toán có các chức năng như sau:

Bảng 4.1 - Danh sách linh kiện khối nguồn.

Linh kiện	Giá trị	Chức năng
C1	07D471K	Tụ chống sét.
HL	Hi-link 5V 3W	Biến áp 220VAC/5VDC 3W.
D1	M7	Diode chỉnh lưu chống dòng ngược.
C2	100uF	Tụ nhôm lọc nguồn ngõ vào cho ASM1117.
C3	100nF	Tụ gốm lọc nguồn ngõ vào cho ASM1117.
ASM1117		Ổn áp tuyến tính 3.3VDC.
C4	100nF	Tụ gốm lọc ngõ ra của ASM1117.
R1	1 kOhm	Điện trở hạn dòng cho LED L1.
L1		Đèn led báo trạng thái nguồn.

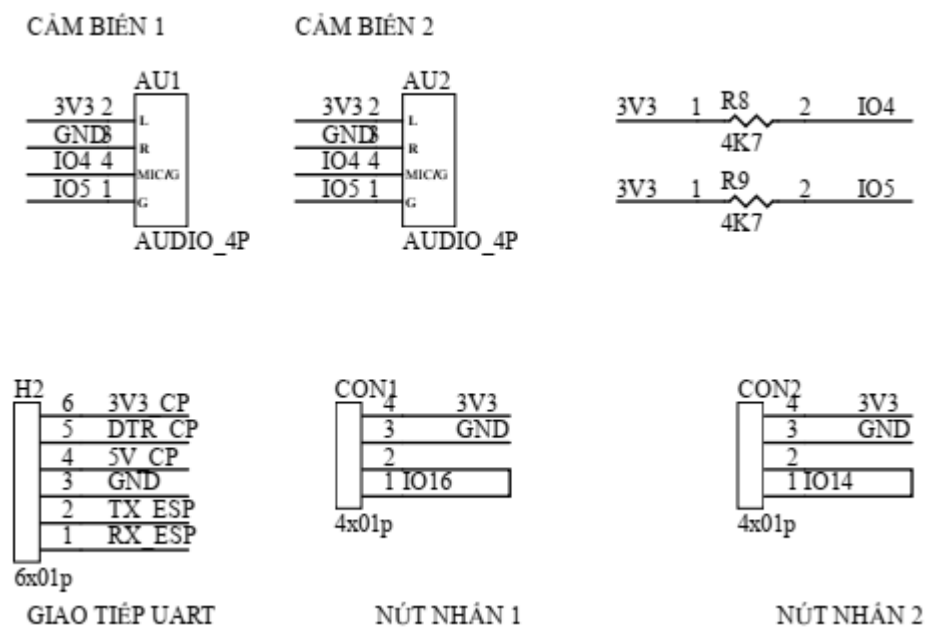
4.3.2.2 Khối vi điều khiển ESP8266 ESP-12F

Khối vi điều khiển được thiết kế để đảm bảo cho vi điều khiển ESP8266 ESP-12F cùng với các ngoại vi hoạt động ổn định. Nguồn nuôi của khối vi điều khiển là nguồn 3.3VDC được lấy từ ngõ ra của IC ổn áp tuyến tính ASM1117 3.3VDC. Ngoài ra khối điều khiển còn có một đèn LED được điều khiển bằng chân IO2 trên module.

R4	10 kOhm	Điện trở kéo lên cho chân IO2 dùng để điều khiển LED L4.
R5	10 kOhm	Điện trở kéo lên cho chân IO0 dùng để chọn giữa chức năng hoạt động hoặc nạp chương trình.
R6	1 kOhm	Điện trở kéo lên cho chân IO15.
R7	10 kOhm	Điện trở hạn dòng cho LED L4.
H1		Header dùng để nối IO0 và GND trong quá trình nạp chương trình cho vi điều khiển.

4.3.2.3 Khối nút nhấn và giao tiếp

Khối nút nhấn và giao tiếp cảm biến thông qua chuẩn giao tiếp I2C bao gồm 02 jack cắm audio 3.5mm 4 chân dùng để cắm các loại cảm biến, 02 header cắm 4 chân dùng để cắm module cảm biến chạm điện dung, ngoài ra khối này còn có cổng giao tiếp UART để giúp nạp chương trình dễ dàng hơn trong quá trình phát triển.



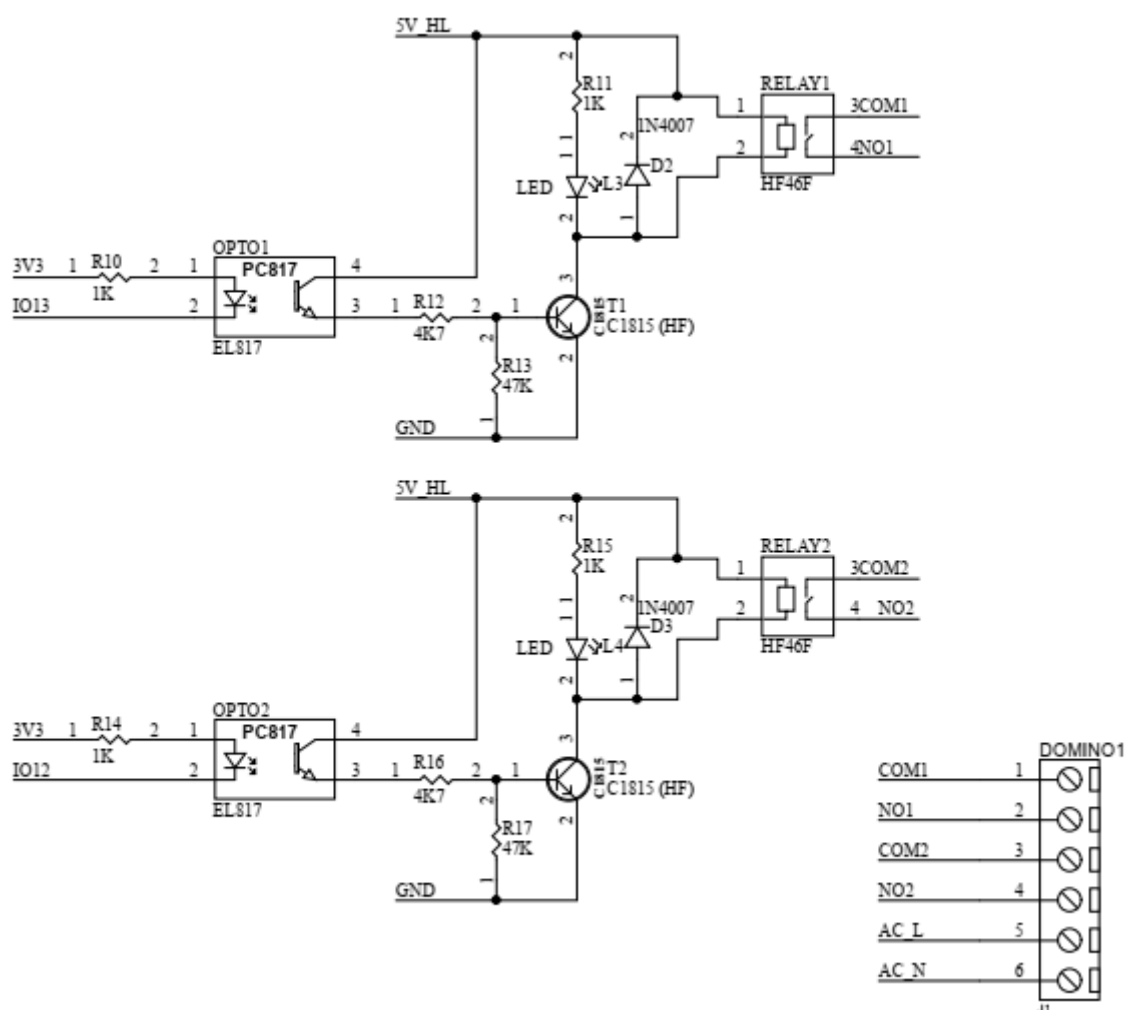
Hình 4.12 - Sơ đồ nguyên lý khối nút nhấn và cảm biến.

Các linh kiện được thiết kế và tính toán có các chức năng như sau:

Bảng 4.3 - Danh sách linh kiện khối nút nhấn và cảm biến.

Linh kiện	Giá trị	Chức năng
AU1, AU2		2 Jack Audio 3.5mm dùng để cắm cảm biến giao tiếp I2C.
H2		Header 6 chân dùng để cắm mạch giao tiếp UART CP2102.
CON1, CON2		2 Connector 4 chân dùng để cắm các nút nhấn chạm điện dung.
R8, R9	4.7 kOhm	2 điện trở kéo lên cho chân SDA và SCL.

4.3.2.4 Khối relay và opto



Hình 4.13 - Sơ đồ nguyên lý khối relay và opto.

Khối relay và opto được thiết kế trên nguyên tắc cách ly tín hiệu điều khiển với mạch relay bằng opto cách ly quang học PC817 nhằm chống nhiễu và dòng ngược đi về phía chân điều khiển, khối này còn được thiết kế giúp relay hoạt động ổn định và tránh các tình trạng đánh thủng phân cực thuận trên transistor, gây hại đến mạch điều khiển và toàn bộ mạch điện. Ngoài ra khối relay và opto còn tích hợp LED báo trạng thái bật tắt của relay giúp người dùng dễ dàng kiểm tra được trạng thái bật tắt của thiết bị trên mô hình.

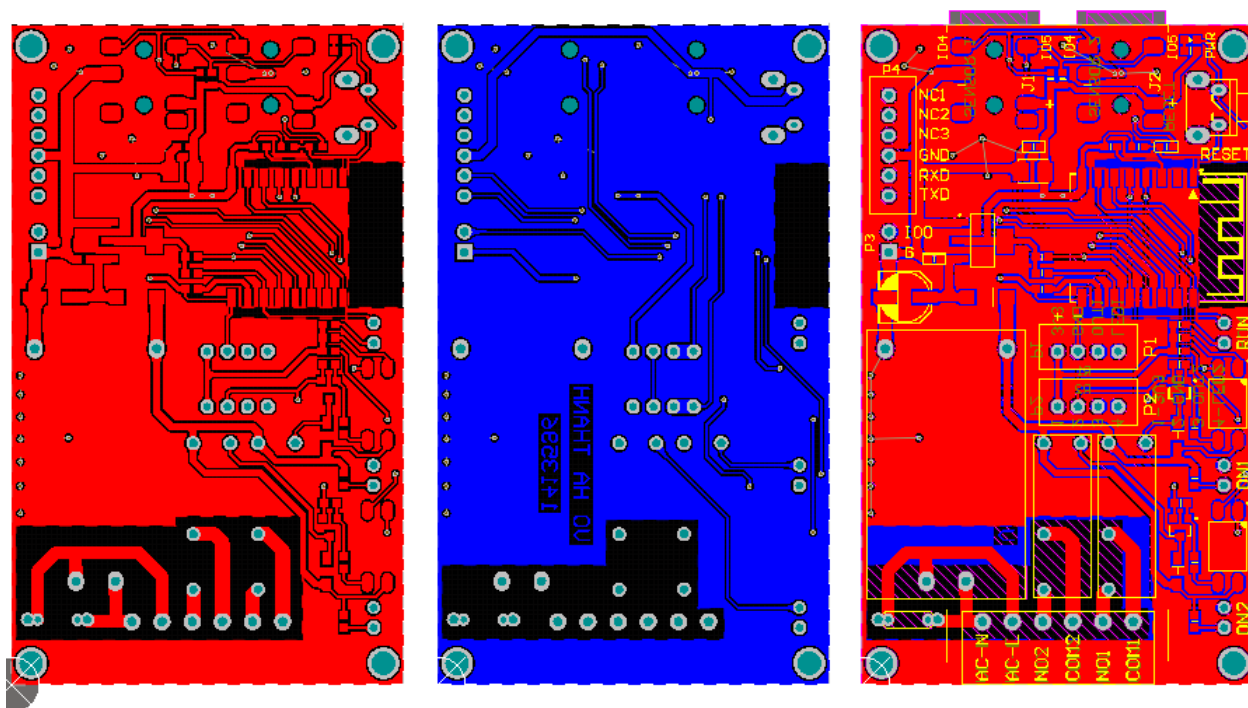
Các linh kiện được thiết kế và tính toán có các chức năng như sau:

Bảng 4.4 - Danh sách linh kiện trong khối relay và opto.

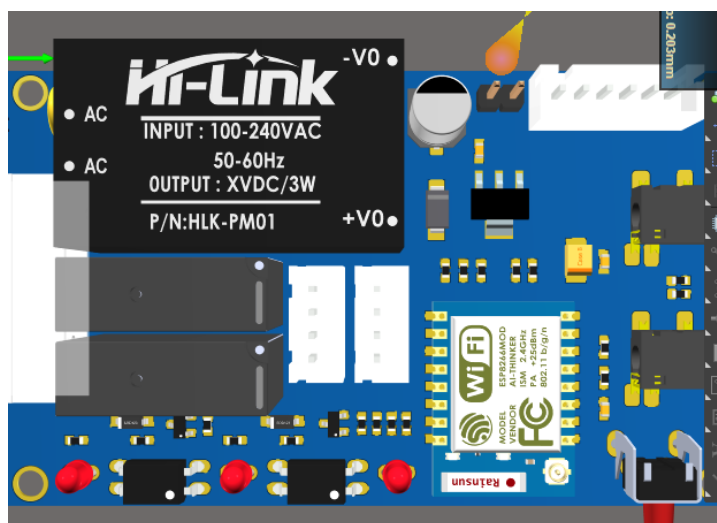
Linh kiện	Giá trị	Chức năng
R10, R14	1 kOhm	Điện trở hạn dòng chân tín hiệu kéo lên 3.3VDC.
OPTO1, OPTO2	EL817	Opto cách ly quang học, dùng để cách ly tín hiệu điều khiển và mạch relay.
R12, R16	4.7 kOhm	Điện trở phân cực cho cực B của transistor.
R13, R17	47 kOhm	Điện trở phân áp cho cực B của transistor.
R11, R15	1 kOhm	Điện trở hạn dòng cho LED L3 và L4.
L3		Đèn LED báo trạng thái bật tắt cho RELAY1.
L4		Đèn LED báo trạng thái bật tắt cho RELAY2.
D2, D3	1N4007	Diode chỉnh lưu giúp bảo vệ transistor không bị relay đánh thủng phân cực thuận.
T1, T2	HF (C1815)	Transistor C1815 NPN dùng để điều khiển bật tắt RELAY1 và RELAY2.

4.3.3 Thiết kế và vẽ PCB

Sau khi hoàn thành vẽ sơ đồ nguyên lý của mạch điện, tiến hành tạo PCB Project trong Altium Designer để đi layout cho mạch nguyên lý đã vẽ.



Hình 4.14 - PCB 2 lớp sau khi được thiết kế.



Hình 4.15 - Mô hình mạch 3D xây dựng trên Altium Designer.

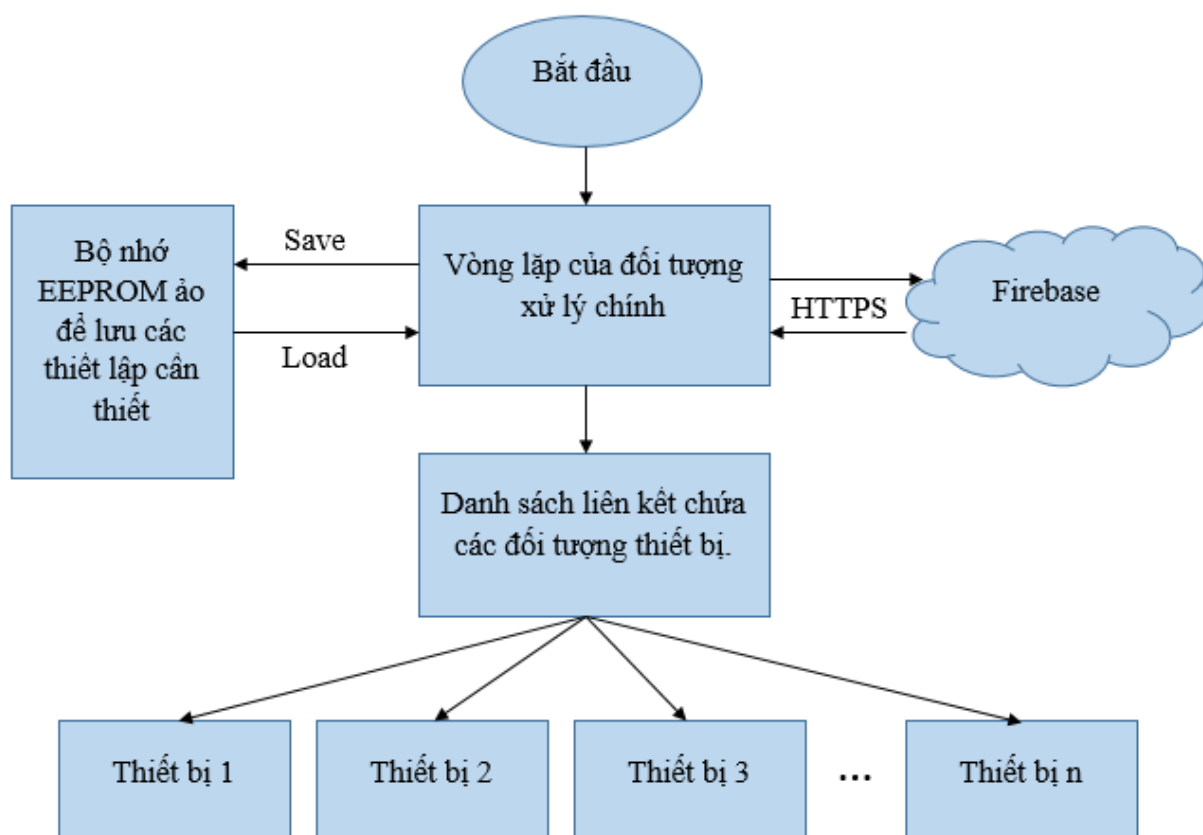
Chương 5: Thiết kế phần mềm

5.1 Yêu cầu kỹ thuật

- Hệ thống người dùng được xây dựng dựa trên Firebase Authentication giúp người dùng bảo mật được thông tin và được cấp quyền điều khiển các thiết bị trong phạm vi cho phép. Ngoài ra hệ thống người dùng còn hỗ trợ số lượng người dùng lớn và được liên kết đến các thiết bị điều khiển của mình.
- Hệ thống người dùng bao gồm đăng ký, đăng nhập và tạo lại mật khẩu thông qua email.
- Người dùng có thể điều khiển thiết bị được cấu hình qua mạng internet thông qua Web App hoặc Mobile App.
- Hỗ trợ trang cấu hình cho thiết bị giúp người dùng dễ dàng cấu hình Wifi và kết nối thiết bị vào tài khoản của mình.
- Nhấn giữ các nút nhấn chạm trong vòng 5 giây để vào trang cấu hình.
- Trang cấu hình tự động mở sau khi điện thoại hay máy tính đăng nhập vào mạng được phát ra từ ESP8266 Access Point.
- Trang điều khiển được thiết kế có đủ tính năng để có thể quản lý thiết bị theo cụm hoặc nhóm, giúp việc quản lý trở nên dễ dàng hơn.
- Trang điều khiển hiển thị đầy đủ trạng thái của thiết bị được điều khiển bao gồm On, Off, Offline, Error.

5.2 Thiết kế và lập trình firmware cho ESP8266

5.2.1 Sơ đồ khối tổng quan



Hình 5.1 - Sơ đồ khối tổng quan chương trình trên ESP8266.

5.2.2 Thư viện Firebase-ESP8266

Đây là bộ thư viện được xây dựng bởi Mobizt dựa trên nền tảng của API được hỗ trợ bởi Firebase. Thư viện này cho phép người dùng tạo ra kết nối giữa ESP8266 và Firebase Realtime-Database giúp người dùng có thể dễ dàng kết nối vi điều khiển ESP8266 của mình đến cơ sở dữ liệu và thực hiện các chức năng tương tự như API được hỗ trợ.

Thư viện được xây dựng theo hướng đối tượng và tạo ra một đối tượng Firebase để người sử dụng có thể quản lý thông qua đối tượng đó. Các hàm trong

thư viện được sử dụng để xây dựng nền tảng giao tiếp với Firebase Realtime-Database như sau:

Hàm khởi tạo kết nối:

*void **begin**(const String &host, const String &auth);*

- **host**: Địa chỉ của database.
- **auth**: Mã xác thực của database.

Hàm thiết lập phương thức kết nối:

*void **enableClassicRequest**(FirebaseData &dataObj, bool flag);*

- **dataObj**: Là kiểu dữ liệu hướng đối tượng được định nghĩa trong thư viện chứa các dữ liệu được gửi đi hay nhận dữ liệu trả về trong quá trình kết nối.
- **flag**: Cờ dùng để đảm bảo kết nối không bị chặn bởi Firewall.

Hàm tự động kết nối lại Wifi:

*void **reconnectWiFi**(bool reconnect);*

- **reconnect**: Cờ để bật tự động kết nối lại.

Hàm cài đặt thời gian huỷ gói tin khi thời gian phản hồi quá lâu:

*void **setReadTimeout**(FirebaseData &dataObj, int millisec);*

- **dataObj**: Là kiểu dữ liệu hướng đối tượng được định nghĩa trong thư viện chứa các dữ liệu được gửi đi hay nhận dữ liệu trả về trong quá trình kết nối.
- **millisec**: Thời gian tính bằng mili giây.

Hàm cài đặt kích thước của gói tin:

void setwriteSizeLimit(FirebaseData &dataObj, const String &size);

- **dataObj**: Là kiểu dữ liệu hướng đối tượng được định nghĩa trong thư viện chứa các dữ liệu được gửi đi hay nhận dữ liệu trả về trong quá trình kết nối.
- **size**: Kích thước dữ liệu tính bằng bytes.

Hàm tạo ra cập nhật trả về từ dữ liệu được chỉ định:

bool beginStream(FirebaseData &dataObj, const String &path);

- **dataObj**: Là kiểu dữ liệu hướng đối tượng được định nghĩa trong thư viện chứa các dữ liệu được gửi đi hay nhận dữ liệu trả về trong quá trình kết nối.
- **path**: Đường dẫn trỏ đến dữ liệu.

Các hàm lấy và gửi thông tin đến dữ liệu được chỉ định:

bool get(FirebaseData &dataObj, const String &path);

bool setString(FirebaseData &dataObj, const String &path);

bool getString(FirebaseData &dataObj, const String &path);

bool updateNodeSilent (FirebaseData &dataObj, const String &path);

- **dataObj**: Là kiểu dữ liệu hướng đối tượng được định nghĩa trong thư viện chứa các dữ liệu được gửi đi hay nhận dữ liệu trả về trong quá trình kết nối.
- **path**: Đường dẫn trỏ đến dữ liệu.

5.2.3 Xây dựng thư viện quản lý và điều khiển thiết bị

Sau khi tìm hiểu về thư viện hỗ trợ, tiến hành thực hiện thư viện mới dựa trên thư viện có sẵn. Thư viện mới được xây dựng theo hướng đối tượng nhằm giúp việc quản lý trong lập trình dễ dàng hơn và dễ dàng mở rộng mô hình hay nâng cấp firmware cho thiết bị. Thư viện mới bao gồm 2 Class là ***FirestoreDevice*** và ***FirestoreSmartControlClass***.

FirestoreDevice là Class được xây dựng nhằm tượng trưng và quản lý thông số cho một thiết bị được điều khiển. Các đối tượng được thiết lập theo kiểu Class lưu trữ các thông tin bao gồm các thông tin như chân điều khiển ngoại vi, chân nhận tín hiệu nút nhấn, ngoài ra các đối tượng này còn được điều khiển thông qua các hàm như:

Khởi tạo và lưu lại các thiết lập:

`void load(String params);`

- ***params***: Các thiết lập được lưu ở dạng chuỗi.

Thay đổi trạng thái để bật tắt thiết bị:

`void setDevice(bool device, bool force);`

- ***device***: Trạng thái on/off của thiết bị.
- ***force***: Cờ bắt buộc thay đổi.

Cập nhật trạng thái lên cơ sở dữ liệu:

`void update();`

FirestoreSmartControlClass là cũng là Class được xây dựng theo hướng đối tượng, đối tượng thuộc class này đóng vai trò quan trọng trong việc quản lý dữ liệu cũng như điều khiển các đối tượng thuộc class ***FirestoreDevice***. Đối tượng của ***FirestoreSmartControlClass*** lưu giữ các thông tin về kết nối như thông tin Wifi, thông tin kết nối Firebase được kế thừa từ thư viện hỗ trợ. Ngoài ra, đây cũng

là đối tượng giúp quản lý tất cả về mặt chương trình chính của firmware cho ESP8266. Các hàm chính của đối tượng thuộc ***FirebaseESPSmartControlClass*** bao gồm các chức năng như sau:

Hàm khởi tạo ban đầu:

*static void **begin**(long **baud**);*

- **baud**: Baudrate dùng để kết nối với máy tính.

Hàm xác định chế độ khởi động là sử dụng hoặc thiết lập:

*static bool **modeDetect**();*

Hàm đọc các giá trị thiết lập về kết nối và trạng thái trước đó:

*static void **loadConfig**();*

Hàm lưu lại các giá trị thiết lập về kết nối và trạng thái:

*static void **saveConfig**(String **wifiSSID**, String **wifiPassword**, String **UserID**);*

- **wifiSSID**: Tên Wifi.
- **WifiPassword**: Mật khẩu Wifi.
- **UserID**: Mã người dùng.

Hàm quản lý HTTP Webserver dùng để xử lý yêu cầu truy xuất trang thiết lập:

*static void **handleRoot**();*

Hàm cài đặt thời gian cập nhật trạng thái mới lên cơ sở dữ liệu:

*static void **update**();*

Hàm xử lý cập nhật và yêu cầu cập nhật tất cả các đối tượng ***FirebaseESPDevice***:

*static void **setFirebaseUpdate**(int time, void (***_updateFunction**)(void));*

- **time**: Thời gian tính bằng mili giây.
- ***_updateFunction**: Con trỏ hàm tới hàm cập nhật,

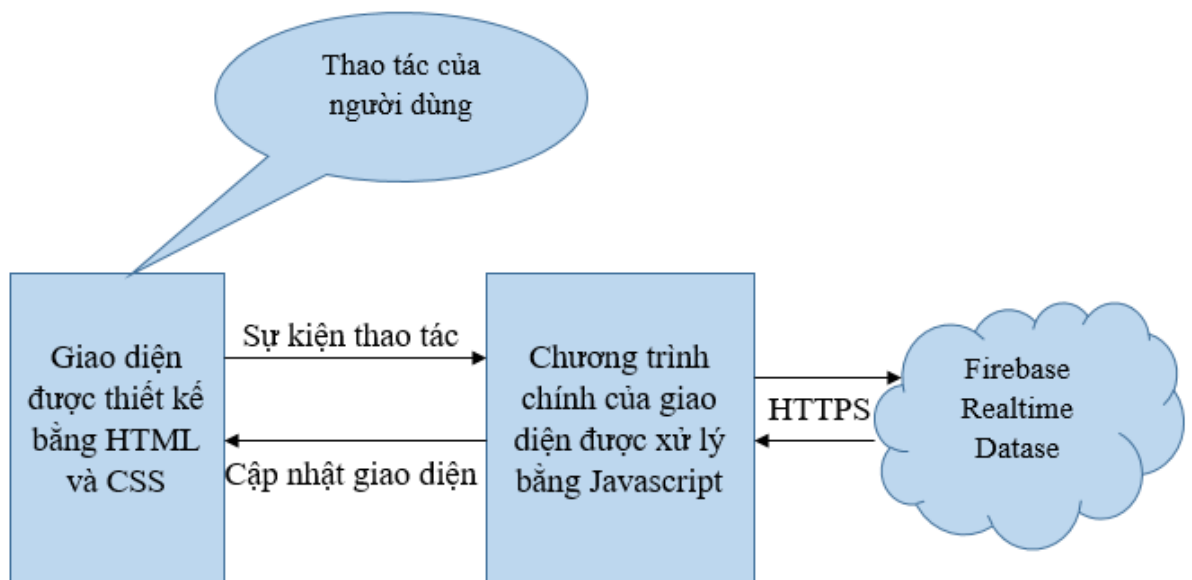
Hàm xử lý chính là vòng lặp dùng để quản lý luồng cho chương trình hoạt động:

```
static bool onProcess();
```

Ngoài các hàm chính và dữ liệu nêu trên, thư viện còn một số hàm phụ và biến để lưu trữ trong quá trình sử dụng.

5.3 Thiết kế và lập trình giao diện người dùng trên website

Giao diện người dùng được thiết kế là một trang web giúp cho người sử dụng có thể sử dụng được ở nhiều nơi và nhiều thiết bị. Trang giao diện được xây dựng trên HTML và được bố trí và trang trí bằng CSS, ngoài ra trang giao diện được lập trình sử dụng Javascript để thực hiện các thao tác người dùng, hiển thị và giao tiếp với Firebase Realtime Database.



Hình 5.2 - Sơ đồ giao diện trang web điều khiển.

Giao diện điều khiển được thiết kế là một trang web nên Javascript đóng vai trò chủ chốt trong quá trình xử lý giao tiếp với con người đồng thời trao đổi thông tin với cơ sở dữ liệu và cập nhật hiển thị lên giao diện. Một số hàm chính trong quá trình điều khiển bao gồm:

Hàm xử lý đăng nhập và truyền nhận dữ liệu đến cơ sở dữ liệu của người dùng:

firebase.auth().onAuthStateChanged(firebaseUser);

- ***firebaseUser***: Đối tượng chứa thông tin người dùng.

Các hàm tạo, đăng nhập, đăng xuất và làm mới mật khẩu:

onLogin = function();

onLogout = function();

onRegister = function();

onReset = function();

Các hàm đổi giao diện người dùng:

accountpanel = function();

registerpanel = function();

resetpanel = function();

back = function(id);

- ***id***: id của giao diện trước đó.

openPanel = function(id);

- ***id***: ID của giao diện trước đó.

Các hàm thiết lập và điều khiển thiết bị:

getDeviceByInfo = function(deviceinfo, macAddress);

- ***deviceinfo***: Thông thiết bị dạng chuỗi.
- ***macAddress***: Địa chỉ MAC của ESP8266 trên thiết bị.

createDevicePanel = function(device, motherRoot);

- ***device***: Đối tượng chứa thông tin của thiết bị.
- ***motherRoot***: Đường dẫn trỏ đến dữ liệu trên Firebase:

deviceChange = function(motherRoot, id);

- ***motherRoot***: Đường dẫn trỏ đến dữ liệu trên Firebase:
- ***id***: ID của relay điều khiển.

roomChange = function(motherRoot, id);

- ***motherRoot***: Đường dẫn trỏ đến dữ liệu trên Firebase:
- ***id***: ID của relay điều khiển.

addRoom = function(motherRoot, id);

- ***motherRoot***: Đường dẫn trỏ đến dữ liệu trên Firebase:
- ***id***: ID của relay điều khiển.

renameDevice = function(motherRoot, id);

- ***motherRoot***: Đường dẫn trỏ đến dữ liệu trên Firebase:
- ***id***: ID của relay điều khiển.

deleteDevice = function(motherRoot, id);

- ***motherRoot***: Đường dẫn trỏ đến dữ liệu trên Firebase:
- ***id***: ID của relay điều khiển.

Chương 6: Kết quả và hướng phát triển

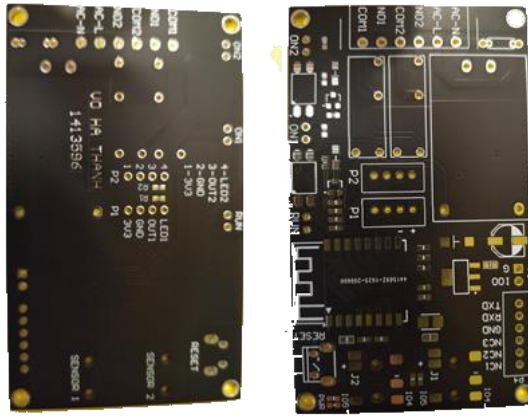
6.1 Kết quả thiết kế phần cứng

Sau khi thiết kế nguyên lý mạch điện cũng như thiết kế PCB cho sản phẩm và tiến hành đặt mạch từ nước ngoài, kết quả đạt được của mạch điện đã được thực hiện gồm các tính năng sau:

- Mạch sử dụng nguồn 220VAC từ điện lưới để hoạt động.
- Mạch có 2 relay HF46F cùng opto dùng để bật tắt thiết bị, hỗ trợ ra cổng kết nối với NO và COM của các relay.
- Mạch có 2 cổng port 4 chân dùng để kết nối với nút nhấn chạm điện dung, ngoài ra còn 1 nút nhấn reset sử dụng trong quá trình phát triển.
- Mạch có hàng rào 6 chân dùng để nạp chương trình cho ESP8266 trong quá trình thực hiện, phát triển cũng như nâng cấp sau này.
- Mạch có 2 cổng Audio 3.5mm 4 chân có kết nối I2C dùng để gắn cảm biến cho việc hỗ trợ giao tiếp cảm biến trong tương lai.
- Mạch có đèn LED thông báo trạng thái relay cũng như đèn LED thông báo tình trạng thiết bị.



Hình 6.1 - Mạch sau khi được thi công.



Hình 6.2 - Mạch in sau khi thi công.

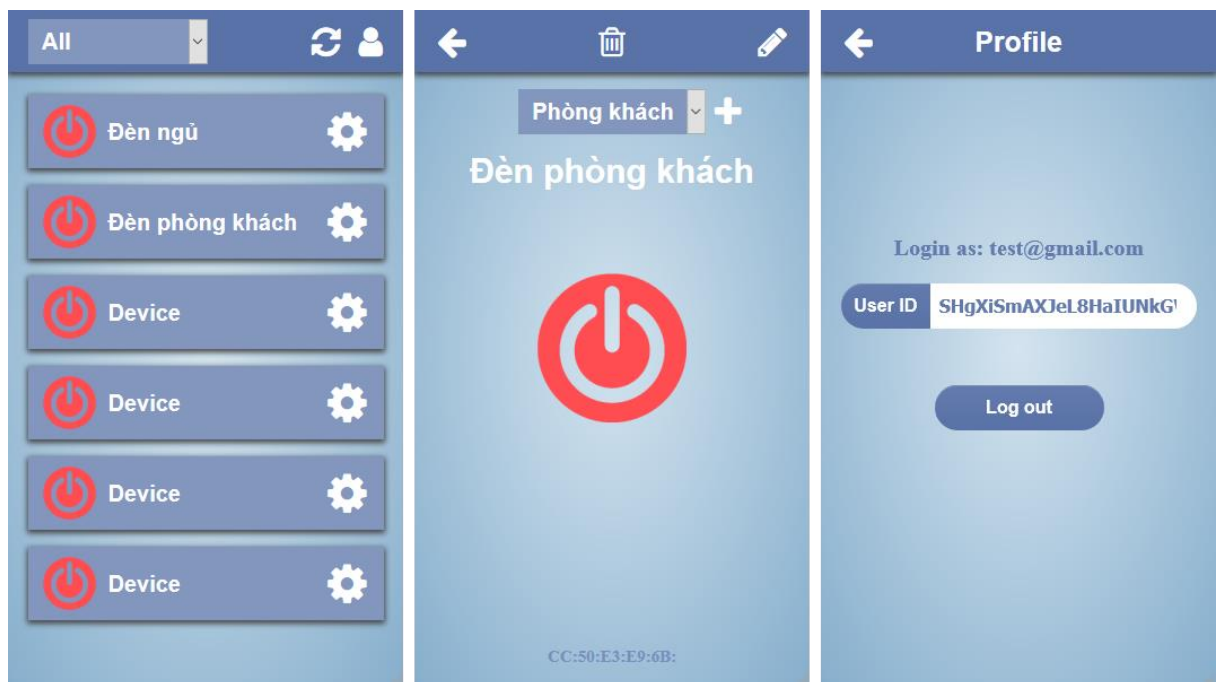
6.2 Kết quả thiết kế phần mềm

Sau khi tiến hành thực hiện phần mềm, kết quả đạt được ban đầu đã có thể giúp người dùng điều khiển được các thiết bị từ xa thông qua trang web điều khiển.

Kết quả thực hiện đạt được bao gồm:

- Đã xây dựng được hệ thống người dùng có đầy đủ các chức năng quản lý tài khoản như đăng ký, đăng nhập sử dụng email, ngoài ra hệ thống còn hỗ trợ thay đổi mật khẩu thông qua email.
- Hệ thống cho phép nhiều người dùng sử dụng, mỗi người dùng có thể điều khiển nhiều thiết bị rời rạc của mình sau khi cấu hình liên kết thiết bị đến cơ sở dữ liệu có Firebase thành công.
- Đã xây dựng được trang quản lý thiết bị có các tính năng điều khiển thiết bị qua mạng internet, đồng thời có chức năng phân chia thiết bị theo nhóm và đặt tên cho nhóm lần thiết bị giúp người sử dụng dễ dàng quản lý hơn.
- Xây dựng thành công Mobile App sử dụng Flutter để nhúng trang web quản lý đã có vào ứng dụng.
- Trên thiết bị vật lý, thiết kế được trang cấu hình dùng để cấu hình Wifi cũng như liên kết đến cơ sở dữ liệu của tài khoản người dùng.

- Người sử dụng có thể vào trang cấu hình bằng cách nhấn giữ tất cả các nút nhấn chạm trong vào 5 giây sau đó đăng nhập vào Wifi của Access Point phát ra từ ESP8266, người dùng cần đăng nhập vào Wifi đó để mở trang cấu hình.



Hình 6.3 - Giao diện điều khiển trên điện thoại.



Hình 6.4 - Giao diện điều khiển trên máy tính.

6.3 Kết luận và phương hướng phát triển

Sản phẩm được tạo ra cơ bản đã đáp ứng được nhu cầu điều khiển không dây thông qua mạng internet và đáp ứng hầu hết các yêu cầu kỹ thuật đã đề ra, song vẫn còn nhiều mặt hạn chế cần được nâng cấp cũng như phát triển thêm các tính năng mới trong tương lai. Phương hướng phát triển chính sẽ bao gồm các tính năng mới cũng như sửa lại các mặt hạn chế của sản phẩm nhằm giúp các thiết bị trở nên hiện đại hơn và phù hợp với nhu cầu của đại đa số người dùng.

Các phương hướng phát triển cụ thể như sau:

- Sửa lại mạch điện tương thích với các vỏ hộp đã được thiết kế theo chuẩn sử dụng, tối ưu giá thành sản phẩm và lựa chọn các loại linh kiện phù hợp hơn và đầy đủ chức năng hơn.
- Hỗ trợ đầy đủ các tính năng quản lý tình trạng thiết bị như On/Off/Offline/Error.
- Hỗ trợ được nhiều loại cảm biến như nhiệt độ, độ ẩm, áp suất, ánh sáng..., đồng thời cho phép điều khiển thiết bị theo giá trị của cảm biến.
- Sử dụng ngắt trên chân nút nhấn chạm để có phản hồi tốt hơn.
- Thêm tính năng hẹn giờ bật tắt thiết bị.
- Thêm tính năng chia sẻ thiết bị với người dùng khác.
- Mobile App hỗ trợ tự động kết nối và cấu hình thiết bị ngay trong ứng dụng và không cần phải có thao tác đăng nhập Wifi để thiết lập.

Ngoài ra, do còn là mô hình nên sản phẩm cần được phát triển nhiều thêm và định hướng lâu dài để có thể trở thành sản phẩm công nghiệp có thể được sử dụng rộng rãi trong cuộc sống và mang lại sự tiện nghi cho người sử dụng.

TÀI LIỆU THAM KHẢO

- [1] "Sonoff Basic," [Online]. Available: <https://sonoff.tech/product/wifi-diy-smart-switches/basicr2>.
- [2] "Sonoff Wall Switches," [Online]. Available: <https://sonoff.tech/product-category/wifi-smart-wall-swithes>.
- [3] "Sonoff TH16," [Online]. Available: <https://sonoff.tech/product/wifi-diy-smart-switches/th10-th16>.
- [4] "Sonoff POW R2," [Online]. Available: <https://sonoff.tech/product/wifi-diy-smart-switches/powr2>.
- [5] "Smart Life," [Online]. Available: <https://vn.smartlife.asia/cong-tac-dieu-khien-dien-thong-minh/>.
- [6] "Shelly 1," [Online]. Available: <https://shelly.cloud/products/shelly-1-smart-home-automation-relay/>.
- [7] "Shelly 2.5," [Online]. Available: <https://shelly.cloud/products/shelly-25-smart-home-automation-relay/>.
- [8] "Introduction to Firebase," [Online]. Available: <https://hackernoon.com/introduction-to-firebase-218a23186cd7>.
- [9] "HTTP vs HTTPS: What's the Difference?," [Online]. Available: <https://www.guru99.com/difference-http-vs-https.html>.
- [10] "Firebase Database REST API," [Online]. Available: <https://firebase.google.com/docs/reference/rest/database#section-api-usage>.
- [11] "RESTful API là gì? Cách thiết kế RESTful API," [Online]. Available: <https://topdev.vn/blog/restful-api-la-gi/>.
- [12] "Introduction to ESP8266," [Online]. Available: <https://www.theengineeringprojects.com/2018/08/introduction-to-esp8266.html>.
- [13] "ESP8266EX Datasheet," [Online]. Available: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf.
- [14] "IC ESP8266EX," [Online]. Available: https://www.alibaba.com/product-detail/chip-RF-wifi-wireless-ic-QNF32_60775933319.html.

- [15] "NodeMCU ESP8266: Details and Pinout," [Online]. Available: <https://www.instructables.com/id/NodeMCU-ESP8266-Details-and-Pinout/>.
- [16] "What is Arduino?," [Online]. Available: <https://www.arduino.cc/en/guide/introduction>.
- [17] "Giới thiệu về HTML," [Online]. Available: <https://timoday.edu.vn/bai-1-gioi-thieu-ve-html/>.
- [18] "Giới thiệu về CSS," [Online]. Available: <https://quantrimang.com/gioi-thieu-ve-css-152825>.
- [19] "Giới thiệu về JavaScript và đoạn mã JavaScript đầu tiên," [Online]. Available: <https://xuanthulab.net/gioi-thieu-ve-javascript-va-doan-ma-javascript-dau-tien.html>.
- [20] "Giới thiệu Ngôn ngữ lập trình C," [Online]. Available: <http://itprotraining.vn/vi/lap-trinh-c-11/gioi-thieu-ngon-ngu-lap-trinh-c>.
- [21] "Giới thiệu về ngôn ngữ lập trình C++," [Online]. Available: <https://daynhauhoc.com/t/gioi-thieu-ve-ngon-ngu-lap-trinh-c/29448>.
- [22] "Module nguồn AC-DC Hi-Link HLK-PM01 5VDC 3W," [Online]. Available: <https://hshop.vn/products/module-nguon-ac-dc-hi-link-hlk-pm01-5vdc-3w>.
- [23] "AMS1117 1A LDO Regulator – Fixed/Variable," [Online]. Available: <https://components101.com/regulators/ams1117-1a-ldo-regulator-pinout-datasheet>.
- [24] "Mạch thu phát Wifi SoC ESP8266 ESP-12F Ai-Thinker," [Online]. Available: <https://hshop.vn/products/mach-thu-phat-wifi-soc-esp8266-v12>.
- [25] "Cảm ứng 1 chạm điện dung Capacitive Touch Button 3~5VDC," [Online]. Available: <https://hshop.vn/products/camung-1-chamdien-dung>.
- [26] "HF46F-G 5-HS1 Relay Không Chốt SPST-NO 5V 7A," [Online]. Available: <https://www.thegioiic.com/products/hf46f-g-5-hs1-relay-khong-chot-spst-no-5v-7a>.
- [27] "PJ-342S Cổng Audio 3.5mm 4 Cực 6 Chân Dán SMD," [Online]. Available: <https://www.thegioiic.com/products/pj-342s-cong-audio-3-5mm-4-cuc-6-chan-dan-smd>.