**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**ĐỀ TÀI: NGHIÊN CỨU VỀ IoT VÀ ỨNG DỤNG ĐỂ GIÁM SÁT VÀ ĐIỀU KHIỂN NHÀ THÔNG MINH**

Giảng viên hướng dẫn: ThS. TRẦN PHONG NHÃ

Sinh viên thực hiện: ĐỖ HỮU XUYÊN

Lớp : CQ.55.CNTT

Khoá :55

Tp. Hồ Chí Minh, Tháng 01, Năm 2021

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**ĐỀ TÀI: NGHIÊN CỨU VỀ IoT VÀ ỨNG DỤNG ĐỂ GIÁM SÁT VÀ ĐIỀU KHIỂN NHÀ THÔNG MINH**

Giảng viên hướng dẫn: ThS. TRẦN PHONG NHÃ

Sinh viên thực hiện: ĐỖ HỮU XUYÊN

Lớp : CQ.55.CNTT

Khoá :55

Tp. Hồ Chí Minh, Tháng 01, Năm 2021

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH Độc lập – Tự do – Hạnh phúc**

# NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP

BỘ MÔN: **CÔNG NGHỆ THÔNG TIN**

-------\*\*\*-------

**Mã Sinh Viên: 5551074048 Họ Tên SV: ĐỖ HỮU XUYÊN**

**Khoá: 55 Lớp: CQ.55.CNTT**

1. **Tên đề tài**

Nghiên cứu về IoT và ứng dụng để giám sát và điều khiển nhà thông minh.

1. **Mục đích, yêu cầu**

* Mục đích:

Xây dựng mô hình hệ thống điều khiển các thiết bị trong nhà qua internet.

* Yêu cầu:

Sử dụng các thiết bị và giải pháp IoT kết hợp với Mạng Máy Tính, Database, Web-Interface, Restful-Api để kết hợp thực hiện.

* Thiết bị phần cứng:
* Raspberry Pi 3B+, Arduino Wemos D1 R2, DHT11, Pir HC-SR501, Magnetic Door Switch Sensor, MQ-135.
* Kết nối các thiết bị với Mạng Máy Tính, giao thức MQTT.
* Database:
* Xây dựng thiết bị Raspberry 3 B+ trở thành một Gateway để truyền phát internet và cài đặt các phần mềm liên quan để nó trở thành một Database di động.
* Đo đạc thông tin của môi trường xung quanh sau đó đẩy dữ liệu trực tiếp về Database.
* Thiết kế web interface:

Thiết kế web application với giao diện dễ sử dụng, thân thiện với người dùng hiển thị các thông tin thu thập được từ cảm biến lên bề mặt web application.

* Chức năng chính:
* Điều khiển các thiết bị trong nhà qua mọi thiết bị thông minh.
* Hiển thị nhiệt độ, độ ẩm.
* Hệ thống chống trộm.
* Hệ thống cảnh báo khí ga.
* Hệ thống bật tắt đèn.

1. **Nội dung và phạm vi đề tài**

* Nội dung đề tài:
* Tổng quang về phần cứng trên các thiết bị như: Raspberry, Wemos D1 R2, DHT11, MQ-135, LED, Pir HC-SR501, Magnetic Door Switch Sensor.
* Tổng quan về lập trình nhúng cơ bản lên trên các thiết bị Arduino, Raspberry 3 B+.
* Phân tích tìm hiểu về giao thức MQTT, truyền nhận dữ liệu và điều khiển cơ bản liên quan đến giao thức.
* Tổng quan về phần mềm hỗ trợ: Database (SQLite), Node-Red, Python Flask.
* Phạm vi đề tài:
* Tìm hiểu Giao thức MQTT và ứng của nó trong hệ thống IoT hiện nay.
* Xây dựng hệ thống IOT điều khiển và giám sát ngôi nhà.

1. **Công nghệ, công cụ và ngôn ngữ lập trình**

* Công cụ: Raspberry Pi 3 B+, Arduino Wemos D1 R2, DHT11, MQ-135, LED, Pir HC-SR501, Magnetic Door Switch Sensor.
* Ngôn ngữ lập trình: Lập trình Arduino, Python, Node-Red.
* Database: SQLite.

1. **Các kết quả chính dự kiến sẽ đạt được và ứng dụng**

* Hiểu được cấu trúc, cách thức hoạt động và phát triển của Internet of Thing.
* Biết lập trình nhúng trên các thiết bị phần cứng, hiểu sơ bộ về SQLite.
* Bổ sung thêm kiến thức về lập trình.
* Xây dựng một Web application để hiện thị các thông số, số liệu đo được từ các cảm biến lên Web applicatioin và có thể điều khiển ngược lại.
* Quyển báo cáo đề tài tốt nghiệp.

1. **Giáo viên và cán bộ hướng dẫn**

Họ tên: ThS. Trần Phong Nhã

Đơn vị công tác: Trường Đại học Giao Thông Vận Tải-Phân hiệu tài TP.HCM

Điện thoại: Email:

|  |  |
| --- | --- |
| **Ngày tháng 01 năm 2021**  **Trưởng BM Công nghệ Thông tin** | **Đã giao nhiệm vụ TKTN**  **Giáo viên hướng dẫn** |

Đã nhận nhiệm vụ TKTN

Sinh viên: Đỗ Hữu Xuyên Ký tên:

Điện thoại: 0703393996 Email: dohuuxuyen96@gmail.com

# LỜI CẢM ƠN

Em xin chân thành cảm ơn khoa Công Nghệ Thông Tin, trường đại học Giao Thông Vận Tải-Phân hiệu tại TP.HCM đã tạo điều kiện tốt cho em thực hiện tốt đề tài này. Đặc biệt, em xin chân thành cám ơn thầy Trần Phong Nhã là người trực tiếp hướng dẫn, chỉ bảo, dẫn lối cho em trong suốt quá trình em thực hiện đề tài.

Em cũng xin gửi lời cảm ơn sâu sắc đến quý thầy cô trong khoa đã tận tình giảng dạy, trang bị cho em những kiến thức trong những năm vừa qua.

Mặc dù em đã cố gắng hoàn thành đồ án trong phạm vi và khả năng cho phép, nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong sự cảm thông và sự chỉ bảo tận tình của quý thầy cô.

Cuối cùng, em xin kính chúc quý thầy cô mạnh khỏe, hạnh phúc và thành công hơn nữa trong công việc cũng như trong cuộc sống.

Em xin chân thành cảm ơn!

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

|  |
| --- |
| ***Tp. Hồ Chí Minh, ngày ….… tháng ….… năm ….…***  **Giáo viên hướng dẫn**  **ThS. Trần Phong Nhã** |

**Mục lục**

[NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP i](#_Toc62549848)

[LỜI CẢM ƠN iv](#_Toc62549849)

[DANH MỤC CHỮ VIẾT TẮT ix](#_Toc62549850)

[DANH MỤC HÌNH ẢNH xi](#_Toc62549851)

[DANH MỤC BẢNG BIỂU xii](#_Toc62549852)

[LỜI MỞ ĐẦU xiii](#_Toc62549853)

[1 Lý do chọn đề tài 1](#_Toc62549854)

[2 Mục tiêu nghiên cứu 2](#_Toc62549855)

[3 Phạm vi đề tài 2](#_Toc62549856)

[4 Yêu cầu chức năng và phi chức năng 2](#_Toc62549857)

[5. Cấu trúc báo cáo đồ án tốt nghiệp 2](#_Toc62549858)

[CHƯƠNG 1. CƠ SỞ LÝ THUYẾT 4](#_Toc62549859)

[1.1 Giới thiệu về Internet of Thing (IoT) 4](#_Toc62549860)

[1.1.1 Định nghĩa: 4](#_Toc62549861)

[1.1.2 Ứng dụng của IoT 6](#_Toc62549862)

[1.1.3 Ưu điểm, khuyết điểm 7](#_Toc62549863)

[1.2 Giới thiệu về Raspberry Pi 8](#_Toc62549864)

[1.2.1 Định nghĩa 8](#_Toc62549865)

[1.2.2 Sử dụng Raspberry Pi 9](#_Toc62549866)

[1.2.3 Lý do sử dụng Raspberry Pi trong đồ án 10](#_Toc62549867)

[1.3 Tổng quan về một số cảm biến môi trường liên quan 10](#_Toc62549868)

[1.3.1 Wemos D1 R2 10](#_Toc62549869)

[1.3.2 DHT11 12](#_Toc62549870)

[1.3.3 MQ-135 13](#_Toc62549871)

[1.3.4 Magnetic Door Switch Sensor 14](#_Toc62549872)

[1.3.5 Pir HC-SR501 14](#_Toc62549873)

[1.3.6 Arduino IDE 15](#_Toc62549874)

[1.4 Giới thiệu về giao thức MQTT 16](#_Toc62549875)

[1.4.1 Khái quát chung 16](#_Toc62549876)

[1.4.2 Các khái niệm đáng chú ý trong giao thức MQTT 18](#_Toc62549877)

[1.5 Giới thiệu về Flask Python 20](#_Toc62549878)

[1.5.1 Python 20](#_Toc62549879)

[1.5.2 Flask – Python 22](#_Toc62549880)

[1.6 Giới thiệu về RESTful – API 22](#_Toc62549881)

[1.6.1 Resource 22](#_Toc62549882)

[1.6.2 Restful Api 23](#_Toc62549883)

[1.7 Tổng quan về SQLite 24](#_Toc62549884)

[1.7.1 Định nghĩa SQLite: 24](#_Toc62549885)

[1.7.2 Ưu điểm: 24](#_Toc62549886)

[1.7.3 Lịch sử: 25](#_Toc62549887)

[1.7.4 Hạn chế: 25](#_Toc62549888)

[1.7.5 Lệnh trong SQLite 26](#_Toc62549889)

[1.8 Giới thiệu về Node-Red 26](#_Toc62549890)

[CHƯƠNG 2. TRIỂN KHAI ỨNG DỤNG NHÀ THÔNG MINH VỚI MQTT 28](#_Toc62549891)

[2.1 Giới thiệu tổng quan bài toán 28](#_Toc62549893)

[2.2 Thiết lập Arduino IDE 30](#_Toc62549894)

[2.2.1 Thiết lập cơ bản trên Arduino IDE 30](#_Toc62549895)

[2.2.2 Nhúng thiết bị với giao thức MQTT 33](#_Toc62549896)

[2.3 Thiết lập Raspberry Pi 3 B+ 39](#_Toc62549900)

[2.3.1 Thiết lập Raspbperry Pi thành Gateway 40](#_Toc62549901)

[2.3.2 Sử dụng MQTT trong Raspberry 43](#_Toc62549902)

[CHƯƠNG 3: THIẾT KẾ, CHẾ TẠO MÔ HÌNH THỰC NGHIỆM 44](#_Toc62549903)

[3.1 Giao diện giám sát và điều khiển 44](#_Toc62549905)

[3.1.1 Phần điều Khiển và hiển thị: 44](#_Toc62549906)

[3.1.2 Phần hiển thị nhiệt độ, độ ẩm trong SQLite Database: 48](#_Toc62549907)

[KẾT QUẢ VÀ KIẾN NGHỊ 51](#_Toc62549908)

[TÀI LIỆU THAM KHẢO 52](#_Toc62549908)

# DANH MỤC CHỮ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Mô tả | Thuật ngữ | Ghi chú |
|  | MQTT | Message Queuing Telemetry Transport | Giao thức truyền thông điệp |
|  | IoT | Internet of Thing | Internet vạn vật |
|  | Api | Application Programming Interface | Giao diện lập trình ứng dựng |
|  | LED | Light-emitting Diode | Điốt phát quang |
|  | Wemos D1 R2 |  | Board mạch phát triển trên Module ESP8266 |
|  | DHT11 |  | Cảm biến nhiệt độ, độ ẩm |
|  | MQ-135 |  | Cảm biến khí ga |
|  | MISO | Master Input Slave Output | Chân nhận dữ liệu khi cấu hình là Slave |
|  | MOSI | Master Ouput Slave Input | Chân truyền dữ liệu khi cấu hình là Slave |
|  | VCC |  | Nguồn điện vô |
|  | GND |  | Nguồn điện ra |
|  | MCU | Multipoint Control Unit | Thiết bị điều khiển đa điểm |
|  | Gateway |  | Truyền thông |
|  | CSDL |  | Cớ sở dữ liệu |
|  | IPv6 | Internet protocol version 6 | Giao thức liên mạng thế hệ 6 |
|  | USB | Universal Serial Bus | Chuẩn kết nối tuần tự đa dụng trong máy tính |
|  | ATM | Automated Teller Machine | Máy rút tiền tự động |
|  | CPU | Central Processing Unit | Đơn vị xử lý trung tâm |
|  | GPU | Graphics Processing Unit | Đơn vị xử lý đồ hoạ |
|  | RAM | Random Access Memory | Bộ nhớ truy xuất ngẫu nhiên |
|  | Wi-Fi | Wireless Fidelity | Internet không dây |
|  | VPN | Virtual Private Network | Mạng riêng ảo |
|  | NAS | Network Attached Storage | Thiết bị lưu trữ gắn vào mạng |
|  | I/O | Input/output | Đầu vào / Đầu ra |
|  | Arduino IDE | Arduino Integrated Development Environment | Trình soạn thảo văn bản |
|  | QoS | Quality of Service | Chất lượng dịch vụ |
|  | Pi | Resbpberry Pi | Vi máy tính |
|  | IE | Internet Explorer | Trình duyệt web |
|  | LPG | Liquefied Petroleum Gas | Khí gas |
|  | RDBMS | Relational Database Management System | Hệ quản trị cơ sở dữ liệu quan hệ |
|  | SSH | Secure Shell | Môi trường an toàn |
|  | DHCP | Dynamic Host Configuration Protocol | Giao thức cấu hình động máy chủ |
|  | REST | Representational State Transfer | Dạng chuyển đổi cấu trúc dữ liệu |

# DANH MỤC HÌNH ẢNH

|  |  |  |
| --- | --- | --- |
| STT | Tên hình ảnh | Trang |
|  | Hình 1.1: Internet of Thing | 5 |
|  | Hình 1.2: Ứng dụng điều khiển đèn LED | 6 |
|  | Hình 1.3: Ngôi nhà thông minh | 6 |
|  | Hình 1.4: Thành phố thông minh | 7 |
|  | Hình 1.5: Raspberry Pi 3 B+ | 9 |
|  | Hình 1.6: Bo mạch Wemos D1 R2 | 11 |
|  | Hình 1.7: Cảm biến nhiệt độ, độ ẩm - DHT11 | 12 |
|  | Hình 1.8: Cảm biến khí ga – MQ-135 | 13 |
|  | Hình 1.9: Cảm biến từ cửa – Magnetic Door Switch Sensor | 14 |
|  | Hình 1.10: Cảm biến thân nhiệt chuyển động Pir HC-SR501 | 15 |
|  | Hình 1.11: Hướng dẫn nạp Sketch | 16 |
|  | Hình 1.12: Kiến trúc mức cao của MQTT | 17 |
|  | Hình 1.13: SQLite Database | 24 |
|  | Hình 1.14: Node-Red dạng liên kết cơ bản | 27 |
|  | Hình 1.15: Node-Red đóng vài trò là server và client | 27 |
|  | Hình 1.16: Node-Red có vai trò là server, còn lại là client | 27 |
|  | Hình 2.1: Triển khai tổng quan mô hình theo thực tế | 29 |
|  | Hình 2.2: Triển khai mô hình MQTT trong thực tế. | 29 |
|  | Hình 2.3: Mô hình hoạt động của Paho-Mqtt | 30 |
|  | Hình 2.4: Preferences của Arduino IDE | 31 |
|  | Hình 2.5: Cài đặt thư viện Arduino | 32 |
|  | Hình 2.6: Chình thông số Wenmos D1 R2 trên Arduino IDE | 32 |
|  | Hình 2.7: Kết nối và gửi dữ liệu lên MQTT | 39 |
|  | Hình 2.8: Interfacing Options trong Rasberry Pi | 40 |
|  | Hình 2.9: SSH trong Rasberry Pi | 40 |
|  | Hình 2.10: Sơ đồ thiết lập raspberry wifi | 41 |
|  | Hình 2.11: Kết quả Raspberry Pi thành gateway | 43 |
|  | Hình 3.1: Khỏi động Node-Red | 44 |
|  | Hình 3.2: Manage palette | 45 |
|  | Hình 3.3: Tìm kiếm node-red-dashboard để cài đặt | 45 |
|  | Hình 3.4: Cài đặt Dashboard thành công | 46 |
|  | Hình 3.5: Thiết lập Đèn với MQTT | 46 |
|  | Hình 3.6: Thiết lập nhiệt độ, độ ẩm | 47 |
|  | Hình 3.7: Giao diện điều khiển và hiển thị nhiệt độ, độ ẩm | 47 |
|  | Hình 3.8: Trang hiển thị nhiệt độ, độ ẩm trong SQLite Database | 48 |
|  | Hình 3.9: Kết quả khởi chạy Restful Api | 50 |

## **DANH MỤC BẢNG BIỂU**

|  |  |  |
| --- | --- | --- |
| STT | Tên bảng biểu | Trang |
|  | Bảng 1.1: Thông số chân của Wenmos D1 R2 | 12 |
|  | Bảng 1.2: Thông số chân của cảm biến DHT11 | 13 |
|  | Bảng 1.3: Thông số chân của cảm biến MQ-135 | 13 |
|  | Bảng 1.4: Thông số chân của cảm biến Magnetic Door Switch Sensor | 14 |
|  | Bảng 1.5: Thông số chân của cảm biến Pir HC-SR501 | 15 |

## **LỜI MỞ ĐẦU**

Trong thời đại công nghệ thông tin như hiện nay, đặc biệt là công cuộc cách mạng công nghệ 4.0 thì mạng máy tính cũng tiếp bước phát triển theo. Ngoài việc kết nối tất cả mọi người lại với nhau, giúp nhau giao tiếp truyền đạt những thông tin, chia sẻ hình ảnh, buôn bán v.v…thì bây giờ internet còn thể giúp ta làm được nhiều việc hơn như điều khiển đồ vật, xây dựng thiết bị thông minh, smart home….

Internet of Thing hay được gọi là internet vạn vật, mạng lưới vạn vật kết nối internet là một liên mạng, trong đó các thiết bị, phương tiện vận tải, phòng ốc và các thiết bị khác được nhúng với các thiết bị điện tử, phần mềm, cảm biến, cơ cấu chấp hành cùng với khả năng kết nối mạng máy tính giúp cho các thiết bị thu thập và truyền tải dữ liệu.

Từ những xu hướng phát triển và những nhu cầu trên, đồ án này sẽ khảo sát và nghiên cứu về Internet of Thing đặc biệt trong đó là giao thức MQTT giúp ta lấy dữ liệu từ cảm biến, sau đó đẩy dữ liệu vào database và thể hiện thông tin lên website và từ đó ta đưa ra những phương án nghiên cứu thích hợp.

Các chương đầu trong tài liệu sẽ trình bày lần lượt các khái niệm chung về IoT, sau sẽ trình bày nội dung, cách thức xây dựng và hoạt động của giao thức MQTT. Xây dựng RestFul-API. Cuối cùng là chương tổng kết về các khó khăn và những thành quả trong suốt quá trình thực hiện phát triển về kiến thức và áp dụng vào thực tiễn của IoT cũng như giao thức MQTT.

**PHẦN TỔNG QUAN**

## **1 Lý do chọn đề tài**

Internet of Thing (IoT) – Internet vạn vật hay còn được gọi là mạng lưới vạn vật dường như đang trong giai đoạn phát triển bùng nổ là xu hướng mới đang được các doanh nghiệp trrong lĩnh vực công nghệ quan tâm và đầu tư nghiên cứu. Thực chất Internet of Thing đã xuất hiện ngày từ thời kỳ sơ khai của internet, khi các nhà phát minh mong muốn kết nối tất cả mọi thứ qua một lưới đồng nhất để có thể điều khiển chúng phục vụ mục đích của con người.

Cùng với sự bùng nổ của IPv6 và mạng internet 4G, cũng như ngày càng có nhiều tablet và mobile thì sự phát triển của IoT là điều tất yếu. Theo dự đoán của trang Gartner (Công ty nghiên cứu và tư vấn công nghệ thông tin hàng đầu thế giới), ngày nay có hơn 3,1 tỉ vật được kết nối internet và năm 2020 sẽ đặt đến ngưỡng 7,6 tỉ. Chẳng phải tự nhiên mà các doanh nghiệp hay tập đoàn, hay người người đổ xô vào sử dụng IoT. Vì chúng sẽ giúp chúng ta tăng năng suất và giảm chi phí sản xuất. Hay như thay vì mỗi buổi tối con người phải tắt mở từng cái đèn từng vật dụng một cách thủ công thì nay nhờ IoT người sử dụng có thể điều khiển chúng từ xa theo một quy trình đã định sẵn v.v…

MQTT (Message Queuing Telemetry Transport) là một giao thức truyền thông điệp theo mô hình publish/subscribe (xuất bản – theo dõi) sử dụng ở băng thông thấp, có độ tin cậy cao và có khả năng hoạt động trong điều kiện không ổn đinh. Được sử dụng phổ biến, đặc biệt là trong lĩnh vực IoT này.

Cũng là một người đam mê nghiên cứu công nghệ, cũng như theo đuổi con đường lập trình viên chuyên nghiệp, để bắt kịp với xu hướng thể giới. Tác giả thấy IoT như một tiềm năng phát triển tương lai, giúp thay đổi cuộc sống con người. Ngoài ra nó còn giúp nâng cao kiến thức cũng như ôn lại toàn bộ những kiến thức cũ trong những năm học tại trường. Áp dụng chúng vào thực thế, phát triển và củng cố bản thân. Đây là những lý do chính để tác giả thử sức mình ở lĩnh vực mới với mốc thời gian cố định là ba tháng trong khóa đồ án tốt nghiệp này.

**2 Mục tiêu nghiên cứu**

* Tổng quan về lập trình nhúng trên các thiết bị phần cứng IoT như Raspberry, Arduino Wemos D1 R1, DHT11, MQ-135, LED, Pir HC-SR501, Magnetic Door Switch Sensor.
* Nghiên cứu và tìm hiểu về giao thức MQTT trong IoT. Đưa ra các ứng dụng cơ bản và xây dựng một ứng dụng thực tế của MQTT. Sử dụng thư viện hỗ trợ để đưa dữ liệu trực tiếp vào SQLite.
* Thiết kế một RestFul Api với các giao thức hỗ trợ như Get, Post…Đưa dữ liệu lên website và hiển thị dữ liệu lên nó.
* Xây dựng một Web application bằng Node-Red sau đó lấy dữ liệu từ database lên và hiện thị nó.

## **Phạm vi đề tài**

* Tìm hiểu Giao thức MQTT và ứng của nó trong hệ thống IoT hiện nay.
* Nghiên cứu IoT giám sát và điều khiển nhà thông minh.

## **4 Yêu cầu chức năng và phi chức năng**

1. **Yêu cầu chức năng:**

* Hệ thống sử dụng Kit Arduino và cảm biến nhiệt độ, độ ẩm cho phép thu thập các thông số tương ứng với môi trường xung quanh. Dữ liệu thu được tự động lưu trữ vào cơ sở dữ liệu.
* Chức năng điều khiển thiết bị từ xa, thực hiện với việc bật tắt điều khiển LED

qua web application.

1. **Yêu cầu phi chức năng:**

* Hệ thống hoạt động ổn định với các cảm biến cố môi trường khác nhau.
* Có khả năng tương thích với nhiều trình duyệt khác nhau như IE, Chrome,Firefox…

- Hiểu và nắm bắt được những công nghệ cơ bản khi triển khai ứng dụng IoT.

**5. Cấu trúc báo cáo đồ án tốt nghiệp**

* **Phần tổng quan**
* Lý do chọn đề tài.
* Mục tiêu nghiên cứu.
* Phạm vi đề tài.
* Các yêu cầu chức năng và phi chức năng.
* Cấu trúc báo cáo đồ án tốt nghiệp.
* **Chương 1: Cơ sở lý thuyết**
* Giới thiệu về Internet of Thing.
* Giới thiệu về Raspberry Pi 3 B+.
* Tổng quan về một số cảm biến môi trường liên quan.
* Giới thiệu về giao thức MQTT.
* Giới thiệu về Python Flask.
* Giới thiệu về RESTful-API.
* Tổng quan về SQLite.
* Giới thiệu về Node-Red.
* **Chương 2: Triển khai ứng dụng nhà thông minh với MQTT**
* Giới thiệu tổng quan bài toán.
* Thiết lập Arduino IDE.
* Thiết lập Raspberry Pi 3 B+.
* **Chương 3: Thiết kế, chế tạo mô hình thực nghiệm**
* Giao diện giám sát và điều khiển.
* **Kết quả và kiến nghị**
* Kết quả đạt đượcThiết lập Arduino IDE.
* Hạn chế.
* Hướng phát triển.
* **Tài liệu tham khảo**

# CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

## **Giới thiệu về Internet of Thing (IoT)**

* + 1. **Định nghĩa:**

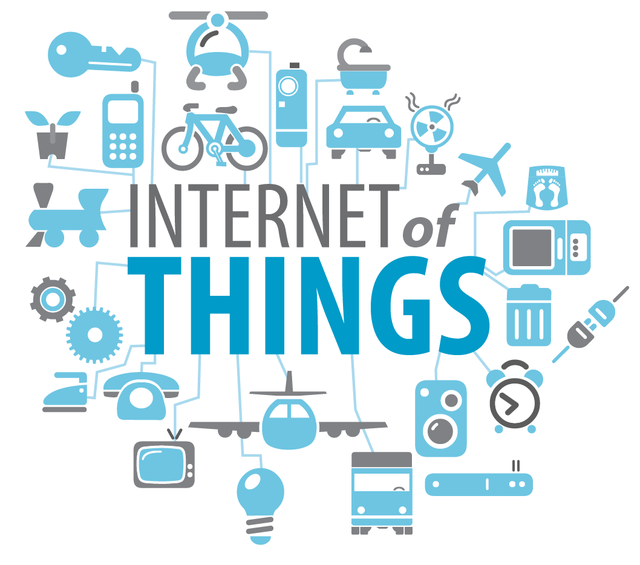
Biểu hiện của Internet of Things (IoT) đã xuất hiện ngay từ thời kỳ sơ khai của Internet, khi các nhà phát minh mong muốn kết nối tất cả mọi thứ qua một mạng lưới đồng nhất để có thể điều khiển chúng phục vụ cho mục đích của con người.

Trong các tư liệu về IoT, người ta thường nhắc đến một chiếc máy bán nước giải khát tự động tại trường Đại học Carnegie Melon (Mỹ) vào đầu những năm 1980 như là một thiết bị đầu tiên mở màn cho xu hướng này, chiếc máy được lập trình để có thể kết nối với người điều khiển qua Internet, nhằm kiểm tra tình trạng của máy và bổ sung nước khi cần thiết mà không cần sự tiếp xúc kiểm tra trực tiếp.

IoT là thuật ngữ dùng để chỉ các đối tượng có thể được nhận biết (identifiable) cũng như chỉ sự tồn tại của chúng trong một kiến trúc mang tính kết nối. Cụm từ này được đưa ra bởi Kevin Ashton vào năm 1999. Ông là một nhà khoa học đã sáng lập ra Trung tâm Auto-ID ở đại học MIT, nơi thiết lập các quy chuẩn toàn cầu cho RFID (một phương thức giao tiếp không dây dùng sóng radio) cũng như một số loại cảm biến khác. IoT sau đó cũng được dùng nhiều trong các ấn phẩm đến từ các hãng và nhà phân tích.

Theo định nghĩa từ Wikipedia: Internet of Thing (IoT) là một kịch bản của thế giới, khi mà mỗi đồ vật được con người cung cấp mốt định danh của riêng mình, và tất cả có khả năng truyền tài, trao đổi thông tin, dữ liệu quan một mạng duy nhất mà không cần đến sự tương tác trực tiếp giữa người với người, hay người với máy tính. IoT đã phát triển từ sự hội tụ của công nghệ không dây, công nghệ vi cơ điện tử và internet.

Như vậy có thể tạm hiểu, Internet of Thing là khi tất cả mọi thứ đều được kết nối với nhau qua mạng internet, người dùng có thể kiểm soát mọi đồ vật của mình qua mạng mà chỉ bằng một thiết bị thông minh, chẳng hạn như smartphone, tablet, pc hay thậm chí chỉ bằng một chiếc smartwatch nhỏ bé trên tay.



Hình 1.1: Internet of Thing.

Ví dụ đơn giản như sau: chiếc tủ lạnh thông thường của người dùng không được kết nối với thiết bị nào khác. Nếu chúng ta muốn ghi lại nhiệt độ ở từng thời điểm của tủ, ta chỉ có cách ghi lại thủ công rồi nhập vào một máy tính hay thiết bị lưu trữ nào đó. Hay như bóng đèn neon ở nhà chẳng hạn, chúng ta muốn thu thập, điều chỉnh độ sáng của nó thì phải đo thủ công rồi ghi lại.

Còn nếu như máy tính có khả năng giúp con người thu thập tất cả những dữ liệu về mọi thứ xung quanh, chúng ta có thể "theo dõi và đếm mọi thứ, giúp giảm hao phí, chi phí và lỗ. Chúng ta sẽ biết chính xác khi nào các vật dụng cần phải sửa chữa, thay thế, khi nào chúng còn mới và khi nào thì chúng hết hạn sử dụng. Chưa kể đến việc chúng ta có thể kiểm soát chúng mọi lúc mọi nơi. IoT có tiềm năng thay đổi thế giới, giống như cách mà Internet đã thay đổi cuộc sống của chúng ta. Ngôi nhà thông minh với các bóng đèn thông minh, máy giặt thông minh, tủ lạnh thông minh… có thể xem là bước đầu của IoT bởi chúng đều được liên kết với nhau và/hoặc liên kết vào Internet.

### **Ứng dụng của IoT**



Hình 1.2: Ứng dụng điều khiển đèn LED

IoT có ứng dụng rộng vô cùng, có thể kể ra một số thứ như sau:

* Quản lý chất thải
* Tự động hóa ngôi nhà



Hình 1.3: Ngôi nhà thông minh.

* Quản lý và lập kế hoạch quản lý đô thị



Hình 1.4: Thành phố thông minh.

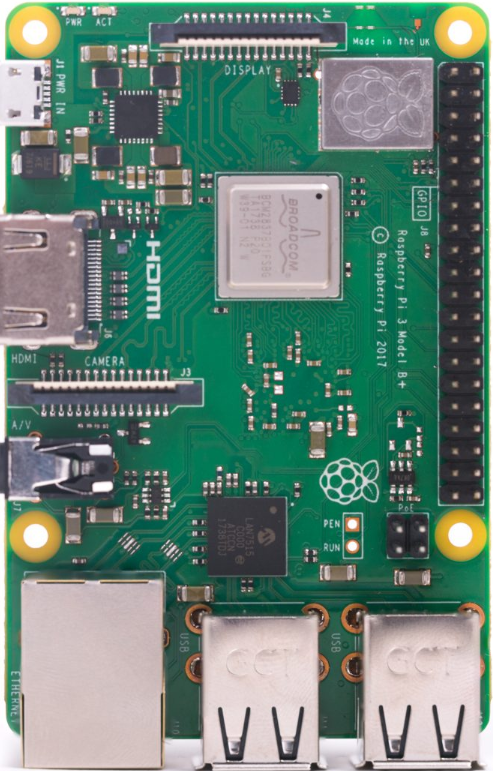
* Quản lý môi trường
* Phản hồi trong các tình huống khẩn cấp
* Mua sắm thông minh
* Quản lý các thiết bị cá nhân
* Đồng hồ đo thông minh

### **Ưu điểm, khuyết điểm**

* Ưu điểm:
* Information (Thông tin): Càng có nhiều thông tin càng dễ dàng đưa ra một quyết định chính xác, IoT giúp con người biết được những gì họ cần phải mua tại các cửa hàng tạp hóa khi họ bước ra ngoài mà bản thân không cần kiểm tra việc đó, nó không chỉ giúp tiết kiệm thời gian mà còn vô cùng thuận tiện.
* Tracking (Theo dõi): Các thiết bị điện toán sẽ theo dõi cả chất lượng và tính sẵn có của mọi thứ trong nhà, nơi làm việc hay bất cứ đâu. Biết được những sản phẩm hết hạn sử dụng trước khi có người sử dụng chúng, giúp cải thiện an toàn và chất lượng cuộc sống, cùng với đó, sẽ giúp chúng ta không bao giờ rơi vào hoàn cảnh cần một thứ gì đó mà nó đã hết, IoT đã nhắc nhở ta việc đó trước khi nó xảy ra.
* Time (Thời gian): Tiết kiệm thời gian, như đã trình bày ở trên, khi IoT đi vào cuộc sống, nó sẽ giúp con người theo dõi và làm những công việc thay thế con người, do đó sẽ tiết kiệm được một lượng lớn thời gian và công việc.
* Money (Tiền): Một trong những lợi ích lớn nhất mà nó đem lại đó là về khía cạnh tài chính. Với công nghệ này nó sẽ thay thế cho những người chịu trách nhiệm về giám sát hay bảo tồn những vật tư hay nguồn cung cấp.
* Nhược điểm:
* Compatibility (Tính tương thích): Hiện tại, không có tiêu chuẩn quốc tế tương thích cho thiết bị gắn thẻ và giám sát, một khái niệm thống nhất như USB hoặc Bluetooth là cần thiết và điều này các công ty sản xuất các thiết bị này sẽ dễ dàng có thể thực hiện nhưng không dễ dàng để có một chuẩn tương thích cho các thiết bị.
* Complexity (Sự phức tạp): Với các hệ thống phức tạp, khả năng xảy ra các trường hợp thất bại sẽ cao hơn. Với Internet of Things nó có thể không đoán trước được. Ví dụ: giả sử tác giả và một người bạn cùng nhận được thông báo rằng thực phẩn trong nhà đã hết và cả hai đều vào cửa hàng và mua thực phẩm, kết quả là tác giả và bạn kia đã mua gấp 2 lần số lượng cần thiết...
* Privacy/Security (Vấn đề riêng tư và bảo mật): Bảo mật là vấn đề lớn với IoT, IoT là một mạng lưới toàn cầu, dữ liệu nó có thể thu thập và chia sẻ ở khắp nơi với bất cứ ai, do đó tất cả dữ liệu phải được mã hóa để những thông tin về tình trạng tài chính hay số lượng thực phẩm mà tiêu thụ không phải là thông tin public tại nơi làm việc hay với bạn bè có thể biết được.
* Safety (Sự an toàn): Có thể phần mềm sẽ bị tấn công và thông tin cá nhân của người dùng sẽ bị rò rỉ, do đó có thể gặp rủi ro. Tất cả các rủi ro này người tiêu dùng sẽ chịu trách nhiệm.

## **Giới thiệu về Raspberry Pi**

**1.2.1 Định nghĩa**



Hình 1.5: Raspberry Pi 3 B+.

Raspberry Pi là một máy vi tính rất nhỏ gọn, kích thước hai cạnh chỉ cỡ một cái thẻ ATM. Nhà sản xuất đã tích hợp mọi thứ cần thiết trong đó để người dùng sử dụng như một cái máy vi tính. Trên bo mạch của Raspberry Pi có CPU, GPU, RAM, khe cắm thẻ microSD, Wi-Fi, Bluetooth và 4 cổng USB 2.0. Khi mua Raspberry Pi về, chúng ta chỉ việc cài hệ điều hành (thực ra là copy/paste các thư mục vào thẻ nhớ), gắn chuột, bàn phím và màn hình là bắt đầu sử dụng được rồi (hoặc cao cấp hơn nữa là remote desktop từ một máy khác qua, hoặc SSH).

### **Sử dụng Raspberry Pi**

* Đầu coi phim HD giống như Android Box, hỗ trợ KODI đầy đủ.
* Máy chơi game cầm tay, console, game thùng. Chơi như máy điện tử băng ngày xưa, giả lập được nhiều hệ máy.
* Cắm máy tải Torrent 24/24.
* Dùng làm VPN cá nhân.
* Biến ổ cứng bình thường thành ổ cứng mạng (NAS).
* Làm camera an ninh, quan sát từ xa.
* Hiển thị thời tiết, hiển thị thông tin mạng nội bộ...
* Máy nghe nhạc, máy đọc sách.
* Làm thành một cái máy Terminal di động có màn hình, bàn phím, pin dự phòng để sử dụng mọi lúc mọi nơi, dò pass Wi-Fi...
* Làm thiết bị điều khiển Smart Home, điều khiển mọi thiết bị điện tử trong nhà.
* Điều khiển robot, máy in không dây từ xa, Airplay...

### **Lý do sử dụng Raspberry Pi trong đồ án**

Như những gì đã được nêu ở trên thì trong đồ án này Raspberry Pi có nhiệm vụ là:

* Trở thành gateway, địa điểm connect mạng không dây cho tất cả các thiết bị như pc, arduino sau khi được set-up.
* Trở thành một database di động với tính năng lưu tất cả các giá trị hay số đo của cảm biến trong cở sở dữ liệu.

Vậy lý do lại chọn nó trong đồ án này: câu trả lời:

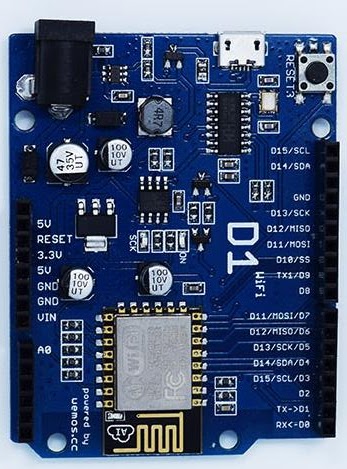
* Giá thành rẻ: chỉ với 990.000 đ đã có thể có một computer mini nhỏ gọn.
* Có thể dễ dàng biến Pi 3 B+ thành một gateway với tốc độ internet chuẩn g, có khả năng phát sóng wifi khá tốt.
* Trở thành một ổ cứng di động có thể di chuyển ở mọi nơi.
* Môi trường lập trình python chuyên nghiệp với nền tảng linux trên hệ điều hành Raspbian.
* Bước đệm cho những nghiên cứu sau này vì Raspberry Pi hỗ trợ rất tốt cho nền tảng IoT.

Với nhứng lý do nêu trên thì tác giả quyết định chọn Raspberry Pi 3 B+ để thực hiện đồ án này.

## **Tổng quan về một số cảm biến môi trường liên quan**

### **Wemos D1 R2**

Wemos D1 R2 là kit phát triển phiên bản mới từ Wemos, kit được thiết kế với hình dáng tương tự Arduino Uno nhưng trung tâm lại là module wifi Soc ESP8266EX được xây dựng lại firmware để có thể chạy với chương trình Arduino.  Kit thích hợp và dễ dàng thực hiện các ứng dụng thu thập dữ liệu và điều khiển qua Wifi.



Hình 1.6: Bo mạch Wemos D1 R2.

**Thông số kỹ thuật:**

* Vi điều khiển: ESP8266EX
* Điện áp hoạt động: 3.3V
* Số chân I/O: 11 (tất cả các chân I/O đều có Interrupt/PWM/I2C/One-wire, trừ chân D0)
* Số chân Analog Input: 1 (điện áp vào tối đa 3.3V)
* Bộ nhớ Flash: 4MB
* Điện áp vào: 9-24V
* Điện áp ra: 5V - Dòng max: 1A
* Giao tiếp: [Cable Micro USB](https://iotmaker.vn/cable-micro-usb-b-80cm.html)
* Wifi: 2.4 GHz
* Hỗ trợ bảo mật: WPA/WPA2
* Tích hợp giao thức TCP/IP
* Kích thước: 68.6mm x 53.4mm (2.701" x 2.102")
* Lập trình trên các ngôn ngữ: C/C++, Micropython, NodeMCU – Lua

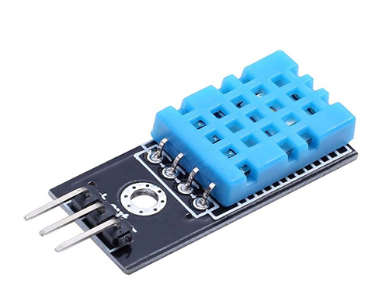
**Thông số chân cắm**

|  |  |  |
| --- | --- | --- |
| **Pin** | **Tính năng** | **ESP8266 Pin** |
| TX | TXD | TXD |
| RX | RXD | RXD |
| A0 | Analog Input | A0 |
| D0 | I/O | GPIO16 |
| D1 | I/O, SCL | GPIO5 |
| D2 | I/O, SDA | GPIO4 |
| D3 | I/O, 10K pull-up | GPIO0 |
| D4 | I/O, 10K pull-up, BUILTIN\_LED | GPIO2 |
| D5 | I/O, SCK | GPIO14 |
| D6 | I/O, MISO | GPIO12 |
| D7 | I/O, MOSI | GPIO13 |
| D8 | I/O, 10K pull-down, SS | GPIO15 |
| GND | Ground | GND |
| 5V | 5V |  |
| 3V3 | 3.3V | 3.3V |
| RST | Reset | RST |

Bảng 1.1: Thông số chân của Wenmos D1 R2.

### **DHT11**

DHT11 là cảm biến rất thông dụng hiện nay vì chi phí rẻ và rất dễ lấy dữ liệu thông qua giao tiếp 1-wire (giao tiếp digital 1-wire truyền dữ liệu duy nhất). Cảm biến được tích hợp bộ tiền xử lý tín hiệu giúp dữ liệu nhận về được chính xác mà không cần phải qua bất kỳ tính toán nào.



Hình 1.7: Cảm biến nhiệt độ, độ ẩm - DHT11.

**Thông số kỹ thuật**

* Điện áp hoạt động: 3V - 5V (DC)
* Dãi độ ẩm hoạt động: 20% - 90% RH, sai số ±5%RH
* Dãi nhiệt độ hoạt động: 0°C ~ 50°C, sai số ±2°C
* Khoảng cách truyển tối đa: 20m
* Kết nối:

|  |  |
| --- | --- |
| **Chân** | **Tính năng** |
| 5V | VCC |
| GND | GND |
| Digital I/O | OUT - Data |

Bảng 1.2: Thông số chân của cảm biến DHT11.

### **MQ-135**

Cảm biến khí ga MQ-135 là một trong những loại cảm biến được sử dụng để nhận biết: LPG, i-butan, Propane, Methane , Alcohol, Hydrogen, Smoke và khí ga. Được thiết kế với độ nhạy cao, thời gian đáp ứng nhanh. Giá trị đọc được từ cảm biến sẽ được đọc về từ chân Analog của vi điều khiển.



Hình 1.8: Cảm biến khí ga – MQ-135

**Thông số kỹ thuật**

* Nguồn hoạt động: 5V (DC)
* Dòng: 150mA
* Tính hiệu tương tự (analog)
* Hoạt động trong thời gian dài, ổn định
* Thứ tự chân:

|  |  |
| --- | --- |
| **Chân** | **Tính năng** |
| VCC | Nguồn |
| GND | GND |
| A0 | Analog |
| D0 | Digital |

Bảng 1.3: Thông số chân của cảm biến MQ-135.

### **Magnetic Door Switch Sensor**

Cảm biến từ cửa magnetic door switch sensor là loại công tắc từ : khi 2 miếng đặt gần nhau thì sẽ đóng tiếp điểm ( miếng có 2 sợi dây ), cảm biến chuyên dùng để cảnh báo chống trộm mở cửa nhờ thiết kế 2 cặp vuông góc, cảnh báo mở két sắt, tủ điện.v..v...



Hình 1.9: Cảm biến từ cửa – Magnetic Door Switch Sensor

**Thông số kỹ thuật**

* Khoảng cách hoạt động: 18 +- 6mm
* Điện áp giữa 2 tiếp điểm: tối đa 100 VDC
* Dòng tiêu thụ: tối đa 300mA
* Tuổi thọ: 100 triệu lần đóng/ngắt
* Dạng ngõ ra: thường mở ( khi để gần 2 miếng thì đóng)

|  |  |
| --- | --- |
| **Chân** | **Tính năng** |
| GND | GND |
| Digital I/O | OUT - Data |

Bảng 1.4: Thông số chân của cảm biến Magnetic Door Switch Sensor

### **1.3.5 Pir HC-SR501**

Cảm biến thân nhiệt chuyển động PIR (Passive infrared sensor) HC-SR501 được sử dụng để phát hiện chuyển động của các vật thể phát ra bức xạ hồng ngoại (con người, con vật, các vật phát nhiệt,...), 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 (giữ tín hiệu bao lâu sau khi kích hoạt) qua biến trở tích hợp sẵn.

Cảm biến thân nhiệt chuyển động PIR HC-SR501 tại Hshop.vn có cảm biến, thấu kính và board mạch chất lượng tốt cho độ nhạy và độ bền cao nhất.



Hình 1.10: Cảm biến thân nhiệt chuyển động Pir HC-SR501

**Thông số kỹ thuật**

* Phạm vi phát hiện: góc 360 độ hình nón, độ xa tối đa 6m.
* Nhiệt độ hoạt động: 32-122 ° F ( 050 ° C)
* Điện áp hoạt động: DC 3.8V - 5V
* Mức tiêu thụ dòng: ≤ 50 uA
* Thời gian báo: 30 giây có thể tùy chỉnh bằng biến trở.
* Độ nhạy có thể điều chỉnh bằng biến trở.
* Kích thước: 1,27 x 0,96 x 1.0 ( 32,2 x 24,3 x 25,4 mm)

|  |  |
| --- | --- |
| **Chân** | **Tính năng** |
| VCC | Nguồn |
| GND | GND |
| Digital I/O | OUT - Data |

Bảng 1.5: Thông số chân của cảm biến Pir HC-SR501

### **1.3.6 Arduino IDE**

Arduino IDE (Arduino Integrated Development Environment) là một trình soạn thảo văn bản, giúp chúng ta viết code để nạp vào bo mạch arduino.

Một trương trình viết bởi Arduino IDE được gọi là sketch, sketch được lưu dưới định dạng .ino.

**Nạp Sketch**



Hình 1.11: Hướng dẫn nạp Sketch

Trước khi nạp Sketch chúng ta phải chọn tên bo mạch, và cổng com: Từ menu chính chọn **Tools > Board** và **Tools > Port**

**Libraries** Là những thư viện bổ sung thêm những chức năng để sử dụng trong sketch, ví dụ làm việc với màn hình LCD người dùng phải cần có một thư viện để điều khiển nó. Để dùng library, ta chọn **Sketch->import Library**.

Đây là danh sách những thư viện thông dụng ta nên biết. Một số thư viện có sẵn với Arduino IDE. Một số khác chúng ta có thể tải thêm từ internet hoặc thông qua **Library Manager.**

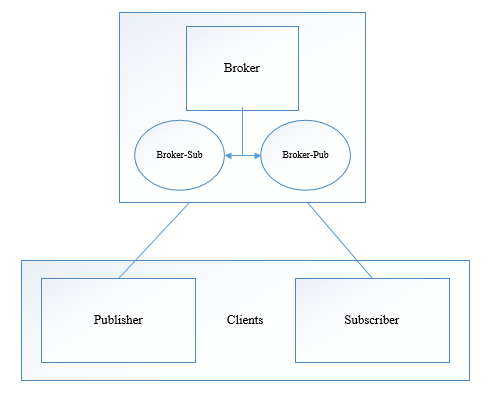
**Serial Monitor** Là hộp thoại để hiển thị dữ liệu được gửi từ bo mạch Arduino, hoặc gửi dữ liệu từ máy tính tới bo mạch.

## **Giới thiệu về giao thức MQTT**

### **1.4.1 Khái quát chung**

Message Queuing Telemetry Transport (MQTT) (tạm dịch là Giao vận tầm xa) là giao thức truyền message theo mô hình cung cấp/thuê bao (publish/subcribe). Nó dựa trên một Broker (điểm trung gian) "nhẹ" (khá ít xử lý), và được thiết kế có tính mở (tức là không đặc trưng cho ứng dụng cụ thể nào), đơn giản, "nhẹ", và dễ implement. Những đặc trưng này khiến MQTT rất lý tưởng để sử dụng trong các môi trường bị giới hạn tài nguyên tính toán và truyền dữ liệu như:

* Những nơi mà giá network đắt hoặc băng thông thấp, hay thiếu tin cậy
* Khi chạy trên một thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ
* Kiến trúc mức cao (high-level) của MQTT gồm 2 phần chính là Broker và Clients.



Hình 1.12: Kiến trúc mức cao của MQTT.

Trong đó, broker được coi như trung tâm, nó là điểm giao của tất cả các kết nối đến từ client. Nhiệm vụ chính của broker là nhận mesage từ publisher, xếp các message theo hàng đợi rồi chuyển chúng tới một địa chỉ cụ thể. Nhiệm vụ phụ của broker là nó có thể đảm nhận thêm một vài tính năng liên quan tới quá trình truyền thông như: bảo mật message, lưu trữ message, logs,...

Client thì được chia thành 2 nhóm là publisher và subscriber. Client là các software components hoạt động tại edge device nên chúng được thiết kế để có thể hoạt động một cách linh hoạt (lightweight). Client chỉ làm ít nhất một trong 2 việc là publish các message lên một topic cụ thể hoặc subscribe một topic nào đó để nhận message từ topic này.

MQTT Clients tương thích với hầu hết các nền tảng hệ điều hành hiện có: MAC OS, Windows, LInux, Androids, iOS...

Chúng ta có thể tưởng tượng broker giống như một sạp báo. Publisher là các tòa soạn báo. Tòa soạn in báo và chuyển cho sạp báo. Người đọc báo đến sạp báo, chọn tờ báo mình cần đọc (subscriber).

### **Các khái niệm đáng chú ý trong giao thức MQTT**

Trong một hệ thống sử dụng giao thức MQTT, nhiều client kết nối tới một server ( Trong MQTT, server được gọi là MQTT Broker). Mỗi client sẽ đăng ký theo dõi các kênh thông tin (topic) hoặc gửi dữ liệu lên kênh thông tin đó. Quá trình đăng ký này gọi là “subscribe” và hành động một client gửi dữ liệu lên kênh thông tin được gọi là “publish”. Mỗi khi kênh thông tin đó được cập nhật dữ liệu (dữ liệu này có thể đến từ các client khác) thì những client nào đã đăng ký theo dõi kênh này sẽ nhận được dữ liệu cập nhật đó.

* **Message**

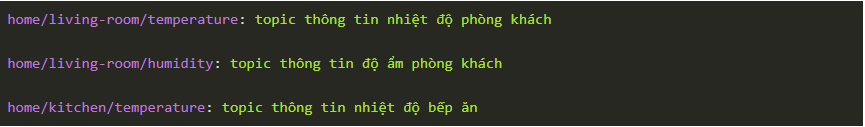
Trong giao thức MQTT, message còn được gọi là "message payload", có định dạng mặc định là plain-text (chữ viết người đọc được), tuy nhiên người sử dụng có thể cấu hình thành các định dạng khác.

* **Topic**

Topic có thể coi như một "đường truyền" logic giữa 2 điểm là publisher và subscriber. Về cơ bản, khi message được publish vào một topic thì tất cả những subscriber của topic đó sẽ nhận được message này.

Giao thức MQTT cho phép khai báo các topic kiểu phân cấp.

Giả sử người dùng có một hệ thống cảm biến đo thông tin môi trường trong ngôi nhà của họ. Một ngôi nhà thường sẽ có nhiều phòng và mỗi phòng lại có bộ cảm biến môi trường riêng. Như vậy, các topic phục vụ truyền tải thông tin môi trường cho ngôi nhà của họ có thể được khai báo như sau:



Bên cạnh đó, MQTT cũng hỗ trợ wildcards trong cú pháp khai báo topic, cụ thể đó là **"+" và "#"**. Trong đó, **"+"** bao gồm các topic ở cùng 1 level và **"#"** bao gồm tất cả các topic nằm dưới topic hiện tại.



Cụ thể như sau, khi muốn biết thông tin nhiệt độ của từng phòng, ta chỉ cần khai báo topic cần theo dõi là:



* **MQTT hỗ trợ 3 mức QoS**

**QoS-0** là mức đảm bảo thấp nhất, tất cả các message có QoS 0 sau khi được gửi đi bởi publisher sẽ không được kiểm tra xem đã đến broker hay chưa (fire - and - forget).

**QoS-1**: message được đảm bảo rằng đã đến nơi nhận ít nhất 1 lần (tức là sự trùng lặp vẫn có thể xảy ra).

**QoS-2**: đây là mức đảm bảo cao nhất, broker sẽ đảm bảo các message có QoS-2 sẽ đến nơi nhận chỉ 1 lần duy nhất, không trùng lặp, không thất lạc. Tất nhiên việc xác nhận với QoS-2 sẽ tốn băng thông hơn 2 cách còn lại.

* **Retain**

Retain là một cờ (flag) được gắn cho một message của giao thức MQTT. Retain chỉ nhận giá trị 0 hoặc 1 (tương ứng 2 giá trị logic false hoặc true). Nếu retain = 1, broker sẽ lưu lại message cuối cùng của 1 topic kèm theo mức QoS tương ứng. Khi client bắt đầu subscribe topic có message được lưu lại đó, client ngay lập tức nhận được message.

* **MQTT Bridge**

MQTT Bridge là một tính năng của MQTT Broker cho phép các MQTT Broker có thể kết nối và trao đổi dữ liệu với nhau. Để sử dụng tính năng này, người sử dụng cần tối thiểu 2 Broker. Trong đó, một Broker bất kỳ sẽ được cấu hình thành Bridge. Khi cấu hình MQTT bridge, ta cần lưu ý tới các thông số sau:

* **address**: địa chỉ của broker cần kết nối.
* **port:** cổng kêt nối của brokervà cổng kết nối mặc định của MQTT sẽ là 1883. Ngoài ra ta cũng có thể tạo thêm port kết nối cho MQTT.
* **bridge\_protocol\_version**: phiên bản của giao thức MQTT đang sử dụng chung cho 2 broker.
* **topic**: phần này định nghĩa 3 thông số: tên topic được trao đổi giữa 2 broker, chiều trao đổi (1 chiều hay 2 chiều) và topic mapping giữa 2 broker.
* **Bảo mật**

MQTT được thiết kế một cách nhẹ và linh hoạt nhất có thể. Do đó nó chỉ có 1 lớp bảo mật ở tầng ứng dụng: bảo mật bằng xác thực (xác thực các client được quyền truy cập tới broker).

Tuy vậy, MQTT vẫn có thể được cài đặt kết hợp với các giải pháp bảo mật đa tầng khác như kết hợp với VPN ở tầng mạng hoặc SSL/TLS ở tầng transport.

MQTT được thiết kế nhằm phục vụ truyền thông machine-to-machine nhưng thực tế chứng minh nó lại linh hoạt hơn mong đợi. Nó hoàn toàn có thể áp dụng cho các kịch bản truyền thông khác như: machine-to-cloud, cloud-to-machine, app-to-app. Chỉ cần có một broker phù hợp và MQTT client được cài đặt đúng cách, các thiết bị xây dựng trên nhiều nền tảng khác nhau có thể giao tiếp với nhau một cách dễ dàng.

Giao thức MQTT ra đời năm 1999 và tính đến thời điểm hiện tại, MQTT phiên bản (3.1.1) được công nhận chuẩn OASIS.

* 1. **Giới thiệu về Flask Python**
     1. **Python**

Python là một ngôn ngữ khá cũ được tạo ra bởi Guido Van Rossum. Thiết kế bắt đầu vào cuối những năm 1980 và được phát hành lần đầu tiên vào tháng 2 năm 1991.

Vào cuối những năm 1980, Guido Van Rossum làm việc trong Amoeba, phân phối một nhóm hệ điều hành. Ông muốn sử dụng một ngôn ngữ thông dịch như ABC (ABC có cú pháp rất dễ hiểu) để truy cập vào những cuộc gọi hệ thống Amoeba. Vì vậy, ông quyết định tạo ra một ngôn ngữ mở rộng. Điều này đã dẫn đến một thiết kế của ngôn ngữ mới, chính là Python sau này.

Rossum là fan của một sê-ri chương trình hài cuối những năm 1970, và cái tên “Python” được lấy từ tên một phần trong sê-ri đó “Monty Python’s Flying Circus”.

* **Tính năng chính của Python:**
* **Ngôn ngữ lập trình đơn giản, dễ học:** Python có cú pháp rất đơn giản, rõ ràng. Nó dễ đọc và viết hơn rất nhiều khi so sánh với những ngôn ngữ lập trình khác như C++, Java, C#. Python làm cho việc lập trình trở nên thú vị, cho phép tập trung vào những giải pháp chứ không phải cú pháp.
* **Miễn phí, mã nguồn mở:** có thể tự do sử dụng và phân phối Python, thậm chí là dùng cho mục đích thương mại. Vì là mã nguồn mở, người dùng không những có thể sử dụng các phần mềm, chương trình được viết trong Python mà còn có thể thay đổi mã nguồn của nó. Python có một cộng đồng rộng lớn, không ngừng cải thiện nó mỗi lần cập nhật.
* **Khả năng di chuyển:** Các chương trình Python có thể di chuyển từ nền tảng này sang nền tảng khác và chạy nó mà không có bất kỳ thay đổi nào. Nó chạy liền mạch trên hầu hết tất cả các nền tảng như Windows, macOS, Linux.
* **Khả năng mở rộng và có thể nhúng:** Giả sử một ứng dụng đòi hỏi sự phức tạp rất lớn, người dùng có thể dễ dàng kết hợp các phần code bằng C, [C++](https://quantrimang.com/cplusplus) và những ngôn ngữ khác (có thể gọi được từ C) vào code Python. Điều này sẽ cung cấp cho ứng dụng của chúng ta những tính năng tốt hơn cũng như khả năng scripting mà những ngôn ngữ lập trình khác khó có thể làm được.
* **Ngôn ngữ thông dịch cấp cao:** Không giống như [C](https://quantrimang.com/lap-trinh-c)/C++, với Python, không phải lo lắng những nhiệm vụ khó khăn như quản lý bộ nhớ, dọn dẹp những dữ liệu vô nghĩa,... Khi chạy code Python, nó sẽ tự động chuyển đổi code sang ngôn ngữ máy tính có thể hiểu. Không cần lo lắng về bất kỳ hoạt động ở cấp thấp nào.
* **Thư viện tiêu chuẩn lớn để giải quyết những tác vụ phổ biến:** Python có một số lượng lớn thư viện tiêu chuẩn giúp cho công việc lập trình của trở nên dễ thở hơn rất nhiều, đơn giản vì không phải tự viết tất cả code. Ví dụ: ta cần kết nối [cơ sở dữ liệu](https://quantrimang.com/co-so-du-lieu) SQLite trên Web server? Chúng ta có thể nhập thư viện SQLiteDB và sử dụng nó. Những thư viện này được kiểm tra kỹ lưỡng và được sử dụng bởi hàng trăm người. Vì vậy, có thể chắc chắn rằng nó sẽ không làm hỏng code hay ứng dụng của mình.
* **Hướng đối tượng:** Mọi thứ trong Python đều là hướng đối tượng. [Lập trình hướng đối tượng](https://quantrimang.com/steve-jobs-dinh-nghia-lap-trinh-huong-doi-tuong-khien-ca-the-gioi-than-phuc-131900" \t "_blank" \o "Steve Jobs định nghĩa lập trình hướng đối tượng khiến cả thế giới thán phục) (OOP) giúp giải quyết những vấn đề phức tạp một cách trực quan. Với OOP, có thể phân chia những vấn đề phức tạp thành những tập nhỏ hơn bằng cách tạo ra các đối tượng.
  + 1. **Flask – Python**

Flask là một web frameworks, nó thuộc loại micro-framework được xây dựng bằng ngôn ngữ lập trình Python. Flask cho phép người dùng xây dựng các ứng dụng web từ đơn giản tới phức tạp. Nó có thể xây dựng các Api nhỏ, ứng dụng web chẳng hạn như các trang web, blog, trang wiki hoặc một website dựa theo thời gian hay thậm chí là một trang web thương mại. Flask cung cấp cho chúng ta công cụ, các thư viện và các công nghệ hỗ trợ người dùng làm những công việc trên.

Flask là một micro-framework. Điều này có nghĩa Flask là một môi trường độc lập, ít sử dụng các thư viện khác bên ngoài. Do vậy, Flask có ưu điểm là nhẹ, có rất ít lỗi do ít bị phụ thuộc cũng như dễ dàng phát hiện và xử lý các lỗi bảo mật.

**Cài đặt thư viện flask python**

Để có thể sử dụng flask framework trong lập trình python. Việc đầu tiên là chúng ta cần cài bộ công cụ này.

**Lưu ý:**Tác giả khuyến khích người dùng sử dụng phiên bản mới nhất của python 3. Thư viện Flask chỉ hỗ trợ cho các phiên bản Python 3.4 hoặc mới hơn.

Nếu chúng ta đã cài Python và thiết lập môi trường thành công.Ta có thể sử dụng command sau đây để cài Flask:

*Pip3 install flask*

Chỉ với 1 lệnh chúng ta đã cài xong thư viện Flask này rồi.

* 1. **Giới thiệu về RESTful – API**
     1. **Resource**

Quản lý resource (tài nguyên) là một phần quan trọng và chiếm phần lớn trong việc phát triển website. Trong đó resource của các website khác nhau có thể sẽ khác nhau. Với các trang mạng xã hội như Facebook thì resource thường là danh sách người dùng (user hoặc account), danh sách các bài viết (post hoặc article), các ảnh được đăng (photo hoặc image), các trang fanpage (fanpage)... Đối với một trang chia sẻ ảnh như Instagram thì resource có thể là các ảnh được đăng (photo), danh sách người dùng (user)... Đối với các trang bán hàng thì resource có thể là danh sách sản phẩm (product), danh sách các người bán (seller), danh sách khách hàng (user hay customer)...

Việc quản lý resource của một website bao gồm 4 tác vụ chính:

* Tạo mới một resource (create)
* Lấy thông tin một resource (read)
* Cập nhật một resource (update)
* Xoá một resource (delete)

Và sau một thời gian dài thì người ta đã thống nhất ra các tiêu chuẩn khác nhau để thực hiện việc quản lý resource. Các tiêu chuẩn này (hay còn được gọi là Web API hoặc HTTP API) quy định một cách thống nhất việc quản lý các resource của web. Và RESTful là một trong các web API được sử dụng phổ biến ngày nay.

* + 1. **Restful Api**

RESTful API là một tiêu chuẩn dùng trong việc thết kế các thiết kế API cho các ứng dụng web để quản lý các resource. *RESTful* là một trong những kiểu thiết kế API được sử dụng phổ biến nhất ngày nay.

Trọng tâm của REST quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE...) và cách dịch dạng các URL cho ứng dụng web để quản các resource.

Ví dụ với một trang blog để quản lý các bài viết chúng ta có các URL đi với HTTP method như sau:

URL tạo bài viết: http://my-blog.xyz/posts/123. Tương ứng với HTTP method là POST

URL cập nhật bài viết với ID là 123: http://my-blog.xyz/posts/123. Tương ứng với HTTP method là PUT.

Với các ứng dụng web được thiết kế sử dụng RESTful, lập trình viên có thể dễ dàng biết được URL và HTTP method để quản lý một resource. Cần lưu ý bản thân RESTful không quy định logic code ứng dụng và RESTful cũng không giới hạn bởi ngôn ngữ lập trình ứng dụng. Bất kỳ ngôn ngữ lập trình (hoặc framework) nào cũng có thể áp dụng RESTful trong việc thiết kế API cho ứng dụng web.

* 1. **Tổng quan về SQLite**
     1. **Định nghĩa SQLite:**

SQLite là một thư viện phần mềm mà triển khai một SQL Database Engine truyền thống, không cần Server, không cần cấu hình và nhỏ gọn. SQLite là một trong những Database Engine phát triển nhanh nhất, nhưng sự phát triển của nó là về mặt tính phổ biến, không phải là về mặt kích cỡ. SQLite là mã nguồn mở.



Hình 1.13: SQLite Database

Nói một cách đơn giản SQLite là phần mềm quản lý cơ sở dữ liệu (DBMS) tương tự như Mysql, PostgreSQL... Đặc điểm của SQLite là gọn, nhẹ, đơn giản. Chương trình gồm 1 file duy nhất vỏn vẹn chưa đến 400kB, không cần cài đặt, không cần cấu hình hay khởi động mà có thể sử dụng ngay. Dữ liệu Database cũng được lưu ở một file duy nhất. Không có khái niệm user, password hay quyền hạn trong SQLite Database.

SQLite không thích hợp với những hệ thống lớn nhưng ở quy mô vừa tầm thì SQLite phát huy uy lực và không hề yếu kém về mặt chức năng hay tốc độ. Với các đặc điểm trên SQLite được sử dụng nhiều trong việc phát triển, thử nghiệm… và là sự lưa chọn phù hợp cho những người bắt đầu học Database.

SQLite Engine không là một Standalone Process giống như các cơ sở dữ liệu khác, bạn có thể liên kết nó một cách tĩnh hoặc một cách động tùy theo yêu cầu với ứng dụng của bạn. SQLite truy cập các file lưu giữ của nó một cách trực tiếp.

* + 1. **Ưu điểm:**
* SQLite không yêu cầu một tiến trình Server riêng rẽ để hoạt động.
* SQLite không cần cấu hình, nghĩa là không cần thiết phải cài đặt.
* Một SQLite Database đầy đủ được lưu giữ trong một disk file đơn.
* SQLite là rất nhỏ gọn, nhỏ hơn 400kB đã đươc cấu hình đầy đủ hoặc nhỏ hơn 250kB khi đã bỏ qua các tính năng tùy ý.
* SQLite là tự chứa, nghĩa là không có sự phụ thuộc vào ngoại vi.
* Các Transaction trong SQLite là tuân theo đầy đủ chuẩn ACID, đảm bảo truy cập an toàn từ nhiều tiến trình hoặc thread.
* SQLite hỗ trợ hầu hết các tính năng của một ngôn ngữ truy vấn trong chuẩn SQL92.
* SQLite được viết bằng ANSI-C và cung cấp API đơn giản và dễ dàng sử dụng.
* SQLite là có sẵn trên UNIX (Linux, Mac OS-X, Android, iOS) và Windows (Win32, WinCE, WinRT).
  + 1. **Lịch sử:**
* Năm 2000: D.Richard Hipp đã thiết kế SQLite với mục đích không cần quản trị để điều hành một chương trình.
* Vào tháng 8, SQLite 1.0 được công bố với GNU Database Manager.
* Năm 2011: Hipp bổ sung UNQI Interface cho SQLite DB và để phát triển UNQLite (là một Document Oriented Database).
  + 1. **Hạn chế:**

SQLite không hỗ trợ một số tính năng của chuẩn SQL92, bao gồm:

* RIGHT OUTER JOIN: Chỉ có LEFT OUTER JOIN được triển khai.
* FULL OUTER JOIN: Chỉ có LEFT OUTER JOIN được triển khai.
* ALTER TABLE: Các biến thể RENAME TABLE và ADD COLUMN của lệnh ALTER TABLE được hỗ trợ. Trong khi DROP COLUMN, ALTER COLUMN, ADD CONSTRAINT: Không được hỗ trợ.
* Hỗ trợ Trigger: Hỗ trợ FOR EACH ROW trigger nhưng không hỗ trợ FOR EACH STATEMENT.
* VIEWs: View trong SQLite là read-only. Bạn không thể thực thi một lệnh DELETE, INSERT, hoặc UPDATE trên một View
* GRANT và REVOKE: Chỉ hỗ trợ các quyền truy cập mà áp dụng trên normal file dưới hệ điều hành.
  + 1. **Lệnh trong SQLite**

Các lệnh SQLite chuẩn để tương tác với cơ sở dữ liệu quan hệ là giống như SQL. Chúng là CREATE, SELECT, INSERT, UPDATE, DELETE và DROP.

Những lệnh này có thể được phân chia thành 3 nhóm tùy theo tính năng hoạt động của chúng.

Nhóm DDL (Data Definition Language):

* CREATE: Tạo một bảng mới, một View của một bảng hoặc đối tượng khác trong Database
* ALTER: Sửa đổi một đối tượng cơ sở dữ liệu đang tồn tại, ví dụ một bảng
* DROP: Xóa cả một bảng, một View của một bảng hoặc đối tượng khác trong Database

Nhóm DML (Data Manipulation Language):

**•** INSERT: Tạo một bản ghi

**•** UPDATE: Sửa đổi các bản ghi

**•** DELETE: Xóa các bản ghi

Nhóm DQL (Data Query Language):

**•** SELECT: Lấy các bản ghi cụ thể từ một hoặc nhiều bảng

* 1. **Giới thiệu về Node-Red**

Node-RED là một công cụ lập trình để kết nối các thiết bị phần cứng, API và online services theo những cách mới và thú vị.

Nó cung cấp một trình soạn thảo dựa trên trình duyệt giúp dễ dàng kết nối các luồng với nhau bằng cách sử dụng một loạt các Node trong bảng màu (palette) có thể được triển khai chỉ bằng một cú nhấp chuột.

Nếu như ngày xưa phải biết ngôn ngữ C, C++ hay các ngôn ngữ máy bậc thấp. Thì giờ đây ta có thể code nó bằng javascript do Node-RED được xây dựng trên NodeJS

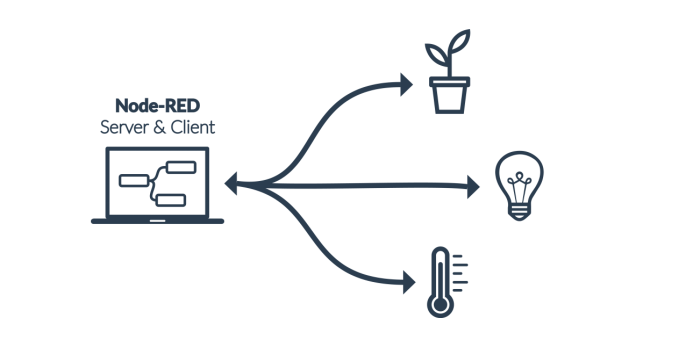
Node-RED được dựa trên Node.js, nó có thể được xem như một web server mà bạn có thể cấu hình tùy chỉnh các chức năng gọi là “flow” từ bất kỳ trình duyệt nào trên máy tính. Mỗi ứng dụng Node-RED bao gồm các node có thể liên kết được với nhau với các dạng là input, output và operation.

Một ví dụ đơn giản để chúng ta có thể hình dung được các node khác nhau sẽ tương tác như thế nào.



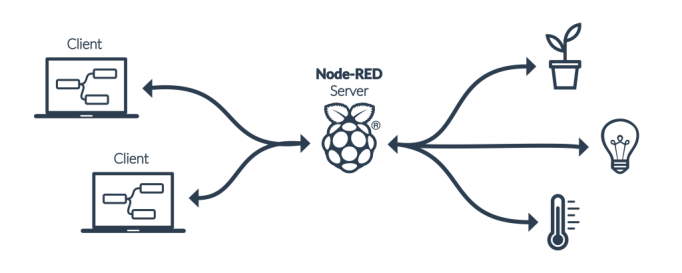
Hình 1.14: Node-Red dạng liên kết cơ bản

Với Node-RED ta có thể hình dung cách tương tác và giao tiếp với các thiết bị một cách tổng quan như hình dưới. Ở đây máy tính của mình sẽ đóng vai trò là server và client.



Hình 1.15: Node-Red đóng vai trò là server và client

Nếu dùng Raspberry thì thiết bị này sẽ đóng vai trò là Server, còn lại sẽ là client như hình.



Hình 1.16: Node-Red có vai trò là server, còn lại là client

# CHƯƠNG 2. TRIỂN KHAI ỨNG DỤNG NHÀ THÔNG MINH VỚI MQTT

1. 1. **Giới thiệu tổng quan bài toán**

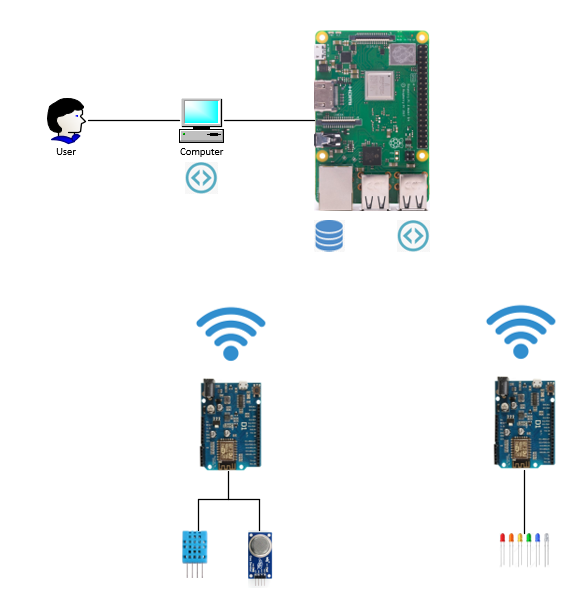
Internet of Thing hay được gọi là internet vạn vật, mạng lưới vạn vật kết nối internet là một liên mạng, trong đó các thiết bị, phương tiện vận tải, phòng ốc và các thiết bị khác được nhúng với các thiết bị điện tử, phẩn mềm, cảm biến, cơ cấu chấp hành cùng với khả năng kết nối mạng máy tính giúp cho các thiết bị thu thập và truyền tải dữ liệu.

Trong nội dung của bài đồ án tốt nghiệp thì ta cần phải xác định rõ, để xây dựng ứng dụng thì có hai điểm chính cần lưu ý: Thu thập và điều khiển.

Thu thập: là dùng Raspberry trờ thành một gateway sau đó phát internet cho các kit Arduino ESP8266 và ta sử dụng ở đây là Wemos D1 R2 thu để thu thập dữ liệu từ các cảm biến như DHT11 là cảm biến nhiệt độ và độ ẩm. Khi ta có được thông số thì sẽ storing vào database một cách trực tiếp để lưu lại dữ liệu theo thời gian. Từ đó có những hướng phát triển tiếp theo.

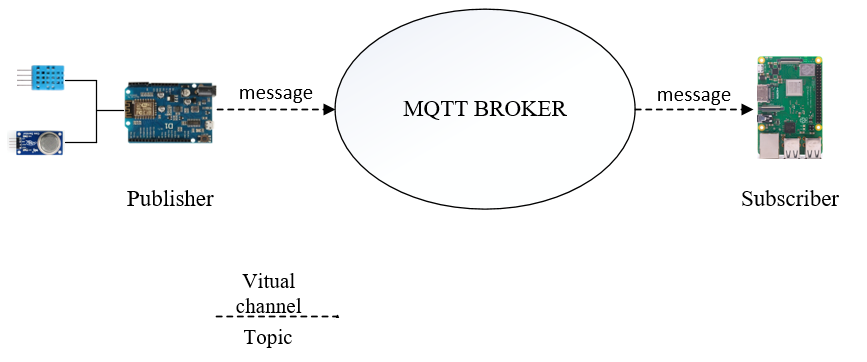
Điều khiển: Khi ta bấm một công tắc hay một cái nút để bật đèn hay một dụng cụ nào đó thì việc thực thi lệnh sẽ diễn ra ngay tức khắc khi ta tác động lên nút điều khiển. Vậy có nghĩa là ở đây độ trễ thời gian hầu như sẽ không có (real time). Để làm được việc đó ta cần phải xây dựng một Web-application. Trong đó, kit Arduino và Web-Site chính là các client.

Các cảm biến tự động như: Hệ thống chống trộm bằng cửa từ (Magneic Door Switch Sensor) thực hiện nhiệm vụ khi có người mở cửa thì còi sẽ báo động. Hệ thống chống trộm bằng chuyển động thân nhiệt (Pir HC-SR501) thực hiện nhiệm vụ khi phát hiện thân nhiệt của kẻ trộm đột nhập vào, sau đó còi sẽ báo động liên tục. Hệ thống cảm biến khí Gas (MQ-135) thực hiện nhiệm vụ khi phát hiện rò rỉ khí gas, lúc đó còi sẽ báo động và chúng ta có thể xử lý kịp thời để được an toàn.



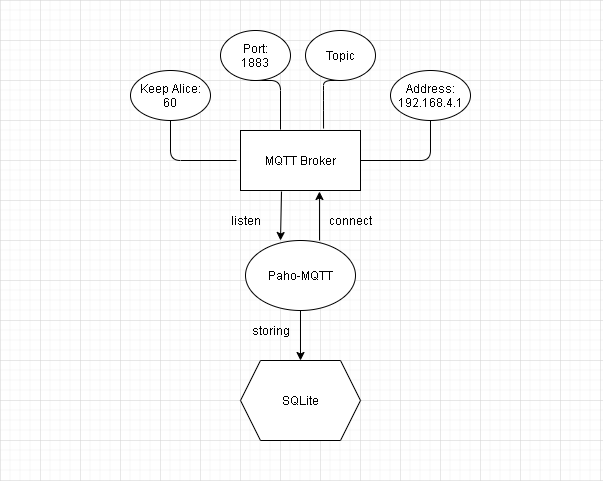
Hình 2.1: Triển khai tổng quan mô hình theo thực tế.

Trong module thu thập: đây sẽ là phần mà ta áp dụng MQTT. Khi một sensor (cảm biến) là DHT11 thu thập được thông số từ môi trường như nhiệt độ, độ ẩm thì sẽ được Arduino Wemos D1 R2 xử lý. Trong phần code nhúng, con Wemos sẽ kết nối không dây với còn Raspberry Pi mà ta đã thiết lập nó thành gateway đồng thời ta cũng kết nối giao thức MQTT. Ta sẻ gửi tín hiệu Publisher từ thiết bị Wemos với các dữ liệu từ sensor lên MQTT Broker với địa chỉ address của PI là 192.168.4.1 và cổng port mặc định của MQTT là 1883.



Hình 2.2: Triển khai mô hình MQTT trong thực tế.

Khi dữ liệu của bên gửi là Publisher đã gửi lên MQTT Broker thành công thì cùng lúc đó bên nhận là Subcriber chính là con Raspberry Pi sẽ nhận được dữ liệu. Nhưng vấn đề không dừng lại ở đây, điều mà tác giả muốn là dữ liệu sẽ được đẩy thẳng trực tiếp vào cơ sở dữ liệu (SQLite). Vậy thư viện Paho-Mqtt sẽ làm những công việc gì? Thư viện Paho-Mqtt sẽ truy cập đến các thông số mà ta đã khai báo trong giao thức MQTT. Ở đó Paho-Mqtt sẽ lắng nghe từ port mà giao thức ta sử dụng để gửi dữ liệu lên và ngay sau đó trực tiếp đẩy dữ liệu đến cơ sở dữ liệu. Nói ra một cách chính xác trong bài đồ án này thì Paho-Mqtt sẽ lắng nghe từ port 1883 và storing vào SQLite



Hình 2.3: Mô hình hoạt động của Paho-Mqtt..

Theo những gì em tìm hiểu được về giao thức MQTT thì nó vẫn có thể được sử dụng để điều khiển các thiết bị, nhưng bị hạn chế là mỗi lần điều khiển ta phải khai báo port, địa chỉ, topic, v.v…. và trên nền web-browser thì việc điều khiển trên đó hầu như rất khó và rất ít hỗ trợ cho các browser hiện nay.. Đối với người lập trình thì không có vấn đề gì, nhưng với người bình thường thì là cả một vấn đề và không tối ưu. Thế nên tác giả đã mở rộng thêm trong đồ án của mình, thử sức tìm hiểu đến một thứ gọi là web application.

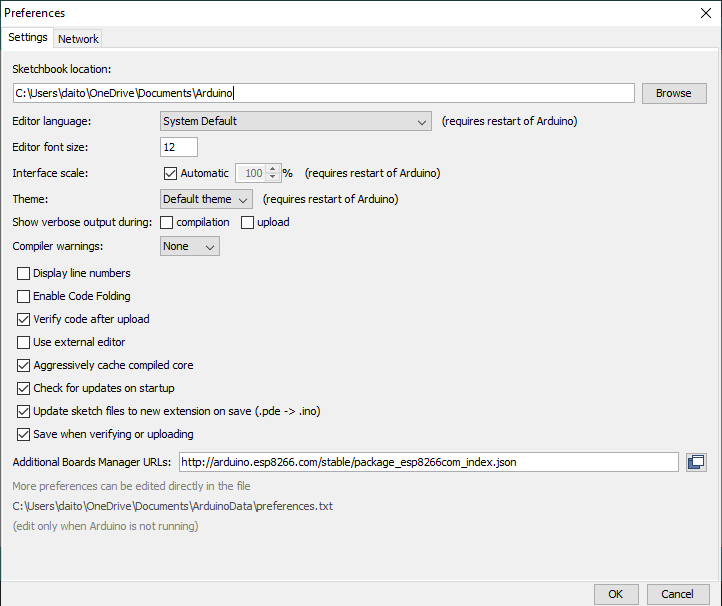
## **Thiết lập Arduino IDE**

* + 1. **Thiết lập cơ bản trên Arduino IDE**

Để lập trình trên Arduino IDE, đặt biệt là trên thiết bị phần cứng chuẩn bị là Wemos D1 R1 thì ta cần phải chuẩn bị với các bước sau:

Bước 1: cài đặt bộ công cụ, trình biên dịch, SDK hỗ trợ chip ESP8266 trong Arduino IDE

Với bộ công cụ này, chúng ta có thể dễ dàng lập trình, biên dịch và sử dụng các thư viện dành cho ESP8266 trực tiếp trên Arduino IDE. Mở Arduino IDE, trên thanh Menu chọn **File → Preferecens**, trong tab **settings** chọn các tùy chọn như hình dưới:



Hình 2.4: Preferences của Arduino IDE.

**Sketckbook location** là đường dẫn mà chúng ta muốn lưu Sketch (file chương trình).

Mục **Additional Board Manager URLs** nhập đường dẫn:

http://arduino.esp8266.com/stable/package\_esp8266com\_index.json.

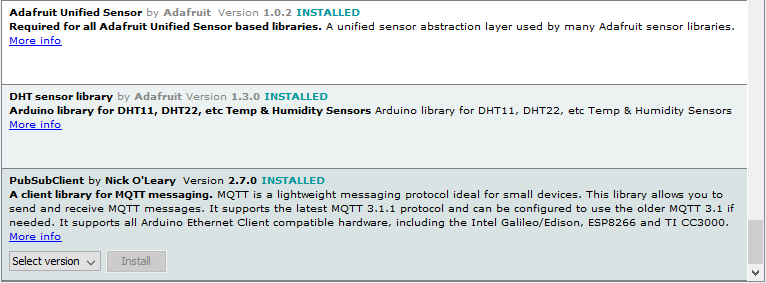
Bước 2: Cài đặt board ESP8266.

Mở **Boards Manager** ở mục **Tools** trên thanh menu-bar → tìm board cần sử dụng với keyword **ESP8266** → chọn board cần cài đặt như hình và nhấn vào install.

Bước 3: Cài đặt thư viện Arduino.

Một số thư viện do các nhà phát triển khác công bố và được tự do sử dụng có thể cài đặt trực tiếp bằng công cụ Library Manager của Arduino.

Khởi động arduino IDE và chọn mục Sketch => include library => Manager library:

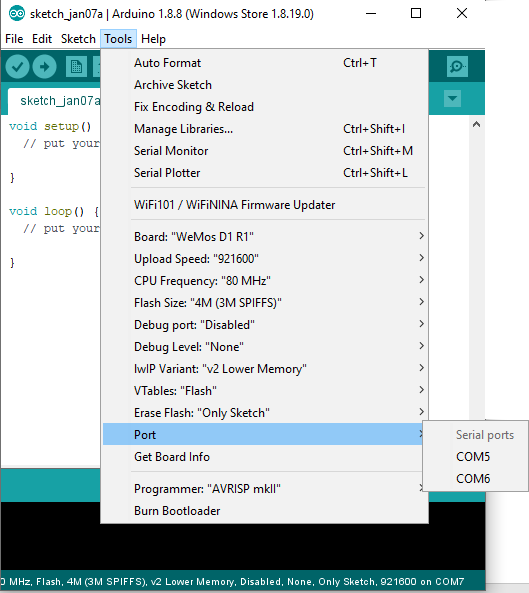


Hình 2.5: Cài đặt thư viện Arduino.

Bước 4: Thiết lập thông số với broad Wemos D1 :

Vào Tool => Board: Wemos D1 và thiết lập các thông số cần thiết: Thiết lập thông số cần thiết cho Arduino.

Khi cắm USB kết nối thiết bị nhúng là Wenmos D1 thì phần Port trong Tool sẽ hiện lên, lúc ta hãy chọn phù hợp để nhúng thiết bị.



Hình 2.6: Chình thông số Wenmos D1 trên Arduino IDE.

### **Nhúng thiết bị với giao thức MQTT**

Khởi tạo kết nối trong MQTT với các khai báo cần thiêt như: thư viện, port, address, topic …

**Code:**

***//Khai báo thư viện ESP8266 WiFi, DHT, MQTT***

*#include <ESP8266WiFi.h>*

*#include <PubSubClient.h>*

*#include "DHT.h"*

***//Khai báo loại cảm biến DHT***

*#define DHTTYPE DHT11*

***//Kết nối wifi Raspberry Pi***

*const char\* ssid = "utc2hcm";*

*const char\* password = "123456789aA";*

***//Kết nối MQTT***

*const char\* mqtt\_server = "192.168.4.1";*

***//Khởi tạo ESPClient***

*WiFiClient espClient;*

*PubSubClient client(espClient)*

***//Kết nối các chân cắm cảm biến đến ESP8266***

*const int DoorPin = 2;*

*const int PirPin = 13;*

*const int GasPin = 15;*

*const int buzzer = 12;*

*const int LedPhongNgu = 4;*

*const int LedPhongKhach = 5;*

*const int LedPhongBep = 0;*

*const int LedPhongTam = 15;*

*const int DHTPin = 14;*

***//Khởi tạo cảm biến DHT***

*DHT dht(DHTPin, DHTTYPE);*

***//Kết nối ESP8266 tới wifi Raspberry Pi***

*void setup\_wifi() {*

*delay(10);*

*Serial.println();*

*Serial.print("Connecting to ");*

*Serial.println(ssid);*

*WiFi.begin(ssid, password);*

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

*delay(500);*

*Serial.print(".");*

*}*

*Serial.println("");*

*Serial.print("WiFi connected - ESP IP address: ");*

*Serial.println(WiFi.localIP());*

*}*

***//Hàm xử lý dữ liệu***

*void callback(String topic, byte\* message, unsigned int length) {*

*Serial.print("Message arrived on topic: ");*

*Serial.print(topic);*

*Serial.print(". Message: ");*

*String messageTemp;*

*for (int i = 0; i < length; i++) {*

*Serial.print((char)message[i]);*

*messageTemp += (char)message[i];*

*}*

*Serial.println();*

***//Hàm nhận thông báo topic, nếu thông báo là 1 thì đèn sáng(on) là 0 thì đèn tắt(off). Dùng giao thức MQTT đề điều khiển.***

*if(topic=="room/ledphongngu"){*

*Serial.print(" Led phong ngu ");*

*if(messageTemp == "1"){*

*digitalWrite(LedPhongNgu, HIGH);*

*Serial.print("On");*

*}*

*else if(messageTemp == "0"){*

*digitalWrite(LedPhongNgu, LOW);*

*Serial.print("Off");*

*}*

*}*

*if(topic=="room/ledphongkhach"){*

*Serial.print("Led phong khach ");*

*if(messageTemp == "1"){*

*digitalWrite(LedPhongKhach, HIGH);*

*Serial.print("On");*

*}*

*else if(messageTemp == "0"){*

*digitalWrite(LedPhongKhach, LOW);*

*Serial.print("Off");*

*}*

*}*

*if(topic=="room/ledphongbep"){*

*Serial.print("Led phong bep ");*

*if(messageTemp == "1"){*

*digitalWrite(LedPhongBep, HIGH);*

*Serial.print("On");*

*}*

*else if(messageTemp == "0"){*

*digitalWrite(LedPhongBep, LOW);*

*Serial.print("Off");*

*}*

*}*

*if(topic=="room/ledphongtam"){*

*Serial.print("Led phong tam ");*

*if(messageTemp == "1"){*

*digitalWrite(LedPhongTam, HIGH);*

*Serial.print("On");*

*}*

*else if(messageTemp == "0"){*

*digitalWrite(LedPhongTam, LOW);*

*Serial.print("Off");*

*}*

*}*

*Serial.println();*

*}*

***//Hàm kết nối lại ESP8266 tới MQTT nếu bị mất mạng***

*void reconnect() {*

***// Lặp lại cho đến khi được kết nối lại***

*while (!client.connected()) {*

*Serial.print("Attempting MQTT connection...")*

*if (client.connect("ESP8266Client")) {*

*Serial.println("connected");*

***// Đăng ký các topic để điều khiển đèn LED***

*client.subscribe("room/ledphongngu");*

*client.subscribe("room/ledphongkhach");*

*client.subscribe("room/ledphongbep");*

*client.subscribe("room/ledphongtam");*

*} else {*

*Serial.print("failed, rc=");*

*Serial.print(client.state());*

*Serial.println(" try again in 5 seconds");*

***// Wait 5 seconds before retrying***

*delay(5000);*

*}*

*}*

*}*

***//Callback là hàm gọi lại là hàm nhận tin nhắn và điều khiển đèn LED***

*void setup() {*

*dht.begin();*

*pinMode(DoorPin, INPUT\_PULLUP);*

*pinMode(PirPin, INPUT\_PULLUP);*

*pinMode(GasPin,INPUT\_PULLUP);*

*pinMode(buzzer, OUTPUT);*

*pinMode(LedPhongNgu, OUTPUT);*

*pinMode(LedPhongKhach, OUTPUT);*

*pinMode(LedPhongBep, OUTPUT);*

*pinMode(LedPhongTam, OUTPUT);*

*Serial.begin(115200);*

*setup\_wifi();*

*client.setServer(mqtt\_server, 1883);*

*client.setCallback(callback);*

*}*

***//Hàm vòng lặp***

*void loop() {*

*if (!client.connected()) {*

*reconnect();*

*}*

*if(!client.loop())*

*client.connect("ESP8266Client");*

*now = millis();*

***//Xuất ra nhiệt độ, độ ẩm mới sau 30s***

*if (now - lastMeasure > 10000) {*

*lastMeasure = now;*

*float h = dht.readHumidity();*

*float t = dht.readTemperature();*

*float f = dht.readTemperature(true);*

***//Kiểm tra xem có bị lỗi không, nếu có sẽ thoát ra và thử lại***

*if (isnan(h) || isnan(t) || isnan(f)) {*

*Serial.println("Failed to read from DHT sensor!");*

*return;*

*}*

***//Tính giá trị nhiệt độ, độ ẩm***

*float hic = dht.computeHeatIndex(t, h, false);*

*static char temperatureTemp[7];*

*dtostrf(hic, 6, 2, temperatureTemp);*

*static char humidityTemp[7];*

*dtostrf(h, 6, 2, humidityTemp);*

***//Xuất ra giá trị nhiệt độ, độ ẩm***

*client.publish("/room/temperature", temperatureTemp);*

*client.publish("/room/humidity", HumidityTemp);*

*Serial.println(data);*

*Serial.print("Humidity: ");*

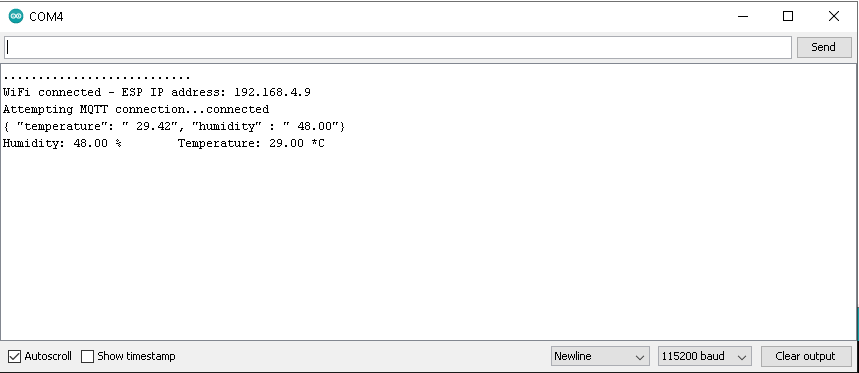
*Serial.print(h);*

*Serial.print(" %\t Temperature: ");*

*Serial.print(t);*

*Serial.print(" \*C ");*

**Kết quả:**



Hình 2.7: Kết nối và gửi dữ liệu lên MQTT.



## **Thiết lập Raspberry Pi 3 B+**

Chuẩn bị:

Phần cứng: Raspberry Pi 3 B+, cáp nguồn, thẻ nhớ SD 16Gb trở lên, cáp HDMI, màn hình, bàn phím, chuột….

Phần mềm: Hệ điều hành Raspbian, MobaXtem, SD card Formatter, Win32DiskImager, Postman

Thiết lập ban đầu:

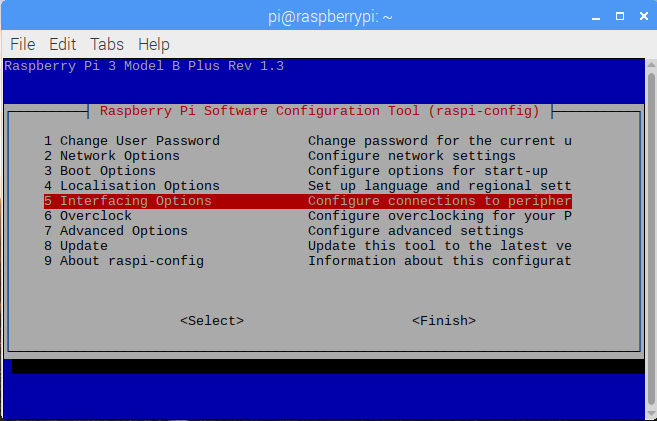
* Download hệ điều hành Raspbian ở link:

<https://www.raspberrypi.org/downloads/raspbian/>

* Format thẻ nhớ bằng phần mền SD card Formatter
* Dùng BalenaEtcher để cài đặt hệ điều hành Raspbian vào trong thẻ nhớ, gắn vào khe thẻ nhớ của Raspberry Pi và cắm nguồn khởi động.
* Khi khởi động lên màn hình ban đầu thể hiện như thế này là khâu chuẩn bị đã thành công.
* Thiết lập Raspberry Pi thành Gateway

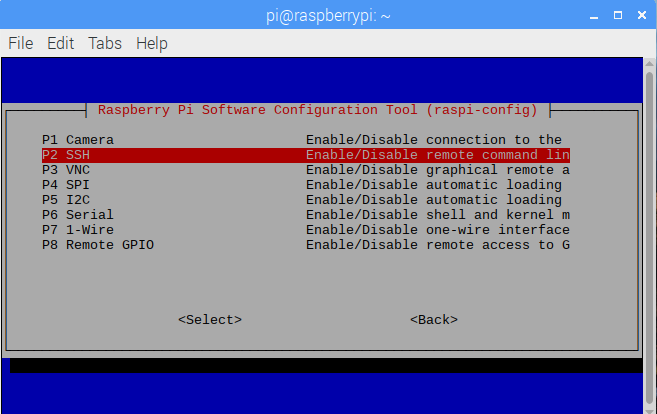
Mở kết nối SSH và dùng MobaXTerm để kết nối.

Dùng câu lệnh: sudo raspi-config để truy cập vào hệ thống để thiết lập:



Hình 2.8: Interfacing Options trong Raspberry Pi.

Vào Interfacing Options sau đó vào SSH và enable cho Pi



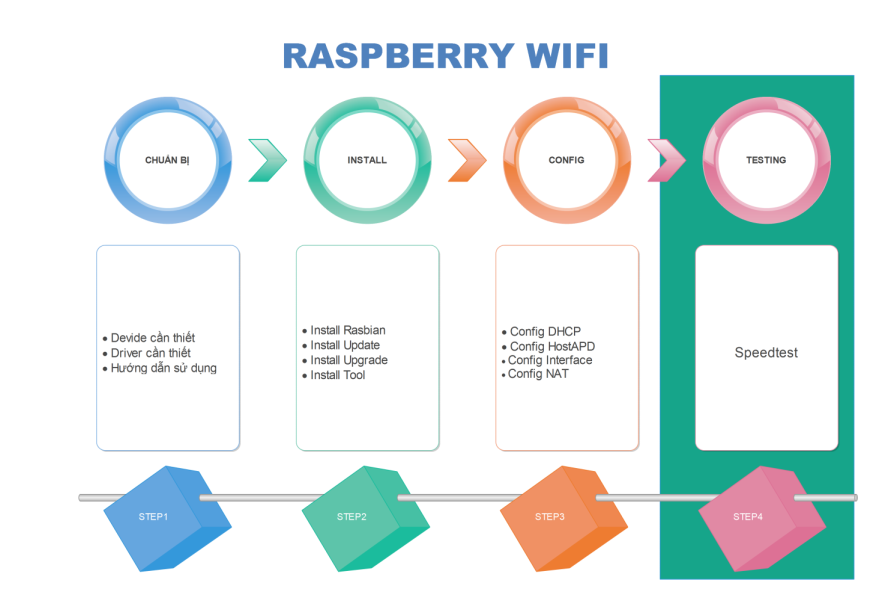
Hình 2.9: SSH trong Raspberry Pi.

Khi đã mở SSH thì chúng ta đã có thể dùng MobaXterm để kết nối thông qua SSH và bắt đầu khâu thiết lập Pi thành Gateway.

Trong phần mền MobaXterm mở kết nối mới với SSH với địa chỉ eth0 mà raspberry đã kết nối. Với tên đăng nhập là: pi và mật khẩu mặc định là: raspberry.

### **2.3.1 Thiết lập Raspbperry Pi thành Gateway**

* Sau đây là các bước để thiết lập Raspberry Pi trở thành Gateway:



Hình 2.10: Sơ đồ thiết lập raspberry wifi.

* Bước 1:Cài đặt tool cần thiết:

sudo apt-get update

Sudo apt-get upgrade

* Bước 2: Cài đặt điểm truy cập và phần mềm quản lý mạng:

Sudo apt-get install hostapd

Sudo systemctl unmask hostapd

Sudo systemctl enable hostapd

sudo apt install dnsmasq

Sudo apt-get instalsudo DEBIAN\_FRONTEND=noninteractive apt install -y netfilter-persistent iptables-persistentl dnsmasq

* Bước 3: Cấu hình DHCPCD

Câu lệnh hệ thống: sudo nano /etc/dhcpcd.conf

interface wlan0

static ip\_address=192.168.4.1/24

nohook wpa\_supplicant

* Bước 4: Cấu hình DHCP và DNS cho mạng không dây:

Đổi tệp cấu hình mặc định và chỉnh sửa tệp mới

sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig

sudo nano /etc/dnsmasq.conf

Thêm phần bên dưới và lưu nó:

interface=wlan0 # Listening interface

dhcp-range=192.168.4.2,192.168.4.20,255.255.255.0,24h

# Pool of IP addresses served via DHCP

domain=wlan # Local wireless DNS domain

address=/gw.wlan/192.168.4.1

# Alias for this router

* Bước 5: Cấu hình Hostapd:

sudo nano /etc/hostapd/hostapd.conf

country\_code=VN

interface=wlan0

ssid=utc2hcm

hw\_mode=g

channel=7

macaddr\_acl=0

auth\_algs=1

ignore\_broadcast\_ssid=0

wpa=2

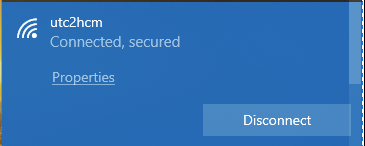
wpa\_passphrase=123456789aA

wpa\_key\_mgmt=WPA-PSK

wpa\_pairwise=TKIP

rsn\_pairwise=CCMP

**Kết quả:**



Hình 2.11: Kết quả Raspberry Pi thành Gateway.

* + 1. **Sử dụng MQTT trong Raspberry**
* Cài đặt tool cần thiết:

sudo apt-get update

sudo apt-get upgrade

* Cài đặt môi trường mosquitto cho MQTT:

sudo apt-get install mosquitto

* Test và khởi động lại mosquito:

service mosquitto status

sudo service mosquitto stop

mosquitto –v

* Cài đặt client để quản lý và sử dụng:

sudo apt-get install mosquitto-clients

Trong MQTT có hỗ trợ tính năng bảo mật khi ta thiết lập user và password cho chủ đề (topic) gửi nhận.

Để thiết lập được ta cần chỉnh sửa cấu hình cho mosquitto:

Dùng câu lệnh hệ thống: sudo nano /etc/mosquitto/mosquitto.conf. Sau đó ta thêm thông số cấu hình như dưới đây:

allow\_anonymous false

password\_file /etc/mosquitto/pwfile

listener 188s

# CHƯƠNG 3: THIẾT KẾ, CHẾ TẠO MÔ HÌNH THỰC NGHIỆM



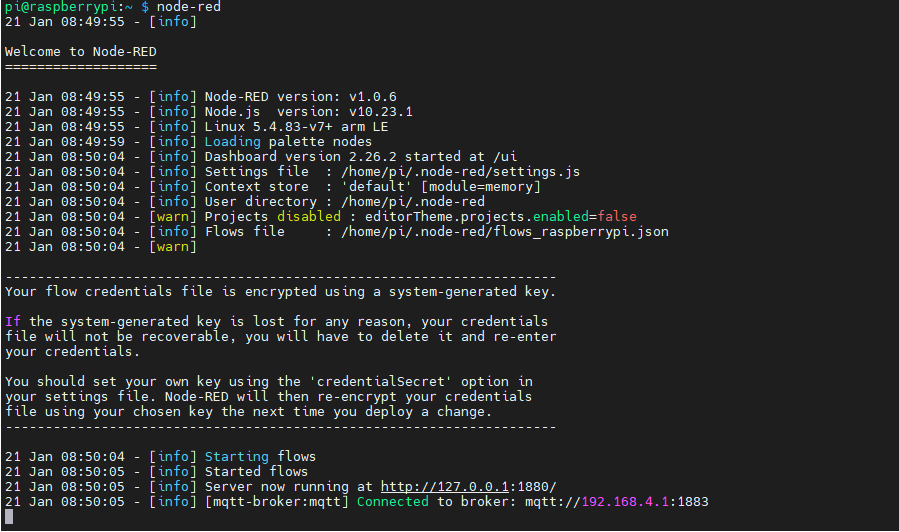
## **Giao diện giám sát và điều khiển**

Với những yêu cầu ở phạm vi đồ án phần web application chỉ cần xây dựng để hiển thị và điều khiển ngược lại ở phần thiết bị. Ở đây em dùng Node-Red

### **Phần điều Khiển và hiển thị:**

Đầu tiên cần khỏi động chương trình Node-Red trên Raspberry. Để khởi động chúng ta truy cập vào board Raspberry mở Terminal lên và gõ lệnh: Node-Red

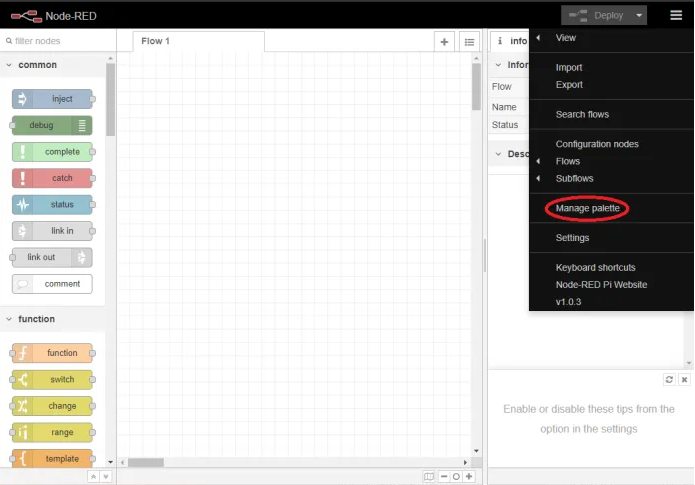
Nếu chạy Node-Red thành công chúng ta sẽ nhận được địa chỉ ở cuối dòng Terminal



Hình 3.1: Khỏi động Node-Red

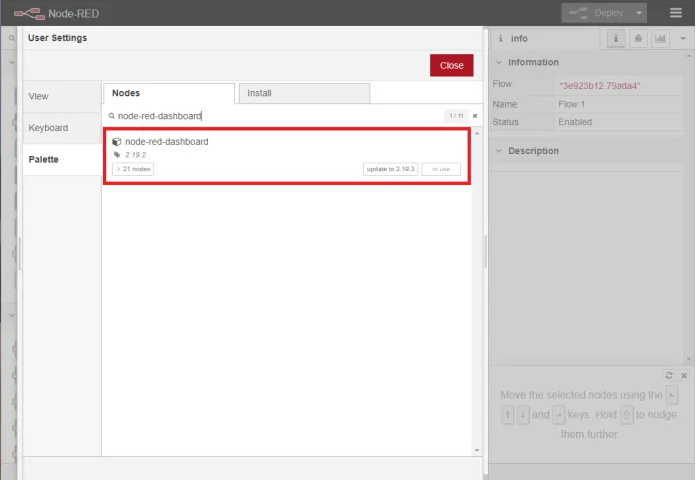
Bây giờ chúng ta sẽ dùng trình duyệt để truy cập vào trang giao diện của Node-Red là <http://192.168.4.1:1880/>

Sau đó chúng ta cần cài thêm Node-red-dashboard để hiển thị giao diện . Click vàio dấu 3 gạch ở góc bên phải -> Manage palette



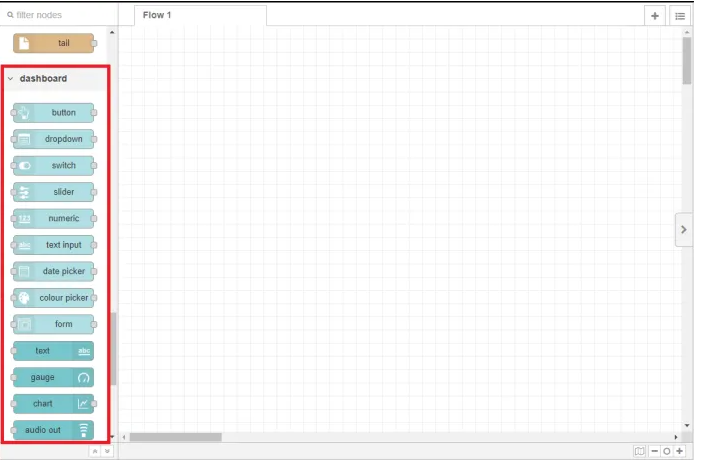
Hình 3.2: Manage palette

Nhập vào ô tìm kiếm: node-red-dashboard -> install



Hình 3.3: Tìm kiếm node-red-dashboard để cài đặt

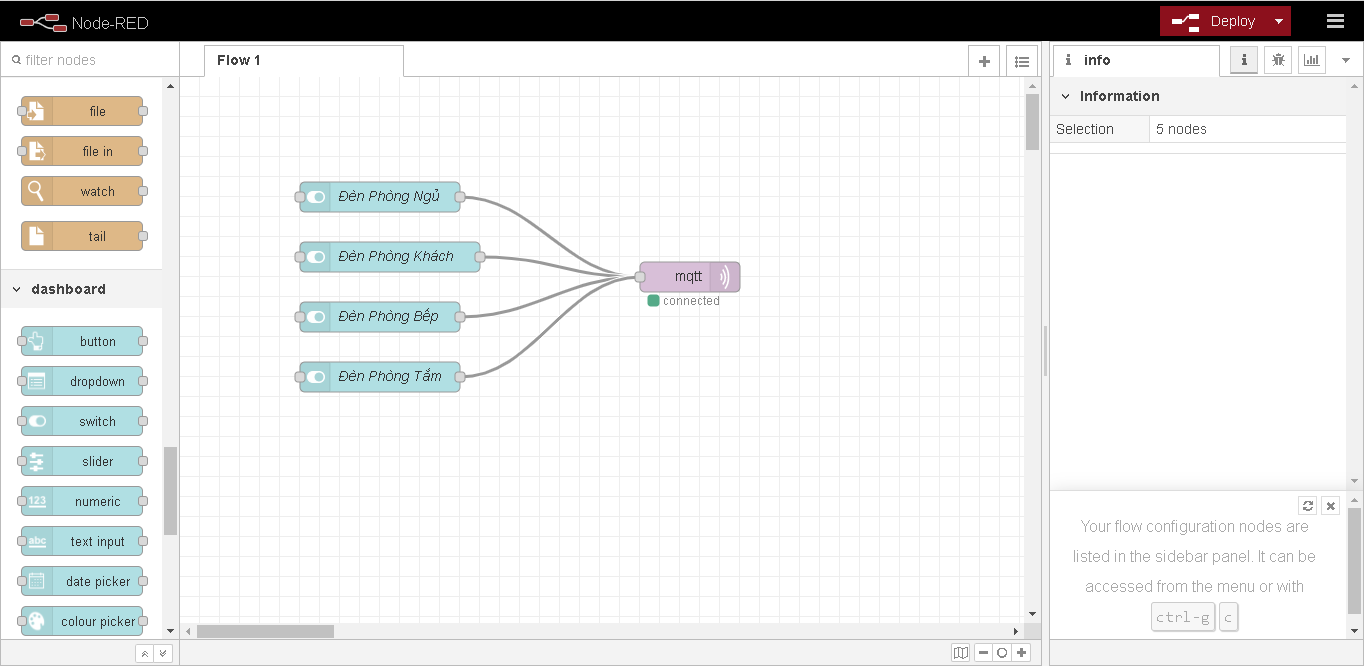
Sau khi cài đặt thành công chúng ta sẽ thấy được giao diện dashboard ở phía dưới bên trái.



Hình 3.4: Cài đặt Dashboard thành công

**Thiết lập điều khiển Đèn**

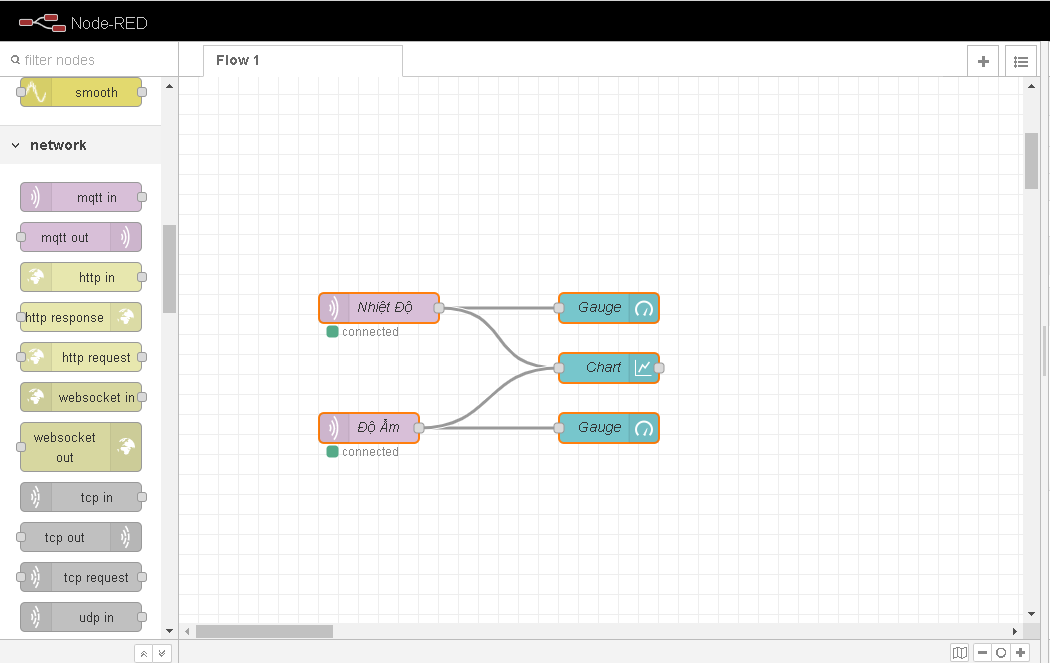
Trong Filter nodes dashboard chúng ta kéo các nút switch vào đổi tên thành: Đèn Phòng Ngủ, Đèn Phòng Khách, Đèn Phòng Bếp và Đèn Phòng Tắm. Rồi thiết lập chúng với giao thức MQTT input để xử lý thông tin. Sau đó chúng ta nối chúng lại với nhau để kết nối.



Hình 3.5: Thiết lập Đèn với MQTT

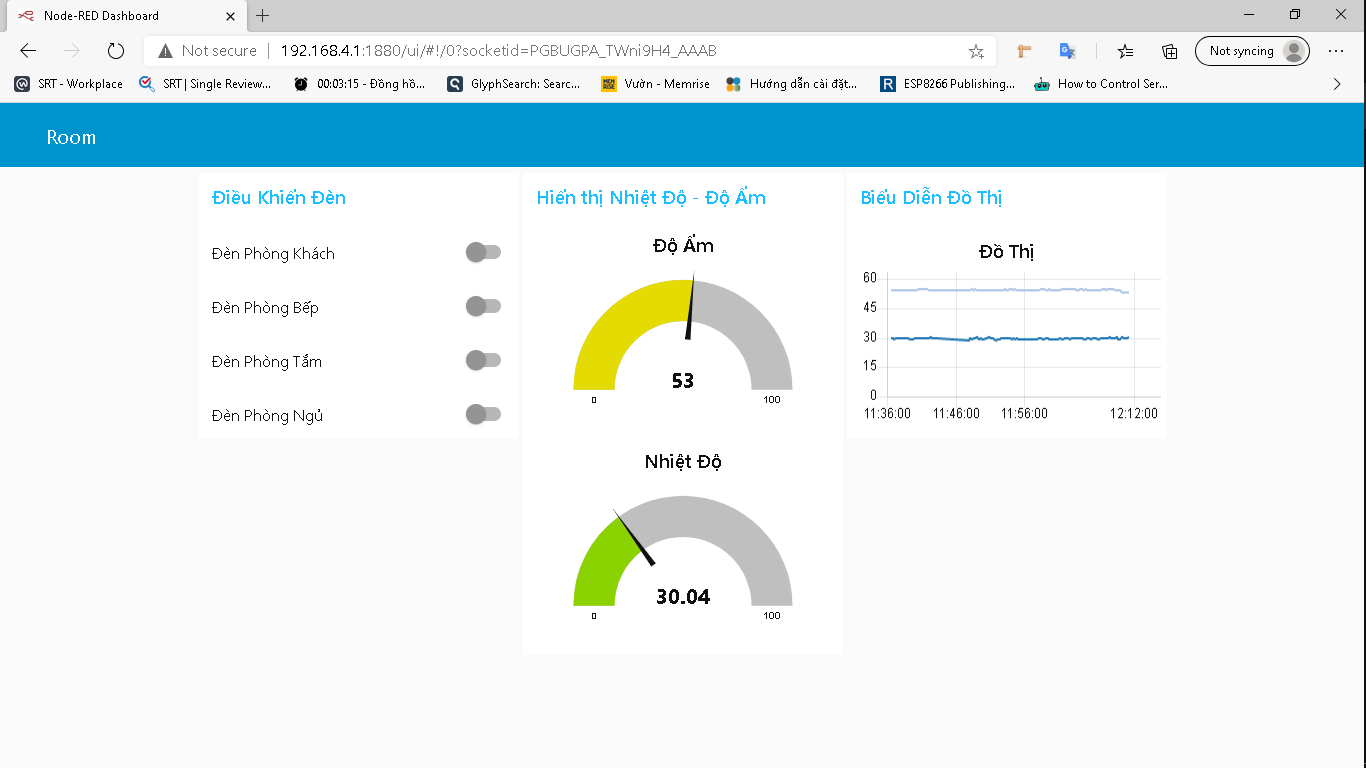
**Thiết lập hiển thị nhiệt độ, độ ẩm**

Ở đây chúng ta sẽ kéo 2 nút MQTT output vào đổi tên thành: Nhiệt độ, độ ẩm. Rồi thiết lập chúng với Gauge để hiển thị biểu đồ Gauge và Chart để hiển thị biểu đồ đồ thị.



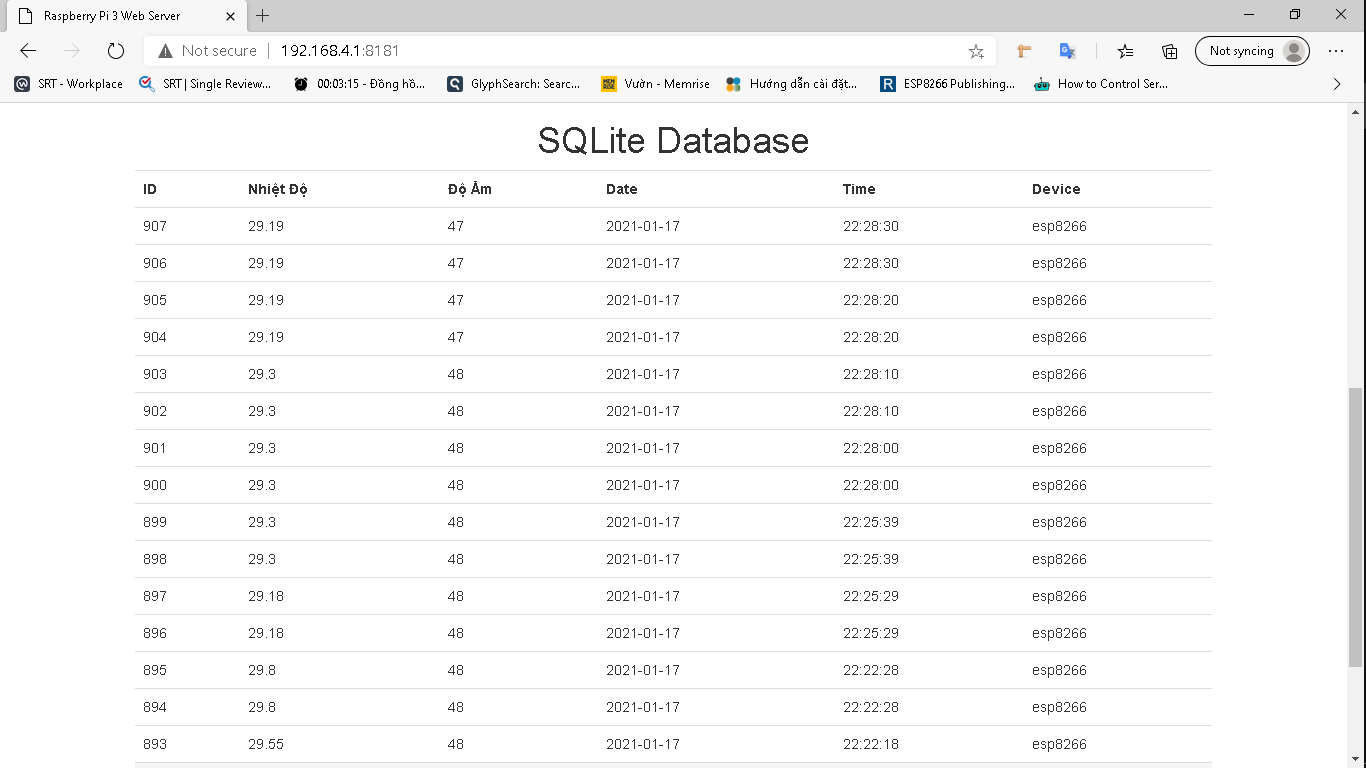
Hình 3.6: Thiết lập nhiệt độ, độ ẩm

Và kết quả cuối cùng chúng ta đã có giao diện điều khiển cũng như hiện thị các thông số nhiệt độ, độ ẩm. Ngoài ra chúng ra còn có thể theo dõi trên điện thoại.



Hình 3.7: Giao diện điều khiển và hiển thị nhiêt độ, độ ẩm

### **3.1.2 Phần hiển thị nhiệt độ, độ ẩm trong SQLite Database:**



Hình 3.8: Trang hiển thị nhiệt độ, độ ẩm trong SQLite Database

Trang hiển thị nhiệt độ, độ ẩm sẽ thu thập được theo thời gian thực(realtime) được cập nhật liên tục lưu lại trong cơ sở dữ liệu và hiển thị toàn bộ các thông số thu thập được trong cơ sở dự liệu theo: ID, Nhiệt Độ, Độ Ẩm, Date(Ngày/tháng/năm), Time(thời gian) và Device(tên thiết bị)

**Restful API bằng Python Flask để đẩy dữ liệu xuống SQLite**

**Code:**

*import paho.mqtt.client as mqtt*

*from flask import Flask, render\_template, request*

*import json*

*import sqlite3*

*app = Flask(\_\_name\_\_)*

*def dict\_factory(cursor, row):*

*d = {}*

*for idx, col in enumerate(cursor.description):*

*d[col[0]] = row[idx]*

*return d*

***//Hàm gọi lại khi client được server phản hồi***

*def on\_connect(client, userdata, flags, rc):*

*print("Connected with result code " + str(rc))*

*client.subscribe("/esp8266/dhtreadings")*

*mqttc=mqtt.Client()*

*mqttc.on\_connect = on\_connect*

*mqttc.on\_message = on\_message*

*mqttc.connect("192.168.4.1",1883,60)*

*mqttc.loop\_start()*

***//Hàm gọi lại khi nhận thông báo từ ESP8266***

*def on\_message(client, userdata, message):*

*if message.topic == "/esp8266/dhtreadings":*

*print("DHT readings update")*

*dhtreadings\_json = json.loads(message.payload)*

***//Hàm kết nối SQLite database***

*conn=sqlite3.connect('sensordata.db')*

*c=conn.cursor()*

*c.execute("""INSERT INTO dhtreadings (temperature,*

*humidity, currentdate, currentime, device) VALUES((?), (?), date('now'),*

*time('now', 'localtime'), (?))""", (dhtreadings\_json['temperature'],*

*dhtreadings\_json['humidity'], 'esp8266') )*

*conn.commit()*

*conn.close() }*

***//Đẩy dữ liệu xuống database***

*@app.route("/")*

*def main():*

*conn=sqlite3.connect('sensordata.db')*

*conn.row\_factory = dict\_factory*

*c=conn.cursor()*

*c.execute("SELECT \* FROM dhtreadings ORDER BY id DESC LIMIT 20")*

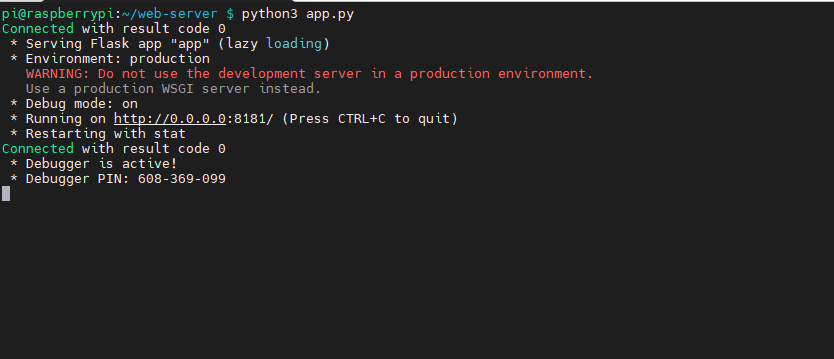
*readings = c.fetchall()*

*return render\_template('main.html', readings=readings, \*\*templateData)*

*)*

*if \_\_name\_\_ == "\_\_main\_\_":*

*app.run(host='0.0.0.0', port=8181, debug=True)*



Hình 3.9: Kết quả khởi chạy Restful Api

# KẾT QUẢ VÀ KIẾN NGHỊ

**Kết quả đạt được**

Sau thời gian tìm hiểu, nghiên cứu đến hiện thực hóa thì đề tài phần nào đó đã đáp ứng được những yêu cầu mà IoT hướng tới:

Xây dựng được hệ thống thu thập dữ liệu với bộ giao thức MQTT cùng với Arduino Wemos D1 R2 và Raspberrry Pi 3 B+. Xây dựng được hệ thống điều khiển từ xa và ở đây là điều khiển đèn LED những nội dung sau:

* Lập trình và xây dựng đưa một computer mini Raspberry Pi 3 B+ trở thành một Gateway có thể truyền phát wifi và trờ thành database để chứa dữ liệu.
* Tiết cận với hệ điều hành linux ở đây là hệ điều Raspbian và sử dùng một cách tổng quan với kiểu điều khiển trên command line.
* Lập trình nhúng với các thiết bị thực với ứng dụng Arduino IDE và xây dựng được hệ thống thu thập dữ liệu với DHT11 sử dụng giao thức MQTT để truyền phát dữ liệu từ thiết bị đến thiết bị, đẩy dữ liệu xuống trực tiếp cơ sỡ dữ liệu SQLite. Ngoài ra có các hệ thống cảnh báo tự động như hệ thống chống trộm bằng cảm biến cửa từ (Magnetic Door Switch Sensor) và cảm biên chuyển động thân nhiệt (Pir HC-SR501), hệ thông cảnh báo khí gas (MQ-135).
* Lập trình xây dựng một Restfui-Api đơn giản bằng Flask-Python.
* Lập trình web-application băng Node-Red với các chức năng chính:

+ Hiển thị các giá trị nhiệt độ, độ ẩm.

+ Điều khiển bật tắt LED từ xa.

**Hạn chế**

Do thời gian nghiên cứu còn hạn chế nên không tránh khỏi những thiếu xót nhất định.

**Hướng phát triển**

Xây dựng một hệ thống Web-site như Firebase: Lấy dữ liệu và lưu về cơ sở dữ liệu với MQTT, ngoài ra có thể hiển thị và điều khiển trực tiếp với Socket.IO.

Xây dựng một hệ thống báo động với dữ liệu khí ga đã lưu trữ được. Phân tích và kiểm tra độ bền của thiết bị.

TÀI LIỆU THAM KHẢO

1. Phạm Minh Tuấn, *Internet of Things Cho người mới bắt đầu.* NXB Đại Học Bách Khoa, 2014
2. Phan Văn Ca, Trương Quang Phúc, *Cơ sở ứng dụng Internet of Thing*, NXB Đại Học Quốc Gia TPHCM, 2017
3. Nguyễn Tất Bảo Thiện, *Arduino và Lâp Trình IoT,* NXB Thanh Niên, 2020.
4. Pham Quang Huy, *Stem với Arduino Dành Cho Người Tự Học*, NXB Thanh Niên, 2020.
5. Bùi Việt Hà, *Python Cơ Bản,* NXB Đại Học Quốc Gia Hà Nội, 2020
6. Lars, *A self hosted MQTT environment for Internet of Things, 2016*

<https://thingsmatic.com/2016/06/24/a-self-hosted-mqtt-environment-for-internet-of-things-part-1/>

1. Ngô Huỳnh Ngọc Khánh, *ESP8266 kết nối Internet Phần 1*, 2017

<http://arduino.vn/bai-viet/1496-esp8266-ket-noi-internet-phan-1-cai-dat-esp8266-lam-mot-socket-client-ket-noi-toi>

1. Hocarm, *MQTT và ESP8266*, 2017

<https://hocarm.org/mqtt-va-esp8266/>

1. Trần Văn Cường, *Tìm hiểu về iptables, 2015*

https://viblo.asia/p/network-tim-hieu-ve-iptables-n7prv348RKod

1. Rui Santos, *How to Install Mosquitto Broker on Raspberry Pi*, 2017

https://randomnerdtutorials.com/how-to-install-mosquitto-broker-on-raspberry-pi/

1. Ngothisaoly, *Giới thiệu về SQLite*, 2019

https://freetuts.net/gioi-thieu-sqlite-sqlite-la-gi-1719.html