

ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ
BỘ MÔN ĐIỆN TỬ

-----o0o-----



LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

MÔ HÌNH NHÀ THÔNG MINH ĐIỀU KHIỂN
THIẾT BỊ QUA INTERNET

GVHD: Th.S VÕ TẤN THÔNG

SVTH: TRẦN LÝ QUỲNH

MSSV: 41303318

TP. HỒ CHÍ MINH, THÁNG 12 NĂM 2017

----- ☆ -----

----- ☆ -----

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

NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

1. HỌ VÀ TÊN: TRẦN LÝ QUỲNH MSSV: 41303318
 2. NGÀNH: **ĐIỆN TỬ - VIỄN THÔNG** LỚP : DD13DV04
 3. Đề tài: Mô hình nhà thông minh điều khiển thiết bị qua Internet.
 4. Nhiệm vụ (Yêu cầu về nội dung và số liệu ban đầu):
Thiết kế hệ thống nhà thông minh điều khiển thiết bị qua Internet gồm các nội dung:
 - Điều khiển các thiết bị bằng relay từ một trang web và hiển thị trạng thái cầu thiết bị trên trang web.
 - Đọc giá trị của các cảm biến, hiển thị trên trang web và ghi lại thời gian phát hiện có chuyển động hoặc rò khí gas.
 - Trang web được truy cập thông qua module ESP8266 đã thiết lập để kết nối Internet.
 5. Ngày giao nhiệm vụ luận văn: 15/8/2017
 6. Ngày hoàn thành nhiệm vụ: 18/12/2017
 7. Họ và tên người hướng dẫn: **Th.S VÕ TẤN THÔNG**
Phản hướng dẫn
.....
.....
- Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

Tp.HCM, ngày tháng năm 2017
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

Bách Khoa lấy của ta những gì, cho ta lại bao nhiêu, liệu có thể cân đo đong đếm? Chỉ biết rằng, Bách Khoa là những ngày tháng của thanh xuân, là kỉ niệm, là bạn bè, là những điều đáng để nhớ.

Cảm ơn Bách Khoa! Cảm ơn gã bạn đời của tuổi trẻ. Hơn bốn năm, không đủ dài để nói một phần lớn của đời người, cũng chẳng ngắn để bảo chỉ là một cơn gió thoảng qua. Những gì có cùng Bách Khoa thật là đáng trân trọng, để lưu giữ, để đọng lại trong con tim nhỏ bé này.

Chặng đường với Bách Khoa đầy gập ghềnh dai dẳng biết mấy, đi đến cuối con đường là một thành công của chúng mình. Nhưng để đạt được điều đó không chỉ dựa vào nỗ lực và sự cố gắng của bản thân mà đó còn là những sự giúp đỡ, hỗ trợ, chia sẻ của những người luôn bên cạnh.

Đầu tiên, em xin chân thành cảm ơn thầy Võ Tấn Thông, người đã giúp đỡ, hướng dẫn em tận tình trong suốt quá trình thực hiện Đồ án môn học 1, Đồ án môn học 2, Thực tập tốt nghiệp và nhất là trong quá trình thực hiện Luận văn tốt nghiệp này.

Để đạt được đến ngày hôm nay, không thể nhắc đến sự dạy bảo, hướng dẫn tận tình của các thầy, cô của Trường Đại học Bách khoa TP. HCM, đặc biệt là các thầy cô trong khoa Điện – Điện tử đã giúp em có được các kiến thức để thực hiện Luận văn tốt nghiệp này.

Cuối cùng, em xin được cảm ơn tất cả bạn bè, những người đã bên cạnh động viên, giúp đỡ em trong suốt quá trình học tập tại trường và thời gian thực hiện Luận văn tốt nghiệp.

Tp. Hồ Chí Minh, ngày tháng năm .

Sinh viên

TÓM TẮT LUẬN VĂN

Đồ án này trình bày về hệ thống điều khiển và giám sát từ xa qua mạng Internet ứng dụng công nghệ Wifi kết hợp internet. Hệ thống có thể giúp cho người sử dụng có thể điều khiển các thiết bị cũng như giám sát trạng thái các thiết bị từ xa trên các thiết bị có thể kết nối Internet như máy vi tính, điện thoại di động, ...

Nguyên lý hoạt động cụ thể của hệ thống này như sau:

- ESP8266 V1 kết nối internet thông qua wifi đọc dữ liệu từ cơ sở dữ liệu trên webserver xử lý dữ liệu và gửi về dữ liệu về vi xử lý.
- Dữ liệu nhận được từ ESP dưới dạng text được vi điều khiển xử lý và thực hiện nhiệm vụ và gửi lại kết quả dưới dạng text cho ESP.
- ESP nhận dữ liệu từ vi xử lý rồi gửi lại kết quả lên cơ sở dữ liệu.

Trang web sẽ có nhiệm vụ hiển thị trạng thái của các thiết bị và thời gian thay đổi trạng thái của các thiết bị. Người dùng cũng sẽ điều khiển các thiết bị thông qua các lựa chọn trên trang web này.

MỤC LỤC

1. GIỚI THIỆU	1
1.1. Tổng quan	1
1.2. Tình hình nghiên cứu trong và ngoài nước	1
1.3. Nhiệm vụ luận văn	2
2. LÝ THUYẾT.....	3
2.1. Một số khái niệm cơ bản.....	3
2.2. Một số dịch vụ mạng phổ biến.....	7
2.2.1 Dịch vụ DNS (Domain Name Service)	7
2.2.2. Dịch vụ FTP (File Transfer Protocol)	7
2.2.3. Dịch vụ WEB	8
2.2.4. Một số dịch vụ khác	10
2.3. Ưu – Nhược điểm của WiFi.....	10
2.3.1. Ưu điểm:	10
2.3.2. Nhược điểm	10
2.4. Giới thiệu ESP8266	11
2.4.1. Giới thiệu ESP8266	11
2.4.2. Vấn đề lập trình trên ESP8266 V1	14
2.4.3. Giới thiệu về Arduino và Atmega328	16
2.4.4. Một số linh kiện khác	19
3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG	23
3.1. Yêu cầu thiết kế phần cứng.....	23
3.2. Sơ đồ khối tổng quát.	23
3.2.1 Thiết kế khối Relay	23
3.2.2 Thiết kế khối nguồn.....	25
3.2.2 Khối xử lý trung tâm	26

3.3	Sơ đồ toàn mạch.....	26
4.	THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM	27
4.1.	Yêu cầu thiết kế	27
4.2.	Phần mềm cho vi điều khiển và ESP8266	27
4.2.1.	Phần mềm dành cho ATMEGA328.....	27
4.2.2.	Phần mềm cho ESP8266	30
4.3.	Trang Web và Server	31
4.3.1.	Phần mềm Microsoft WebMatrix và ASP.NET framework.....	31
4.3.2.	Tạo cơ sở dữ liệu	33
4.3.3.	Trang hiển thị chính của trang Web	35
4.3.4.	Trang hiển thị trạng thái của cảm biến và thiết bị	37
4.3.5.	Trang Web Server.....	39
4.3.6.	Trang hiển thị yêu cầu cho thiết bị	43
5.	KẾT QUẢ THỰC HIỆN.....	44
5.1.	Đăng tải trang Web lên mạng Internet.....	44
5.2.	Phần mềm của vi điều khiển	44
5.3.	Hoạt động của toàn hệ thống	45
6.	KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	46
6.1.	Kết luận.....	46
6.2.	Hướng phát triển	46
7.	TÀI LIỆU THAM KHẢO.....	47
8.	PHỤ LỤC.....	47
8.1	Mã lệnh của trang “Default.cshtml”	47
8.2	Mã lệnh của trang “Sensor.cshtml”	54
8.3	Mã lệnh của trang “SendRequest.cshtml”	55
8.4	Mã lệnh của trang “GetStatus.cshtml”.....	56
8.5	Mã lệnh của trang “DisplayStatus.cshtml”	60

8.6 Mã lệnh của trang “DisplaySensor.cshtml”	61
--	----

DANH SÁCH HÌNH MINH HỌA

Hình 1.2- 1 Mô hình Internet of things.....	2
Hình 2.1- 1 Mô hình OSI 7 lớp.....	4
Hình 2.1- 2 So sánh OSI và TCP/IP.	5
Hình 2.2.3- 1 Mô hình Web Server và các máy client.....	8
Hình 2.4.1- 1 Tập lệnh AT chung.....	12
Hình 2.4.1- 2 Các lệnh AT cấu hình Modele WiFi.	12
Hình 2.4.1- 3 Các lệnh AT với Module WiFi cấu hình là Access Point.	12
Hình 2.4.1- 4 Các lệnh AT đối với Module WiFi cấu hình là Station/Client.....	13
Hình 2.4.1- 5 Sơ đồ chân của ESP8266 V1.....	13
Hình 2.4.3- 1 Thông số của Arduino Uno R3.....	17
Hình 2.4.3- 2 Chân của vi điều khiển Atmega328.	18
Hình 2.4.4.1- 1 Relay	19
Hình 2.4.4.2- 1 Cảm biến nhiệt độ và độ ẩm	20
Hình 2.4.4.3- 1 Cảm biến ánh sáng.....	21
Hình 2.4.4.4- 1 Cảm biến khí gas.....	21
Hình 2.4.4.5- 1 Cảm biến chuyển động	22
Hình 3.2.1- 1 Sơ đồ mạch Relay.....	24
Hình 3.2.2- 1 Sơ đồ mạch nguồn 5V.....	25
Hình 3.2.2- 2 Sơ đồ mạch nguồn 3.3V.	25
Hình 3.2.3- 1 Sơ đồ mạch xử lý trung tâm.....	26
Hình 3.3- 1 Sơ đồ toàn mạch.....	26
Hình 4.2.1- 1 Lưu đồ giải thuật.....	28
Hình 4.2.1- 2 Lưu đồ giải thuật xuất chuỗi.	29
Hình 4.2.1- 3 Sơ đồ khối so sánh.	30
Hình 4.3.1- 1 Phần mềm Webmatrix.....	32

Hình 4.3.2- 1 Bảng Devices.....	33
Hình 4.3.2- 2 Bảng Sensor.....	34
Hình 4.3.3- 1 Trang web server.....	35
Hình 5.1- 1 Trang web hoạt động.....	44
Hình 5.2- 1 Phần mềm IDE 1.8.5.....	45
Hình 5.3- 1 Mạch thực tế.....	45

DANH SÁCH BẢNG SỐ LIỆU

1. GIỚI THIỆU

1.1. Tổng quan

Nhà thông minh là ngôi nhà được trang bị các hệ thống tự động thông minh cùng với cách bố trí hợp lý, các hệ thống này có khả năng tự động điều phối các hoạt động trong ngôi nhà theo thói quen sinh hoạt và nhu cầu cá nhân của gia đình. Chúng ta cũng có thể hiểu ngôi nhà thông minh là một hệ thống mà trong đó, tất cả các thiết bị điện tử gia dụng đều được liên kết với thiết bị điều khiển trung tâm và có thể phối hợp với nhau để cùng thực hiện một chức năng. Các thiết bị này có thể tự đưa ra cách xử lý tình huống được lập trình trước, hoặc là được điều khiển và giám sát từ xa.

Giải pháp nhà thông minh sẽ biến những món đồ điện tử bình thường trong ngôi nhà trở nên thông minh và gần gũi với người dùng hơn, chúng ta được kiểm soát thông qua các thiết bị truyền thông như điều khiển từ xa, điện thoại di động,... ngôi nhà thông minh đơn giản nhất có thể được hình dung bao gồm một mạng điều khiển liên kết một số lượng cố định các thiết bị điện tử gia dụng trong ngôi nhà và chúng được điều khiển thông qua một chiếc điều khiển từ xa. Chỉ với kết nối đơn giản như trên cũng đủ để hài lòng một số lượng lớn các cá nhân có nhu cầu nhà thông minh ở mức bình thường.

Vậy liệu nhà thông minh có làm thay đổi các thói quen vốn đã rất gắn bó từ trước đến nay với hầu hết mọi người?

Chúng ta đều biết phần lớn căn hộ từ mức trung bình đến mức cao cấp đều sử dụng các loại điều khiển từ xa để điều khiển máy lạnh, ti vi...còn lại phần lớn các thiết bị khác như đèn, quạt, bình nước nóng lạnh...phải điều khiển bằng tay. Những việc như vậy đôi lúc sẽ mang lại sự bất tiện, khi mà chúng ta mong muốn có một sự tiện nghi và thoải mái hơn, vừa có thể tận hưởng nằm trên giường xem ti vi vừa có thể kiểm soát được hệ thống các thiết bị trong nhà chỉ với một chiếc smartphone hay máy tính bảng.

1.2. Tình hình nghiên cứu trong và ngoài nước

Với sự phát triển của Internet, smartphone và đặc biệt là các thiết bị cảm biến, Internet of Things (IoT) đang trở thành xu hướng mới của thế giới. IoT được định nghĩa là những vật dụng có khả năng kết nối Internet. Bật vào nhà, mở khóa cửa, đèn sẽ tự động sáng chỗ bạn đứng, điều hòa sẽ tự động điều chỉnh nhiệt độ, nhạc sẽ tự động bật để chào đón bạn... những điều chỉ có trong phim khoa học viễn tưởng, đang dần trở thành hiện thực với công nghệ IoT.



Hình 1.2- 1 Mô hình Internet of things.

Các thiết bị IoT được vận hành nhờ những bộ vi xử lý SOC bên trong. Không như những bộ vi xử lý thông thường, SOC giống như một máy tính trọn vẹn được thu gọn trong diện tích của một con chip điện tử, có kết nối không dây và đảm bảo tiết kiệm điện. Dù nhỏ gọn, sức mạnh của các vi xử lý SOC là không phải bàn cãi khi nó hoàn toàn có thể vận hành trơn tru những hệ điều hành nặng nề như Windows hay Linux. SOC rất phổ biến trong bên trong các linh kiện điện thoại.

Theo dự báo của IDC, thị trường IoT được dự báo sẽ tăng gấp 3 lần, đạt 1,7 nghìn tỉ USD vào năm 2020. Không ít các doanh nghiệp lớn đã nhìn thấy tiềm năng của IoT và mạnh dạn đầu tư vào đây. Tuy nhiên, cũng giống như bất kỳ một công nghệ mới nào, IoT sẽ cần một nền tảng để vận hành. Và các doanh nghiệp công nghệ hiểu rằng, ai tạo ra được nền tảng dẫn đầu, họ sẽ là người chiến thắng trong xu hướng mới này.

1.3. Nhiệm vụ luận văn

Nội dung 1: Tìm hiểu về mạng Wifi và internet

Nội dung 2: Tìm hiểu về Esp8266 v1 nguyên lý hoạt động, cách kết nối wifi và kết nối với mạng internet.

Nội dung 3: Tìm hiểu về Kit vi điều khiển Arduino.

Nội dung 5: Thiết kế và thực hiện mạch Relay dùng để điều khiển các thiết bị điện.

Nội dung 6: Thiết kế trang web và phần server để liên kết với cơ sở dữ liệu.

Nội dung 7: Viết phần mềm cho vi điều khiển và hoàn thành kết nối hệ thống

2. LÝ THUYẾT

2.1. Một số khái niệm cơ bản

Mạng máy tính là tập hợp các máy tính được với nhau bởi đường truyền theo cấu trúc nào đó và thông qua đó các máy tính trao đổi thông tin qua lại với nhau.

Đường truyền là hệ thống các thiết bị truyền dẫn có dây hay không dây dùng để chuyển các tín hiệu điện tử từ máy tính này đến máy tính khác. Các tín hiệu điện tử đó biểu thị các giá trị dữ liệu dưới dạng các xung nhị phân. Tất cả các tín hiệu được truyền giữa các máy tính đều thuộc một số dạng sóng điện từ. Tùy theo tần số của sóng điện từ có thể dùng các đường truyền vật lý khác nhau để truyền các tín hiệu. ở đây đường truyền được kết nối có thể là dây cáp đồng trục, cáp xoắn, cáp quang, dây điện thoại, sóng vô tuyến... các đường truyền dữ liệu tạo nên cấu trúc của mạng. hai khái niệm đường truyền và cấu trúc là những đặc trưng cơ bản của mạng máy tính.

Cần phải phân biệt sự trao đổi qua lại giữa máy tính này với máy tính khác trong mạng máy tính với các hệ thống thu phát một chiều như đài truyền hình, phát thông tin từ vệ tinh xuống các trạm thu thụ động.. vì tại đây chỉ có thông tin một chiều từ nơi phát đến nơi thu mà không quan tâm đến có bao nhiêu nơi thu, có thu tốt hay không.

Đặc trưng cơ bản của đường truyền vật lý là giải thông. Giải thông của một đường truyền chính là độ đo phạm vi tần số mà nó có thể đáp ứng được. Tốc độ truyền dữ liệu trên đường truyền còn được gọi là thông lượng của đường truyền- thường được tính bằng số lượng bit được truyền đi trong một giây (Bps). Thông lượng còn được đo bằng đơn vị khác là Baud. Baud biểu thị số lượng thay đổi tín hiệu trong một giây.

ở đây Baud và Bps không phải bao giờ cũng đồng nhất. ví dụ: nếu trên đường dây có 8 mức tín hiệu khác nhau thì mỗi mức tín hiệu tương ứng với 3 bit hay là 1 Baud tương ứng với 3 bit. Chỉ khi có 2 mức tín hiệu trong đó mỗi mức tín hiệu tương ứng với 1 bit thì 1 Baud mới tương ứng với 1 bit.

Hai máy tính không thể hiểu nhau nếu không có cách thức truyền/ nhận dữ liệu giống nhau, ngoài ra còn phải đảm bảo dữ liệu an toàn, chính xác, nhanh chóng. Từ thực tế đó mô hình phân tầng ra đời, dựa trên ý tưởng phân chia công việc thành từng tầng, mỗi tầng thực hiện chức năng riêng như xử lý, đóng gói, truyền, nhận dữ liệu v.v.. tuy mô hình phân tầng

giải quyết vấn đề thông tin giữa hai máy tính nhưng lại nảy sinh vấn đề khác, đó là có quá nhiều chuẩn giao thức ra đời dẫn đến tình trạng mạng máy tính này không thể giao tiếp với mạng máy tính khác và nhiều tác hại kèm theo. Và mô hình OSI ra đời.

Mô hình OSI là cơ sở dành cho việc chuẩn hóa các hệ thống truyền thông, nó được nghiên cứu và xây dựng với ISO. Việc nghiên cứu về mô hình OSI được bắt đầu tại ISO vào năm 1971 với mục tiêu nhằm tới việc kết nối các sản phẩm của các hãng sản xuất khác nhau và phối hợp hoạt động chuẩn hóa trong các lĩnh vực viễn thông và hệ thống thông tin. Theo mô hình OSI chương trình truyền thông được chia ra thành 7 tầng với những chức năng phân biệt cho từng tầng:

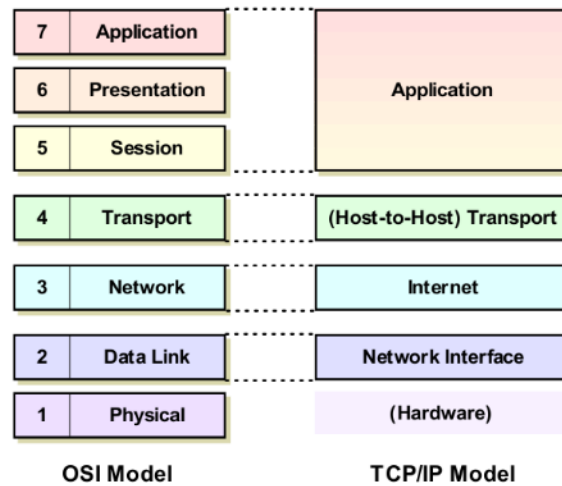
TT	Tên gọi	Ý nghĩa
7	Application (ứng dụng)	Truy cập tới các dịch vụ truyền thông bằng các chương trình ứng dụng
6	Presentation (Trình bày)	Định nghĩa các quy tắc cho truyền thông
5	Session (phiên làm việc)	Xác lập và ngắt các kết nối bằng cách đồng bộ và tổ chức qua đối thoại.
4	Transport (Vận chuyển)	Quy định các thuộc tính kết nối đầu cuối với chất lượng đường truyền cần thiết
3	Network (mạng)	Trao đổi dữ liệu qua lại giữa 2 thực thể
2	Data Link (liên kết dữ liệu)	Truy cập vào môi trường vật lý, phát hiện và xử lý các lỗi đường truyền.
1	Physical (vật lý)	Truyền tải các dòng bit dữ liệu thông qua môi trường vật lý.

Hình 2.1- 1 Mô hình OSI 7 lớp.

Mô hình TCP/IP được xây dựng và phát triển dựa trên mô hình OSI. Mô hình này về cơ bản là dạng rút gọn của mô hình OSI, nó làm đơn giản quá trình xử lý dữ liệu.

Hiện nay, mô hình TCP/IP được sử dụng rộng rãi. Nó là giao thức truyền/ nhận trên mạng Internet.

So sánh sự tương đồng của 2 mô hình này



Hình 2.1- 2 So sánh OSI và TCP/IP.

Mô hình TCP/IP gom tầng Application, Presentation, Session của mô hình OSI thành một tầng là Application, gom hai tầng Data Link và Physical thành tầng Network Access. Trong đó:

Tầng Application: đây là tầng cao nhất trong cấu trúc phân lớp của TCP/IP. Tầng này bao gồm tất cả các chương trình ứng dụng sử dụng các dịch vụ sẵn có thông qua một chồng giao thức TCP/IP. Các chương trình ứng dụng tương tác với một trong các giao thức của tầng giao vận để truyền hoặc nhận dữ liệu. Mỗi chương trình ứng dụng lựa chọn một kiểu giao thức thích hợp cho công việc của nó. Chương trình ứng dụng chuyển dữ liệu theo mẫu mà tầng giao vận yêu cầu.

Một số giao thức thông dụng như:

- DHCP (Dynamic Host Configuration Protocol)
- DNS (Domain Name System)
- SNMP (Simple Network Management Protocol)
- FTP (File Transfer Protocol)
- TFTP (Trivial File Transfer Protocol)
- SMTP (Simple Mail Transfer Protocol)
- TELNET

Tầng Transport: nhiệm vụ trước tiên của tầng giao vận là cung cấp sự giao tiếp thông tin giữa các chương trình ứng dụng. Mỗi sự giao tiếp được gọi là end-to-end. Tầng giao vận cũng có thể điều chỉnh lưu lượng luồng thông tin. Nó cũng cung cấp một sự vận chuyển tin

cây, đảm bảo rằng dữ liệu đến mà không bị lỗi. Để làm như vậy, phần mềm giao thức hỗ trợ để bên nhận có thể gửi lại các thông báo xác nhận về việc thu dữ liệu và bên gửi có thể truyền lại các gói tin bị mất hoặc bị lỗi. Phần mềm giao thức chia dòng dữ liệu ra thành những đơn vị dữ liệu nhỏ hơn (thường được gọi là các Packet) và chuyển mỗi Packet cùng với địa chỉ đích tới tầng tiếp theo để tiếp tục quá trình truyền dẫn.

Có hai giao thức chính trong tầng này gồm:

UDP (User Datagram Protocol): còn gọi là Giao Thức Gói Người Dừng. UDP cung cấp các kênh truyền thông phi kết nối nên nó không đảm bảo truyền dữ liệu 1 cách tin cậy. Các ứng dụng dùng UDP thường chỉ truyền những gói có kích thước nhỏ, độ tin cậy dữ liệu phụ thuộc vào từng ứng dụng.

TCP (Transmission Control Protocol): ngược lại với UDP, TCP cung cấp các kênh truyền thông hướng kết nối và đảm bảo truyền dữ liệu 1 cách tin cậy. TCP thường truyền các gói tin có kích thước lớn và yêu cầu phía nhận xác nhận về các gói tin đã nhận.

Tầng Internet: tầng mạng xử lý giao tiếp thông tin từ một máy này tới một máy khác. Nó chấp nhận một yêu cầu để gửi một gói từ tầng giao vận cùng với một định danh của máy đích mà gói tin sẽ được gửi tới. Ví dụ với giao thức TCP hay UDP của tầng giao vận, nó sẽ bọc gói tin trong một IP Datagram, điền đầy vào trong phần Header, sử dụng giải thuật chọn đường để quyết định là giao phát gói tin trực tiếp hay là gửi nó tới một Router, và chuyển Datagram tới giao diện phối ghép mạng thích hợp cho việc truyền dẫn. Tầng mạng cũng xử lý các Datagram đến, kiểm tra tính hợp lệ của chúng, và sử dụng giải thuật chọn đường để quyết định là Datagram sẽ được xử lý cục bộ hay là sẽ được chuyển đi tiếp. Đối với các Datagram có địa chỉ đích cục bộ, thì phần mềm tầng mạng sẽ xóa phần Header của các Datagram đó, và chọn trong số các giao thức tầng giao vận một giao thức thích hợp để xử lý Packet.

Bốn giao thức quan trọng nhất trong tầng này gồm:

- IP (Internet Protocol): có chức năng gán địa chỉ cho dữ liệu trước khi truyền và định tuyến chúng tới đích.
- ARP (Address Resolution Protocol): có chức năng biên dịch địa chỉ IP của máy đích thành địa chỉ MAC.
- ICMP (Internet Control Message Protocol): có chức năng thông báo lỗi trong trường hợp truyền dữ liệu bị hỏng.
- IGMP (Internet Group Management Protocol): có chức năng điều khiển truyền đa hướng (Multicast).

Tầng Network Access: là tầng thấp nhất của bộ giao thức TCP/IP, chịu trách nhiệm về việc chấp nhận các Datagram của tầng trên (ví dụ IP datagram) và việc truyền phát chúng trên một mạng xác định. Theo quan điểm hiện nay mô hình TCP/IP không còn bao gồm các đặc tả vật lý, nói cách khác tầng Network Access cũng không còn bao gồm vấn đề về phần cứng hay việc truyền tín hiệu vật lý nữa.

2.2. Một số dịch vụ mạng phổ biến

2.2.1 Dịch vụ DNS (Domain Name Service)

Mỗi máy tính trong mạng muốn liên lạc, trao đổi dữ liệu hay thông tin cần phải biết rõ địa chỉ IP của nhau. Nếu số lượng máy tính nhiều thì việc nhớ những địa chỉ IP này rất là khó khăn.

Mỗi máy tính ngoài địa chỉ IP còn có tên máy còn gọi là Computer Name. Đối với con người việc nhớ tên máy dù sao cũng dễ hơn vì chúng có tính trực quan và gợi nhớ hơn so với địa chỉ IP. Ban đầu người ta chỉ lưu tên máy dưới dạng tập in TXT nhằm ánh xạ tên máy và địa chỉ IP nhưng khi nhu cầu về máy nhiều nên tập tin như vậy không đáp ứng được. Do đó dịch vụ DNS ra đời nhằm khắc phục nhược điểm này.

Dịch vụ DNS thực sự là cơ sở dữ liệu phân tán, có nhiệm vụ phân giải địa chỉ IP thành tên máy và ngược lại. DNS hoạt động theo mô hình Server/Client. Phần Server gọi là máy phục vụ hay còn gọi là NameServer. Còn phần Client là chương trình yêu cầu phân giải gọi là Resolver.

Ví dụ: 172.0.0.1 là địa chỉ IP tương ứng của www.vieclam.com.vn

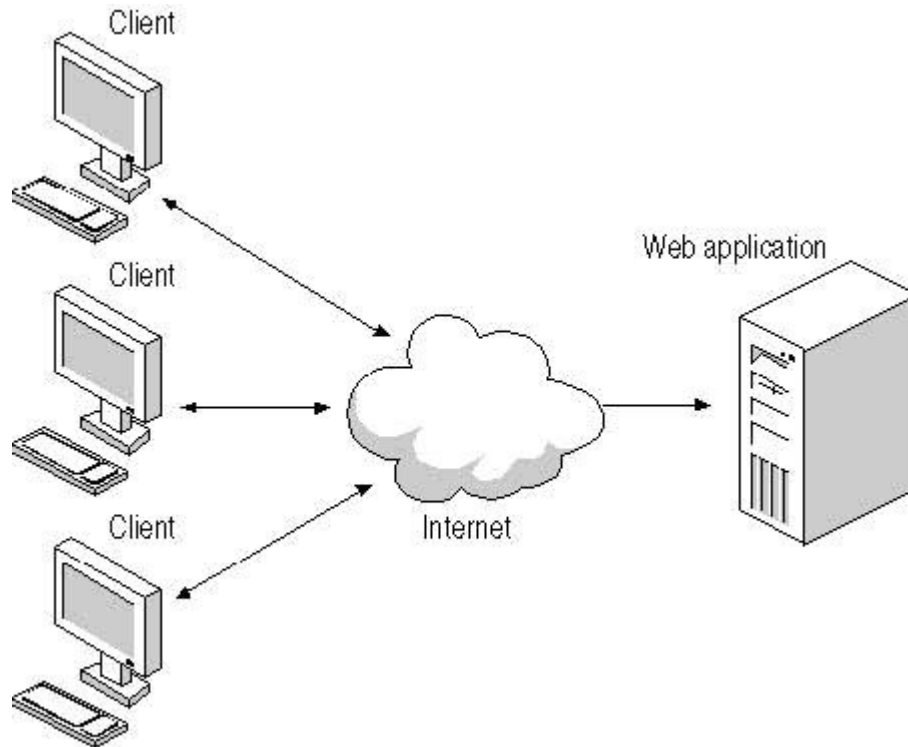
2.2.2. Dịch vụ FTP (File Transfer Protocol)

Là giao thức hoạt động trên chuẩn TCP/IP, FTP cung cấp cơ chế truyền file qua lại giữa các máy tính. FTP là dịch vụ đặc biệt vì nó dùng đến hai cổng. Cổng 20 dùng để truyền dữ liệu và cổng 21 dùng để truyền lệnh.

FTP hoạt động ở 2 chế độ: Active FTP và Passive FTP. Điểm khác biệt cơ bản giữa hai chế độ này là cơ chế Active thì Server phải tạo kết nối tới Client, còn Passive giải quyết được vấn đề này nhưng gây ra vấn đề ở phía Server do cho phép kết nối ở xa với bất kỳ cổng nào lớn hơn 1024, điều này khá nguy hiểm vì dễ bị tấn công.

2.2.3. Dịch vụ WEB

Web Server hoạt động trên cơ sở giao thức HTTP. HTTP là giao thức cho phép trình duyệt Web và Server có thể giao tiếp với nhau. Nó giao thức đơn giản, thông tin điều khiển được truyền đi dưới dạng văn bản thô thông qua kết nối TCP, trên port 80.



Hình 2.2.3- 1 Mô hình Web Server và các máy client.

Giao thức này thực thi đơn giản 2 thao tác Request và Response. Các yêu cầu và đáp ứng thực chất là tải file (thường là HTML), còn được gọi là các trang Web. Các file này có vai trò quan trọng, nó chứa đựng thông tin về trang Web mà chúng ta truy cập. Ngoài ra, nó còn cung cấp các dịch vụ học tập, giải trí như học tiếng anh, nghe nhạc, chơi game, đọc truyện... Cấu trúc file này gồm các thẻ (còn gọi là Tag), khi có yêu cầu thì trình duyệt sẽ tải và hiển thị cho người dùng.

Web Server ban đầu chỉ đơn thuần là truyền các file HTML thông thường chứa thông tin và hình ảnh v.v... Tất cả các trang HTML này gọi chung là Web tĩnh, và không có khả năng thêm bớt nội dung một cách tự động. Do nhu cầu này cần hệ thống xử lý thông tin linh hoạt hơn, mô hình Web động ra đời cho phép chúng ta nhập dữ liệu lưu trữ lại trên máy chủ hay tìm thông tin hàng hóa trực tiếp hay gián tiếp. Cách cổ điển là dùng các trang CGI

(Common Gateway Interface) – nó định nghĩa cách thức Web Server chạy chương trình cục bộ ở phía Server, sau đó nhận kết quả trả về cho trình duyệt Web.

Khi nhận được lệnh GET, POST của giao thức HTTP từ Client, Server có thể cho thực thi chương trình và tự động tạo ra nội dung dữ liệu, sau đó trả về file (thường là thẻ HTML) cho Client. Như vậy lúc này nội dung trả về cho Client sẽ không phải là file tĩnh tồn tại trong ổ cứng của Server. Nó được tạo ra từ chương trình mà chương trình này thường được gọi là CGI. Chương trình CGI của Server có thể sản sinh ra hàng ngàn trang Web theo yêu cầu của Client. Thật ra chương trình CGI đơn thuần chỉ là chương trình *.exe (trên Windows) hoặc chương trình thực thi (trên Unix) bình thường. Chúng ta có thể xây dựng CGI bằng các ngôn ngữ như C, Pascal, Visual Basic...

Cùng với việc sản sinh các trang Web chương trình CGI còn tính toán xử lý, truy xuất số liệu từ cơ sở dữ liệu, cập nhật vào trang Web gửi về Web Server. Cuối cùng Web Server gửi trả về trình duyệt.

Cơ chế hoạt động của CGI rất hiệu quả đối với các trình duyệt hiện nay. Ngày nay CGI vẫn còn được dùng rộng rãi trên các Web Server chạy trên các hệ Unix. Nhược điểm duy nhất là tiêu tốn tài nguyên và phần giới hạn về tốc độ thực thi. Mỗi khi có request thì CGI tại Server thực hiện:

- Nạp chương trình vào bộ nhớ.
- Thực thi chương trình.
- Trả kết quả về Client.

Đối với ngôn ngữ kịch bản thì MicroSoft đề xuất phương án thiết kế các trang ASP. ASP là các trang chứa mã lệnh viết bằng ngôn ngữ Visual Basic kết hợp các thẻ HTML. Nó sẽ được thông dịch từ các Client. Sau đó MicroSoft lại cải tiến bằng cách đưa ra ngôn ngữ ASP.NET. Về cơ bản nó cũng giống như ASP nhưng có nhiều cải tiến đáng kể, ví dụ các trang ASP là trang phải thông dịch thì các trang ASP.NET đã thực sự được thông dịch, điều này cải thiện tốc độ truy cập Web...

Tóm lại, dịch vụ Web ngày càng đáp ứng nhu cầu thực tiễn trong cuộc sống. Hiện nay hầu như nó có mặt khắp mọi nơi và đóng vai trò cực kỳ quan trọng trong việc truyền thông trên thế giới.

Luận văn này sử dụng board Arduino như 1 Server đơn giản, khi Client là các trình duyệt Web kết nối tới và gửi Request thì Server sẽ dùng các lệnh như ethernet.print(), lệnh

này dùng để gửi cho các trình duyệt Web thẻ HTML. Các chương trình như Internet Explorer, Firefox hay Chrome sẽ biên dịch các thẻ và hiển thị trực quan, dễ dàng thao tác bất, tắt các đèn, đồng thời sẽ báo cho người điều khiển biết trạng thái của các đèn, nhiệt độ, độ ẩm...

2.2.4. Một số dịch vụ khác

DHCP: dịch vụ cung cấp địa chỉ IP tự động, có thể thay thế địa chỉ IP trong khoảng thời gian cố định, được ứng dụng nhiều trong công nghiệp.

Mail: hiện nay rất phổ biến là dịch vụ thư điện tử nhanh chóng trên mạng trao đổi thông tin, chia sẻ dữ liệu v.v... Ví dụ như gmail, yahoo mail v.v...

Các dịch vụ chia sẻ tài nguyên như máy in, máy fax. Giúp tiết kiệm chi phí.

2.3. Ưu – Nhược điểm của WiFi

2.3.1. Ưu điểm:

- Mạng không dây cũng như hệ thống mạng thông thường. Nó cho phép người dùng truy xuất tài nguyên mạng ở bất kỳ nơi đâu trong khu vực được triển khai (nhà hay văn phòng). Với sự gia tăng số người sử dụng máy tính xách tay, máy tính bảng, smartphone, đó là một điều rất thuận lợi.
- Với sự phát triển của các mạng không dây công cộng, người dùng có thể truy cập Internet ở bất cứ đâu.
- Mạng không dây có thể đáp ứng tức thì khi gia tăng số lượng người dùng. Với hệ thống mạng dùng cáp cần phải gắn thêm cáp.
- Mạng Internet không dây (Wi-Fi) đã đem lại nhiều lợi ích cho cộng đồng, nó giúp cho việc truyền tải, tiếp nhận thông tin cực kỳ nhanh chóng và tiện lợi, giúp người sử dụng công nghệ tiết kiệm thời gian nâng cao hiệu quả công việc, chúng ta cũng có thể truy cập Internet để theo dõi tin tức bằng sử dụng điện thoại có kết nối Wi-Fi.

2.3.2. Nhược điểm

- Chủ yếu là trong mô hình mạng nhỏ và trung bình, với những mô hình lớn phải kết hợp với mạng có dây. Một mạng chuẩn 802.11g với các thiết bị chuẩn chỉ có thể hoạt động tốt trong phạm vi vài chục mét. Nó phù hợp trong 1 căn nhà, nhưng với một tòa nhà lớn thì không đáp ứng được nhu cầu.

- Bị ảnh hưởng bởi các yếu tố bên ngoài như môi trường truyền sóng, có thể nhiễu do thời tiết.
- Dễ bị các cuộc tấn công đa dạng, phức tạp, nguy hiểm của những kẻ phá hoại vô tình và cố tình, nguy cơ cao hơn mạng có dây.
- Tốc độ của mạng không dây (1- 125 Mbps) chậm hơn so với mạng sử dụng cáp (100Mbps đến hàng Gbps).

2.4. Giới thiệu ESP8266

2.4.1. Giới thiệu ESP8266

ESP8266 là một chip tích hợp được thiết kế dùng cho chuẩn kết nối mới. Có thể dùng nó để đưa những dự án của bạn kết nối đến Internet. Đơn giản nó sử dụng ngõ giao thức nối tiếp với tốc độ Baud 115200 (mặc định). Kết nối mạng không dây, giống như một máy chủ hoặc một cầu nối trung gian và có thể download dữ liệu từ Internet.

Module ESP8266 V1 có bộ nhớ là 1M, được cài firmware 1.01 và có tốc độ mặc định là 115200 và bạn có thể thay đổi bằng lệnh AT.

Thông số kỹ thuật

- Hỗ trợ chuẩn 802.11 b/g/n.
- Wi-Fi 2.4 GHz, hỗ trợ WPA/WPA2.
- Chuẩn điện áp hoạt động: 3.3V.
- Chuẩn giao tiếp nối tiếp UART với tốc độ Baud lên đến 115200.
- Có 3 chế độ hoạt động: Client, Access Point, Both Client and Access Point.
- Hỗ trợ các chuẩn bảo mật như: OPEN, WEP, WPA_PSK, WPA2_PSK, WPA_WPA2_PSK.
- Hỗ trợ cả 2 giao tiếp TCP và UDP.
- Làm việc như các máy chủ có thể kết nối với 5 máy con.

Các lệnh tập lệnh AT

AT command	Description	Parameter	Example
AT	Kiểm tra lệnh, luôn trả về "OK"		<i>AT</i>
AT+RST	Khởi động lại module		<i>AT+RST</i>
AT+GMR	Truy vấn phiên bản Firmware		<i>AT+GMR</i>

Hình 2.4.1- 1 Tập lệnh AT chung.

AT command	Description	Parameter	Example
AT+CWMODE = <mode>	Cài đặt chế độ	1 = Station 2 = Access Point 3 = Both	<i>AT+CWMODE=1</i>
AT+CWMODE?	Truy vấn chế độ đã cài đặt		<i>AT+CWMODE?</i>
AT+CWMODE=?	Truy vấn các chế độ có thể cài đặt		<i>AT+CWMODE=?</i>
AT+CIPMUX = <mode>	Cài đặt số lượng các kênh kết nối	0 = 1 kênh kết nối 1 = Nhiều kênh kết nối	<i>AT+CIPMUX=1</i>
AT+CIPMODE = <mode>	Cài đặt chế độ dữ liệu	0 = transparent 1 = Data	<i>AT+CIPMODE=1</i>
AT+CIPMODE?	Truy vấn chế độ dữ liệu cài đặt		<i>AT+CIPMODE?</i>

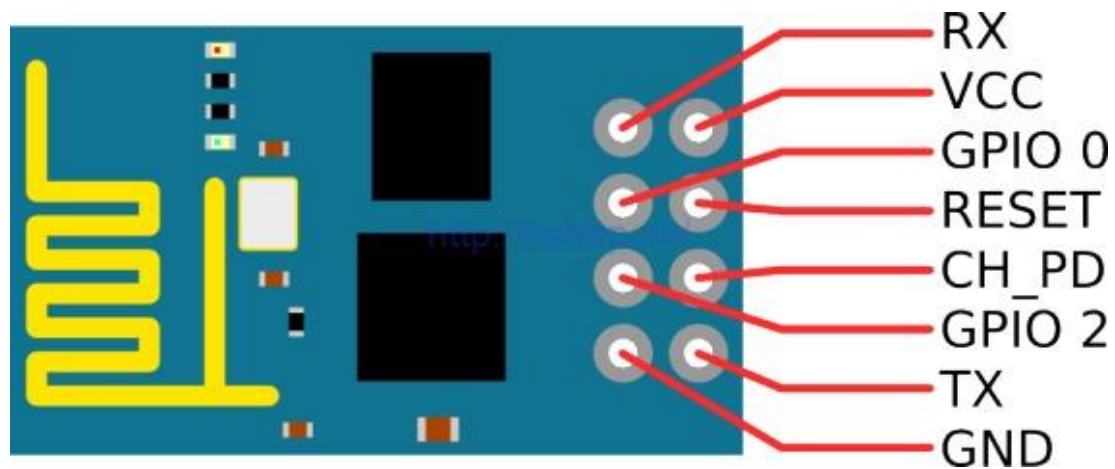
Hình 2.4.1- 2 Các lệnh AT cấu hình Modele WiFi.

AT command	Description	Parameter	Example
AT+CWJAP = <ssid>, <password>	Kết nối với 1 mạng wifi	ssid "SSID" pass "password"	<i>AT+CWJAP = "MLAB", "1235678"</i>
AT+CWJAP?	Truy vấn mạng wifi đang kết nối		<i>AT+CWJAP?</i>
AT+CWLAP	Truy vấn các mạng wifi có thể kết nối		<i>AT+CWLAP</i>
AT+CWQAP	Đồng kế nối wifi với một Access Point		<i>AT+ CWQAP</i>
AT+CIFSR	Xem địa chỉ IP của module		<i>AT+CIFSR</i>

Hình 2.4.1- 3 Các lệnh AT với Module WiFi cấu hình là Access Point.

AT command	Description	Parameter	Example
AT+CWSAP=<ssid>,<password> , <chan>, <enc>	Cài đặt các thông số cho Access Point	ssid "SSID" pass "password" chan "channel" enc "Encryption" (0 = Open 1 = WEP 2 = WPA_PSK 3 = WPA2_PSK 4 = WPA_WPA2_PSK)	AT+CWSAP= "MLAB","12345678",5,3
AT+CWSAP?	Xem cài đặt hiện tại của Access Point		AT + CWSAP?
AT+CWLIF	Danh sách các station đang kết nối		AT + CWLIF

Hình 2.4.1- 4 Các lệnh AT đối với Module WiFi cấu hình là Station/Client.



Hình 2.4.1- 5 Sơ đồ chân của ESP8266 V1.

- VCC: 3.3V lên đến 300mA
- GND: Mass
- Tx: Chân Tx của giao thức UART, kết nối đến chân Rx của vi điều khiển.
- Rx: Chân Rx của giao thức UART, kết nối đến chân Tx của vi điều khiển.
- RST: chân reset, kéo xuống mass để reset.
- CH_PD: Kích hoạt chip, sử dụng cho Flash Boot và updating lại module
- GPIO0: kéo xuống thấp cho chế độ update.
- GPIO2: không sử dụng.

2.4.2. Vấn đề lập trình trên ESP8266 V1

2.4.2.1. Lập trình ESP8266 V1 trên IDE Arduino

ESP8266 được tích hợp vi điều khiển Tenslica L106 32 bit (MCU) có tính năng tiêu thụ điện năng thấp và RSIX 16 bit, đạt tốc độ 160MHz. Với hệ thống hoạt động thời gian thực và chức năng ngăn xếp Wi-Fi, khoảng 80% sức mạnh của vi xử lý vẫn còn sẵn cho người dùng lập trình và phát triển.

Để lập trình được cho ESP ta cần thư viện cho ESP. Trên IDE ta có thể tải về qua Boards Manager.

2.4.2.2. Thư viện ESP8266WiFi

Gọi `WiFi.softAP(ssid)` để thiết lập một open network.

Gọi `WiFi.softAP(ssid, password)` để thiết lập WPA2-PSK (mật khẩu ít nhất 8 ký tự).

`WiFi.macAddress(mac)` cho STA, `WiFi.softAPmacAddress(mac)` cho AP.

`WiFi.localIP()` cho STA, `WiFi.softAPIP()` cho AP.

Để chuyển đổi sang chế độ station, ta dùng hàm `begin`. Các tham số cần thiết sẽ là SSID và password, để module có thể kết nối đến một Access Point (AP) cụ thể.

`WiFi.begin(ssid, password)`

Gọi hàm này module sẽ chuyển sang chế độ station và kết nối với điểm truy cập cuối cùng được sử dụng dựa trên cấu hình được lưu trong bộ nhớ flash. Để thiết lập tất cả các thông số, ta có thể dùng lệnh:

`WiFi.begin(ssid, password, channel, bssid, connect)`

Các thông số:

`ssid` - tên WiFi của điểm truy cập mà chúng ta muốn kết nối đến, có thể có tối đa lên đến 32 ký tự.

`password` - mật khẩu của điểm truy cập, có độ dài từ 8 đến 64 ký tự.

`channel` - thiết lập kênh cho WiFi, tham số này có thể bỏ qua.

`bssid` - địa chỉ MAC của AP.

`connect` - nếu giá trị là false, module sẽ lưu các tham số nhưng không thiết lập kết nối đến điểm truy cập.

2.4.2.3. WiFi Access point

Access Point (AP - Điểm truy cập) cung cấp khả năng truy cập mạng WiFi cho các thiết bị khác (Station) và kết nối chúng với mạng có dây. ESP8266 có thể cung cấp chức năng tương tự nhưng nó không kết nối có dây với một mạng. Chế độ hoạt động như vậy gọi là **soft-AP**. Số lượng trạm tối đa kết nối với soft-AP là 5.

Phần mô tả API này gồm có 3 phần: cách thiết lập soft-AP, quản lý kết nối và lấy thông tin về cấu hình soft-AP.

2.4.2.4. Thiết lập mạng

Phần này mô tả các chức năng để thiết lập và cấu hình ESP8266 ở chế độ điểm truy cập mềm (soft-AP).

2.4.2.5. SoftAP

Cách thiết lập đơn giản nhất chỉ yêu cầu một tham số và được sử dụng để thiết lập một mạng Wi-Fi mở.

`WiFi.softAP(ssid)`

Để thiết lập mạng được bảo vệ bằng mật khẩu, hoặc để cấu hình các thông số mạng bổ sung, sử dụng quá tải sau đây:

`WiFi.softAP(ssid, password, channel, hidden)`

Tham số đầu tiên của hàm này là bắt buộc, còn lại ba tùy chọn.

`ssid` - chuỗi ký tự chứa SSID mạng (tối đa 63 ký tự).

`password` - chuỗi ký tự tùy chọn với mật khẩu. Đối với mạng WPA2-PSK, nó phải có ít nhất 8 ký tự. Nếu không được chỉ định, điểm truy cập sẽ mở ra cho bất kỳ ai kết nối.

`channel` - Tham số tùy chọn để thiết lập kênh Wi-Fi, từ 1 đến 13. Kênh mặc định = 1.

`hidden` - Tham số tùy chọn, thiết lập là true để ẩn SSID.

2.4.2.6. Thư viện ESP8266HTTPClient.h

Chứa các hàm:

`http.begin("http://tranlyquynh123.freeasphost.net/sendrequest?")`. Để gửi kết yêu cầu truy cập đến một địa chỉ.

`http.GET()` nhận dữ liệu trả về từ trang web.

Thư việ ESP8266WebServer.h

```
server.on("/", []()

{

    IPAddress ip = WiFi.softAPIP();

    String ipStr = String(ip[0]) + '.' + String(ip[1]) + '.' + String(ip[2]) + '.'
+ String(ip[3]);

    content = "<!DOCTYPE HTML>\r\n<html><h2>XSwitch</h2>";

    content += "<form method=\"get\" action=\"setting\">";

    content += "<div>Wifi</div>";

    content += "<div><input name=\"ssid\" size=\"40\"></div>";

    content += "<div>Mat Khau</div>";

    content += "<div><input name=\"pass\" size=\"40\"></div>";

    content += "<div><input type='submit'></div>";

    content += "<p>";

    content += st;

    content += "</p>";

    content += "</html>";

    server.send(200, "text/html", content);

});
```

2.4.3. Giới thiệu về Arduino và Atmega328

Arduino là một nền tảng điện tử nguồn mở dựa trên những phần cứng và phần mềm linh hoạt, dễ sử dụng. Arduino dành cho những nghệ sĩ, nhà thiết kế, người thích tìm tòi và tất cả những ai yêu thích sáng tạo các sản phẩm và môi trường có tính tương tác cao.

Mạch arduino được lắp ráp từ các linh kiện dễ tìm và hướng đến đối tượng người dùng đa dạng. Đừng lo nếu bạn là dân nghiệp dư vì arduino có một hệ thống thư viện phong phú, cộng đồng người dùng arduino đông đảo sẵn sàng chia sẻ kiến thức và mã nguồn sẽ giúp bạn tạo nên những dự án thiết thực.

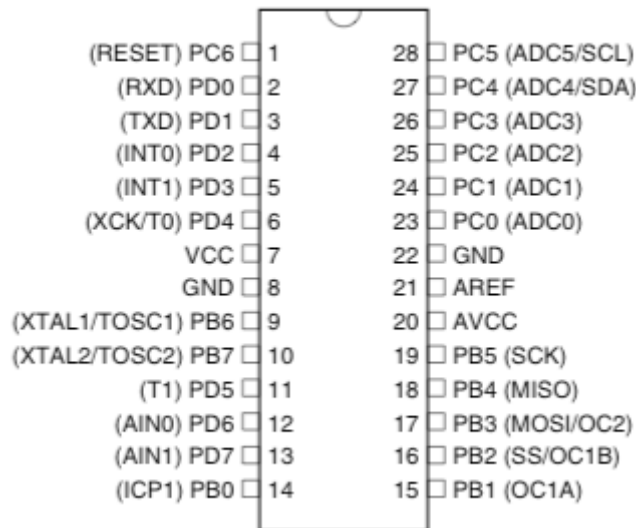
Một vài thông số của Arduino UNO R3

Vi điều khiển	ATmega328 họ 8bit
Điện áp hoạt động	5V DC (chỉ được cấp qua cổng USB)
Tần số hoạt động	16 MHz
Dòng tiêu thụ	khoảng 30mA
Điện áp vào khuyến dùng	7-12V DC
Điện áp vào giới hạn	6-20V DC
Số chân Digital I/O	14 (6 chân hardware PWM)
Số chân Analog	6 (độ phân giải 10bit)
Dòng tối đa trên mỗi chân I/O	30 mA
Dòng ra tối đa (5V)	500 mA
Dòng ra tối đa (3.3V)	50 mA
Bộ nhớ flash	32 KB (ATmega328) với 0.5KB dùng bởi bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)

Hình 2.4.3- 1 Thông số của Arduino Uno R3.

Atmega328 là một chip vi điều khiển được sản xuất bởi hãng [Atmel](#) thuộc họ MegaAVR có sức mạnh hơn hẳn [Atmega8](#). Atmega 328 là một bộ vi điều khiển 8 bit dựa trên kiến trúc RISC bộ nhớ chương trình 32KB ISP flash có thể ghi xóa hàng nghìn lần, 1KB EEPROM, một bộ nhớ RAM vô cùng lớn trong thế giới vi xử lý 8 bit (2KB SRAM).

Với 23 chân có thể sử dụng cho các kết nối vào hoặc ra i/O, 32 thanh ghi, 3 bộ timer/counter có thể lập trình, có các ngắt nội và ngoại (2 lệnh trên một vector ngắt), giao thức truyền thông nối tiếp USART, SPI, I2C. Ngoài ra có thể sử dụng bộ biến đổi số tương tự 10 bit (ADC/DAC) mở rộng tới 8 kênh, khả năng lập trình được watchdog timer, hoạt động với 5 chế độ nguồn, có thể sử dụng tới 6 kênh điều chế độ rộng xung (PWM), hỗ trợ bootloader.



Hình 2.4.3- 2 Chân của vi điều khiển Atmega328.

Atmega328 có khả năng hoạt động trong một dải điện áp rộng (1.8V – 5.5V), tốc độ thực thi (thông lượng) 1MIPS trên 1MHz.

Ngày nay vi điều khiển Atmega328 thực sự được sử dụng phổ biến từ các dự án nhỏ của sinh viên, học sinh với giá thành rẻ, xử lý mạnh mẽ, tiêu tốn ít năng lượng (chế độ hoạt động: 0.2 mA, chế độ ngủ: 0.1 μ A, chế độ tích kiệm: 0.75 μ A) và sự hỗ trợ nhiệt tình của cộng đồng người dùng AVR. Và không thể không nhắc tới sự thành công của Vi điều khiển Atmega328 trong dự án mã nguồn mở Arduino với các modul Arduino Uno (R3), Arduino Nano, Arduino Pro mini những sản phẩm dẫn dắt chúng ta vào thế giới mã nguồn mở để hoàn thành một chương trình trong “nháy mắt”.

Thông số chính Atmega328P:

- Kiến trúc: AVR 8bit.
- Xung nhịp lớn nhất: 20Mhz.
- Bộ nhớ chương trình (FLASH): 32KB.
- Bộ nhớ EEPROM: 1KB.
- Bộ nhớ RAM: 2KB.
- Điện áp hoạt động rộng: 1.8V – 5.5V.
- Số timer: 3 timer gồm 2 timer 8-bit và 1 timer 16-bit.
- Số kênh xung PWM: 6 kênh (1timer 2 kênh).

2.4.4. Một số linh kiện khác

2.4.4.1. Relay

Relay là một **công tắc** (khóa K). Nhưng khác với công tắc ở một chỗ cơ bản, relay được kích hoạt bằng điện thay vì dùng tay người. Relay là một công tắc nên nó có 2 trạng thái: **đóng** và **mở**.



Hình 2.4.4.1- 1 Relay.

Cấu tạo Relay gồm 2 phần:

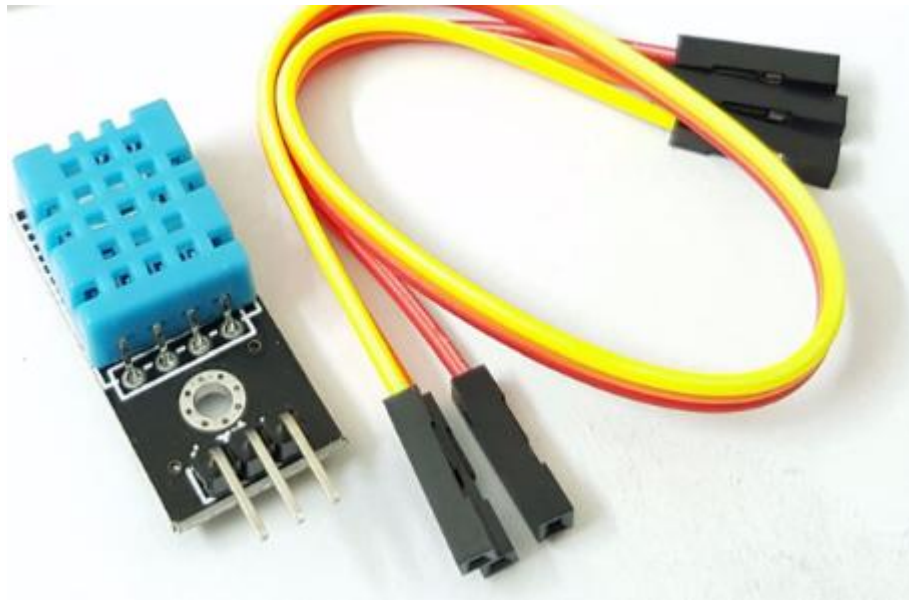
+ Cuộn hút:

- Tạo ra năng lượng từ trường để hút tiếp điểm về phía mình.
- Tùy vào điện áp làm việc người ta chia Relay ra DC: 5V, 12V, 24V-AC: 110V, 220V.

+ Cặp tiếp điểm:

- Khi không có từ trường. Tiếp điểm 1 được tiếp xúc với tiếp điểm 2 nhờ lực của lò xo. Tiếp điểm thường đóng.
- Khi có năng lượng từ trường thì tiếp điểm 1 bị hút chuyển sang 3.
- Trong Relay có thể có 1 cặp tiếp điểm, 2 cặp tiếp điểm hoặc nhiều hơn.

2.4.4.2. Cảm biến nhiệt độ và độ ẩm (DHT11)



Hình 2.4.4.2- 1 Cảm biến nhiệt độ và độ ẩm.

Cảm biến độ ẩm, nhiệt độ DHT11 ra chân được tích hợp sẵn điện trở 5.1K giúp người dùng dễ dàng kết nối và sử dụng hơn so với cảm biến DHT11 chưa ra chân. Bộ tiền xử lý tín hiệu tích hợp trong cảm biến giúp bạn có được dữ liệu chính xác mà không cần phải qua bất kỳ tính toán nào. Module được thiết kế hoạt động ở mức điện áp 5VDC.

Thông số kỹ thuật:

- Điện áp hoạt động: 5VDC.
- Chuẩn giao tiếp: TTL, 1 wire.
- Khoảng đo nhiệt độ: 0-50°C.
- Khoảng đo độ ẩm: 20%-80%.
- Tần số lấy mẫu tối đa 1Hz.
- Kích thước: 28 x 12 x 10mm.

2.4.4.3. Cảm biến ánh sáng (BH1750)



Hình 2.4.4.3- 1 Cảm biến ánh sáng.

Cảm biến cường độ ánh sáng BH1750 được sử dụng để đo cường độ ánh sáng theo đơn vị lux, cảm biến có ADC nội vs bộ tiền xử lý nên giá trị được trả ra là giá trị trực tiếp cường độ ánh sáng lux mà không phải qua bất kỳ xử lý hay tính toán nào thông qua giao tiếp I2C.

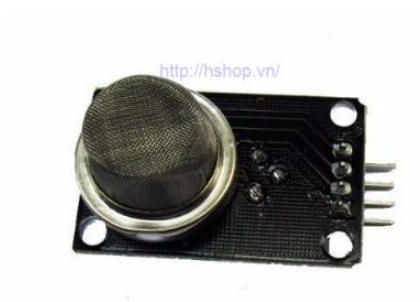
Thông số kỹ thuật:

- Nguồn 3-5VDC.
- Giao tiếp I2C.
- Khoảng đo: 1-65535 lux.
- Kích thước: 21 x 16 x 3mm.

Một số ví dụ về độ rọi của ánh sáng:

- Trời nhiều mây trong nhà: 5-50 lux.
- Trời nhiều mây ngoài trời: 50-500 lux.
- Trời nắng trong nhà: 100-1000 lux.
- Ánh sáng cần thiết để đọc sách: 50-60 lux.

2.4.4.4. Cảm biến khí gas (MQ2)



Hình 2.4.4.4- 1 Cảm biến khí gas.

MQ-2 sử dụng phần tử SnO_2 có độ dẫn điện thấp hơn trong không khí sạch, khi khí dễ cháy tồn tại, cảm biến có độ dẫn điện cao hơn, nồng độ chất dễ cháy càng cao thì độ dẫn điện của SnO_2 sẽ càng cao và được tương ứng chuyển đổi thành mức tín hiệu điện. MQ-2 là cảm biến khí có độ nhạy cao với LPG, Propane và Hydrogen, mê-tan và hơi dễ bắt lửa khác, với chi phí thấp và phù hợp cho các ứng dụng khác nhau.

Cảm biến xuất ra cả hai dạng tín hiệu là Analog và Digital, tín hiệu Digital có thể điều chỉnh mức cảnh báo bằng biến trở.

Nguồn sử dụng: 5VDC.

2.4.4.5. Cảm biến chuyển động



Hình 2.4.4.5- 1 Cảm biến chuyển động.

Cảm biến thân nhiệt chuyển động PIR HC-SR501 được sử dụng để phát hiện chuyển động của các vật thể phát ra bức xạ hồng ngoại, cảm biến có thể chỉnh được độ nhạy để giới hạn khoảng cách bắt xa gần cũng như cường độ bức xạ của vật thể mong muốn, ngoài ra cảm biến còn có thể điều chỉnh thời gian kích trễ qua biến trở tích hợp sẵn.

Thông số kỹ thuật:

- Phạm vi phát hiện: độ xa tối đa 6m.
- Nhiệt độ hoạt động: $0-50^{\circ}\text{C}$.
- Điện áp hoạt động: 3.8-5VDC.
- Mức tiêu thụ dòng: $\leq 50\text{ uA}$.
- Kích thước: 32.2 x 24.3 x 25.4mm.

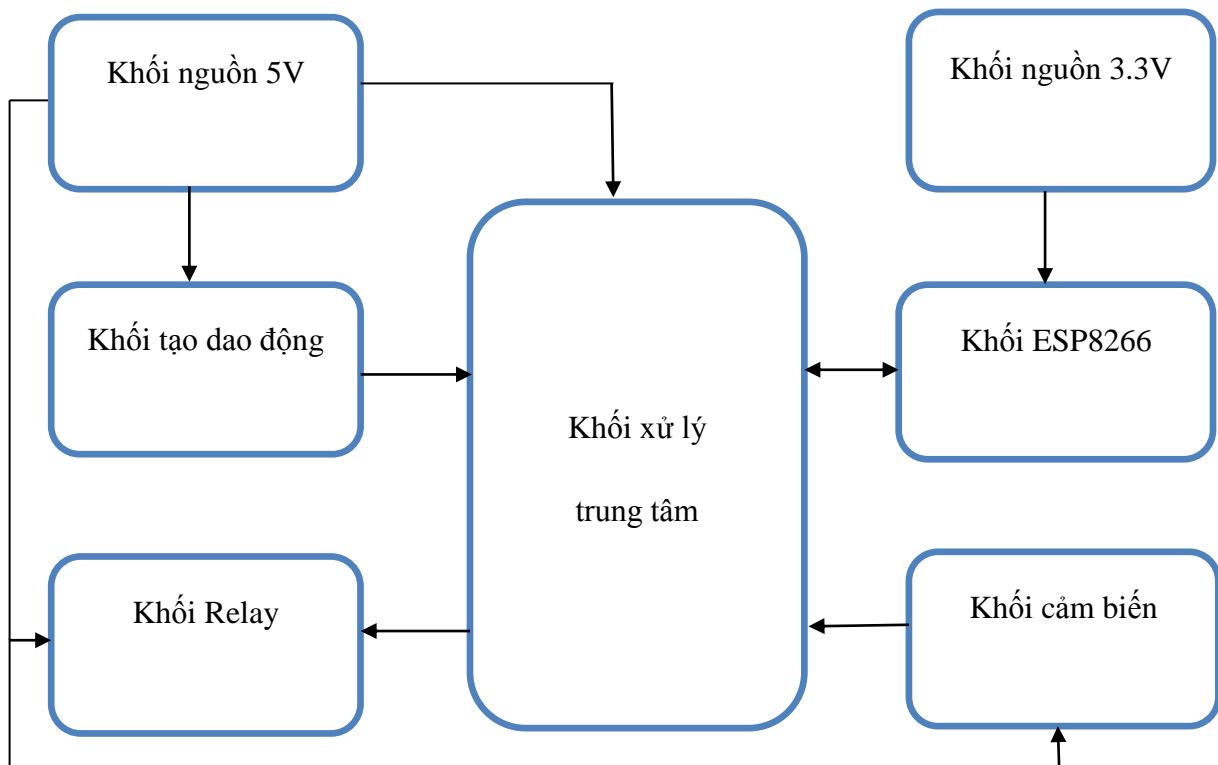
3. THIẾT KẾ VÀ THỰC HIỆN PHẦN CỨNG

3.1. Yêu cầu thiết kế phần cứng

Phần thiết kế có chức năng nhận dữ liệu nhận được từ ESP8266 và xử lý dữ liệu. Chuyển thành lệnh điều khiển các Relay..

Mình chọn IC ATMEGA328 để làm vi xử lý trung tâm vì: trong quá trình thực hiện đồ án và lên ý tưởng. Mình dùng mạch Arduino để nạp code và chạy thử nghiệm. Sau khi mình thử nghiệm thành công mình chuyển sang dùng luôn IC của mạch này cho tiện.

3.2. Sơ đồ khối tổng quát.



3.2.1 Thiết kế khối Relay

Để điều khiển các thiết bị điện thông thường, ta không thể sử dụng trực tiếp các chân ngõ ra của vi điều khiển vì hạn chế của điện áp và dòng ra của vi điều khiển. Muốn điều khiển thông qua vi điều khiển, ta cần một mạch relay. Mạch này bao gồm một BJT đóng vai trò khóa điện tử, nhận tín hiệu điều khiển từ vi điều khiển. BJT sẽ đóng ngắt nguồn điện đi qua cuộn dây của relay, từ đó relay sẽ đóng ngắt các thiết bị điện.

Đèn LED được mắc song song với cuộn dây của relay để chỉ thị trạng thái của relay. Điện áp rơi trên LED là 2.47-3.7V và chọn dòng qua LED là 10 -20mA, ta tính được R10 như sau:

$$R10 = \frac{5 - 3.7}{0.02} \sim \frac{5 - 2.47}{0.01} = 65 \sim 223\Omega$$

Chọn R10 là 220Ω, khi đó dòng qua LED là 11.5 mA, đảm bảo để LED sáng bình thường.

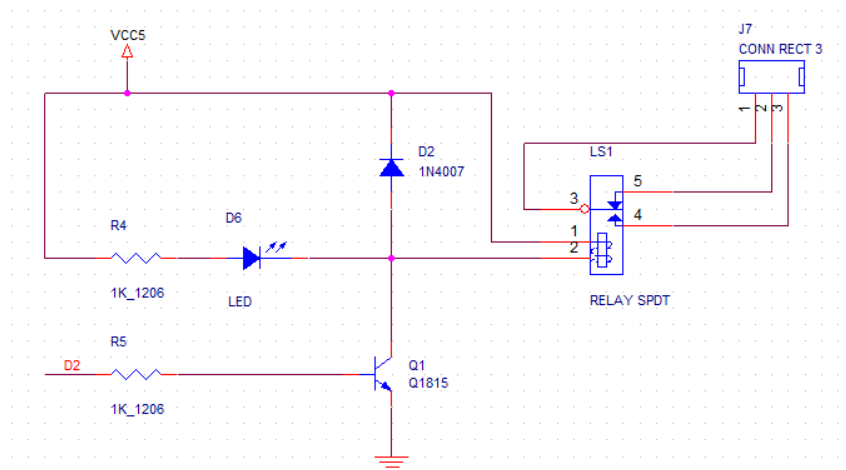
Ở đây sử dụng relay SRD-05VDC-SL-C có điện áp hoạt động của cuộn dây là 5V, có thể đóng ngắt tải có điện áp xoay chiều tối đa là 250V, điện áp một chiều tối đa là 30V và dòng tối đa là 15A. Dòng đi qua cuộn dây lúc hoạt động là 71.4 mA.

BJT Q1815 được phân cực để hoạt động như khóa điện tử. Vì ngõ ra các chân của ATMEGA328 có điện áp là 5 V nên khi có tín hiệu mức cao tương ứng với 5 V thì BJT dẫn bão hòa. Để BJT dẫn bão hòa thì :

$$I_B \geq \frac{I_C}{h_{fe \min}} \text{ với } I_C = 80 \text{ mA và } h_{fe \min} = 30$$

Khi đó $I_B \geq 2.67 \text{ mA}$. Từ đó tính được R1:

$$R1 < \frac{V_{IN1} - V_{BE}}{I_B} = \frac{5 - 0.7}{0.0026} = 1654\Omega. \text{ Chọn } R1 = 1K.$$



Hình 3.2.1- 1 Sơ đồ mạch Relay.

Khi áp ở ngõ vào cực B là 5V thì BJT sẽ dẫn bão hòa, có dòng đi qua cuộn dây và đèn LED. Relay sẽ chuyển sang vị trí thường hở, đèn LED sẽ sáng.

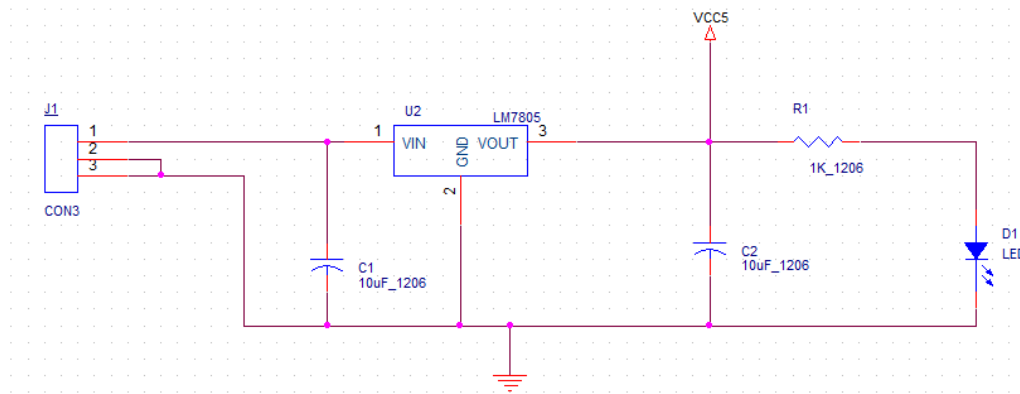
Khi áp ở ngõ vào cực B là 0 V, $V_{IN1} < V_{BE}$ nên BJT tắt, không có dòng đi qua cuộn dây, relay ở vị trí thường đóng.

Diode mắc ngược D1 có nhiệm vụ bảo vệ BJT khỏi các gai xung điện áp khi BJT tắt.

3.2.2 Thiết kế khối nguồn

Khối nguồn 5V:

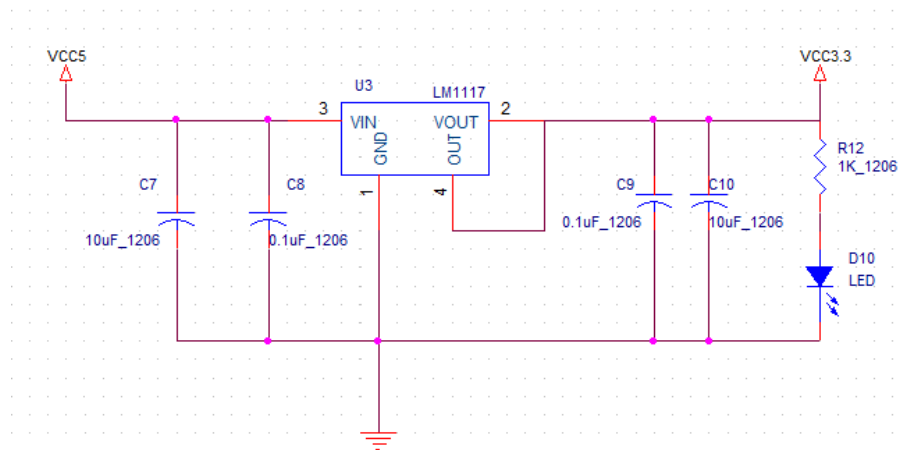
Dùng IC KA7805 để chuyển đổi nguồn vào 9VDC thành nguồn ra 5VDC.



Hình 3.2.2- 1 Sơ đồ mạch nguồn 5V.

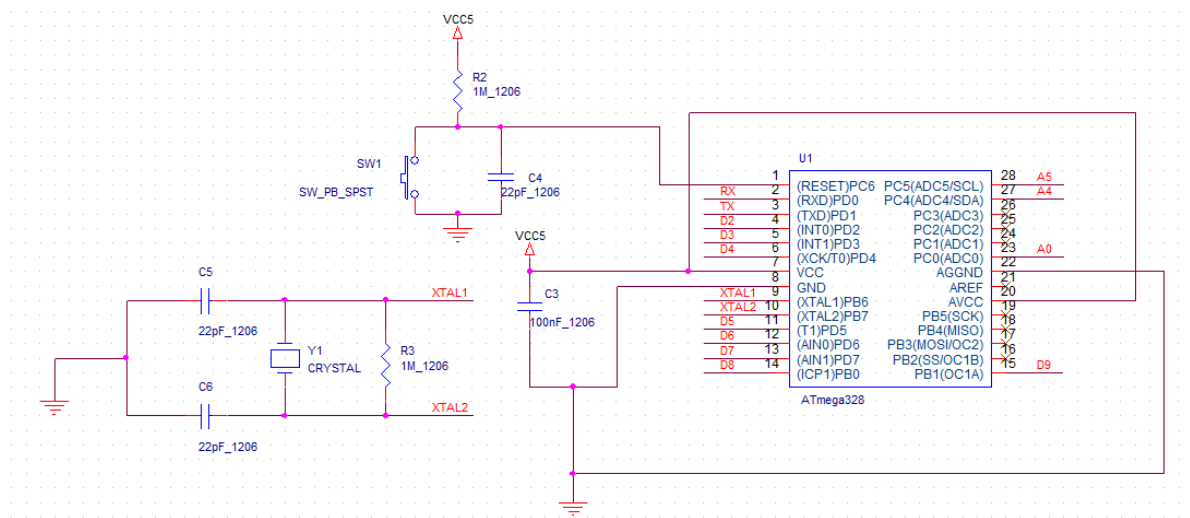
Khối nguồn 3.3V:

Module ESP8266 V1 khi sử dụng cần điện áp 3.3VDC và dòng lớn nên ta phải cấp nguồn riêng cho ESP chứ không sử dụng nguồn 3.3VDC của vi điều khiển. Ta dùng IC LM1117 để chuyển đổi điện áp 5VDC thành 3.3VDC để cấp nguồn cho ESP8266.



Hình 3.2.2- 2 Sơ đồ mạch nguồn 3.3V.

3.2.2 Khối xử lý trung tâm



4. THIẾT KẾ VÀ THỰC HIỆN PHẦN MỀM

4.1. Yêu cầu thiết kế

Yêu cầu thiết kế phần mềm gồm có hai phần chính là: trang Web, phía Server và phần mềm cho vi điều khiển.

Phần trang Web cần hiển thị các lựa chọn để điều khiển các thiết bị, trạng thái của các thiết bị và trạng thái của cảm biến. Ngoài ra còn hiển thị các dữ liệu ghi lại thời gian thay đổi trạng thái của thiết bị và thời gian phát hiện chuyển động của cảm biến. Phía Server có nhiệm vụ nhận các giá trị biến do vi điều khiển gửi lên và lưu vào cơ sở dữ liệu.

4.2. Phần mềm cho vi điều khiển và ESP8266

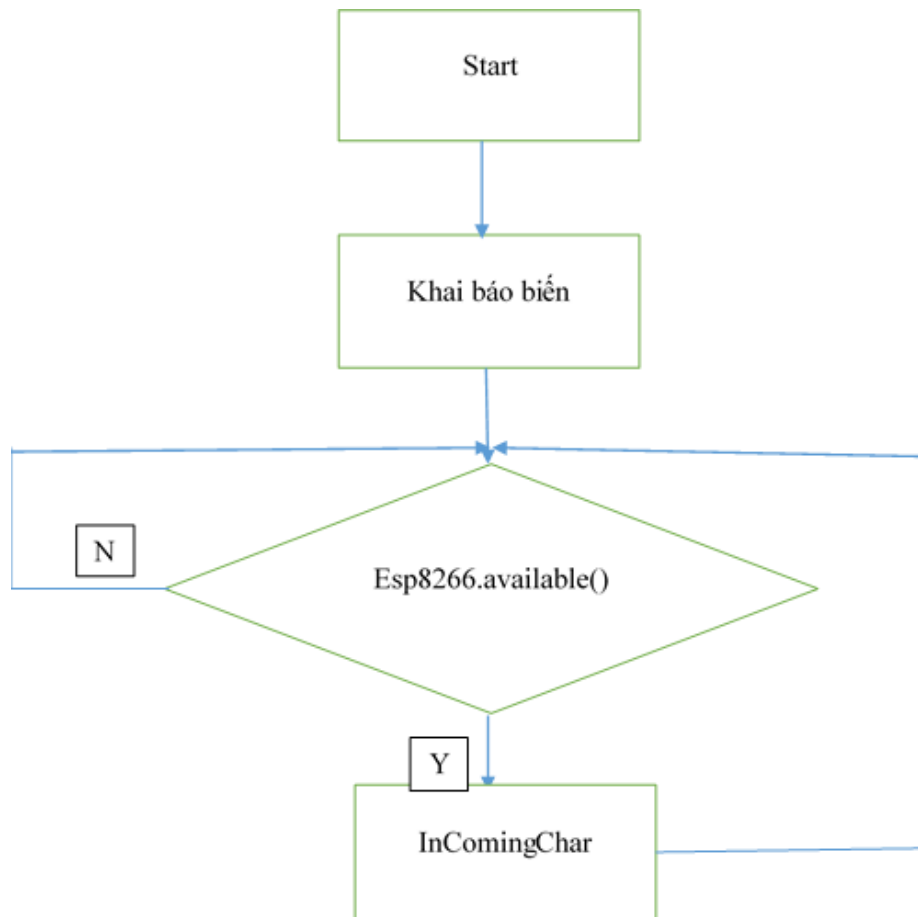
- Vi điều khiển có nhiệm vụ nhận lệnh từ ESP8266 và thực hiện điều khiển thiết bị rồi trả về kết quả cũng như trạng thái của các thiết bị.

- ESP8266 kết nối wifi gửi và nhận lệnh từ cơ sở dữ liệu.

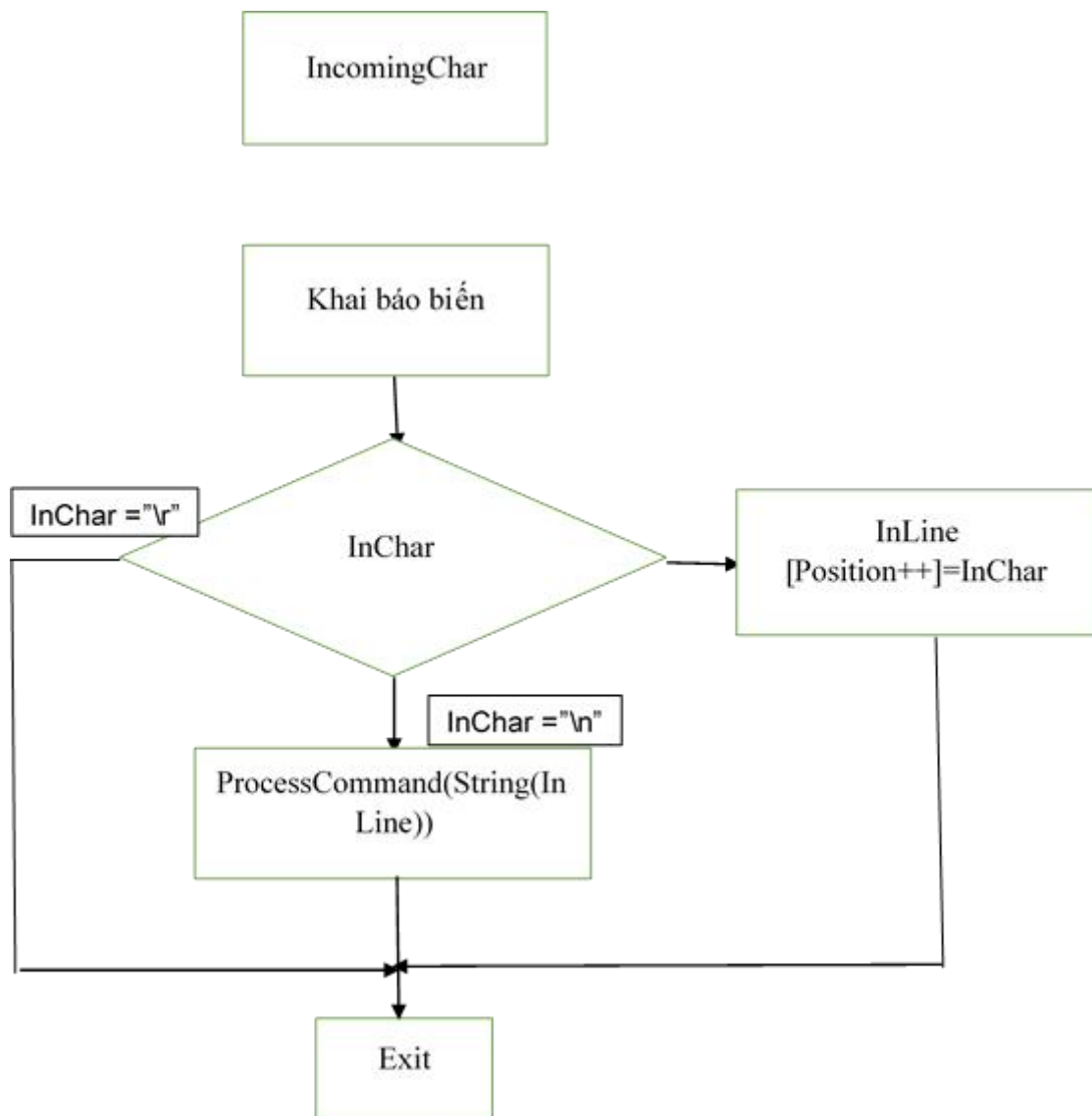
4.2.1. Phần mềm dành cho ATMEGA328

Là nhân của Kit Arduino nên ta lập trình cho ATMEGA này cũng y như lập trình cho Kit Arduino uno R3. Ta cũng dùng phần mềm biên dịch IDE để viết.

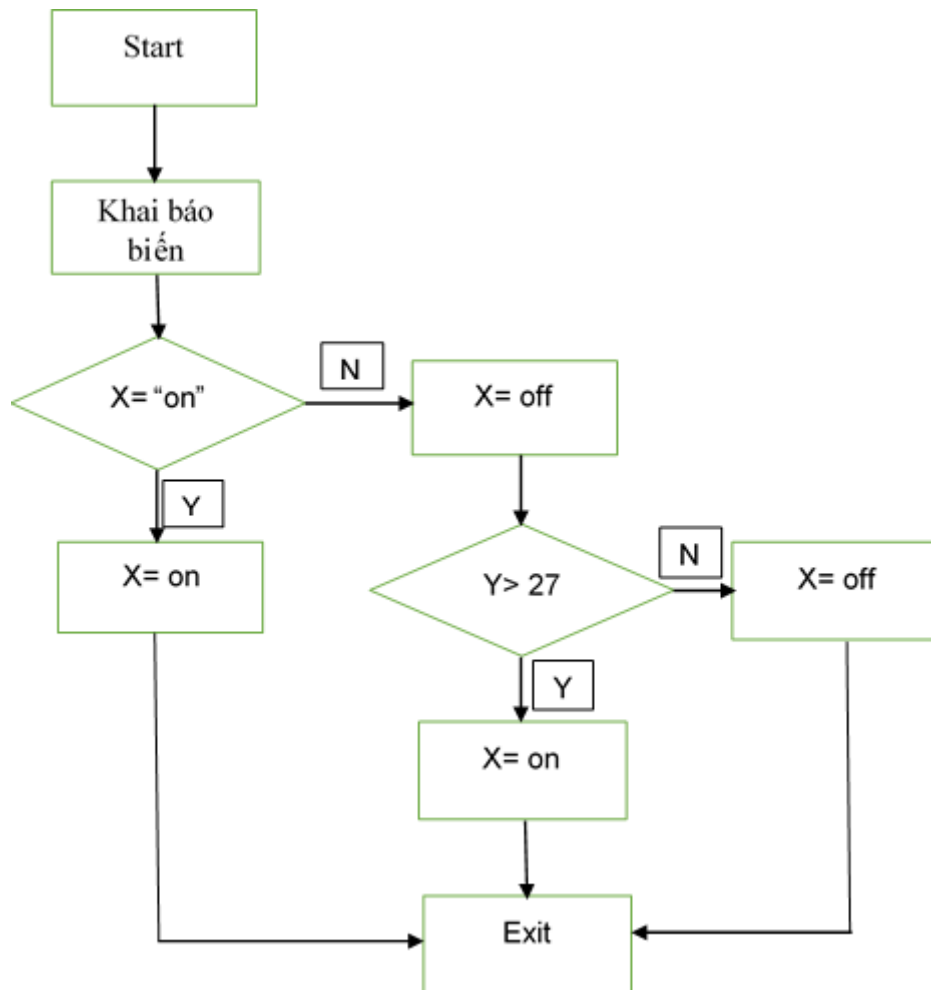
IC này làm việc như sau: Ban đầu chương trình khai báo các biến và các chân tín hiệu ra điều khiển các thiết bị. Chương trình sẽ mở cổng seriport để nhận tín hiệu từ ESP. Khi nhận tín hiệu xong. Chương trình sẽ lại xử lý tín hiệu và điều khiển thiết bị theo nội dung nhận được. Sau khi thực hiện xong chương trình sẽ đọc các tín hiệu độ ẩm đất, nhiệt độ trong phòng, độ ẩm trong phòng và ánh sáng trong phòng để gửi dữ liệu này qua ESP.



Hình 4.2.1- 1 Lưu đồ giải thuật.



Hình 4.2.1- 2 Lưu đồ giải thuật xuất chuỗi.



Hình 4.2.1- 3 Sơ đồ khối so sánh.

4.2.2. Phần mềm cho ESP8266

ESP8266 có thể được lập trình như một vi xử lý bình thường. Với đầy đủ chức năng của một MCU.

Ở đây mình dùng chính IDE của arduino để lập trình cho ESP. Với các thư viện hỗ trợ được cung cấp bởi nhà sản xuất của ESP.

```

#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>

#include <ESP8266WebServer.h>

#include "EEPROM.h"
    
```

Thư viện `#include <ESP8266WiFi.h>` được dùng để thiết lập các chế độ wifi.

- `WiFi.mode(WIFI_STA)`. chế độ điểm truy cập
- `WiFi.scanNetworks()`. Hàm tìm kiếm các điểm truy cập xung quanh

Sơ lược cách thức hoạt động của chương trình:

Khi khởi động chương trình. ESP đọc tên điểm truy cập và mật khẩu đã lưu trong bộ nhớ ERROM và thực hiện kết nối với điểm truy cập đó. Sau một khoảng thời gian. Nếu không truy cập được (có thể do tín hiệu yếu hoặc không còn tồn tại điểm truy cập đó nữa) thì ESP sẽ tự động chuyển sang chế độ phát wifi và là một webserver đơn giản để tương tác với người dùng. Khi người dùng truy cập và điểm phát wifi do ESP tạo ra. Và gõ vào trình duyệt địa chỉ 192.168.4.1 thì giao diện web sẽ mở ra và hiện các điểm truy cập hiện có gần đây. Người dùng có thể chọn và gõ mật khẩu vào để ESP tự lưu mật khẩu và tên truy cập vào ERROM để cho những lần kết nối sau. Sau khi nhập điểm truy cập và mật khẩu. Ta tiến hành khởi động lại ESP. Sau lần khởi động. Nếu truy cập được điểm phát wifi. ESP sẽ chạy vào vòng lặp. Trong vòng lặp này, chương trình sẽ truy cập tới một địa chỉ web đã định sẵn và đọc nội dung từ trang web đó.

Đây là trang web thể hiện các trạng thái mà người dùng muốn các thiết bị đáp ứng. ESP sẽ lọc dữ liệu và đọc các trạng thái đó bằng đoạn code.

Sau khi có được các trạng thái thì ESP sẽ gửi dữ liệu cho ATMEGA328 qua seriporrt bằng lệnh:

Sau đó ESP sẽ chờ phản hồi từ ATMEGA và nhận chuỗi dữ liệu để gửi lệnh web ở địa chỉ:

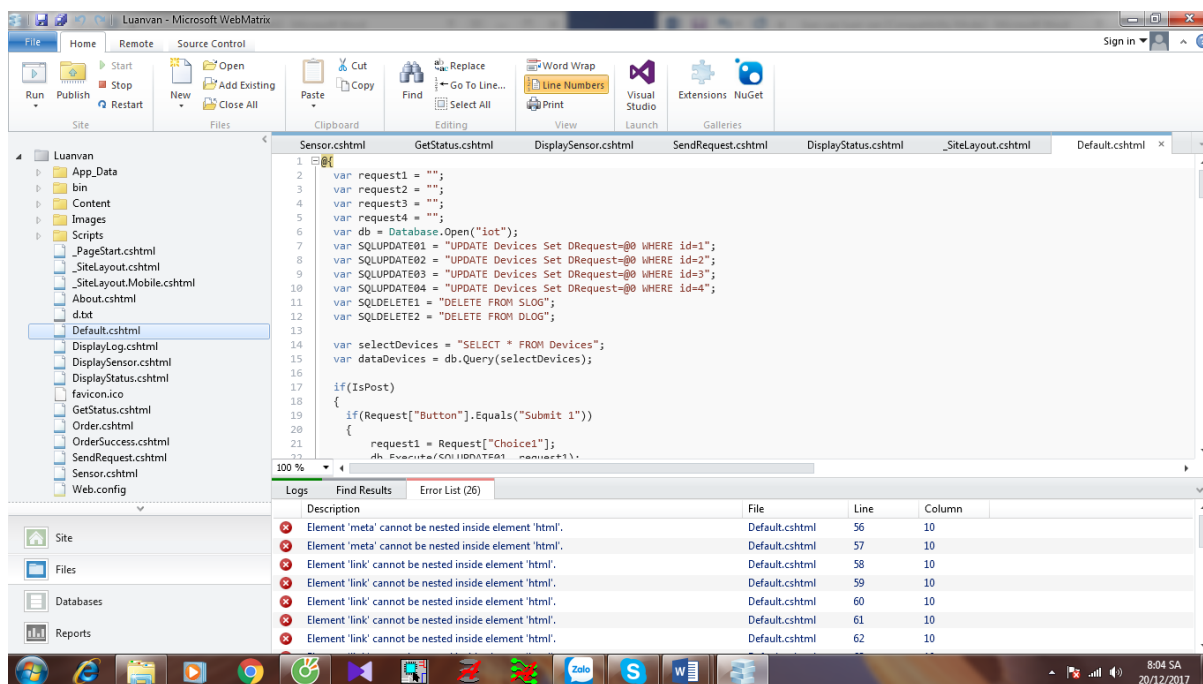
“Inline” là chuỗi dữ liệu mà ESP nhận được từ ATMEGA328.

Sau khi thực hiện xong ESP lại gửi lệnh truy cập lệnh web để lấy nội dung điều khiển của người dùng về và gửi cho ATMEGA328. Cứ như thế lặp đi lặp lại.

4.3. Trang Web và Server

4.3.1. Phần mềm Microsoft WebMatrix và ASP.NET framework

Giao diện trang Web, phía Server và cơ sở dữ liệu trong luận văn này đều được tạo trên phần mềm WebMatrix sử dụng ngôn ngữ HTML kết hợp với ASP.NET Razor.



Hình 4.3.1- 1 Phần mềm Webmatrix.

WebMatrix là một phần mềm ứng dụng phát triển Web miễn phí cho hệ điều hành Windows được phát hành bởi Microsoft. WebMatrix cho phép người phát triển xây dựng trang Web sử dụng các mẫu tích hợp có sẵn hoặc các ứng dụng mã nguồn mở thông dụng với sự hỗ trợ tuyệt đối với ASP.NET, PHP, Node.js và HTML5. Microsoft phát triển WebMatrix cho mục đích cung cấp cho người phát triển Web các tính năng lập trình, chỉnh sửa và phát hành trang Web chỉ trên một phần mềm duy nhất. Phiên bản mới nhất hiện nay là Microsoft WebMatrix 3.

ASP.NET là một chương trình khung ứng dụng Web phía Server thiết kế cho việc phát triển Web để tạo các trang Web động. Được phát triển bởi Microsoft cho phép người phát triển Web xây dựng các trang Web động, ứng dụng Web và dịch vụ Web.

ASP.NET Web pages, được biết chính thức với với tên Web Forms, là các khối xây dựng chính cho phát triển ứng dụng. Web forms được chứa trong các tập tin với phần mở rộng “.aspx”, các tập tin này thường chứa các đánh dấu động HTML cũng như các đánh dấu định nghĩa các điều khiển phía Server và các điều khiển của người dùng. Các mã động được chạy trên Server có thể được đặt trong một khối `<% -- mã động -- %>`.

Phần mềm WebMatrix hỗ trợ ASP.NET Razor. Razor là cú pháp lập trình của ASP.NET sử dụng để tạo trang Web động với ngôn ngữ lập trình C# hoặc VisualBasic.NET. Razor cho phép người dùng sử dụng quy trình xây dựng của HTML. Thay vì sử dụng cú pháp đánh dấu với các ký tự `<% -- %>` để chỉ ra khối mã lệnh, cú pháp Razor bắt đầu khối mã lệnh

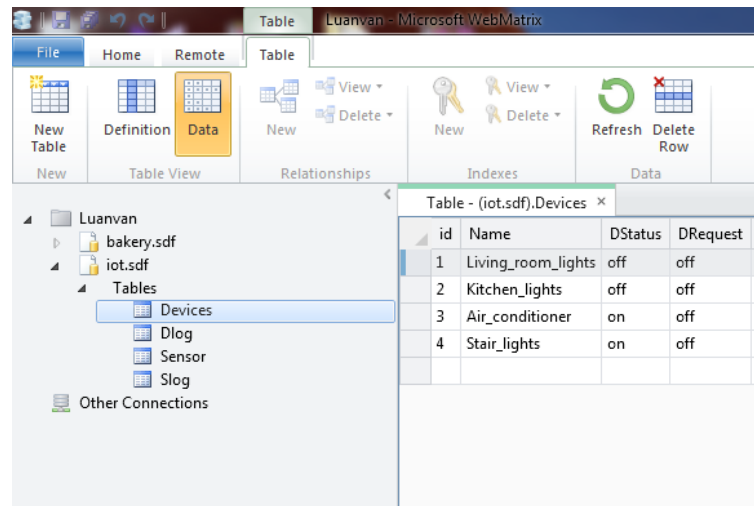
với một ký tự @ duy nhất và không cần thêm ký tự nào để đóng khối mã lệnh. Ưu điểm của Razor là cung cấp một cú pháp tối ưu cho HTML sử dụng một cách tiếp cận tập trung vào mã lệnh, với sự thay đổi nhỏ nhất giữa HTML và mã lệnh.

4.3.2. Tạo cơ sở dữ liệu

Cơ sở dữ liệu được tạo bằng Microsoft SQL Server 2012 trên phần mềm WebMatrix. Cơ sở dữ liệu gồm có các bảng dùng để lưu trữ dữ liệu được gọi lên từ vi điều khiển và ESP8266, đồng thời cũng lưu trữ các yêu cầu được chọn ở giao diện trang Web.

Cơ sở dữ liệu có tên *iot*, gồm có các bảng:

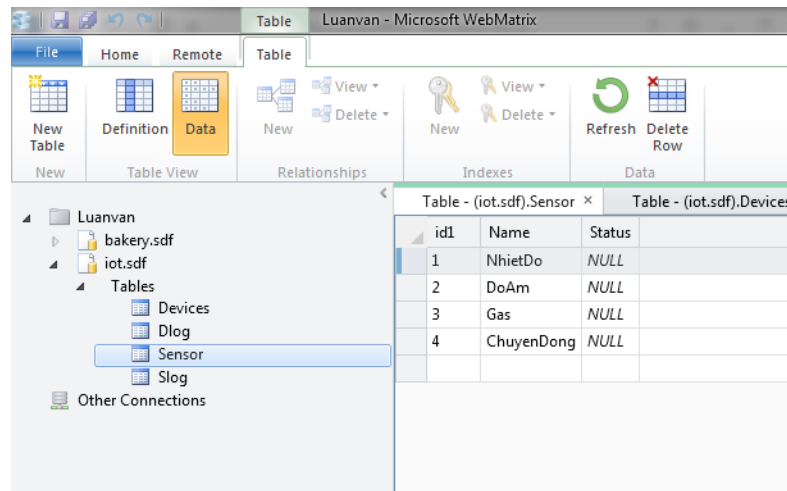
-Bảng *Devices* gồm có các cột: cột *id* chứa số thứ tự, cột *Name* chứa tên của các thiết bị, cột *DStatus* chứa trạng thái của các thiết bị, cột *DRequest* chứa yêu cầu từ dao diện trang Web.



id	Name	DStatus	DRequest
1	Living_room_lights	off	off
2	Kitchen_lights	off	off
3	Air_conditioner	on	off
4	Stair_lights	on	off

Hình 4.3.2- 1 Bảng *Devices*.

- Bảng *Sensor* gồm có các cột: cột *id* chứa số thứ tự, cột *Name* chứa tên cảm biến, cột *Status* chứa trạng thái của cảm biến.



Hình 4.3.2- 2 Bảng Sensor.

- Bảng DLog gồm có các cột: cột id chứa số thứ tự, cột Name chứa tên của thiết bị, cột Status chứa trạng thái vừa thay đổi của thiết bị, cột ATime chứa thời gian thay đổi của thiết bị.

- Bảng SLog gồm các cột: cột id chứa số thứ tự, cột Time chứa thời gian khi có sự thay đổi giá trị của các cảm biến – giá trị cột Status trong bảng Sensor là 1.

4.3.3. Trang hiển thị chính của trang Web

Trang hiển thị chính của trang Web được viết bằng HTML – Ngôn ngữ đánh dấu siêu văn bản, và các phần truy cập vào cơ sở dữ liệu phía Server sử dụng Razor.

Mô hình nhà thông minh điều khiển thiết bị qua Internet

Trạng thái thiết bị

- Living_room_lighta = off.
- Kitchen_lighta = off.
- Air_conditioner = on.
- Stair_lighta = on.

Các Cảm Biến

Nhiệt độ: _____

Độ ẩm: _____

Khí gas: Rò rỉ Gas.

Chuyển động: > Có độc nhập.

Bảng Điều Khiển

Living_room_lighta Submit 1

Kitchen_lighta Submit 2

Air_conditioner Submit 3

Stair_lighta Submit 4

Log Devices

Device	Change to	Time
--------	-----------	------

Log Sensor

Sensor	Change to	Time
--------	-----------	------

Hình 4.3.3- 1 Trang web server.

Trang hiển thị chính của trang Web có 3 phần chính: phần hiển thị trạng thái của cảm biến, phần hiển thị trạng thái của thiết bị và phần hiển thị các lựa chọn để điều khiển thiết bị.

Với phần hiển thị trạng thái cảm biến và hiển thị trạng thái của thiết bị, trang chính sẽ tải từ hai trang phụ là “DisplaySensor.cshtml” và “DisplayStatus.cshtml”. Trang chính sẽ tải hai trang này liên tục với khoảng trễ 500 ms để hiển thị tức thời trạng thái của cảm biến và thiết bị. Để thực hiện việc này, ta sử dụng đoạn mã JavaScript như sau:

<script>

```
function Load_external_content(){  
    $('#status').load('DisplayStatus.cshtml');  
    $('#sensor').load('DisplaySensor.cshtml')  
}  
setInterval('Load_external_content()', 500);  
</script>
```

Đoạn mã này sẽ tải trang “DisplaySensor.cshtml” vào phần được đặt trong phần đánh dấu có id="sensor" và tải trang “DisplayStatus.cshtml” vào phần được đặt trong phần đánh dấu có id="status". setInterval là phương pháp gọi hàm sau một khoảng thời gian tính bằng ms và thực thi hàm đó. Ở đây setInterval sẽ gọi hàm “Load_external_content()” sau khoảng thời gian 500 ms.

Với phần hiển thị các lựa chọn để điều khiển các thiết bị, sử dụng các Form để lựa chọn. Đoạn mã của một form như sau:

```
<form method="post">  
    <select name="Choice1">  
        <option value="on">On</option>  
        <option value="off">Off</option>  
    </select>  
    <input id="button1" type="submit" value="Submit 1" name="Button"/>  
</form>
```

Người dùng có thể lựa chọn hai lựa chọn là On và Off, sau đó nhấn nút Submit để xác nhận lựa chọn. Khi nhấn nút Submit, Form sẽ tạo một yêu cầu POST đến phía Server để thực thi việc lưu giá trị value vào cơ sở dữ liệu.

Nếu nhấn nút Submit 1, phía Server sẽ thực hiện cập nhật cột DRequest của bảng Devices tương ứng với id là 1. Tương tự, sẽ cập nhật cột DRequest tương ứng với id là 2 hoặc 3 nếu nhấn nút Submit 2 hoặc Submit 3.

Để cập nhật giá trị cơ sở dữ liệu ta sử dụng cú pháp câu lệnh SQL:

```
UPDATE table_name
SET column1=value1,column2=value2,...
WHERE some_column=some_value;
```

Phía Server được viết bằng ASP.NET Razor. Để kết nối với cơ sở dữ liệu ta sử dụng phương pháp Database.Open(filename). Để thực thi các câu lệnh SQL, ta sử dụng Database.Execute(SQLstatement[,parameters]). Đoạn mã ASP.NET Razor để thực hiện cập nhật cột DRequest tương ứng với id là 1 khi có yêu cầu POS như sau:

```
@{
    var request1 = "";
    var db = Database.Open("iot");
    var SQLUPDATE01 = "UPDATE Devices Set DRequest=@0 WHERE id=1";

    if(IsPost){
        if(Request["Button"].Equals("Submit 1")){
            request1 = Request["Choice1"];
            db.Execute(SQLUPDATE01, request1);
        }
    }
}
```

4.3.4. Trang hiển thị trạng thái của cảm biến và thiết bị

Để lựa chọn dữ liệu trong một bảng để hiển thị từ cơ sở dữ liệu ta sử dụng cú pháp câu lệnh SQL như sau:

```
SELECT column_name,column_name
FROM table_name;
```

Hoặc để lựa chọn toàn bộ dữ liệu trong một bảng ta sử dụng cú pháp câu lệnh SQL như sau :

```
SELECT * FROM table_name;
```

Với trang hiển thị trạng thái cảm biến “DisplaySensor.cshtml”, ta sẽ lấy dữ liệu từ bảng Sensor. Nếu như giá trị của cột Status là 1 thì sẽ hiển thị “Movement Detected”, nếu giá trị là 0 thì sẽ hiển thị “No movement”. Đoạn mã ASP.NET để lấy dữ liệu từ cơ sở dữ liệu như sau:

```
@{  
  
    var db = Database.Open("iot");  
  
    var sqlQ1 = "SELECT * FROM Sensor WHERE id1=1";  
  
    var sqlQ2 = "SELECT * FROM Sensor WHERE id1=2";  
  
    var sqlQ3 = "SELECT * FROM Sensor WHERE id1=3";  
  
    var sqlQ4 = "SELECT * FROM Sensor WHERE id1=4";  
  
    var selectQueryStrig = "SELECT * FROM SLOG ORDER BY id3";  
  
    var data1 = db.QuerySingle(sqlQ1);  
  
    var sstt1 = data1.Status;  
  
    var Name1 = data1.Name;  
  
    var data2 = db.QuerySingle(sqlQ2);  
  
    var sstt2 = data2.Status;  
  
    var Name2 = data2.Name;  
  
    var data3 = db.QuerySingle(sqlQ3);  
  
    var sstt3 = data3.Status;  
  
    var Name3 = data3.Name;  
  
    var data4 = db.QuerySingle(sqlQ4);  
  
    var sstt4 = data4.Status;  
  
    var Name4 = data4.Name;  
  
    var Display1 = "";  
  
    if(sstt3 == "0")  
    {  
        Display1 = "Bình thường.";  
    }  
}
```



```
if(sstt3 != "0")
{
    Display1 = "Rò khí Gas.";
}

var Display2 = "";

if(sstt4 == "0")
{
    Display2 = "Không có người.";
}

if(sstt4 != "0")
{
    Display2 = "Có đột nhập.";
}
}
```

Để hiển thị giá trị của biến Display trong đoạn mã HTML, chỉ cần gọi @Display.

Đối với trang hiển thị trạng thái thiết bị “[DisplayStatus.cshtml](#)”, ta sẽ lấy các giá trị của cột DStatus trong bảng Devices để hiển thị bằng các dòng lệnh như sau:

```
<ul>

    @foreach(var row in data){

        <li>@row.Name is @row.DStatus.</li>

    }

</ul>
```

Hiển thị thời gian thay đổi trạng thái của các thiết bị bằng phương pháp tương tự như trong trang hiển thị trạng thái cảm biến.

4.3.5. Trang Web Server

Gồm hai trang riêng biệt, giá trị của cảm biến sẽ được gọi lên trang “[Sensor.cshtml](#)” và giá trị của các thiết bị sẽ được gọi lên trang “[GetStatus.cshtml](#)”.

Các giá trị biến được gửi lên các trang này nhờ vào giao thức chuyển tải siêu văn bản (HTTP). Hai phương pháp thường được sử dụng là GET và POST. Ở đây sử dụng phương pháp GET. Chuỗi truy vấn (các cặp tên/giá trị) của phương pháp GET được gửi trong đường dẫn URL như sau:

[/GetStatus.cshtml?name1=value1&name2=value2](#)

Trang GetStatus.cshtml” sẽ đọc giá trị từ đường dẫn và lưu vào cơ sở dữ liệu. Đồng thời sẽ ghi lại thời gian thay đổi của các giá trị vào cơ sở dữ liệu.

Để đọc giá trị biến từ chuỗi truy vấn HTTP, sử dụng cú pháp Request.QueryString(variable).

@{

```
var db = Database.Open("iot");

var ustt1 = Request.QueryString["Living_room_lights"];

var ustt2 = Request.QueryString["Kitchen_lights"];

var ustt3 = Request.QueryString["Air_conditioner"];

var ustt4 = Request.QueryString["Stair_lights"];

var ustt5 = Request.QueryString["NhietDo"];

var ustt6 = Request.QueryString["DoAm"];

var ustt7 = Request.QueryString["Gas"];

var ustt8 = Request.QueryString["ChuyenDong"];

var SQLUPDATE1 = "UPDATE Devices Set DStatus=@0 WHERE id=1";

var SQLUPDATE2 = "UPDATE Devices Set DStatus=@0 WHERE id=2";

var SQLUPDATE3 = "UPDATE Devices Set DStatus=@0 WHERE id=3";

var SQLUPDATE4 = "UPDATE Devices Set DStatus=@0 WHERE id=4";

var SQLUPDATE5 = "UPDATE Sensor Set Status=@0 WHERE id1=1";

var SQLUPDATE6 = "UPDATE Sensor Set Status=@0 WHERE id1=2";

var SQLUPDATE7 = "UPDATE Sensor Set Status=@0 WHERE id1=3";

var SQLUPDATE8 = "UPDATE Sensor Set Status=@0 WHERE id1=4";

var sqlQ1 = "SELECT * FROM Devices WHERE id=1";

var sqlQ2 = "SELECT * FROM Devices WHERE id=2";

var sqlQ3 = "SELECT * FROM Devices WHERE id=3";

var sqlQ4 = "SELECT * FROM Devices WHERE id=4";

var sqlQ5 = "SELECT * FROM Sensor WHERE id1=1";
```

```
var sqlQ6 = "SELECT * FROM Sensor WHERE id1=2";

var sqlQ7 = "SELECT * FROM Sensor WHERE id1=3";

var sqlQ8 = "SELECT * FROM Sensor WHERE id1=4";

var SQLINSERTD = "INSERT INTO DLog (Name,Status,ATime) VALUES (@0,@1,@2)";

var SQLINSERTS = "INSERT INTO DLog (Name,Status,Time) VALUES (@0,@1,@2)";

string vnTimeZoneKey = "SE ASIA Standard Time";

TimeZoneInfo vnTimeZone =
TimeZoneInfo.FindSystemTimeZoneById(vnTimeZoneKey);

DateTime timelog =
TimeZoneInfo.ConvertTimeFromUtc(DateTime.UtcNow,vnTimeZone);

var data1 = db.QuerySingle(sqlQ1);
var data2 = db.QuerySingle(sqlQ2);
var data3 = db.QuerySingle(sqlQ3);
var data4 = db.QuerySingle(sqlQ4);
var data5 = db.QuerySingle(sqlQ5);
var data6 = db.QuerySingle(sqlQ6);
var data7 = db.QuerySingle(sqlQ7);
var data8 = db.QuerySingle(sqlQ8);

if(String.Compare(ustt1,data1.DStatus) != 0)
{
    db.Execute(SQLINSERTD,"Living_room_lights",ustt1,timelog);
    db.Execute(SQLUPDATE1, ustt1);
}

if(String.Compare(ustt2,data2.DStatus) != 0)
{
    db.Execute(SQLINSERTD,"Kitchen_lights",ustt2,timelog);
    db.Execute(SQLUPDATE2, ustt2);
}
```

```
if(String.Compare(ustt3,data3.DStatus) != 0)
{
    db.Execute(SQLINSERTD, "Air_conditioner",ustt3,timelog);
    db.Execute(SQLUPDATE3, ustt3);
}
if(String.Compare(ustt4,data4.DStatus) != 0)
{
    db.Execute(SQLINSERTD, "Stair_lights",ustt4,timelog);
    db.Execute(SQLUPDATE4, ustt4);
}
if(String.Compare(ustt5,data5.Status) != 0)
{
    db.Execute(SQLINSERTS, "NhietDo",ustt5,timelog);
    db.Execute(SQLUPDATE5, ustt5);
}
if(String.Compare(ustt6,data6.Status) != 0)
{
    db.Execute(SQLINSERTS, "DoAm",ustt6,timelog);
    db.Execute(SQLUPDATE6, ustt6);
}
if(String.Compare(ustt7,data7.Status) != 0)
{
    db.Execute(SQLINSERTS, "Gas",ustt7,timelog);
    db.Execute(SQLUPDATE7, ustt7);
}
if(String.Compare(ustt8,data8.Status) != 0)
{
    db.Execute(SQLINSERTS, "ChuyenDong",ustt8,timelog);
    db.Execute(SQLUPDATE8, ustt8);
}
```

```
}
```

4.3.6. Trang hiển thị yêu cầu cho thiết bị

Trang hiển thị yêu cầu “SendRequest.cshml”, có nhiệm vụ hiển thị các yêu cầu từ bảng Devices của cơ sở dữ liệu để ESP8266 có thể đọc và gửi cho vi điều khiển. Yêu cầu của một thiết bị được bắt đầu bằng kí tự <#> và kết thúc bằng ký tự <.> để dễ dàng nhận biết.

Mã của trang này như sau :

```
@{
```

```
var db = Database.Open("iot");  
var sqlQ = "SELECT * FROM Devices";  
var selectQueryString = "SELECT * FROM DLog ORDER BY id";  
var data = db.Query(sqlQ);
```

```
}
```

```
<!DOCTYPE html>  
  
<html lang="en">  
  
  <head>  
  
    <meta charset="utf-8" />  
  
    <title>Status</title>  
  
  </head>  
  
  <body>  
  
    <div id="statusupdate">  
  
      <ul>  
  
        @foreach(var row in data)  
        {  
  
          <li>  
  
            @row.Name = @row.DRequest.  
  
          </li>  
  
        }  
  
      </ul>  
  
    </div>
```

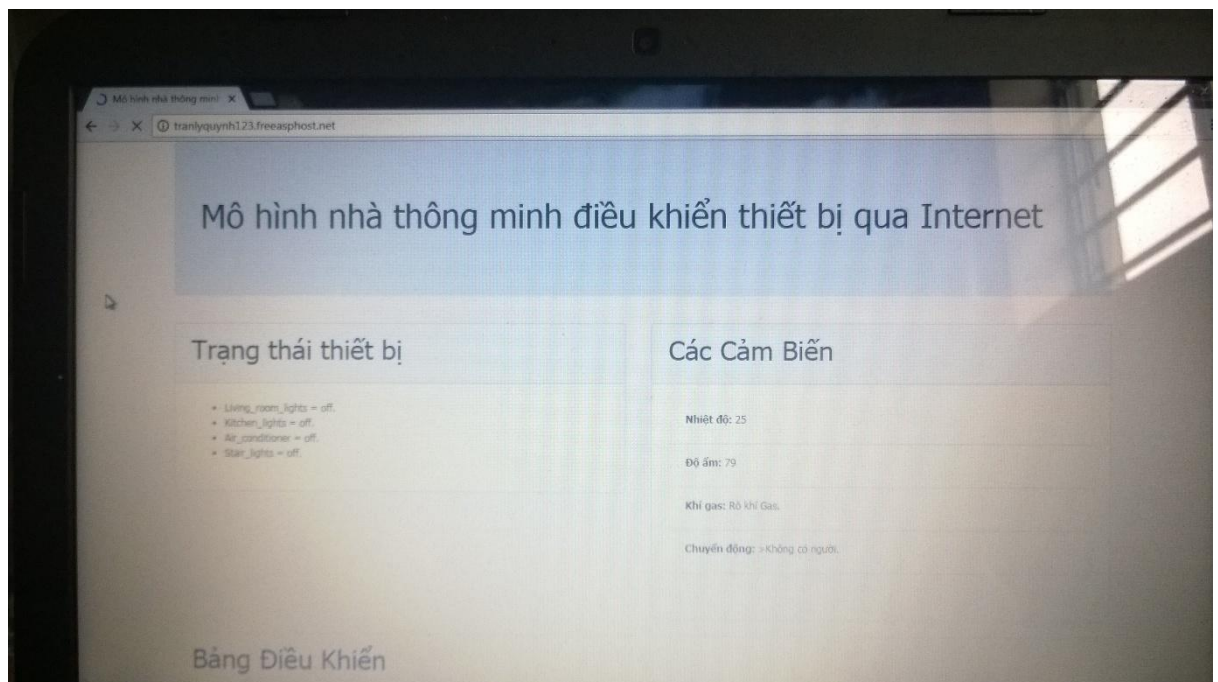
```
</body>  
</html>
```

5. KẾT QUẢ THỰC HIỆN

5.1. Đăng tải trang Web lên mạng Internet

Phần trang Web sau khi được viết trên WebMatrix và chạy thử trên LocalHost được tải lên host để đăng tải lên mạng. Trong đề án này lựa chọn sử dụng dịch vụ host của Freeasphost.net vì có cung cấp dịch vụ host miễn phí đồng thời hỗ trợ ASP.NET và cung cấp một tên miền con cho trang Web và không có quảng cáo.

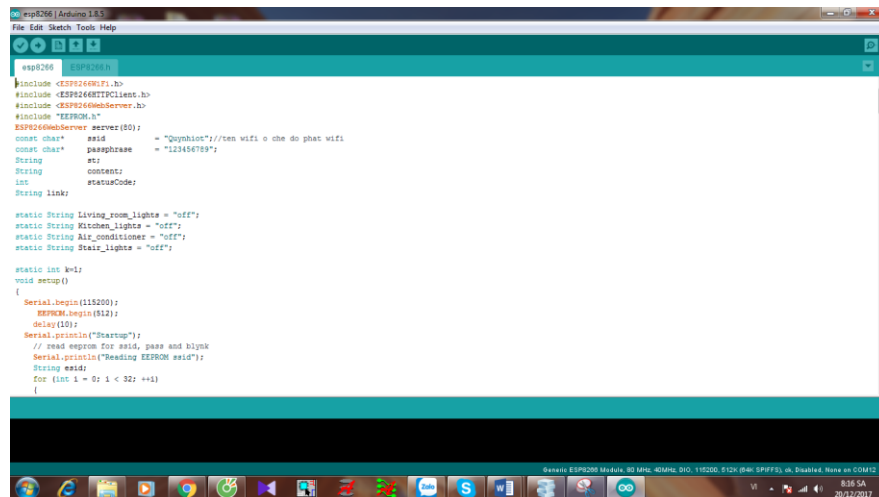
Dịch vụ host của Freeasphost cho phép đăng tải trang Web qua cách tải lên các tập tin một cách đơn giản. Trang Web đã được đưa lên host có tên miền “www.tranlyquynh123.freeasphost.net”.



Hình 5.1- 1 Trang web hoạt động.

5.2. Phần mềm của vi điều khiển

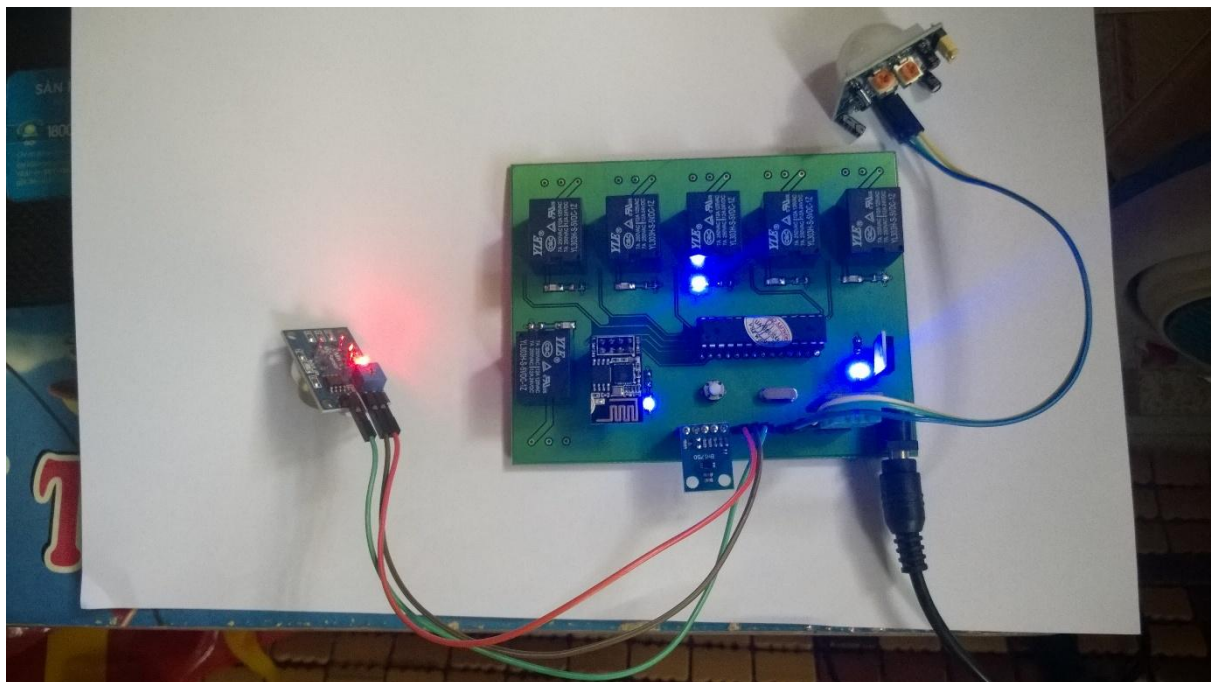
Phần mềm của ATMEGA328 và ESP8266 được viết trên IDE của Arduino. Trên đây có các thư viện hỗ trợ lập trình cũng như tải code lên vi điều khiển một cách đơn giản và thuận tiện trong quá trình thử nghiệm.



Hình 5.2- 1 Phần mềm IDE 1.8.5.

5.3. Hoạt động của toàn hệ thống

Sau khi thiết kế thi công và nạp code vào các vi điều khiển ta tiến hành chạy thử nghiệm.



Hình 5.3- 1 Mạch thực tế.

Lúc mới khởi động do ERROM chưa lưu sẵn nội dụng nên ESP sẽ chuyển sang chế độ Websever và làm một điểm phát wifi có tên "Quynhiot".

Sau khi đăng nhập vào wifi. Ta mở trình duyệt web và truy cập vào địa chỉ 192.168.4.1 thì ta được giao diện sau. Và điền wifi mình muốn cho hệ thống đăng nhập. Sau khi điền chính xác và nhấn nút Submint. Trang web trả về thông tin : {"Success": "Luu vao he

thong. Khoi dong lai ten wifi moi"}). Như vậy hệ thống đã cập nhật lại điểm truy cập Wifi và lưu vào ERROM.

Ta tiến hành khởi động lại hệ thống.

Hệ thống hoạt động tốt. Có delay không đáng kể.

6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1. Kết luận

Trong quá trình thực hiện đề tài, với sự hạn chế về thời gian và tài liệu vì vậy đòi hỏi bản thân phải thật cố gắng tìm tòi và nghiên cứu để hoàn thành đề tài một cách trọn vẹn. Quá trình thực hiện đề tài đã giúp rèn luyện khả năng tự tìm hiểu và tổng hợp các nguồn tài liệu, đồng thời rèn luyện các kỹ năng đã được học trong suốt quá trình học tập tại trường như thiết kế, thi công mạch in, viết chương trình cho vi điều khiển, ... cũng như biết thêm được các kiến thức mới như thiết kế trang Web đơn giản.

Ưu điểm:

- Hệ thống có tính thẩm mỹ cao.
- Có thể thi công thực tế trong ngôi nhà.
- Hoạt động ổn định và chính xác.
- Giá thành rẻ.
- Dễ dàng sử dụng đối với tất cả mọi người

Nhược điểm:

- Tính bảo mật chưa cao.
- Chưa thể biết thiết bị mình điều khiển có bị hư hay không.

6.2. Hướng phát triển

Hệ thống này có thể phát triển phía phần mềm, xây dựng cơ sở dữ liệu dữ liệu chặt chẽ hơn, thêm chức năng đăng nhập để đảm bảo sự bảo mật của người dùng. Có thể kết hợp các chức năng hẹn giờ, tự xử lý các tác vụ...

Với hệ thống wifi ngày càng phát triển và rộng khắp thì đây là một ứng dụng dành cho tất cả mọi người mọi nhà có nhu cầu.

Hệ thống này có thể được sử dụng trong cuộc sống hàng ngày như trong các hệ thống nhà thông minh. Nó còn có thể ứng dụng trong các ứng dụng an ninh để theo dõi cơ quan, nhà xưởng từ xa hoặc ứng dụng trong công nghiệp để theo dõi trạng thái của các thiết bị từ xa, hoặc theo dõi ở những nơi khó tiếp cận, điều kiện khắc nghiệt.

Nhìn chung, đây là một đề tài có tính ứng dụng cao, có tính khả thi, phù hợp với khả năng kinh tế của hầu hết mọi người và góp phần làm cho mọi thứ tiện lợi và thân thiện với con người.

7. TÀI LIỆU THAM KHẢO

- [1] “ASP.NET Tutorial”, [Online] <http://www.w3schools.com/aspnet/default.asp>.
- [2] Hồ Trung Mỹ, “Vi Xử Lý”, Nhà Xuất Bản Đại Học Quốc Gia TP. Hồ Chí Minh, 2006.
- [3] “ESP8266-Arduino”, [Online] <https://github.com/hocarm/ESP8266-Arduino>.
- [4] “Học Iot cơ bản”, [Online] <https://hocarm.org/hoc-iot-voi-esp8266>.
- [5] “Web server với Arduino và ESP8266”, [Online] <http://arduino.vn/bai-viet/1226-web-server-voi-arduino-va-esp8266>.
- [6] “Language Reference”, [Online] <https://www.arduino.cc/reference/en>.

8. PHỤ LỤC

8.1 Mã lệnh của trang “Default.cshtml”

@{

```
var request1 = "";
```

```
var request2 = "";
```

```
var request3 = "";
var request4 = "";
var db = Database.Open("iot");
var SQLUPDATE01 = "UPDATE Devices Set DRequest=@0 WHERE id=1";
var SQLUPDATE02 = "UPDATE Devices Set DRequest=@0 WHERE id=2";
var SQLUPDATE03 = "UPDATE Devices Set DRequest=@0 WHERE id=3";
var SQLUPDATE04 = "UPDATE Devices Set DRequest=@0 WHERE id=4";
var SQLDELETE1 = "DELETE FROM SLOG";
var SQLDELETE2 = "DELETE FROM DLOG";
var selectDevices = "SELECT * FROM Devices";
var dataDevices = db.Query(selectDevices);
if(IsPost)
{
    if(Request["Button"].Equals("Submit 1"))
    {
        request1 = Request["Choice1"];
        db.Execute(SQLUPDATE01, request1);
    }
    if(Request["Button"].Equals("Submit 2"))
    {
        request2 = Request["Choice2"];
        db.Execute(SQLUPDATE02, request2);
    }
    if(Request["Button"].Equals("Submit 3"))
    {
        request1 = Request["Choice3"];
        db.Execute(SQLUPDATE03, request3);
    }
    if(Request["Button"].Equals("Submit 4"))
    {
```

```
        request1 = Request["Choice4"];

        db.Execute(SQLUPDATE04, request4);
    }

    if(Request["Button"].Equals("Submit All"))
    {
        request1 = Request["Choice1"];
        request2 = Request["Choice2"];
        request3 = Request["Choice3"];
        request4 = Request["Choice4"];

        db.Execute(SQLUPDATE01, request1);
        db.Execute(SQLUPDATE02, request2);
        db.Execute(SQLUPDATE03, request3);
        db.Execute(SQLUPDATE04, request4);
    }
}

}

<!DOCTYPE html>

<html lang="en">

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="apple-touch-icon" sizes="57x57" href="/apple-icon-57x57.png">
<link rel="apple-touch-icon" sizes="60x60" href="/apple-icon-60x60.png">
<link rel="apple-touch-icon" sizes="72x72" href="/apple-icon-72x72.png">
<link rel="apple-touch-icon" sizes="76x76" href="/apple-icon-76x76.png">
<link rel="apple-touch-icon" sizes="114x114" href="/apple-icon-114x114.png">
<link rel="apple-touch-icon" sizes="120x120" href="/apple-icon-120x120.png">
<link rel="apple-touch-icon" sizes="144x144" href="/apple-icon-144x144.png">
<link rel="apple-touch-icon" sizes="152x152" href="/apple-icon-152x152.png">
<link rel="apple-touch-icon" sizes="180x180" href="/apple-icon-180x180.png">
```

```
<link rel="icon" type="image/png" sizes="192x192" href="/android-icon-192x192.png">

<link rel="icon" type="image/png" sizes="32x32" href="/favicon-32x32.png">
<link rel="icon" type="image/png" sizes="96x96" href="/favicon-96x96.png">
<link rel="icon" type="image/png" sizes="16x16" href="/favicon-16x16.png">
<link rel="manifest" href="/manifest.json">

    <meta name="msapplication-TileColor" content="#ffffff">
    <meta name="msapplication-TileImage" content="/ms-icon-144x144.png">
    <meta name="theme-color" content="#ffffff">

<title>Mô hình nhà thông minh điều khiển thiết bị qua Internet</title>

<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0
beta.2/css/bootstrap.min.css" integrity="sha384-
PshH8R72JQ3S0dhVi3uxftmaW6Vc51MKb0q5P2rRUPvrszuE4W1povHYgTpBfshb"
crossorigin="anonymous">

<script
src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js"></script>

<script>

    function Load_external_content() {

        $('#status').load('DisplayStatus.cshtml');
        $('#sensor').load('DisplaySensor.cshtml');
        $('#log_devices_sensor').load('DisplayLog.cshtml');
    }

    setInterval('Load_external_content()', 500);

</script>

<style>

    .m-b-15 {

        margin-bottom: 15px;

    }
```

```
</style>

</head>

<body>

<div class="container">

  <div class="jumbotron">

    <h1 class="display-6">Mô hình nhà thông minh điều khiển thiết bị
qua Internet</h1>

  </div>

  <div class="row">

    <div class="col-sm-6">

      <div class="card">

        <div class="card-header">

          <h2 id="h2st">Trạng thái thiết bị</h2>

        </div>

        <div class="card-body">

          <div id="status"></div>

        </div>

      </div>

    </div>

    <div class="col-sm-6">

      <div class="card">

        <div class="card-header">

          <h2 id="h2st">Các Cảm Biến</h2>

        </div>

        <div class="card-body">

          <div id="sensor"></div>

        </div>

      </div>

    </div>

  </div>

</div>
```

```
</div>

</br>

<div class="card">

    <div class="card-header">

        <h2 id="h2st">Bảng Điều Khiển</h2>

    </div>

    <div class="card-body">

        <div id="control">

            <form method="post">

                <div class="control_item m-b-15">

                    <strong>Living_room_lights </strong>

                    <select name="Choice1">

                        <option value="on">On</option>

                        <option value="off">Off</option>

                    </select>

                    <input id="button1" type="submit" value="Submit
1" name="Button" class="btn btn-success"/>

                </div>

                <div class="control_item m-b-15">

                    <strong>Kitchen_lights </strong>

                    <select name="Choice2">

                        <option value="on">On</option>

                        <option value="off">Off</option>

                    </select>

                    <input id="button2" type="submit" value="Submit
2" name="Button" class="btn btn-success"/>

                </div>

                <div class="control_item m-b-15">

                    <strong>Air_conditioner </strong>

                    <select name="Choice3">
```

```
        <option value="on">On</option>
        <option value="off">Off</option>
    </select>

    <input id="button3" type="submit" value="Submit
3" name="Button" class="btn btn-success"/>
</div>

<div class="control_item m-b-15">
    <strong>Stair_lights </strong>
    <select name="Choice4">
        <option value="on">On</option>
        <option value="off">Off</option>
    </select>

    <input id="button4" type="submit" value="Submit
4" name="Button" class="btn btn-success"/>
</div>

    <input id="button7" type="submit" value="Submit All"
name="Button" class="btn btn-primary"/>
</form>
</div>
</div>
</br>
<form method="post">
    <input id="buttondel1" type="submit" value="Clear Devices Log"
name="Button">
    <input id="buttondel2" type="submit" value="Clear Sensor Log"
name="Button">
</form>
<div id="log_devices_sensor"></div>

<div id="footer">
```

```
        <footer>
            footer
        </footer>
    </div>
</div>
</body>
</html>
```

8.2 Mã lệnh của trang “Sensor.cshtml”

@{

```
var db = Database.Open("iot");
var NhietDo = Request.QueryString["NhietDo"];
var DoAm = Request.QueryString["DoAm"];
var Gas = Request.QueryString["Gas"];
var ChuyenDong = Request.QueryString["ChuyenDong"];
var SQLUPDATE1 = "UPDATE Sensor Set Status=@0 WHERE id1=1";
var SQLINSERT1 = "INSERT INTO Slog (Name,Status,Time) VALUES (@0,@1,@2)";
var SQLUPDATE2 = "UPDATE Sensor Set Status=@0 WHERE id1=2";
var SQLINSERT2 = "INSERT INTO Slog (Name,Status,Time) VALUES (@0,@1,@2)";
var SQLUPDATE3 = "UPDATE Sensor Set Status=@0 WHERE id1=3";
var SQLINSERT3 = "INSERT INTO Slog (Name,Status,Time) VALUES (@0,@1,@2)";
var SQLUPDATE4 = "UPDATE Sensor Set Status=@0 WHERE id1=4";
var SQLINSERT4 = "INSERT INTO Slog (Name,Status,Time) VALUES (@0,@1,@2)";
var sqlQ1 = "SELECT * FROM Sensor WHERE id1=1";
var sqlQ2 = "SELECT * FROM Sensor WHERE id1=2";
var sqlQ3 = "SELECT * FROM Sensor WHERE id1=3";
var sqlQ4 = "SELECT * FROM Sensor WHERE id1=4";

var data1 = db.QuerySingle(sqlQ1);
```



```
var data2 = db.QuerySingle(sqlQ2);  
var data3 = db.QuerySingle(sqlQ3);  
var data4 = db.QuerySingle(sqlQ4);  
if(String.Compare(NhietDo,data1.Status) !=0)  
{  
    db.Excute(SQLINSERT1,"NhietDo",NhietDo,timelog);  
    db.Excute(SQLUPDATE1, NhietDo);  
}  
if(String.Compare(DoAm,data2.Status) !=0)  
{  
    db.Excute(SQLINSERT2,"DoAm",DoAm,timelog);  
    db.Excute(SQLUPDATE2, DoAm);  
}  
if(String.Compare(Gas,data3.Status) !=0)  
{  
    db.Excute(SQLINSERT3,"Gas",Gas,timelog);  
    db.Excute(SQLUPDATE3, Gas);  
}  
if(String.Compare(ChuyenDong,data4.Status) !=0)  
{  
    db.Excute(SQLINSERT4,"ChuyenDong",ChuyenDong,timelog);  
    db.Excute(SQLUPDATE4, ChuyenDong);  
}  
}
```

8.3 Mã lệnh của trang “SendRequest.cshtml”

```
@{
```

```
var db = Database.Open("iot");  
var sqlQ = "SELECT * FROM Devices";
```

```
var selectQueryString = "SELECT * FROM DLog ORDER BY id";

var data = db.Query(sqlQ);
}

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8" />

    <title>Status</title>

  </head>

  <body>

    <div id="statusupdate">

      <ul>

        @foreach(var row in data)
        {
          <li>

            @row.Name = @row.DRequest.

          </li>

        }

      </ul>

    </div>

  </body>

</html>
```

8.4 Mã lệnh của trang “GetStatus.cshtml”

```
@{

var db = Database.Open("iot");
```

```
var ustt1 = Request.QueryString["Living_room_lights"];
var ustt2 = Request.QueryString["Kitchen_lights"];
var ustt3 = Request.QueryString["Air_conditioner"];
var ustt4 = Request.QueryString["Stair_lights"];
var ustt5 = Request.QueryString["NhietDo"];
var ustt6 = Request.QueryString["DoAm"];
var ustt7 = Request.QueryString["Gas"];
var ustt8 = Request.QueryString["ChuyenDong"];
var SQLUPDATE1 = "UPDATE Devices Set DStatus=@0 WHERE id=1";
var SQLUPDATE2 = "UPDATE Devices Set DStatus=@0 WHERE id=2";
var SQLUPDATE3 = "UPDATE Devices Set DStatus=@0 WHERE id=3";
var SQLUPDATE4 = "UPDATE Devices Set DStatus=@0 WHERE id=4";
var SQLUPDATE5 = "UPDATE Sensor Set Status=@0 WHERE id1=1";
var SQLUPDATE6 = "UPDATE Sensor Set Status=@0 WHERE id1=2";
var SQLUPDATE7 = "UPDATE Sensor Set Status=@0 WHERE id1=3";
var SQLUPDATE8 = "UPDATE Sensor Set Status=@0 WHERE id1=4";
var sqlQ1 = "SELECT * FROM Devices WHERE id=1";
var sqlQ2 = "SELECT * FROM Devices WHERE id=2";
var sqlQ3 = "SELECT * FROM Devices WHERE id=3";
var sqlQ4 = "SELECT * FROM Devices WHERE id=4";
var sqlQ5 = "SELECT * FROM Sensor WHERE id1=1";
var sqlQ6 = "SELECT * FROM Sensor WHERE id1=2";
var sqlQ7 = "SELECT * FROM Sensor WHERE id1=3";
var sqlQ8 = "SELECT * FROM Sensor WHERE id1=4";
var SQLINSERTD = "INSERT INTO DLog (Name,Status,ATime) VALUES (@0,@1,@2)";
var SQLINSERTS = "INSERT INTO DLog (Name,Status,Time) VALUES (@0,@1,@2)";
```

```
string vnTimeZoneKey = "SE ASIA Standard Time";

TimeZoneInfo vnTimeZone =
TimeZoneInfo.FindSystemTimeZoneById(vnTimeZoneKey);

DateTime timelog =
TimeZoneInfo.ConvertTimeFromUtc(dataTime.UtcNow,vnTimeZone);

var data1 = db.QuerySingle(sqlQ1);
var data2 = db.QuerySingle(sqlQ2);
var data3 = db.QuerySingle(sqlQ3);
var data4 = db.QuerySingle(sqlQ4);
var data5 = db.QuerySingle(sqlQ5);
var data6 = db.QuerySingle(sqlQ6);
var data7 = db.QuerySingle(sqlQ7);
var data8 = db.QuerySingle(sqlQ8);

if(String.Compare(ustt1,data1.DStatus) != 0)
{
    db.Execute(SQLINSERTD,"Living_room_lights",ustt1,timelog);
    db.Execute(SQLUODATE1, ustt1);
}

if(String.Compare(ustt2,data2.DStatus) != 0)
{
    db.Execute(SQLINSERTD,"Kitchen_lights",ustt2,timelog);
    db.Execute(SQLUODATE2, ustt2);
}

if(String.Compare(ustt3,data3.DStatus) != 0)
{
    db.Execute(SQLINSERTD,"Air_conditioner",ustt3,timelog);
```

```
        db.Execute(SQLUODATE3, ustt3);
    }
    if(String.Compare(ustt4,data4.DStatus) != 0)
    {
        db.Execute(SQLINSERTD,"Stair_lights",ustt4,timelog);
        db.Execute(SQLUODATE4, ustt4);
    }
    if(String.Compare(ustt5,data5.Status) != 0)
    {
        db.Execute(SQLINSERTS,"NhietDo",ustt5,timelog);
        db.Execute(SQLUODATE5, ustt5);
    }
    if(String.Compare(ustt6,data6.Status) != 0)
    {
        db.Execute(SQLINSERTS,"DoAm",ustt6,timelog);
        db.Execute(SQLUODATE6, ustt6);
    }
    if(String.Compare(ustt7,data7.Status) != 0)
    {
        db.Execute(SQLINSERTS,"Gas",ustt7,timelog);
        db.Execute(SQLUODATE7, ustt7);
    }
    if(String.Compare(ustt8,data8.Status) != 0)
    {
        db.Execute(SQLINSERTS,"ChuyenDong",ustt8,timelog);
        db.Execute(SQLUODATE8, ustt8);
    }
```

```
}  
}
```

8.5 Mã lệnh của trang “DisplayStatus.cshtml”

```
@{  
    var db = Database.Open("iot");  
    var sqlQ = "SELECT * FROM Devices";  
    var data = db.Query(sqlQ);  
}  
  
<!DOCTYPE html>  
  
<html lang="en">  
    <head>  
        <meta charset="utf-8" />  
        <title></title>  
    </head>  
    <body>  
        <div id="statusupdate">  
            <ul>  
                @foreach (var row in data) {  
                    <li>@row.Name = @row.DStatus.</li>  
                }  
            </ul>  
        </div>
```

```
</body>  
</html>
```

8.6 Mã lệnh của trang “DisplaySensor.cshtml”

```
@{  
  
    var db = Database.Open("iot");  
  
    var sqlQ1 = "SELECT * FROM Sensor WHERE id1=1";  
  
    var sqlQ2 = "SELECT * FROM Sensor WHERE id1=2";  
  
    var sqlQ3 = "SELECT * FROM Sensor WHERE id1=3";  
  
    var sqlQ4 = "SELECT * FROM Sensor WHERE id1=4";  
  
    var selectQueryStrig = "SELECT * FROM SLOG ORDER BY id3";  
  
    var data1 = db.QuerySingle(sqlQ1);  
  
    var sstt1 = data1.Status;  
  
    var Name1 = data1.Name;  
  
    var data2 = db.QuerySingle(sqlQ2);  
  
    var sstt2 = data2.Status;  
  
    var Name2 = data2.Name;  
  
    var data3 = db.QuerySingle(sqlQ3);  
  
    var sstt3 = data3.Status;  
  
    var Name3 = data3.Name;  
  
    var data4 = db.QuerySingle(sqlQ4);  
  
    var sstt4 = data4.Status;  
  
    var Name4 = data4.Name;  
  
    var Display1 = "";  
  
    if(sstt3 == "0")  
    {
```

```
        Display1 = "Bình thường.";
    }

    if(sstt3 != "0")
    {
        Display1 = "Rò khí Gas.";
    }

    var Display2 = "";

    if(sstt4 == "0")
    {
        Display2 = "Không có người.";
    }

    if(sstt4 != "0")
    {
        Display2 = "Có đột nhập.";
    }
}

<!DOCTYPE html>

<html lang="en">

    <head>

        <meta charset="utf-8" />

    </head>

    <body>

        <div id="log">

            <div id="log1" class="list-group list-group-flush">

                <p class="list-group-item"><strong>Nhiệt độ:
```



```
</strong><label>@sst1</label></p>

<p class="list-group-item"><strong>Độ ẩm:

</strong><label>@sst2</label></p>

<p class="list-group-item"><strong>Khí gas:

</strong><label>@Display1</label></p>

<p class="list-group-item"><strong>Chuyển động:

</strong><label>@Display2</label></p>

    <br></br>

</div>

</div>

</body>

</html>
```