# **LỜI CẢM ƠN**

Lời đầu tiên em xin gửi lời cảm ơn sâu sắc đến thầy , người luôn tận tình hướng dẫn, giúp đỡ em rất nhiều trong suốt quá trình thực hiện đồ án tốt nghiệp này.

Em xin chân thành cảm ơn Ban giám đốc Học viện cùng quý thầy cô trong suốt những năm học vừa qua, những người đã tận tâm giảng dạy, trang bị cho em những kiến thức quý báu.

Xin chân thành cảm ơn tới gia đình, anh chị, bạn bè đã giúp đỡ, ủng hộ, động viện em trong suốt thời gian học tập và nghiên cứu.

Mặc dù em đã cố gắng hoàn thiện thật tốt đồ án nhưng do kiến thức có hạn do đó không thể tránh khỏi những thiếu sót, em rất mong nhận được sự cảm thông, ý kiến đóng góp của các quý Thầy cô và các bạn!

Em xin chân thành cảm ơn!

**MỤC LỤC**

[**LỜI CẢM ƠN** 1](#_Toc28071304)

[**MỤC LỤC** 2](#_Toc28071305)

[**DANH MỤC HÌNH ẢNH** 4](#_Toc28071306)

[**DANH MỤC TỪ VIẾT TẮT** 6](#_Toc28071307)

[**LỜI NÓI ĐẦU** 7](#_Toc28071308)

[**CHƯƠNG I: TỔNG QUAN VỀ HỆ THỐNG ĐÈN ĐƯỜNG THÔNG MINH** 8](#_Toc28071309)

[**1.1.** **Khái niệm cơ bản** 8](#_Toc28071310)

[**1.2.** **Thế nào là đèn đường thông minh?** 8](#_Toc28071311)

[**1.3.** **Thành phần cơ bản của hệ thống đèn đường thông minh** 9](#_Toc28071312)

[**CHƯƠNG II: ĐỀ XUẤT GIẢI PHÁT XÂY DỰNG HỆ THỐNG** 11](#_Toc28071313)

[**2.1.** **Đề xuất giải pháp xây dựng** 11](#_Toc28071314)

[2.**1.1. Ý tưởng đề tài** 11](#_Toc28071315)

[**2.1.1.** **Mô hình hệ thống** 13](#_Toc28071316)

[**2.1.2.** **Các tính năng của hệ thống** 13](#_Toc28071317)

[**2.2.** **Giới thiệu các linh kiện sử dụng trong hệ thống** 14](#_Toc28071318)

[**2.2.1.** **NodeMCU** 14](#_Toc28071319)

[**2.2.2.** **Cảm biến độ ẩm, nhiệt độ DHT11** 15](#_Toc28071320)

[**2.2.3.** **Cảm biến ánh sáng** 20](#_Toc28071321)

[**2.2.4. Cảm biến chuyển động** 21](#_Toc28071322)

[**2.3.** **Giao thức truyền tài dữ liệu MQTT** 24](#_Toc28071323)

[**2.3.1.** **MQTT là gì?** 24](#_Toc28071324)

[**2.3.2.** **Ưu điểm của giao thức MQTT?** 24](#_Toc28071325)

[**2.3.3.** **MQTT làm việc như thế nào?** 25](#_Toc28071326)

[**2.3.4.** **Các thành phần của MQTT?** 26](#_Toc28071327)

[**2.4.** **Tổng quan về ThingsBoard IoT Platform** 26](#_Toc28071328)

[**2.4.1.** **Giới thiệu về ThingsBoard IoT Platform** 26](#_Toc28071329)

[**2.4.2.** **Các tính năng của nền tảng ThingsBoard** 28](#_Toc28071330)

[**CHƯƠNG III: NGHIÊN CỨU, XÂY DỰNG HỆ THỐNG PHẦN MỀM VỚI THINGSBOARD IOT PLATFORM** 29](#_Toc28071331)

[**3.1.** **Cài đặt, cấu hình hệ thống phần mềm** 29](#_Toc28071332)

[**3.1.1.** **Cài đặt ThingsBoard trên máy tính Ubuntu** 29](#_Toc28071333)

[**3.1.2.** **Quyền quản lý trong Thingsboard** 32](#_Toc28071334)

[**3.2.** **Truyền, nhận dữ liệu giữa các đèn với hệ thống Iot Gateway** 33](#_Toc28071335)

[**3.2.1.** **Sơ đồ kết nối hệ thống đèn đường thông minh** 33](#_Toc28071336)

[**3.2.2** **Thiết lập hệ thống đèn đường lên giao diện điều khiển ThingsBoard** 35](#_Toc28071337)

[**3.3.** **Cấu hình giao diện hiển thị và bật, tắt đèn trên ThingsBoard** 39](#_Toc28071338)

[**3.3.1.** **Hiển thị dữ liệu cảm biến lên giao diện ThingsBoard** 40](#_Toc28071339)

[**3.3.2.** **Điều khiển đèn với ThingsBoard** 43](#_Toc28071340)

# **DANH MỤC HÌNH ẢNH**

Chương I

[Hình 1. 1 Hệ thống đèn đường thông minh 5](#_Toc28009737)

[Hình 1. 2 Thành phần cơ bản của hệ thống đèn đường thông minh\ 6](#_Toc28009738)

Chương II

[Hình 2. 1 Mô hình hệ thống đèn 12](#_Toc28071253)

[Hình 2. 2 Mô hình hệ thống 13](#_Toc28071254)

[Hình 2. 3 Sơ đồ chân NodeMCU 15](#_Toc28071255)

[Hình 2. 4 Sơ đồ chân DHT11 16](#_Toc28071256)

[Hình 2. 5 Cấu tạo của cảm biến độ ẩm 16](#_Toc28071257)

[Hình 2. 6 Cấu tạo của biến trở nhiệt độ 17](#_Toc28071258)

[Hình 2. 7 Quá trình truyền dữ liệu của cảm biến DHT11 17](#_Toc28071259)

[Hình 2. 8 Start Signal 18](#_Toc28071260)

[Hình 2. 9 Tín hiệu phải hồi 18](#_Toc28071261)

[Hình 2. 10 Các bit dữ liệu truyền nhận trong DHT11 19](#_Toc28071262)

[Hình 2. 11 Dữ liệu nhiệt độ, độ ẩm 19](#_Toc28071263)

[Hình 2. 12 Nguyên lý hoạt động của cảm biến ánh sáng 20](#_Toc28071264)

[Hình 2. 13: Cảm biến ánh sáng quang trở CDS 21](#_Toc28071265)

[Hình 2. 14 Nguyên lý hoạt động của cảm biến chuyển động 22](#_Toc28071266)

[Hình 2. 15 Module cảm biến chuyển động HC SR501 23](#_Toc28071267)

[Hình 2. 16 Sơ đồ chân của module cảm biến chuyển động 24](#_Toc28071268)

[Hình 2. 17 Mô hình giao thức MQTT 24](#_Toc28071269)

[Hình 2. 18 Nguyên lý hoạt động của MQTT 26](#_Toc28071270)

[Hình 2. 19 Sơ đồ hệ thống nền tảng ThingsBoard 27](#_Toc28071271)

[Hình 2. 20 Giao diện ThingsBoard 27](#_Toc28071272)

Chương III

[Hình 3. 1 Giao giện trang chủ ThingsBoard 32](#_Toc28071279)

[Hình 3. 2 Phần quyền trong ThingsBoard 33](#_Toc28071280)

[Hình 3. 3 Sơ đồ kết nối phần cứng 34](#_Toc28071281)

[Hình 3. 4 Sơ đồ hệ thống 34](#_Toc28071282)

[Hình 3. 5 Sơ đồ kết nối Iot Gateway lên ThingsBoard server 35](#_Toc28071283)

[Hình 3. 6 Khởi tạo thiết bị 36](#_Toc28071284)

[Hình 3. 7 Giao diện hiển thị tọa dộ từng đèn 36](#_Toc28071285)

[Hình 3. 8 Giao diện chi tiết của từng node cảm biến 37](#_Toc28071286)

[Hình 3. 9 Cài đặt tọa độ 37](#_Toc28071287)

[Hình 3. 10 Giao diện cấu hình hiển thị thông số tọa độ 38](#_Toc28071288)

[Hình 3. 11. Giao diện chuyển hướng 39](#_Toc28071289)

[Hình 3. 12 Tọa độ của đèn sau khi được thiết lập 39](#_Toc28071290)

[Hình 3. 13 Hiển thị thông số từng đèn 43](#_Toc28071291)

[Hình 3. 14 Accsess token của thiết bị 44](#_Toc28071292)

[Hình 3. 15 Khởi tạo nút nhấn 45](#_Toc28071293)

[Hình 3. 16 Khởi tạo input trạng thái 45](#_Toc28071294)

[Hình 3. 17 Cấu hình cho giao diện nút nhấn 46](#_Toc28071295)

[Hình 3. 18 Cấu hình cho giao diện Input 47](#_Toc28071296)

[Hình 3. 19 Giao diện bật tắt đèn 47](#_Toc28071297)

# **DANH MỤC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| Viết tắt | Tiếng Anh | Tiếng Việt |
| AI | Artificial Intelligence | Trí tuệ nhân tạo |
| IOT | Internet of Things | Internet vạn vật kết nối |
| LED | Light Emitting Diode | Diốt phát quang |
| PIR | Passive InfraRed | Cảm biến thụ động |
| HTML | HyperText Markup Language | Ngôn ngữ đánh dấu siêu văn bản |
| GPIO | General Purpose Input Output | Chân vào/ra tín hiệu |

# **LỜI NÓI ĐẦU**

1. **Lý do chọn đề tài**

Internet of Things (vạn vật kết nối) thực sự đang trở thành tương lai của hệ sinh thái công nghệ trên toàn thế giới, chính vì vậy mà IoT nhận được rất nhiều sự quan tâm của các tổ chức, công ty công nghệ cho đến sự hưởng ứng của chính phủ trong định hướng phát triển nhằm xây dựng nền tảng công nghệ và ứng dụng vào thành phố thông minh, mạng lưới năng lượng và tiện ích thông minh.

Đứng trước làn sóng của cuộc cách mạng công nghiệp 4.0 đầy hứa hẹn và nhằm cung cấp giải pháp thiết yếu mang tính định hướng cho các thành phố lớn cho tương lai. Từ những ý nghĩa thực tiễn đó, em đã chọn đề tài “HỆ THỐNG QUẢN LÝ ĐÈN ĐƯỜNG”

1. **Mục tiêu**:

* Hiểu rõ đặc điểm, tính năng và cấu trúc của hệ thống đèn đường thông minh.
* Thiết kế thành công mô hình hệ thống đèn đường thông minh
* Xây dựng hệ thống quản lý đèn đường bằng nền tảng ThingsBoard có chức năng:
  + Điều khiển bật, tắt đèn từ xa thông qua giao diện web.
  + Hiển thị các thông số môi trường đo được tại các cảm biến tích hợp trong hệ thống đèn.
  + Tự động bật đèn khi trời tối, tắt đèn khi trời sáng.
  + Tiết kiệm năng lượng (tắt đèn khi không có phương tiện giao thông đi lại).

1. **Phạm vi nghiên cứu:**

Đồ án tập trung nghiên cứu và đưa ra giải pháp thiết kế cho hệ thống thông minh dựa trên nền tảng IoT platform ThingsBoard.

1. **Phương pháp nghiên cứu:**

Nghiện cứu lý thuyết thiết kế mô hình thực nghiệm và đánh giá kết quả đạt được.

1. **Bố cục đề tài:**

* Chương 1: Tổng quan về hệ thống đèn đường thông minh.
* Chương 2: Đề xuất giải pháp xây dựng hệ thống.
* Chương 3: Xây dựng hệ thống với nền tảng IoT Platform ThingsBoard
* Chương 4: Đánh giá và kết luận.

.

# **CHƯƠNG I: TỔNG QUAN VỀ HỆ THỐNG ĐÈN ĐƯỜNG THÔNG MINH**

1. **Khái niệm cơ bản**

Đèn đường là một nguồn ánh sáng được dựng lên trên các cạnh của một con đường hoặc lối đi. Cuộc sống ngày càng phát triển thì nhu cầu về hệ thống chiếu sáng cho con người ngày càng lớn. Trong hai mươi năm trở lại đây, hệ thống đèn ngày càng phát triển và được xây dựng một cách rộng rãi. Không chỉ trong nhà mà đường làng ngõ xóm, đường dân sinh, đường quốc lộ cũng được chiếu sáng để đảm bảo an toàn khi tham gia giao thông vào buổi tối.



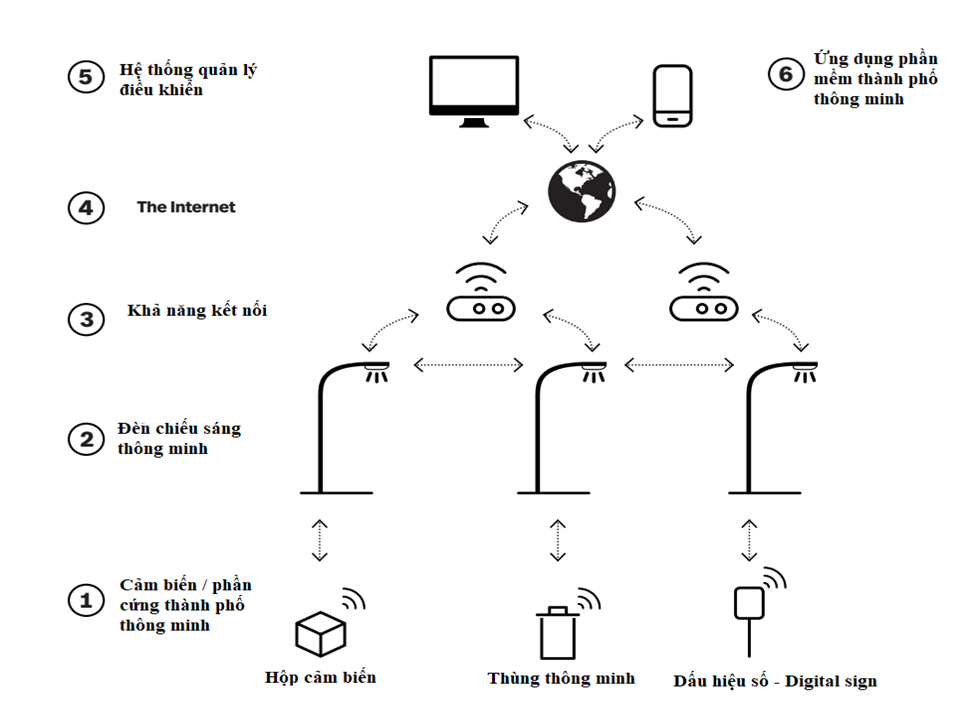
Hình 1. 1 Hệ thống đèn đường thông minh

1. **Thế nào là đèn đường thông minh?**

Trong những năm gần đây, hệ thống đèn đường ngày càng được chú trọng trong cuộc cách mạng 4.0 với các dự án IOT tiêu biểu như thành phố thông minh. Hệ thống đèn đường chiếu sáng hiện nay không chỉ đáp ứng nhu cầu chiếu sáng tối thiểu mà còn được tích hợp nhiều công nghệ tiên tiến trong đó có thể kể đến các chức năng như thu thập dữ liệu cảm biến, bật, tắt tự động, theo dõi mật độ giao thông qua camera an ninh…

Hệ thống đèn đường được kết nối dữ kiệu đến máy chủ qua để điều khiển từ xa, thu thập thông số, lưu trữ dữ liệu từ các cảm biến mang lại sự tiện lợi cho việc quản lý và xây dựng hệ sinh thái IoT.

1. **Thành phần cơ bản của hệ thống đèn đường thông minh**



Hình 1. Thành phần cơ bản của hệ thống đèn đường thông minh\

Một hệ thống chiếu sáng công cộng thông minh đa năng và tổng quát của thành phố thông minh bao gồm các thành phần chính như sau:

(1) Cảm biến/phần cứng thành phố thông minh (ví dụ: Cảm biến nhiệt độ, độ ẩm, không khí…) có thể được kết nối với hệ thống chiếu sáng công cộng thông minh. Với đặc điểm hạ tầng riêng phân tán, hệ thống chiếu sáng công cộng thông minh có thể kích hoạt các khả năng cho các cảm biến/phần cứng này - cung cấp vị trí lắp đặt, kết nối và nguồn điện đáng tin cậy.

(2) Đèn chiếu sáng thông minh với bộ điều khiển, cảm biến khác nhau được nhúng bên trong cho phép quản lý cấu hình chiếu sáng cho toàn hệ thống và cấu hình riêng riêng cho bộ đèn. Ngoài ra nó cũng cung cấp khả năng kết nối và gắn kết cho các cảm biến liên quan khác.

(3) Khả năng kết nối - kết nối giữa các đèn trên toàn hệ thống, bao gồm cả cổng kết nối ứng dụng. Nó có thể bao gồm:

* Có dây cứng (Hard-wired hoặc Ethernet).
* Kết nối không dây trực tiếp với Internet thông qua công nghệ không dây như 3G, 4G hay mạng điện năng thấp (Low Power Networks).
* Truyền thông qua đường dây điện: Thực hiện các tin nhắn trên các đường dây điện hiện có.

(4) Kết nối Internet: Thông qua các điểm máy tính cục bộ tại các điểm khu vực.

(5) Hệ thống quản lý điều khiển – Thu thập, xử lý và phân tích dữ liệu của toàn hệ thống, đảm bảo các chức năng cơ bản sau:

* Điều khiển, tùy chỉnh và giám sát các đèn riêng lẻ hoặc các nhóm đèn đường cho từng khu vực;
* Theo dõi và quản lý năng lượng, sự cố của từng cột đèn, từng khu vực và toàn hệ thống.
* Giám sát hiệu suất;
* Kiểm soát màu sắc của ánh sáng;
* Đáp ứng các trường hợp khẩn cấp;
* Dự báo, lập kế hoạch bảo trì.

(6) Ứng dụng phần mềm thành phố thông minh – Thông qua môi trường Internet để cung cấp các dịch vụ khác nhau. Ví dụ: Dịch vụ kiểm soát ánh sáng thông minh cho người dân, điểm cung cấp năng lượng điên, hệ thống giám sát thùng rác thông minh hoặc dịch vụ SmartParking…

**CHƯƠNG II: ĐỀ XUẤT GIẢI PHÁT XÂY DỰNG HỆ THỐNG**

1. **Đề xuất giải pháp xây dựng**

2.**1.1. Ý tưởng đề tài**

Trong lĩnh vực cơ sở hạ tầng của một thành phố nói chung, hệ thống chiếu sáng đường phố và nơi công cộng là một dịch vụ cộng đồng quan trọng. Tuy nhiên với hệ thống đèn đường cũ và phương pháp quản lý lạc hậu có thể tiêu thụ tới 40% ngân sách dành cho năng lượng của một thành phố. Công nghệ IoT được sử dụng thông qua kết hợp cảm biến chiếu sáng công cộng và công nghệ năng lượng mới đầy hứa hẹn và tiết kiệm đang trở thành một giải pháp “Chiếu sáng công cộng thông minh” thay thế cho toàn bộ hệ thống cung cấp chiếu sáng công cộng thông thường hiện nay.

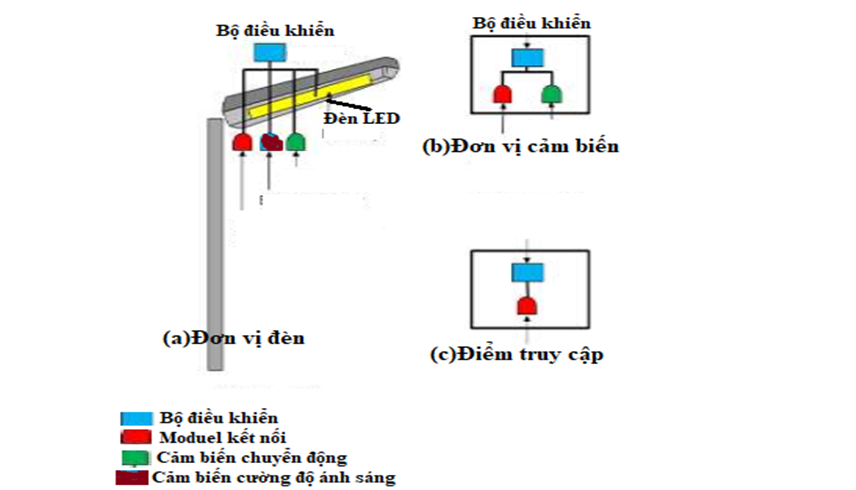
Thuật ngữ “Đèn đường thông minh” đề cập đến cơ sở hạ tầng chiếu sáng nơi công cộng (đường phố, công viên …) thực hiện vai trò cung cấp ánh sáng công cộng truyền thống với những tính năng, công nghệ mới được thiết kế để tăng hiệu quả, năng suất và dịch vụ. Một hệ thống chiếu sáng công cộng thông minh phải đáp ứng được những yêu cầu sau:

* Giảm tiêu thụ năng lượng điện của chiếu sáng công cộng thông qua kiểm soát khoảng thời gian bật tắt của ánh sáng. Giai đoạn này có thể được kiểm soát bằng cách sử dụng cảm biến mức độ ánh sáng môi trường hoặc bộ hẹn giờ tích hợp.
* Điều chỉnh giảm cường độ chiếu sáng thông qua giảm lưu lượng điện áp của mỗi đèn, để tiết kiệm năng lượng điện tiêu thụ.
* Sử dụng đèn và các thiết bị có hiệu suất cao, bao gồm tích hợp các thiết bị thông minh có thể điều chỉnh, tự động phát hiện sự cố nguồn điện và các vấn đề liên quan đến đèn như: Đo và gửi dữ liệu về trạng thái của ánh sáng, mức độ chiếu sáng, mức tiêu thụ năng lượng, điện áp, dòng điện và hệ số công suất; nhận lệnh ngắt và điều khiển ánh sáng…

Mặt khác, hệ thống quản lý trung tâm từ xa phải có chức năng cho phép điều khiển và giám sát từng đèn riêng lẻ. Với hệ thống này, người vận hành có thể theo dõi các thông số chính của bất kỳ cột đèn nào, bóng đèn nào từ phòng điều khiển hoặc thông qua thiết bị di động. Dữ liệu thu được có thể được phân tích xử lý, cho phép tính toán thống kê và dự báo mức tiêu thụ, trạng thái tuổi thọ của bóng đèn, điện áp, lỗi … Thông qua phương pháp này cho phép thay đổi từ qui trình bảo trì khắc phục sang qui trình bảo dưỡng dự phòng với mục tiêu tiết kiệm chi phí, thời gian cho công tác bảo trì hệ thống.

Ở phạm vi rộng hơn, hệ thống chiếu sáng công cộng thông minh cho phép các thành phố và đô thị điều chỉnh, thực thi các chiến lược chiếu sáng phù hợp với các điều kiện cụ thể như: Thay đổi cường độ, màu ánh sáng để phù hợp với từng hoàn cảnh không gian, thời gian và địa điểm phù hợp với từng khu vực, từng nhóm người sử dụng khác nhau.

Một cột đèn chiếu sáng nơi công cộng thông thường chỉ có 1-2 đèn LED và được cung cấp năng lượng thông qua hệ thống cáp điện. Hiện nay, ứng dụng về vật liệu chế tạo cho phép bóng đèn LED có tuổi thọ cao hơn, cường độ ánh sáng tốt hơn, tiêu thụ năng lượng điện ít hơn và có thể điều chỉnh chuyển đổi nhiều màu sắc khác nhau. Ứng dụng công nghệ IoT cho phép một bộ đèn thế hệ mới được tích hợp thêm bộ điều khiển, thiết bị cảm biến ánh sáng, cảm biến chuyển động... và các module kết nối để trở thành thiết bị chiếu sáng thông minh.



Hình 2. 1 Mô hình hệ thống đèn

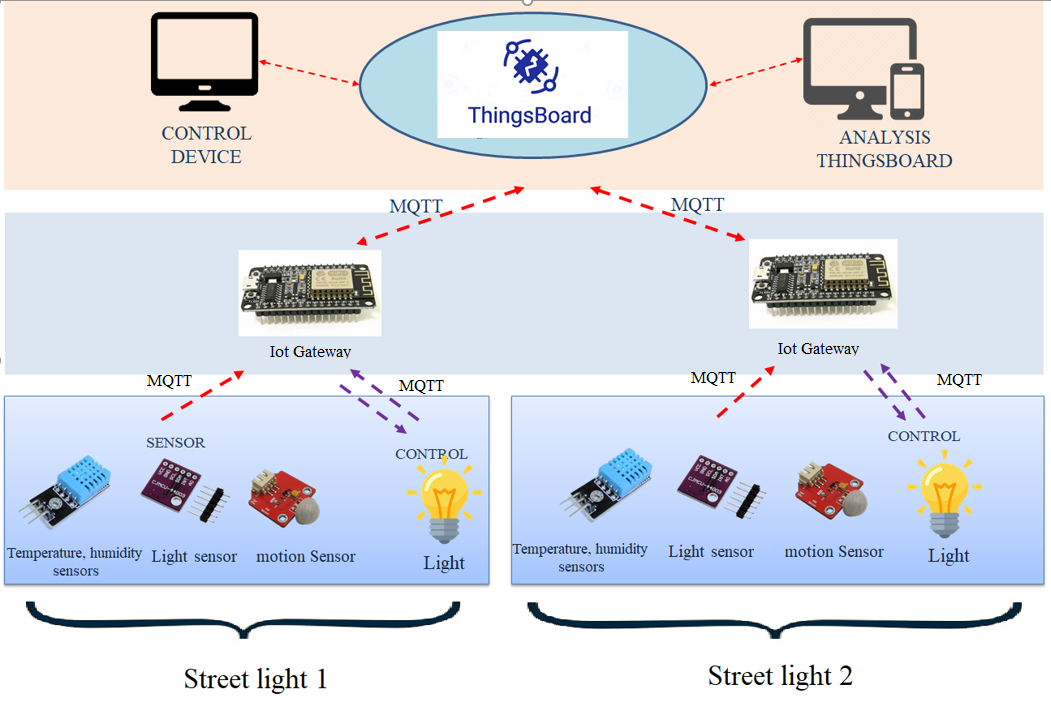
Với đặc điểm hạ tầng phân tán với khoảng cách qui định của hệ thống cột đèn. Tùy theo nhu cầu và nguồn ngân sách đầu tư, các thành phố có thể lắp đặt tích hợp thêm một số hệ thống khác như: Camera giám sát an ninh, hệ thống cảm biến đỗ xe … hoặc có thể tích hợp trực tiếp để hệ thống trở nên toàn diện và đa năng hơn. Các vị trí được tận dụng bao gồm:

- Bên trên cột đèn: Lắp đặt tấm pin năng lượng-Bộ chuyển đổi năng lượng, biển báo điện tử, camera giám sát công cộng, các loại cảm biến đo thông số môi trường …

- Bên dưới cột đèn: Thùng rác thông minh, các bộ cảm biến đỗ xe, điểm sạc năng lượng cho xe điện…

Trong phạm vi của đồ án, em tập trung xây dựng hệ thống đèn đường với các chức năng như: điều khiển bật, tắt đèn từ xa, thu thập dữ liệu từ cảm biến, nhận diện chuyển động để tiết kiệm năng lượng vào ban đêm. Hệ thống được quản lý và điều khiển bởi nền tảng ThingsBoard IoT Platform giúp việc quản lý trở nên trực quan và dễ dàng.

1. **Mô hình hệ thống**



Hình 2. Mô hình hệ thống

Hệ thống được chia làm ba phần chính:

* Phần điều khiển, thu thập dữ liệu cảm biến: Mỗi một cột đèn sẽ được tích hợp các thiết bị như đèn, các loại cảm biến về độ ẩm, ánh sáng, cảm biến chuyển đông.
* Phần truyền tải: các thông số được gửi lên máy chủ quản lý thông qua Node MCU, vừa là nơi chung chuyển phản hồi các tín hiệu bật, tắt đèn từ server đến các đèn và ngược lại
* Phần quản lý: Hệ thống sử dụng nền tảng Thingsboard, giúp cho việc quản lý và điều khiển trở nên dễ dàng và thân thiện thông qua giao diện web

1. **Các tính năng của hệ thống**

Hệ thống đèn đường thông minh có các tính năng chính như sau:

- Bật, tắt đèn từ xa thông qua giao diện ThingsBoard

- Phản hồi trạng thái đèn (khi bật, tắt bằng tay)

- Tự động bật đèn khi trời tối.

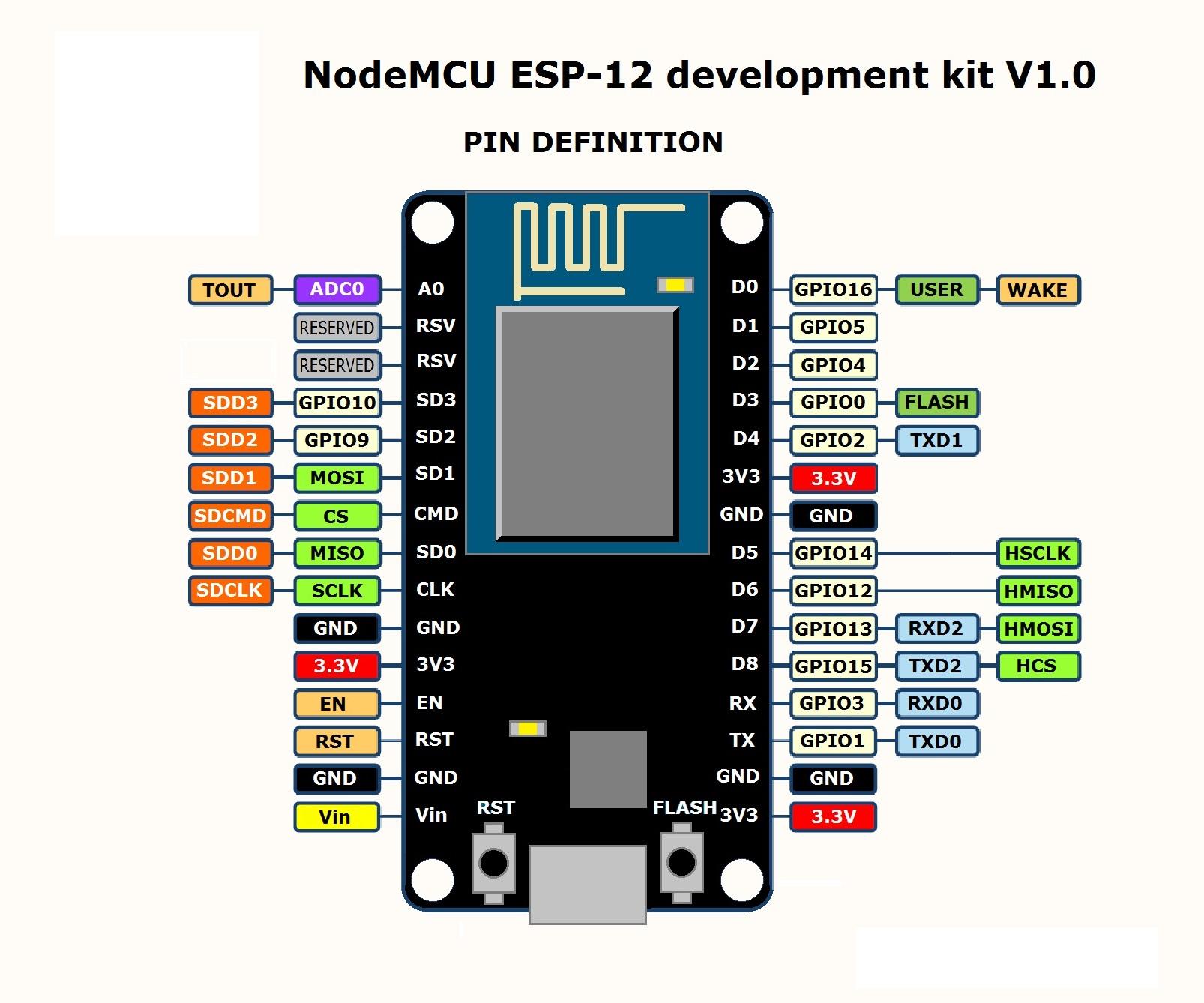
- Xác định được vị trí các đèn, theo dõi các thông số môi trường như nhiệt độ, độ ẩm, ánh sáng.

- Xây dựng mạng lưới đèn, lưu trữ dữ liệu lên server ThingsBoard

1. **Giới thiệu các linh kiện sử dụng trong hệ thống**
2. **NodeMCU**

NodeMCU là một dạng vi điều khiển tích hợp Wifi có khả năng lập trình được. Dùng để điều khiển các tác vụ thông qua Internet. NodeMCU được phát triển dựa trên LUA mã nguồn mở được phát triển cho chip wifi ESP8266.

* Ưu điểm:
* NodeMCU có giá thành rẻ, rẻ nhất trong các mạch tích hợp wifi.
* Sử dụng năng lượng thấp (3,3-5V).
* Kết nối wifi dễ dàng.
* Tích hợp được nhiều ngoại vi được sử dụng nhiều trong các dự án IOT và công nghiệp.
* Thông số kĩ thuật :
* Chip: ESP8266EX.
* WiFi: 2.4 GHz hỗ trợ chuẩn 802.11 b/g/n.
* Điện áp hoạt động: 3.3V.
* Điện áp vào: 5V thông qua cổng USB.
* 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.
* Giao tiếp: Cable Micro USB.
* Hỗ trợ bảo mật: WPA/WPA2.
* Tích hợp giao thức TCP/IP.
* Lập trình trên các ngôn ngữ: C/C++, Micropython, NodeMCU – Lua.
* PinOut:



Hình 2. 3 Sơ đồ chân NodeMCU

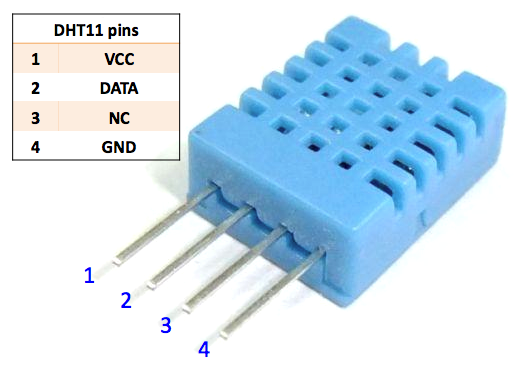
1. **Cảm biến độ ẩm, nhiệt độ DHT11**

DHT11 là cảm biến nhiệt độ,độ ẩm được tích hợp trong một mạch duy nhất. DHT11 có giá thành rẻ, dễ sử dụng,thích hợp trong các ứng dụng yêu cầu độ chính xác không cao,môi trường không khắc nghiệt. 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.

*Thông số kỹ thuật:*

* Điện áp hoạt động: 3V - 5V (DC)
* Dải 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

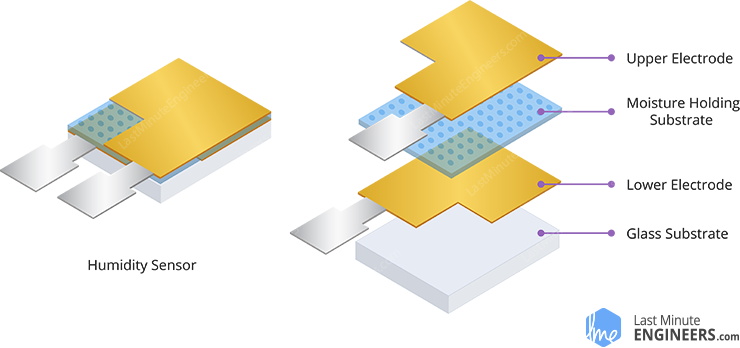
*Sơ đồ chân:*



Hình 2. 4 Sơ đồ chân DHT11

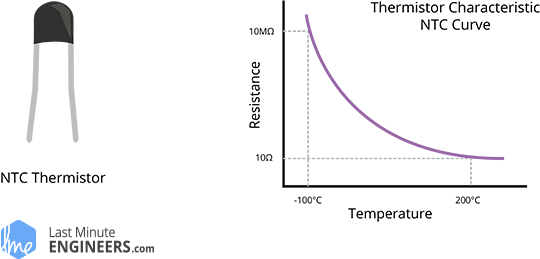
* Cấu tạo cảm biến DHT11: DHT11 được tích hợp hai cảm biến nhiệt dộ và độ ẩm

- Cảm biến độ ẩm



Hình 2. Cấu tạo của cảm biến độ ẩm

Cảm biến độ ẩm được cấu tạo bởi 2 lớp điện cực và nằm ở giữa là một lớp polime giữ ẩm, dẫn điện.Khi độ ẩm tăng các ion trong lớp polime được giải phóng làm tăng độ dẫn điện giữa các điện cực . Điện trở giữa hai cực tỉ lệ nghịch với độ ẩm.

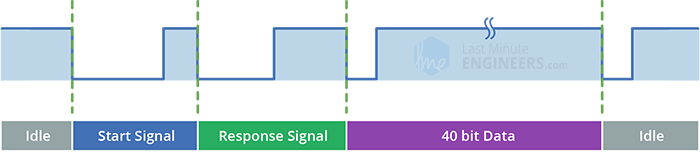


Hình 2. Cấu tạo của biến trở nhiệt độ

Bên trong DHT11 còn có một nhiệt điện trở NTC.Giá trị điện trở NTC tỉ lệ thuận với nhiệt độ ( 100 Ôm /1 độ C). Ngoài ra bên trong nó còn có 1 IC chuyển dổi tín hiệu tương tự sang tín hiệu số.

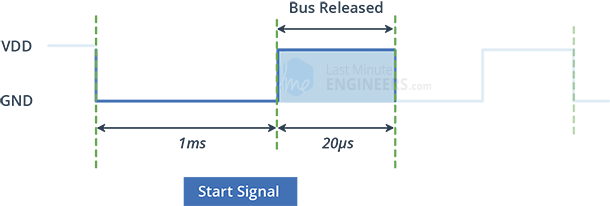
* *DHT11 hoạt động như thế nào?*

Khi cấp nguồn cảm biến cần 2 giây để ổn định. Trong giai đoạn này cảm biến kiểm tra giá trị nhiệt độ,độ ẩm môi trường và chuyển sang chế độ ngủ. Khi MCU gửi tín hiệu bắt đầu ,cảm biến chuyển sang chế độ tốc độ cao và gửi tín hiệu phản hồi.Theo đó nó sẽ gửi một chuỗi dữ liệu gồm 40bit gồm các giá trị nhiệt độ ,độ ẩm tương đối.Sau khi kết thúc cảm biến sẽ tự trở về chế độ ngủ .



Hình 2. Quá trình truyền dữ liệu của cảm biến DHT11

* **Start Signal:**

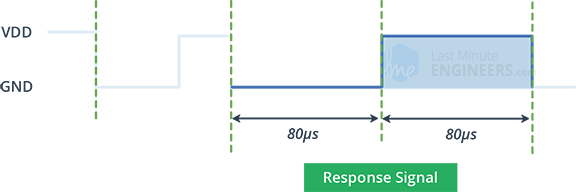


Hình 2. Start Signal

Để yêu cầu đọc, MCU kéo dòng xuống mức thấp khoảng 1ms và dòng được kéo lên cao trong khoảng 20µs có tác dụng đánh thức DHT ở chế độ ngủ.

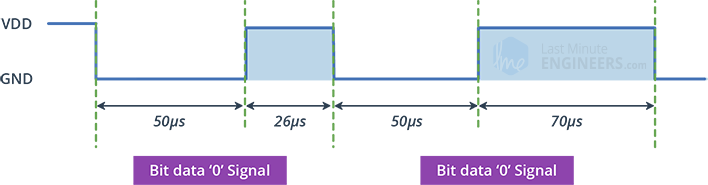
* Response Sinal:

Khi tín hiện bắt đầu kết thúc cảm biến sẽ gửi tín hiệu phản hồi. Cảm biến kéo dòng xuống mức thấp và cao trở lại trong khoảng thời gian 80µs. Trong thời gian này, cảm biến sẽ kiểm tra lại nhiệt độ, độ ẩm môi trường ghi lại dữ liệu tương đối và sẵn sàng cho việc truyền dữ liệu.



Hình 2. Tín hiệu phải hồi

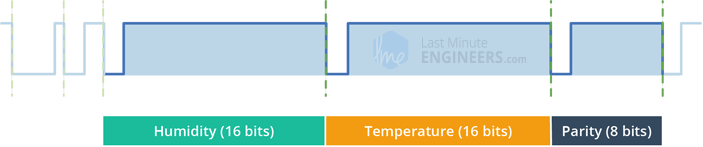
* 40bit Data:



Hình 2. Các bit dữ liệu truyền nhận trong DHT11

Khi tín hiệu phản hồi kết thúc, cảm biến bắt đầu suất dữ liệu nối tiếp 40 bit liên tục. Trước khi gửi dữ liệu cảm biến sẽ kéo dòng xuống mức thấp. Bit dữ liệu 0 hay 1 tùy thuộc vào khoảng thời gian kéo dòng ở mức cao. Với 26µs là mức data 0 còn 70µs là mức data 1. Khi tất cả 40 bit được truyền đi cảm biến sẽ kéo dòng xuống mức thấp trong khoảng thời gian 50µs và chuyển sang chế độ ngủ.

* **Cách đọc nhiệt độ, độ ẩm từ dữ liệu 40 bit**



Hình 2. Dữ liệu nhiệt độ, độ ẩm

Trong 40 bit dữ liệu cảm cảm biến DHT gửi đi chứa 16 bit độ ẩm,16 bit nhiệt độ và 8 bit cuối cùng có chức năng kiểm tra:

|  |  |  |
| --- | --- | --- |
| **0000 0010 1001 0010** | **0000 0001 0000 1101** | **1010 0010** |
| **Relative Humidity** | **Relative Temperature** | **Checksum** |

Giá trị của bit kiểm tra là tổng của 4 byte đầu tiên, điều này chỉ ra rằng dữ liệu đã nhận là hợp lệ.

**0000 0010 + 1001 0010 + 0000 0001 + 0000 1101 = 1010 0010**

Để có được giá trị nhiệt độ, độ ẩm tương đối ta cần chuyển số nhị phân 16 bit thành thập phân. Ví dụ như sau:

**Relative Humidity => 0000 0010 1001 0010(BIN) => 65.8 (% RH)**

**Relative Temperature => 0000 0001 0000 1101(BIN) => 26.9 (°C)**

1. **Cảm biến ánh sáng**
2. Cảm biến ánh sáng là gì?

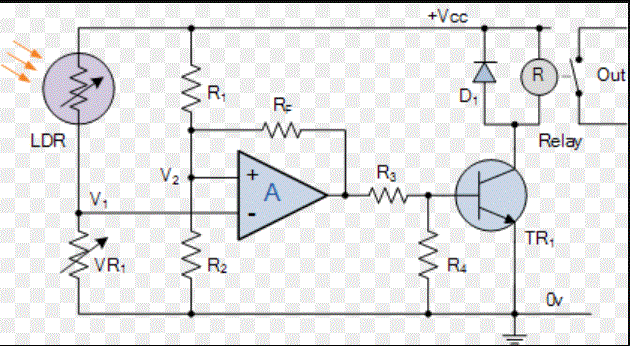
Cảm biến ánh sáng là một thiết bị thụ động chuyển đổi “năng lượng ánh sáng” này cho dù có thể nhìn thấy hoặc trong các phần hồng ngoại của quang phổ thành tín hiệu điện. Cảm biến ánh sáng thường được gọi là “Thiết bị quang điện” hoặc “Cảm biến ảnh” bởi vì năng lượng ánh sáng chuyển đổi (photon) thành điện (electron).

Các thiết bị quang điện có thể được nhóm lại thành hai loại chính, những loại tạo ra điện khi chiếu sáng, chẳng hạn như Photo-voltaics hoặc Photo-emissives vv, và những thứ thay đổi tính chất điện của chúng theo một số cách như Photo-resistors hoặc Photo-conductors.

1. Tế bào quang điện:

Một cảm biến ánh sáng quang điện không tạo ra điện nhưng chỉ thay đổi các tính chất vật lý của nó khi phải chịu năng lượng ánh sáng. Loại phổ biến nhất của thiết bị quang điện là điện trở Photores mà thay đổi điện trở của nó để đáp ứng với những thay đổi cường độ ánh sáng.

Photoresistors là các thiết bị bán dẫn sử dụng năng lượng ánh sáng để điều khiển dòng điện tử, và do đó dòng điện chạy qua chúng. Tế bào quang điện thường được sử dụng được gọi là điện trở phụ thuộc ánh sáng hoặc LDR.

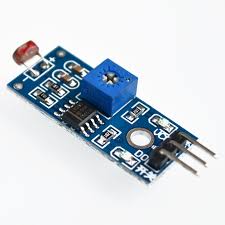


Hình 2. Nguyên lý hoạt động của cảm biến ánh sáng

1. Module cảm biến ánh sáng quang trở CDS

Ngõ ra D0 trên cảm biến được dùng để xác định cường độ sáng của môi trường, khi ở ngoài sáng, ngõ ra D0 là giá trị 0, khi ở trong tối, ngõ ra D0 là 1. Trên cảm biến có 1 biến trở để điều chỉnh cường độ sáng phát hiện, khi văn cùng chiều kim đồng hồ thì sẽ làm giảm cường độ sáng nhận biết của cảm biến, tức là môi trường phải ít sáng hơn nữa thì cảm biến mới đọc gía trị digital là 1.

Cảm biến ánh sáng quang trở có tích hợp sẵn opamp và biến trở so sánh mức tín hiệu giúp cho việc nhận biết tín hiệu trở nên dễ dàng, sử dụng để nhận biết hay bật tắt thiết bị theo cường độ ánh sáng môi trường.

****

Hình 2. 13: Cảm biến ánh sáng quang trở CDS

**Thông số kỹ thuật:**

* Điện áp làm việc: 3.3 ~ 5VDC
* Output: Digital
* Có thể điều chỉnh cường độ ánh sáng phát hiện bằng biến trở gắn trên cảm biến
* Kích thước: 3.2cm x 1.4cm

### **2.2.4. Cảm biến chuyển động**

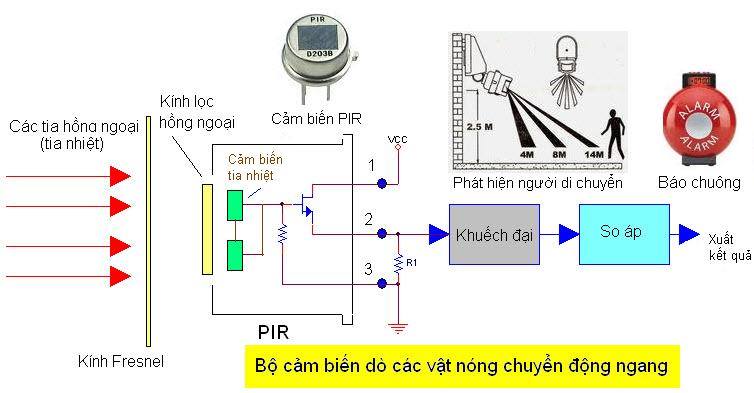
1. Cảm biến chuyển động hồng ngoại PIR là gì?

Cảm biến chuyển động thực chất là cảm biến hồng ngoại thụ động PIR (chữ viết tắt của Passive InfraRed sensor (PIR sensor)), tức là bộ cảm biến thụ động dùng nguồn kích thích là tia hồng ngoại. Tia hồng ngoại (IR) chính là các tia nhiệt phát ra từ các vật thể nóng. Trong các cơ thể sống, trong chúng ta luôn có thân nhiệt (thông thường là ở 37 độ C), và từ cơ thể chúng ta sẽ luôn phát ra các tia nhiệt, hay còn gọi là các tia hồng ngoại, người ta sẽ dùng một tế bào điện để chuyển đổi tia nhiệt ra dạng tín hiệu điện và nhờ đó mà có thể làm ra cảm biến phát hiện các vật thể nóng đang chuyển động. Bản thân công tắc cảm biến không phát tia hồng ngoại, nó chỉ nhận tia hồng ngoại phát ra từ thân thể người (hoặc nguồn nhiệt bất kỳ), sau đó phân tích để xác định điều kiện báo động. Vậy nên nó “thụ động”, nó chỉ nhận chứ không phát tia hồng ngoại.

1. Nguyên lý hoạt động của cảm biến hồng ngoại chuyển động

Các cảm biến PIR luôn có sensor (mắt cảm biến) với 2 đơn vị (element). Sensor là các cảm biến, nó dùng để chuyển đổi các tín hiệu không thuộc điện (vd thân nhiệt) ra dạng tín hiệu điện và đưa vào các dạng mạch điện để xử lý. Chắn trước mắt sensor là một lăng kính (thường làm bằng plastic), chế tạo theo kiểu lăng kính fresnel. Lăng kính fresnel này có tác dụng chặn lại và phân thành nhiều vùng (zone) cho phép tia hồng ngoại đi vào mắt sensor. Chúng ta có thể hiểu đơn giản rằng, nếu không có lăng kính fresnel, toàn bộ bức xạ của môi trường sẽ chỉ coi như có 1 Zone dội hết vào mắt sensor, như vậy thì nó sẽ không có tác dụng phân biệt chuyển động, và sẽ cực kỳ nhạy với bất kỳ sự thay đổi nhiệt độ nào của môi trường.

Hai đơn vị của mắt sensor có tác dụng phân thành 2 điện cực. Một cái là điện cực dương (+) và cái kia là âm (-). Khi 2 đơn vị này được tuần tự kích hoạt (cái này xong rồi mới đến cái kia) thì sẽ sinh ra một xung điện, xung điện này kích hoạt sensor (alarm-báo động). Chính vì nguyên lý này, khi có người đi theo hướng vuông góc với khu vực kiểm soát của sensor (hướng mũi tên), thân nhiệt từ người này (bức xạ hồng ngoại) sẽ lần lượt kích hoạt từng đơn vị cảm biến và làm sensor báo động.



Hình 2. 14 Nguyên lý hoạt động của cảm biến chuyển động

Vậy thì cũng chính do nguyên lý này, nếu người chuyển động theo hướng song song, phát ra cùng lúc 2 luồng bức xạ qua lăng kính fresnel đập vào đồng thời 2 đơn vị cảm biến, xung điện không tạo ra, và lúc này sensor không hề báo động. Do đó nguyên tắc là phải đặt mắt sensor hướng vuông góc với hướng khả dĩ nhất có thể di chuyển. Như vậy mới tăng xác suất chính xác.

1. Các lưu ý cần tránh khi lắp đặt cảm biến PIR

* Không hướng mắt sensor vào nơi có nhiệt độ cao, tia bức xạ hồng ngoại của nó phát ra sẽ gây nhiễu cảm biến, khiến nó hoạt động không chính xác.
* Không hướng trực tiếp mắt sensor về nơi nhiều nắng mặt trời. Tia mặt trời có nhiều bức xạ hồng ngoại, khiến sensor bị nhiễu.
* Không nên đặt sensor gần dây điện nguồn. Cảm biến PIR là một thiết bị điện tử, hoạt động ở điện áp thấp, nên hạn chế đặt gần điện nguồn cao áp.

1. Module cảm biến chuyển động HC SR501

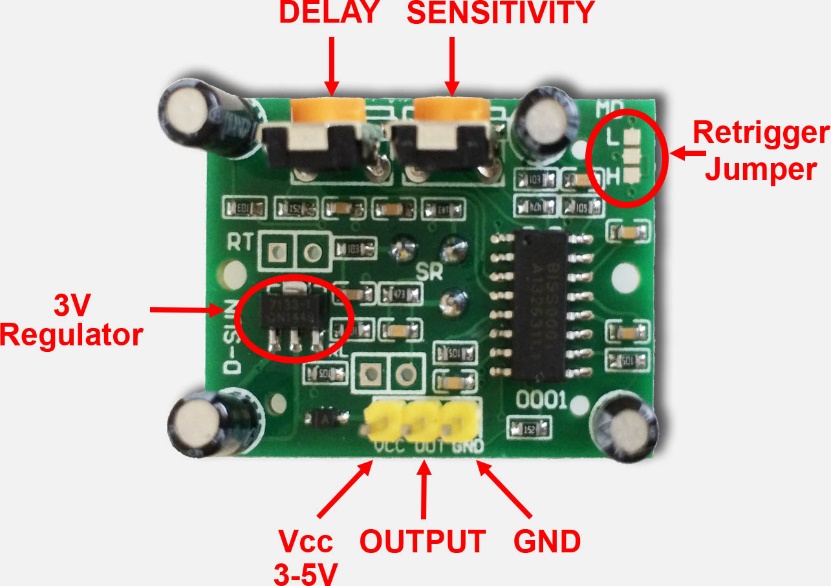


Hình 2. 15 Module cảm biến chuyển động HC SR501

*Thông số kỹ thuật*

* Sử dụng điện áp: 4.5V - 20V DC
* Điện áp đầu ra: 0V - 3.3V DC
* Có 2 chế độ hoạt động:
  + (L) không lặp lại kích hoạt
  + (H) lặp lại kích hoạt
* Thời gian trễ: điều chỉnh trong khoảng 0.5-200S
* Góc quét < 100 độ
* Sử dụng cảm biến: 500BP
* Khoảng các phát hiện: 2m - 4.5m
* Kích thước PCB:32mm x 24mm

*Sơ đồ chân*



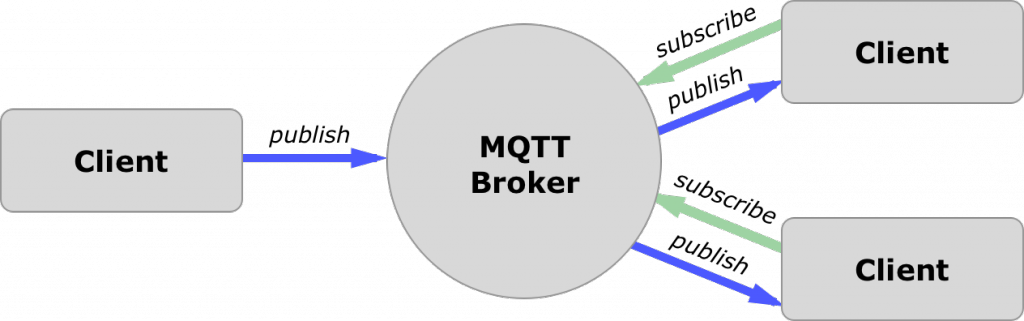
Hình 2. 16 Sơ đồ chân của module cảm biến chuyển động

1. **Giao thức truyền tài dữ liệu MQTT**
   * 1. **MQTT là gì?**

MQTT là một trong những giao thức được sử dụng phổ biến nhất trong các dự án IoT. Đây là một giao thức truyền thông điệp sử dụng băng thông thấp, độ tin cậy cao và có khả năng hoạt động trong điều kiện đường truyền không ổn định.

Ngoài ra, nó được thiết kế như một giao thức nhắn tin nhẹ sử dụng các hoạt động publish/subscribe để trao đổi dữ liệu giữa khách hàng và máy chủ (clients and server). Hơn nữa, kích thước nhỏ, mức sử dụng năng lượng thấp, các gói dữ liệu được tối thiểu hóa và dễ thực hiện khiến cho giao thức trở nên phổ biến trong các giao thức truyền tải dữ liệu hiện nay.

Kiến trúc mức cao (high-level) của MQTT gồm 2 phần chính là Broker và Clients.



Hình 2. Mô hình giao thức MQTT

* + 1. **Ưu điểm của giao thức MQTT?**

MQTT có các tính năng độc đáo mà bạn khó có thể tìm thấy trong các giao thức khác như:

* Đó là một giao thức nhẹ. Vì vậy, thật dễ dàng để thực hiện trong phần mềm và nhanh chóng trong việc truyền dữ liệu.
* Nó dựa trên một kỹ thuật nhắn tin. Tất nhiên, bạn biết tốc độ gửi tin nhắn của bạn nhanh như thế nào.
* Gói dữ liệu tối thiểu hóa. Do đó, sử dụng băng thông thấp.
* Sử dụng điện năng thấp. Kết quả là, nó tiết kiệm pin của thiết bị được kết nối.
* Đó là thời gian thực! Đó là đặc biệt những gì làm cho nó hoàn hảo cho các ứng dụng IoT.

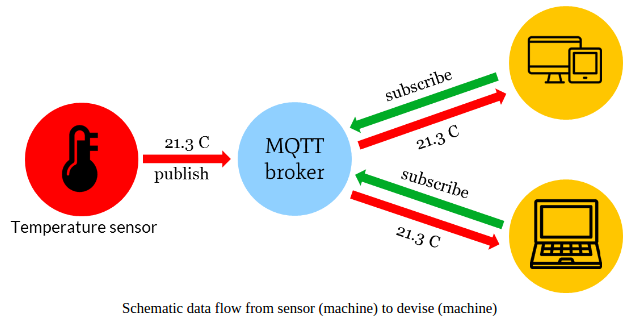
Trong một hệ thống sử dụng giao thức MQTT, nhiều client kết nối tới một server (Trong MQTT, server được gọi là MQTT Broker). Mỗi client sẽ đăng ký theo dõi các kênh thông tin (topic) hoặc gửi dữ liệu lên kênh thông tin đó. Quá trình đăng ký này gọi là “subscribe” và hành động một client gửi dữ liệu lên 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.

* + 1. **MQTT làm việc như thế nào?**

Giống như bất kỳ giao thức internet nào khác, MQTT dựa trên mô hình client và server. Server là người chịu trách nhiệm xử lý các yêu cầu nhận hoặc gửi dữ liệu của các Client với nhau.

Máy chủ MQTT được gọi là nhà môi giới và khách hàng chỉ đơn giản là các thiết bị được kết nối. Vì thế:

* Khi một thiết bị (client) muốn gửi dữ liệu cho broker (máy chủ MQTT), chúng tôi gọi hoạt động này là một ấn phẩm xuất bản.
* Khi một thiết bị (client) muốn nhận dữ liệu từ broker (máy chủ MQTT), chúng tôi gọi hoạt động này là một thuê bao đăng ký trực tuyến.



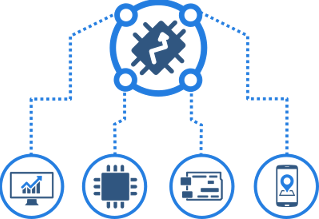
Hình 2. Nguyên lý hoạt động của MQTT

* + 1. **Các thành phần của MQTT?**

Trong MQTT có 5 thành phần cơ bản như sau:

* Broker: là máy chủ xử lý việc truyền dữ liệu giữa các máy khách.
* Topic: là nơi một thiết bị muốn đặt hoặc truy xuất thư gửi hoặc đến.
* Messager: là dữ liệu mà một thiết bị nhận được khi đăng ký tên lửa từ một chủ đề hoặc gửi đi khi xuất bản thành một chủ đề.
* Publish: là quá trình một thiết bị thực hiện để gửi tin nhắn của mình đến nhà môi giới.
* Subscribe: nơi một thiết bị làm để lấy tin nhắn từ broker.
  1. **Tổng quan về ThingsBoard IoT Platform**
     1. **Giới thiệu về ThingsBoard IoT Platform**

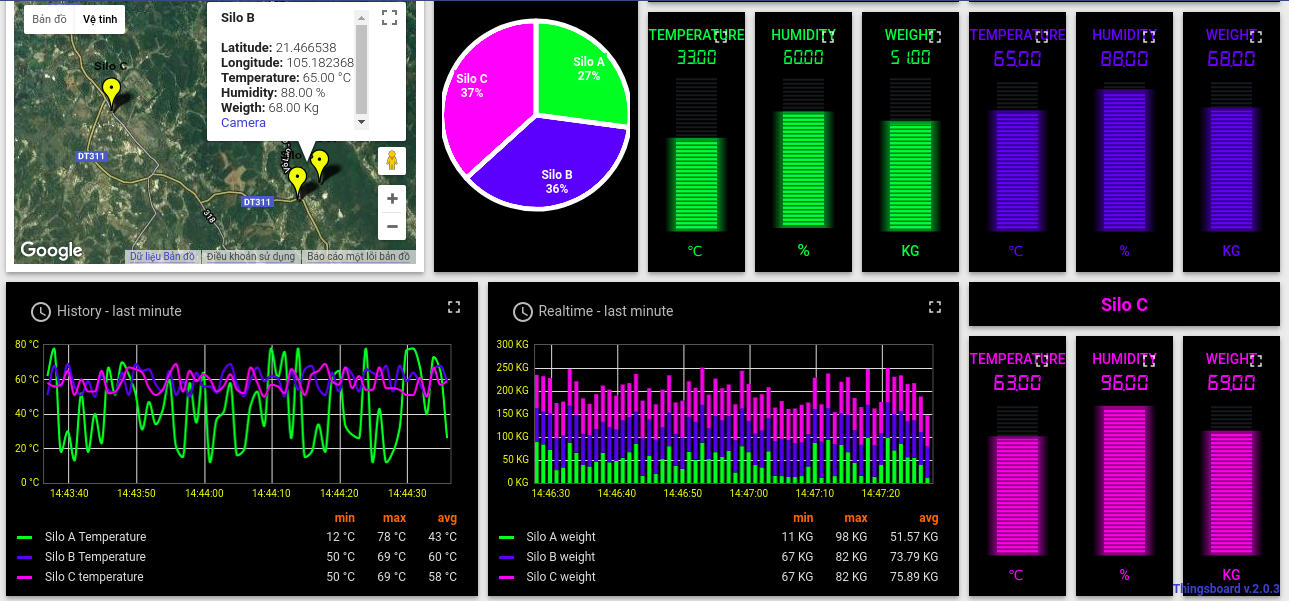
ThingsBoard là một nền tảng IoT mã nguồn mở. Nó cho phép phát triển nhanh chóng, quản lý và mở rộng các dự án IoT. Với nền tảng Thingsboard ta có thể thu thập, xử lý, hiển thị trực quan và quản lý thiết bị.



Hình 2. Sơ đồ hệ thống nền tảng ThingsBoard

Thingsboard cho phép kết nối thiết bị thông qua các giao thức IoT tiêu chuẩn công nghiệp – MQTT, CoAP và HTTP, hỗ trợ cả triển khai đám mây và tại chỗ. Ngoài ra ThingsBoard cho phép tích hợp các thiết bị được kết nối với các hệ thống cũ và bên thứ ba bằng các giao thức hiện có. Kết nối với máy chủ OPC-UA, MQTT broker, Sigfox Backend hoặc Modbus slaves chỉ trong vài phút bằng cách kết nối qua IoT Gateway (xem hình trên).

ThingsBoard cho phép ta tạo các Bảng điều khiển (Dashboard) IoT phong phú để hiển thị dữ liệu và điều khiển thiết bị từ xa trong thời gian thực. Ta có thể xây dựng một bảng điều khiển cho dự án nông trại thông minh để hiển thị trực quan các dữ liệu sản lượng, điều kiện thời tiết trong sản xuất nông nghiệp.



Hình 2. Giao diện ThingsBoard

ThingsBoard còn cho phép ta tạo Chuỗi quy tắc phức tạp để xử lý dữ liệu từ thiết bị của mình và phù hợp với các trường hợp sử dụng ứng dụng cụ.

* + 1. **Các tính năng của nền tảng ThingsBoard**
  1. *Thu thập dữ liệu từ xa*

Thingsboard sẽ hỗ trợ thu thập và lưu trữ dữ liệu từ xa theo cách đáng tin cậy. Ta có thể truy cập dữ liệu đã thu thập bằng cách sử dụng trang tổng quan web tùy chỉnh hoặc API phía máy chủ.

* 1. *Hiển thị trực quan dữ liệu đã thu thập*

Thingsboard cung cấp hơn 30 tiện ích có sẵn để ta sử dụng cho việc hiển thị trực quan các dữ liệu thu thập được của ta. Thingsboard cũng cho phép ta cũng có thể tạo các tiện ích riêng. Các gói tiện ích có sẵn như các tiện ích Google map, đồ thị thời gian thực, các thẻ HTML hiển thị.

* 1. *Công cụ tạo chuỗi quy tắc kéo thả thân thiện*

Xử lý dữ liệu thiết bị đến bằng chuỗi quy tắc linh hoạt dựa trên thuộc tính thực thể hoặc nội dung tin nhắn. Chuyển tiếp dữ liệu tới hệ thống bên ngoài hoặc kích hoạt báo thức bằng cách sử dụng logic tùy chỉnh. Định cấu hình chuỗi thông báo phức tạp trên các thông báo. Làm phong phú thêm chức năng phía máy chủ hoặc điều khiển thiết bị của ta bằng các quy tắc có thể tùy chỉnh cao. Ta có thể làm điều đó bằng cách sử dụng công cụ Rule Engine kéo thả thân thiện.

* 1. *Quản lý thiết bị*

Thingsboard cung cấp khả năng đăng ký và quản lý thiết bị(device). Nó cho phép theo dõi các thuộc tính thiết bị phía máy khách và cung cấp phía máy chủ. Cung cấp API cho các ứng dụng phía máy chủ để gửi các lệnh RPC tới các thiết bị và ngược lại.

* 1. *Quản lý các báo động*

Thingsboard cung cấp khả năng tạo và quản lý các cảnh báo liên quan đến các thực thể: device, asset, … Cho phép giám sát báo động theo thời gian thực và báo động cho việc phân cấp các thực thể liên quan.

* 1. *100% mã nguồn mở*

ThingsBoard được cấp phép theo Giấy phép Apache 2.0. Vì vậy có thể sử dụng bất kỳ sản phẩm nào trong các sản phẩm thương mại của mình miễn phí.

**CHƯƠNG III: XÂY DỰNG HỆ THỐNG VỚI THINGSBOARD IOT PLATFORM**

* 1. **Cài đặt, cấu hình hệ thống phần mềm**
     1. **Cài đặt ThingsBoard trên máy tính Ubuntu**

*Điều kiện tiên quyết*

Để cài đặt ThingsBoard trên Ubuntu Server 18.04 LTS. Yêu cầu phần cứng phải phụ thuộc vào cơ sở dữ liệu được chọn và số lượng thiết bị được kết nối với hệ thống. Để chạy ThingsBoard và PostgreSQL trên một máy duy nhất, sẽ cần ít nhất 1Gb RAM

*Bước 1: Cài đặt Java 8 (OpenJDK)*

Chạy lệnh sau để cài đặt OpenJDK 8:

$ sudo apt update

$ sudo apt install openjdk-8-jdk

Để cấu hình hệ điều hành sử dụng OpenJDK 8 theo mặc định. Có thể cấu hình phiên bản mặc định bằng lệnh sau:

$ sudo update-alternatives --config java

Kiểm tra cài đặt bằng lệnh sau:

$ java -version

Sau khi cài đặt thành công, cửa sổ Terminal sẽ hiển thị:

openjdk version "1.8.0\_xxx"

OpenJDK Runtime Environment (...)

OpenJDK 64-Bit Server VM (build ...)

*Bước 2: Cài đặt ThingsBoard*

Tải gói cài đặt:

Wget https://github.com/thingsboard/thingsboard/releases/download/v2.4.1/thingsboard-2.4.1.deb

Cài đặt ThingsBoard:

sudo dpkg -i thingsboard-2.4.1.deb

*Bước 3: Cấu hình cơ sở dữ liệu ThingsBoard*

Chạy lệnh sau để cài đặt PostgreSQL

sudo apt-get update

sudo apt-get install postgresql postgresql-contrib

sudo service postgresql start

Khi PostgreSQL được cài đặt, có thể tạo người dùng mới hoặc đặt mật khẩu cho người dùng chính.

sudo su - postgres

psql

\password

\q

Sau đó, nhấn “Ctrl + D” để trở về bảng điều khiển người dùng chính và kết nối với cơ sở dữ liệu để tạo bảng điều khiển DB:

psql -U postgres -d postgres -h 127.0.0.1 -W

CREATE DATABASE thingsboard;

\q

Chỉnh sửa tập tin cấu hình ThingsBoard

sudo nano /etc/thingsboard/conf/thingsboard.conf

Thêm các dòng sau vào tập tin cấu hình. Thay ***PUT\_YOUR\_POSTGRESQL\_PASSWORD\_HERE*** bằng mật khẩu người dùng postgres:

# DB Configuration

export DATABASE\_ENTITIES\_TYPE=sql

export DATABASE\_TS\_TYPE=sql

export SPRING\_JPA\_DATABASE\_PLATFORM=org.hibernate.dialect.PostgreSQLDialect

export SPRING\_DRIVER\_CLASS\_NAME=org.postgresql.Driver

export SPRING\_DATASOURCE\_URL=jdbc:postgresql://localhost:5432/thingsboardexport SPRING\_DATASOURCE\_USERNAME=postgres

export SPRING\_DATASOURCE\_PASSWORD=PUT\_YOUR\_POSTGRESQL\_PASSWORD\_HERE

*Bước 4. [Tùy chọn] Cập nhật bộ nhớ cho máy*

Chỉnh sửa tập tin cấu hình ThingsBoard:

sudo nano /etc/thingsboard/conf/thingsboard.conf

Thêm các dòng sau vào tập tin cấu hình.

# Update ThingsBoard memory usage and restrict it to 256MB in /etc/thingsboard/conf/thingsboard.conf

export JAVA\_OPTS="$JAVA\_OPTS -Xms256M -Xmx256M"

*Bước 5. Chạy script cài đặt*

Khi dịch vụ ThingsBoard được cài đặt và cấu hình DB được cập nhật, thực thi đoạn sau:

# --loadDemo option will load demo data: users, devices, assets, rules, widgets.

sudo /usr/share/thingsboard/bin/install/install.sh –loadDemo

*Bước 6. Bắt đầu dịch vụ ThingsBoard*

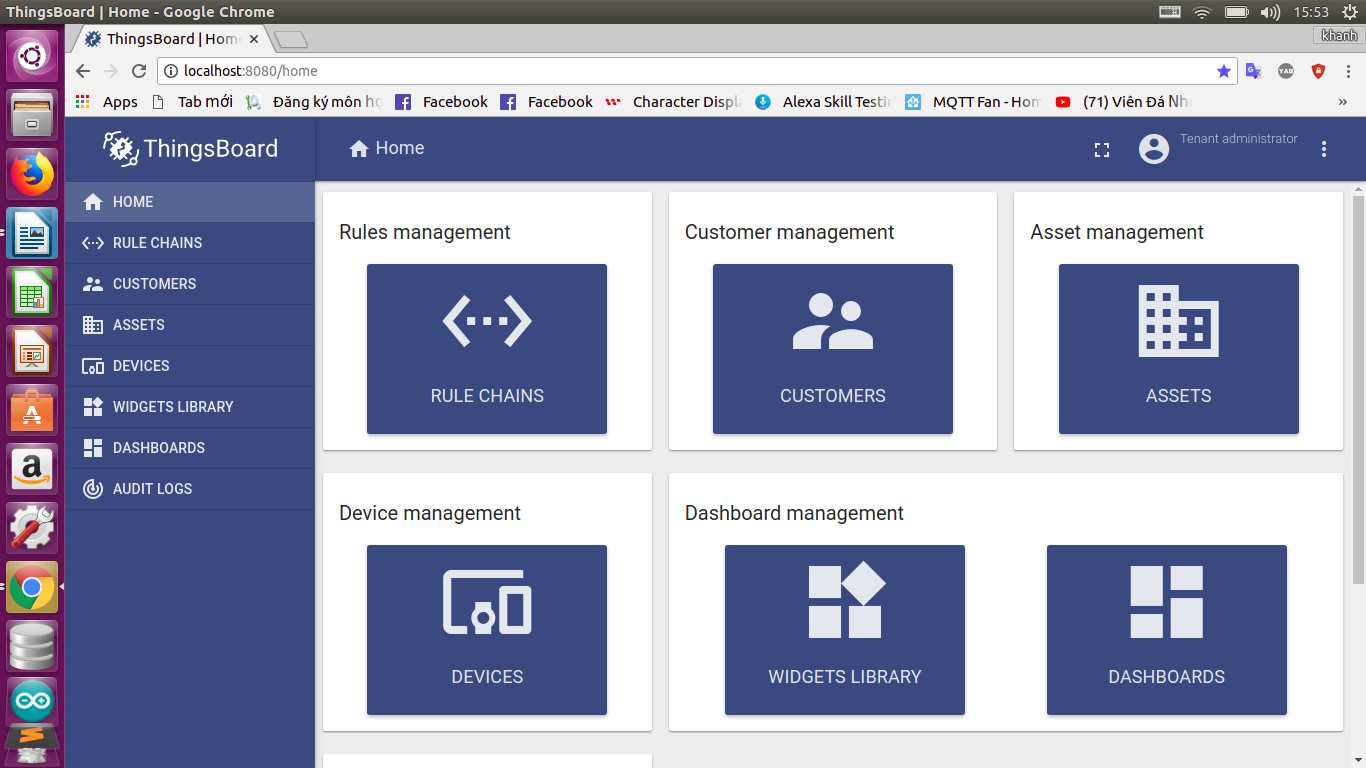
Thực hiện lệnh sau để bắt đầu ThingsBoard:

sudo service thingsboard start

Sau khi bắt đầu, ta sẽ có thể mở giao diện người dùng web bằng liên kết sau:

<http://localhost:8080/>

Login vào giao diện đầu tiên sẽ như sau:



Hình 3. Giao giện trang chủ ThingsBoard

*Chú ý:*

Có thể thay đổi các cấu hình MQTT cho ThingsBoard trong file thingsboard.yml

sudo nano /etc/thingsboard/conf/thingsboard.yml

Ta nhấn tổ hợp phím Ctrl+W để tìm kiếm từ “mqtt” rồi tìm đến phần bind\_port thay đổi port đó khác port 1883 và lưu lại và khởi động lại Thingsboard bằng câu lệnh sau:

sudo service thingsboard restart

* + 1. **Quyền quản lý trong Thingsboard**

Trong Thingsboard cung cấp ba tài khoản mặc định tương ứng với phân cấp quyền cơ bản trong Thingsboard là:

1. *Root*:sysadmin@thingsboard.org / sysadmin

Tài khoản cấp cao nhất trong Thingsboard có nhiệm vụ:

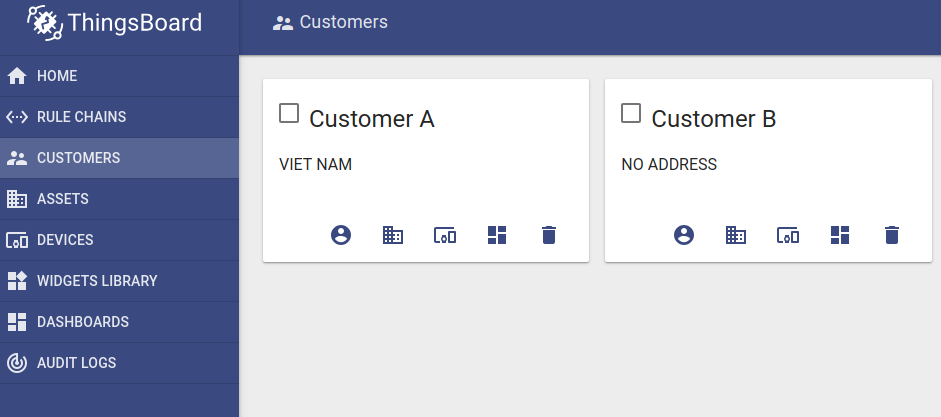
* Tạo các tài khoản khách hàng
* Có quyền được xóa, thêm các tài khoản.
* Là nơi nhận các thông báo từ các tài khoản khác như đổi mật khẩu, thay đổi email,…

1. *Admin:* tenant@thingsboard.org / tenant

Tài khoản người dùng có các chức năng chính:

Quyền làm chủ giao diện:

* Cung cấp và quản lý các Device
* Cung cấp và quản lý các Assets
* Tạo và quản lý các tài khoản khách(Customers)
* Tạo và quản lý các Dashboard
* Cấu hình chuỗi quy tắc (Rule Chain)
* Hoàn toàn có thể thay đổi Username cũng như password của tài khoản.



Hình 3. Phần quyền trong ThingsBoard

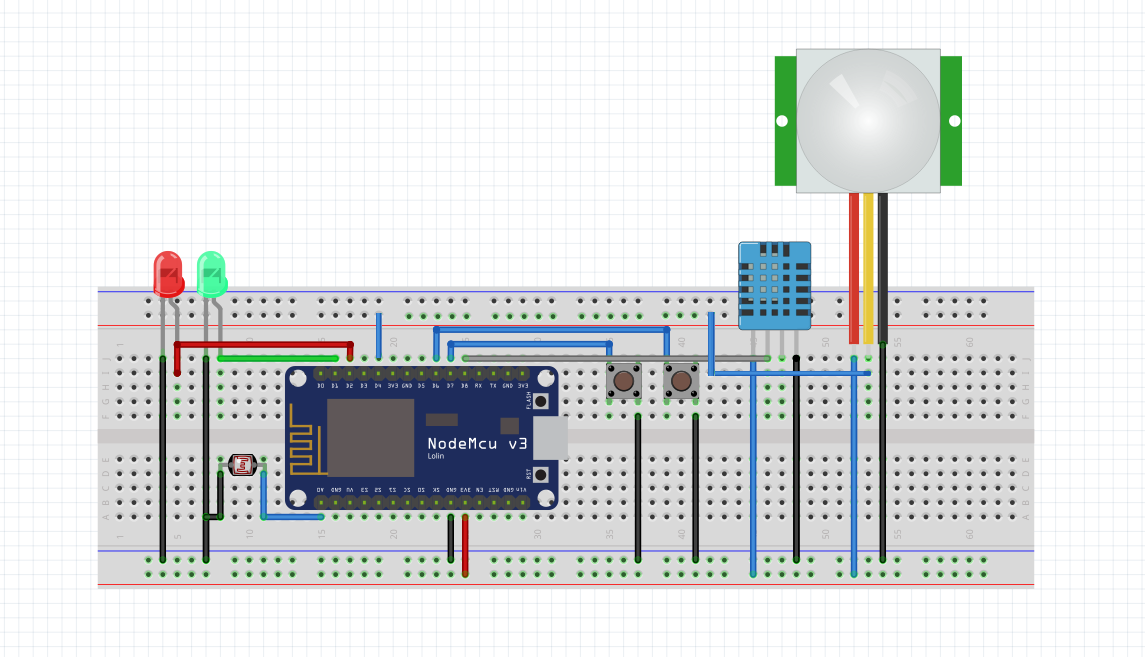
* + - 1. *Customer*: customer@thingsboard.org / customer

Tài khoản khách hàng chỉ có quyền xem các dữ liệu mà không thể thay đổi, thêm mới các device hay chỉnh sửa tài khoản. Ngoài tài khoản root là duy nhất, ta có thể tạo được nhiều các tài khoản admin và khách hàng tùy vào mục đích và đối tượng sử dụng.

* 1. **Truyền, nhận dữ liệu giữa các đèn với hệ thống Iot Gateway**
     1. **Sơ đồ kết nối hệ thống đèn đường thông minh**

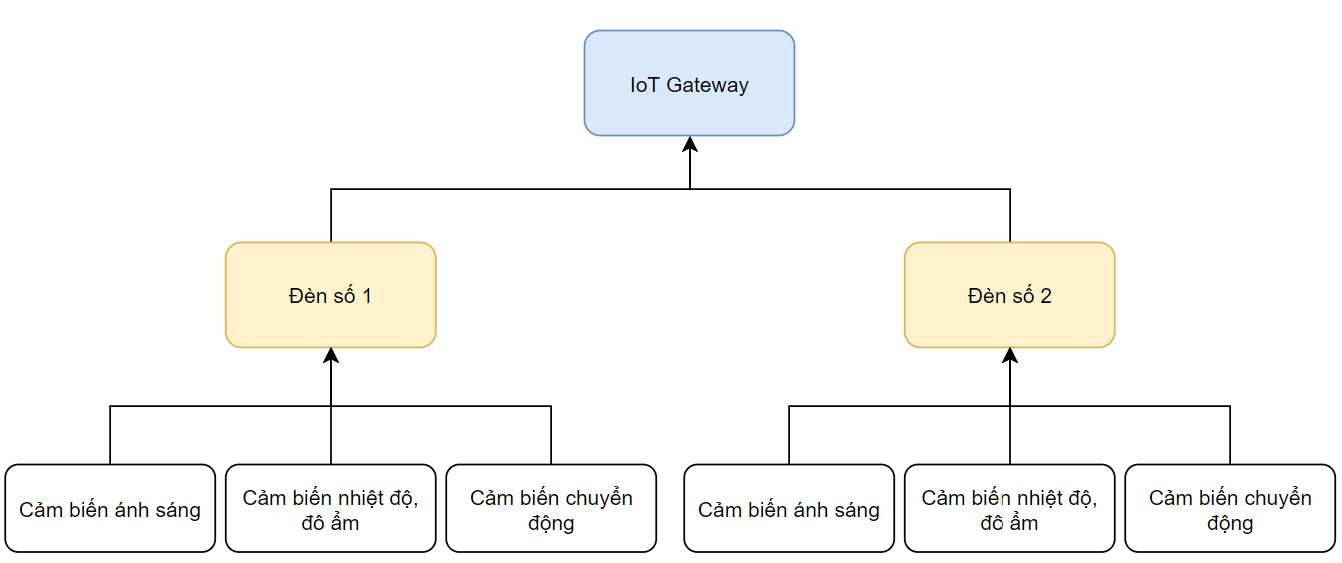
Trong hệ thống đèn đường thông minh, tại mỗi đèn sẽ được tích hợp các cảm biến như nhiệt độ, độ ẩm, cảm biến ánh sáng, cảm biến chuyển động. Dữ liệu được tiếp nhận đến các điểm chung chuyển là các Gateway trước khi được gửi dữ liệu lên Thingsboard thông qua MQTT

* Sơ đồ kết nối phần cứng:



Hình 3. Sơ đồ kết nối phần cứng

* Sơ đồ hệ thống

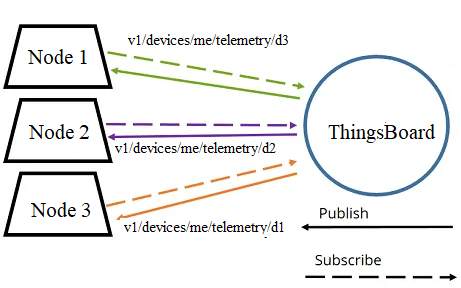


Hình 3. Sơ đồ hệ thống

Trong hệ thống, Node MCU đóng vai trò là một MQTT Broker có chức năng nhận dữ liệu truyền từ các đèn lên hệ thống server hay ngược lại. Các thao tác bật, tắt đèn bằng tay được từ nút nhấn có trong các đèn ngoài việc bật trực tiếp đèn đó còn gửi một thông điệp cho server biết trạng thái cụ thể của thiết bị (việc này là tối quan trọng trong việc phản hồi thiết bị, kiểm tra trạng thái kết nối, …).

Người quản lý cũng có thể bật đèn từ giao diện web thông qua giao thức MQTT gửi đến các IoT Gateway, nó có trách nhiệm xử lý thông điệp được gửi từ server và gửi tín hiệu điều khiển xuống cho các đèn.

Việc quản lý các đèn đưa ra một hệ thống cụ thể như: Một đèn ở vị trí X (20.980938, 105.753209) được xác định bởi một ID cụ thể, server dựa vào ID đó để xác định thiết bị nào được bật cũng như trạng thái của nó.

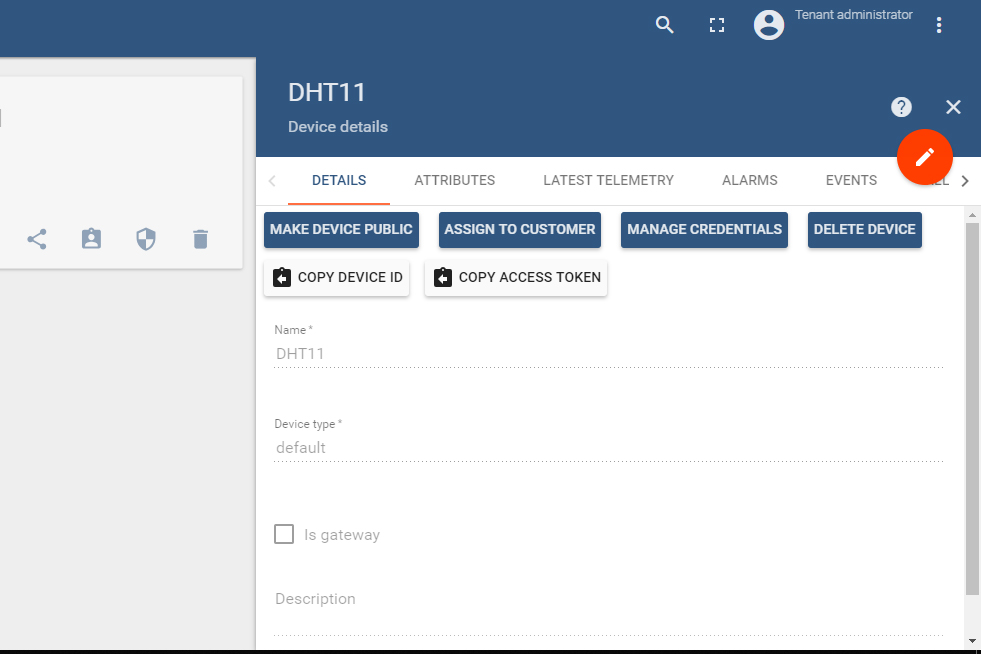


Hình 3. Sơ đồ kết nối Iot Gateway lên ThingsBoard server

* + 1. **Thiết lập hệ thống đèn đường lên giao diện điều khiển ThingsBoard**

1. *Khởi tạo thiết bị*

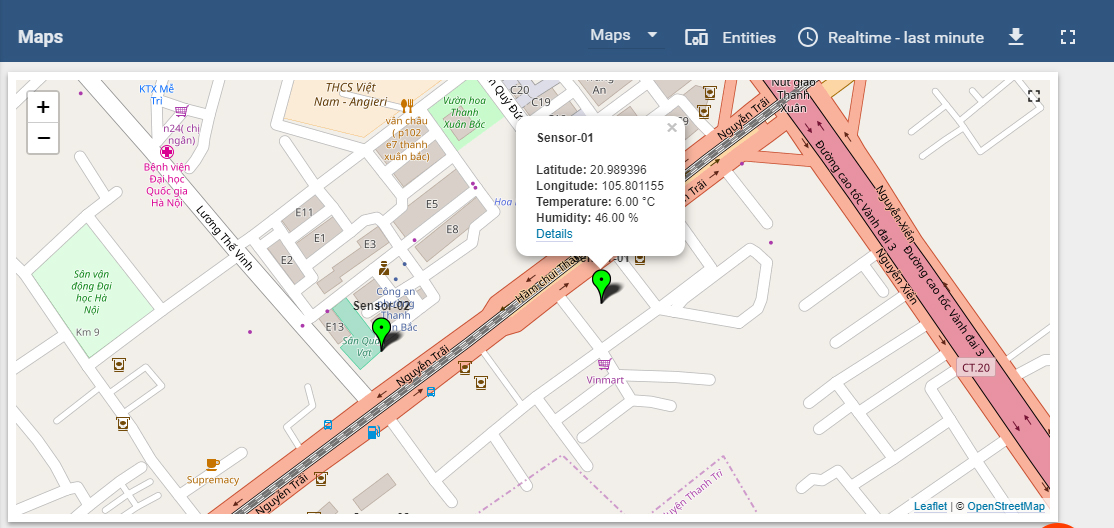
Trên giao diện ThingsBoard cần khởi tạo thiết bị tương ứng với các Gateway, mỗi Gateway được khởi tạo mặc định sẽ có một Access Token riêng để phân biệt giữa các Gateway với nhau.



Hình 3. Khởi tạo thiết bị

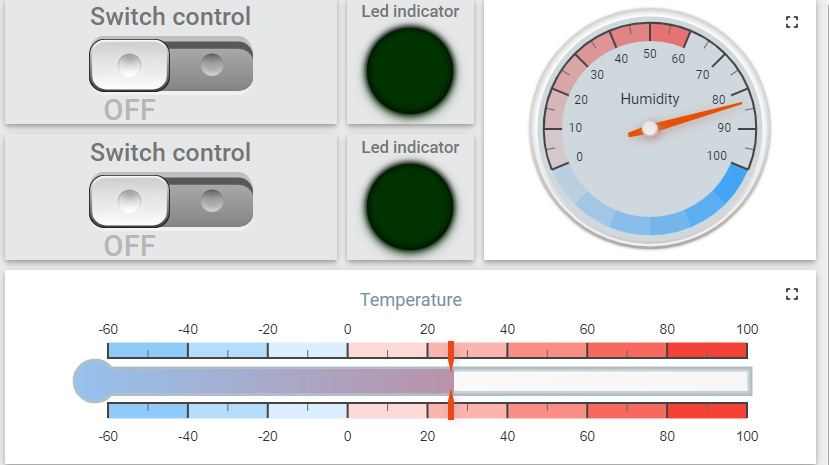
1. *Hiển thị vị trí lên đèn lên giao diện ThingsBoard*

Để hiển thị vị trí các đèn trên giao diện ThingsBoard giúp việc quản lý trở nên dễ dàng và tiện lợi.



Hình 3. Giao diện hiển thị tọa dộ từng đèn

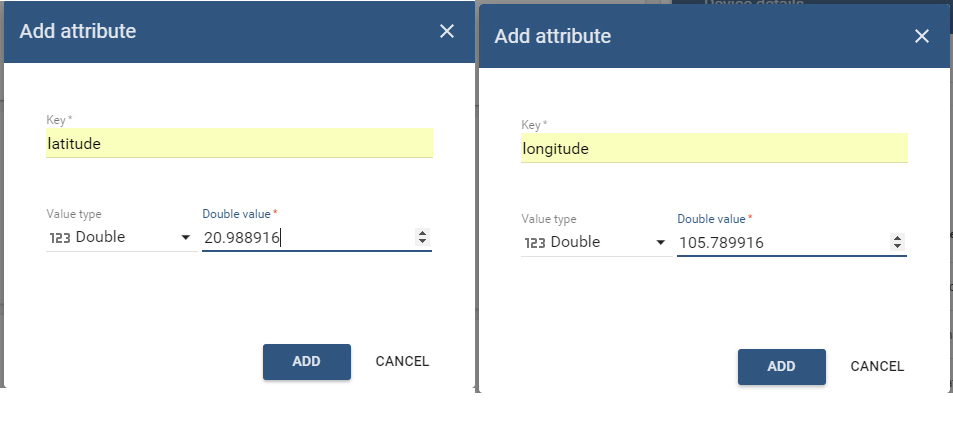
Khi người sử dụng click vào “Details” sẽ mở ra một trang quản lý riêng cho từng thiết bị cụ thể:



Hình 3. Giao diện chi tiết của từng node cảm biến

* Cấu hình tọa độ

Để cung cấp tọa độ cho các thiết bị ta vào phần ATTRIBUTES của mỗi thiết bị, phần entity attributes scope ta chọn Server attributes sau đó tích vào dấu “+” để thêm vĩ độ và kinh độ cho thiết bị.



Hình 3. Cài đặt tọa độ

1. *Xây dựng Dashboard hiển thị*

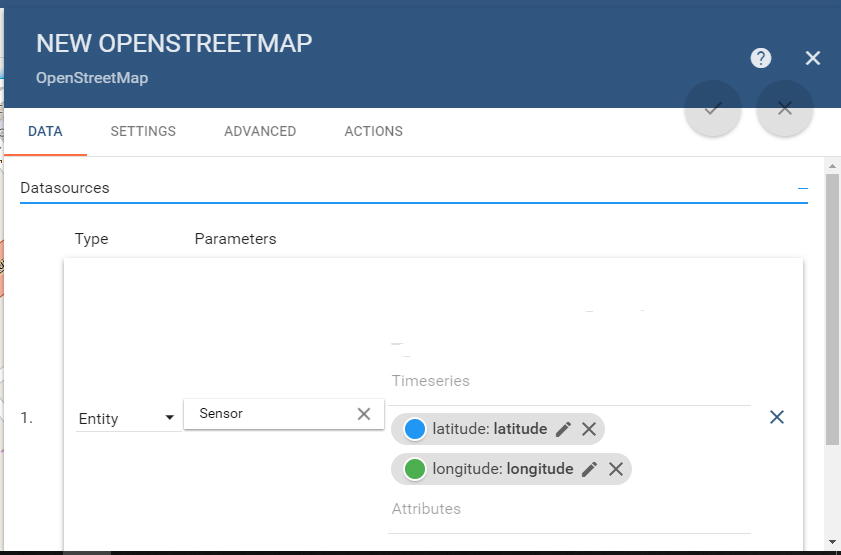
Sau khi đã cung cấp tọa độ cho các thiết bị. Ta sẽ tạo một giao diện để hiển thị ví trí trên bản đồ cũng như hiển thị các giá trị cảm biến đo được tại mỗi thiết bị. Giao diện sẽ có cấu trúc như sau:

* Một trang chính sẽ hiện bản đồ khi người sử dụng mới đăng nhập vàoThingsboard.
* Các trang khác tương ứng với các note cảm biến.

Cấu hình để hiện vị trí lên bản đồ

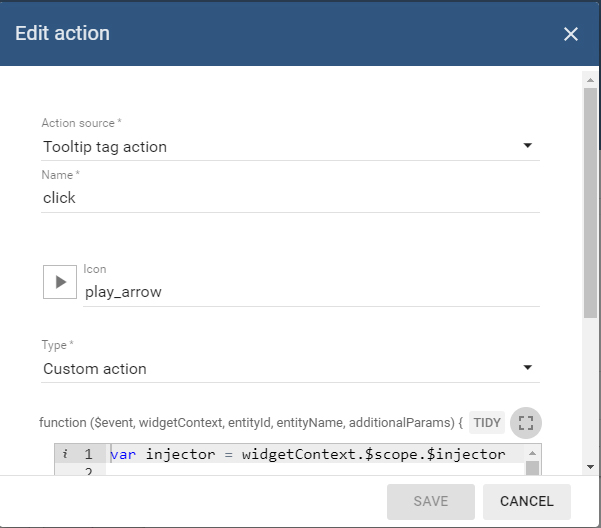
Ta sử dụng tiện ích OpenStreetMaps trong gói tiện ích Maps của Thingboard để hiển thị vị trí các cảm biến. Tiện ích này cần nguồn dữ liệu là một thực thể.

Ở đây nguồn dữ liệu này sẽ được lọc theo loại của thiết bị. Tức là tất cả các thiết bị mà có loại là sensor thì đều sẽ được truy xuất dữ liệu. Điều đó giúp ta có thể hiển thị tất cả các vị trí thiết bị lên bản đồ. Cấu hình giao diện như sau:



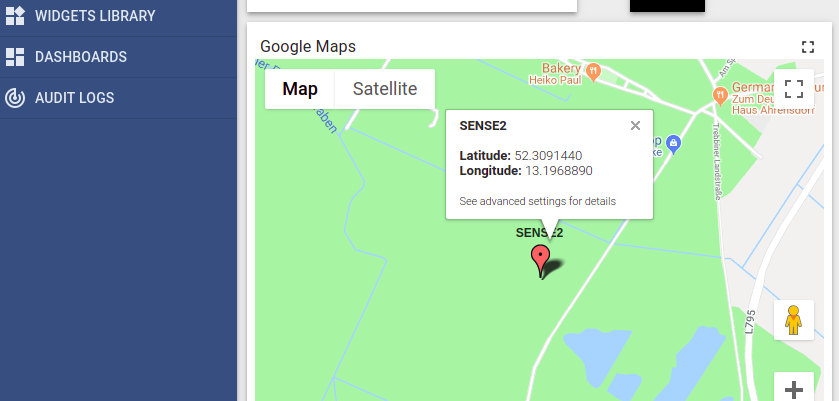
Hình 3. Giao diện cấu hình hiển thị thông số tọa độ

Sau đó vào tab ACTION=> nhấn dấu “+” và cấu hình như sau:



Hình 3. . Giao diện chuyển hướng

Lưu lại cấu hình ta được giao diện như hình:



Hình 3. Tọa độ của đèn sau khi được thiết lập

* 1. **Cấu hình giao diện hiển thị và bật, tắt đèn trên ThingsBoard**

Trong ThingsBoard quy định hai topic cơ bản:

* ATTRIBUTES: lưu trữ thông tin về đèn (tọa độ, tên, …)
* LATEST TELEMETRY: lưu trữ dữ liệu cảm biến và trạng thái đèn.

Các IoT Gateway sẽ kết nối với broker Thingsboard bằng user là $ACCESS\_TOKEN của thiết bị và không có mật khẩu. Dữ liệu thông qua MQTT sẽ được gửi với các thông tin cơ bản như sau:

1. *Telemetry*

* Topic: v1/devices/me/telemetry
* Username: $ACCESS\_TOKEN
* Password: null
* Port: 1883
* Message: {"key1":"value1", "key2":"value2"}

1. *Attributes*

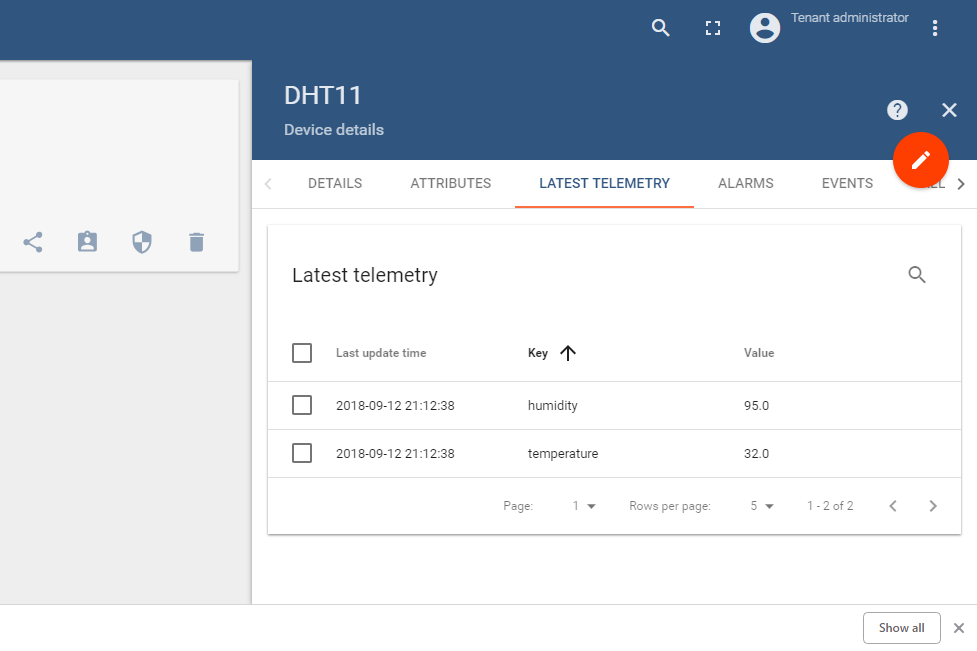
* Topic: v1/devices/me/attributes
* Username: $ACCESS\_TOKEN
* Password: null
* Port: 1883
* Message: {"key1":"value1", "key2":"value2"}

1. *Phản hồi từ phía IoT Gateway lên Server:*

* Topic: v1/devices/me/attributes/request/$request\_id
* Username: $ACCESS\_TOKEN
* Password: null
* Port: 1883
* Message: {"key1":"value1", "key2":"value2"}

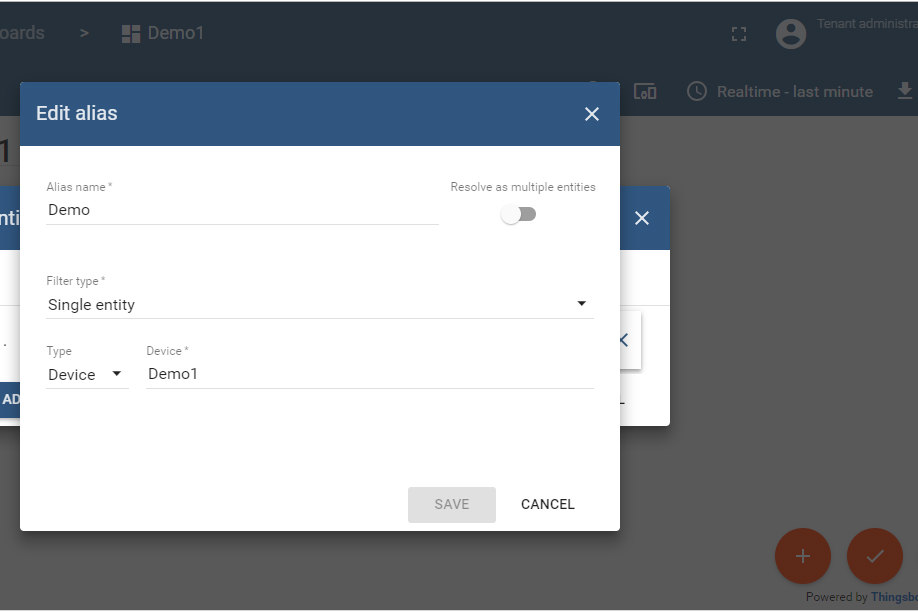
1. *Phản hồi từ Server xuống Iot Gateway*

* Topic: v1/devices/me/attributes/response/+
* Username: $ACCESS\_TOKEN
* Password: null
* Port: 1883
* Message: {"key1":"value1", "key2":"value2"}
  + 1. **Hiển thị dữ liệu cảm biến lên giao diện ThingsBoard**



Hình 4. Dữ liệu sau khi được dẩy lên ThingsBoard

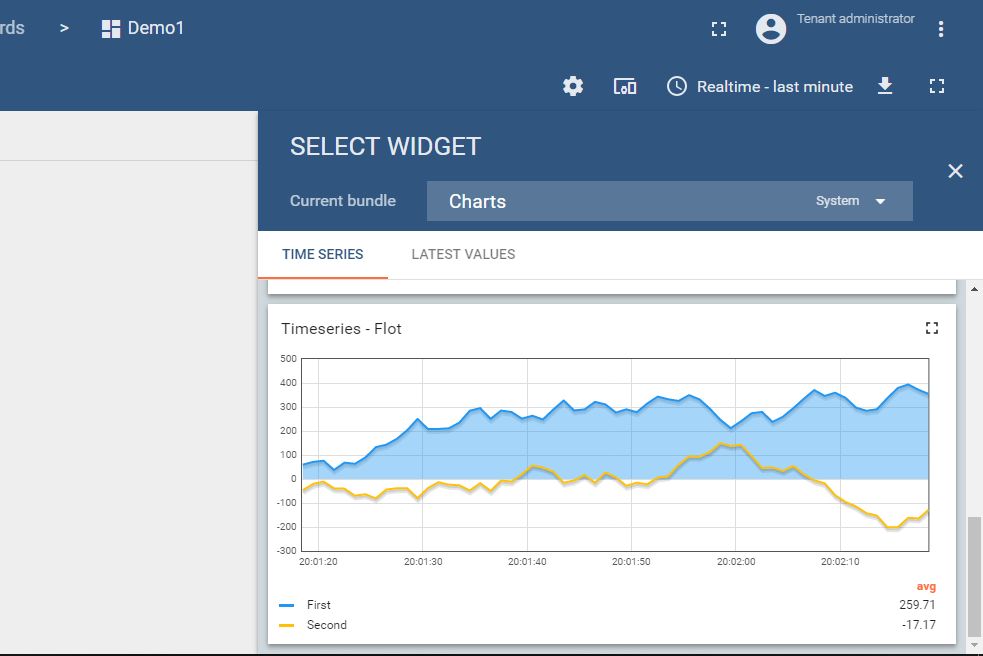
Sau khi nhận dữ liệu, để hiển thị dữ liệu đó trên giao diện web ta thực hiện các bước sau vào phần Entity aliases =>Add Alias =>Edit alias và cấu hình như bên dưới:



Hình 4. Thêm thiết bị vào giao diện

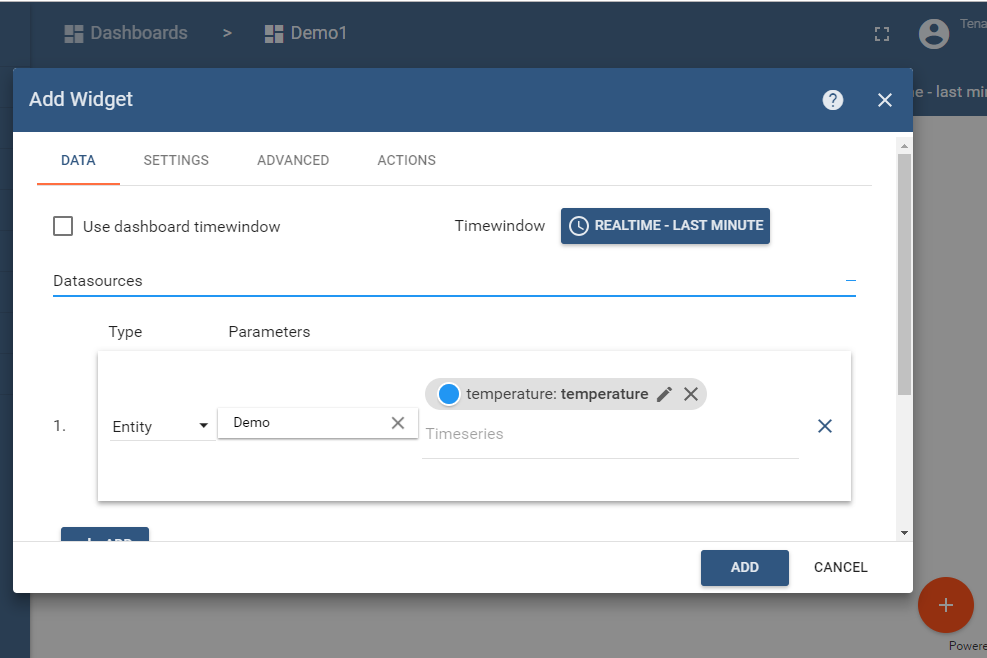
* Chọn các mẫu giao diện có sẵn:

ADD NEW WIDGET => Charts =>Timeseries – Flot.



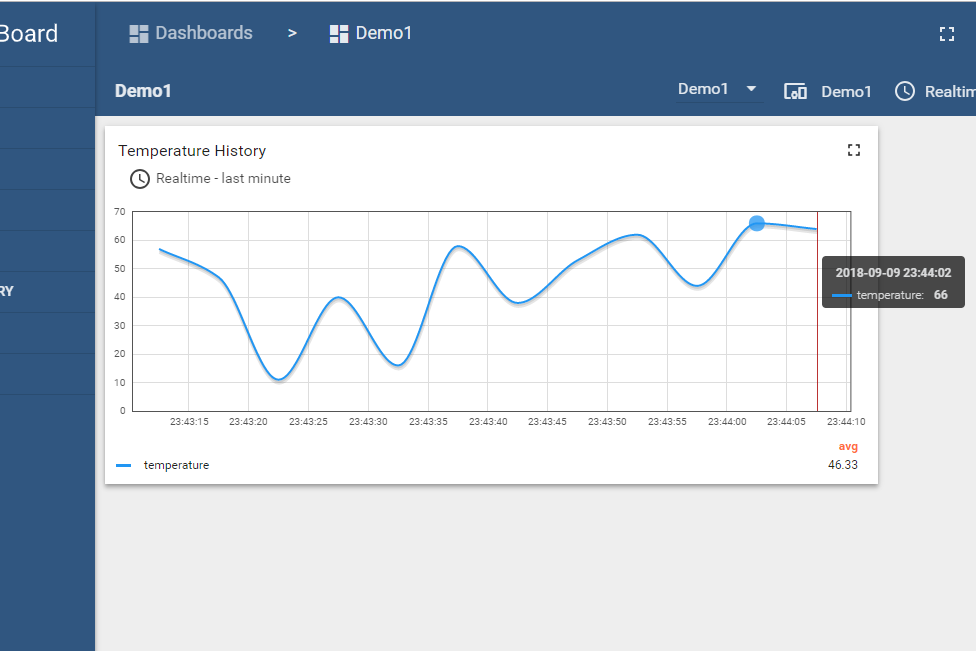
Hình 4. Chọn giao diện phù hợp để hiển thị

* Cấu hình cho mỗi giao diện như sau:



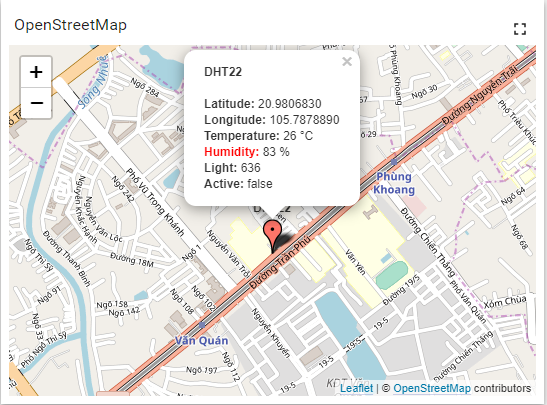
Hình 4. Cấu hình giao diện

* Nhấn ADD để hoàn tất. Dữ liệu đã được gửi lên như hình:



Hình 4. Giao diện hiển thị cảm biến

Làm tương tự cách làm ở bước trước, ta được giao diện hiển thị lên OpenStreetmap như sau:



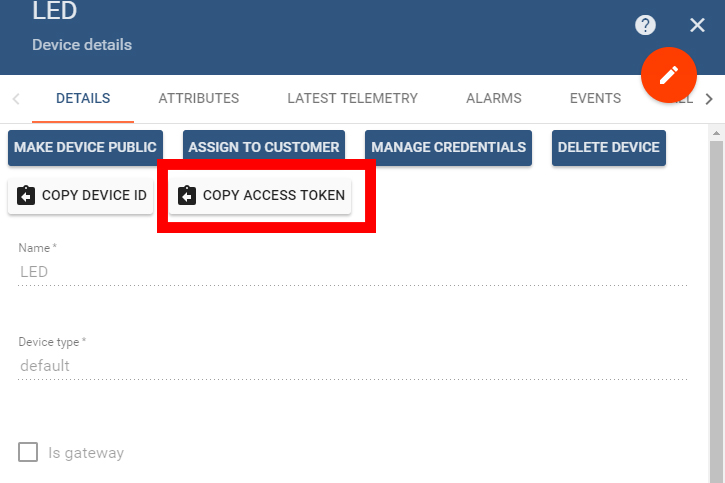
Hình 3. Hiển thị thông số từng đèn

* + 1. **Điều khiển đèn với ThingsBoard**
* Nguyên lý hoạt động

Iot Gateway sẽ kết nối đến Thingsboard bằng giao thức MQTT. Các gateway sẽ subscribe vào topic sau: “v1/devices/me/rpc/request/+” để nhận được tín hiệu điều khiển khi người sử dụng tác động trên giao diện web. Sau đó dựa vào tín hiệu nhận được đó để điều khiển trạng thái các đèn. Sau đó sẽ publish lại trạng thái của các đèn vào topic “v1/devices/me/attributes” để cập nhật trạng thái của nút nhấn trên Thingsboard.

* Cấu hình giao diện ThingsBoard

Trước tiên ta cần khởi tạo thiết bị trên ThingsBoard đã làm được ở bước trước và copy ACCESS TOKEN của thiết bị.



Hình 3. Accsess token của thiết bị

Sau đó tạo một Dashboard (bảng điều khiển) để cho người sử dụng điều khiển. Các tiện ích điều khiển cần được cung cấp một thiết bị đích. Trong trường hợp này thiết bị đích chính là thiết bị ảo (device) tạo trên.



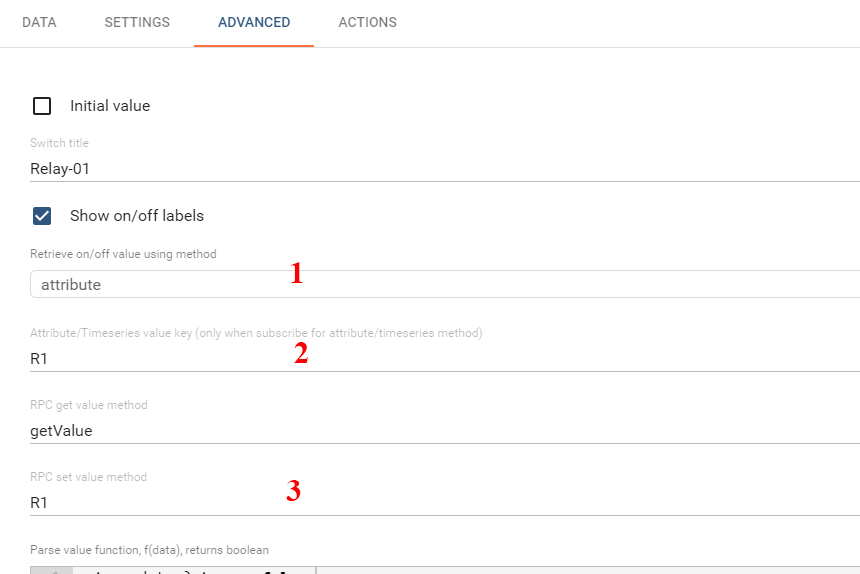
Hình 3. Khởi tạo nút nhấn

Để hiển thị trạng thái kết nối hay không kết nối ta chọn wiget sau:



Hình 3. Khởi tạo input trạng thái

Ta sử dụng một nút nhấn và đầu vào trạng thái tương ứng đối với mỗi đèn. Để kết nối giao diện với thiết bị vừa được khởi tạo ta thay đổi các cấu hình sau:



Hình 3. Cấu hình cho giao diện nút nhấn

Khi người sử dụng điều khiển bật hoặc tắt trên giao diện web thì Thingsboard sẽ gửi vào topic “v1/devices/me/rpc/request/+” một message có dạng như sau: {“method”: “x”, “params”: y} với:

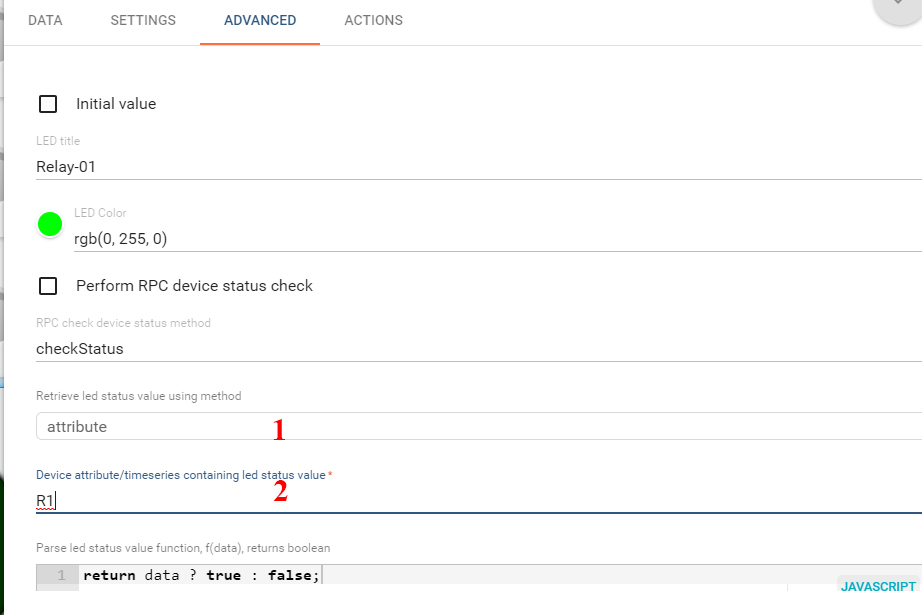
x: là giá trị bạn điền vào ô số 3 trên hình trên

y: là true hoặc false (true nếu bật false nếu tắt)

Thực chất Thingsboard server đang gửi một câu lệnh rpc (remote procedure call). Khi đó Thingsboard sẽ cập nhật trạng thái của nút nhấn trên giao diện dựa trên giá trị bạn điền vào ô số 2, và giá trị này phải được publish tới topic “v1/devices/me/attributes” nếu ô số 1 bạn chọn như hình. Nếu ở ô đó bạn chọn “timeseries” thì giá trị đó cần publish tới topic “v1/devices/me/telemetry”.

* Cấu hình nâng cao cho tiện ích LED Indicator

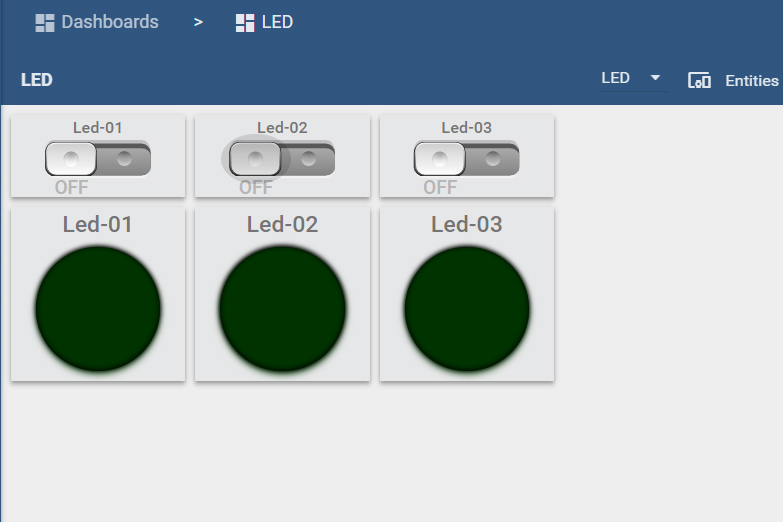
Với tiện ích LED Indicator, ta cài đặt các cấu hình sau:



Hình 3. Cấu hình cho giao diện Input

Trạng thái bật tắt của đèn này sẽ cập nhật dựa trên giá trị điền vào ô số 2 nếu như giá trị này được publish tới topic “v1/devices/me/attributes”

Sau khi đã tạo xong, ta được một dashboard có giao diện như sau:



Hình 3. Giao diện bật tắt đèn

# **CHƯƠNG IV: ĐÁNH GIÁ VÀ KẾT LUẬN**

Đề tài “Đèn đường thông minh” đã hoàn thành các nội dung trong đề cương được phê duyệt. Đề tài đã đạt được một số kết quả chính như sau:

* Hiểu rõ đặc điểm, tính năng và cấu trúc của hệ thống đèn đường thông minh.
* Thiết kế thành công mô hình hệ thống đèn đường thông minh
* Xây dựng hệ thống quản lý đèn đường bằng nền tảng ThingsBoard có chức năng:
  + Điều khiển bật, tắt đèn từ xa thông qua giao diện web.
  + Hiển thị các thông số môi trường đo được tại các cảm biến tích hợp trong hệ thống đèn.
  + Tự động bật đèn khi trời tối, tắt đèn khi trời sáng.
  + Tiết kiệm năng lượng (tắt đèn khi không có phương tiện giao thông đi lại).

**Hướng phát triển của đề tài:**

* Tích hợp với các thiết bi thông minh xây dựng hệ sinh thái IOT
* Tích hợp thêm các giao thức phù hợp trong thực tế như lora, rf, ..

# **TÀI LIỆU THAM KHẢO**

1. *T.S Phạm Thế Quế*, Cơ sở dữ liệu, Bài giảng Cơ sở dữ liệu, Học viện Cộng nghệ Bưu chính Viễn thông, Hà Nội, 2006.
2. *T.S Ngô Đức Thiện*, Điện tử tương tự, Bài giảng Điện tử tương tự, Học viện Cộng nghệ Bưu chính Viễn thông, Hà Nội, 2013.
3. Thingsboard Document, <https://thingsboard.io/>
4. Arduino Document, <http://arduino.vn/>

**PHỤ LỤC**

**Code Arduino:**

**#include <ESP8266WiFi.h> // thư vien dung cho esp8266**

**#include <PubSubClient.h> // thu vien mqtt**

**#include <ArduinoJson.h> // thu vien Arduin Json v5**

**#include "DHT.h" // thư vien DHT sensors**

**// Thông tin về wifi**

**#define ssid "lophocvui.com 2.4GHz"**

**#define password "88889999"**

**// Thông tin về MQTT Broker**

**#define mqtt\_server "demo.thingsboard.io"**

**#define mqtt\_port 1883**

**#define mqtt\_user "dLA8iuJeF5r9zUE83JNY" // user mqtt**

**#define mqtt\_pwd "" //password mqtt**

**#define mqtt\_topic\_sensor "v1/devices/me/telemetry" //topic nhan du lieu cam bien mac dinh cua ThingBoard**

**#define mqtt\_topic\_led\_response "v1/devices/me/attributes" //topic nhan du lieu bat tat den mac dinh cua ThingBoard**

**#define mqtt\_topic\_led\_request "v1/devices/me/rpc/request/+" //topic nhan du lieu bat tat den mac dinh cua ThingBoard**

**#define DeviceID "LHV-01" // id of station x**

**WiFiClient espClient;**

**PubSubClient client(espClient);**

**long now = millis();**

**long lastMeasure = 0;**

**/\*---DHT11---\*/**

**#define DHTPIN D8**

**#define DHTTYPE DHT11**

**DHT dht(DHTPIN, DHTTYPE);**

**/\*---Quang Tro---\*/**

**int quangtro = A0;**

**/\*------LED------\*/**

**#define LED1 D1**

**#define LED2 D2**

**//#define LED3 D8**

**#define BT1 D6**

**/\*----BUTTON-----\*/**

**int buttonStatus = 0;**

**int oldButton = 0;**

**int ledStatus = 0;**

**/\*----SetUp-----\*/**

**void setup(){**

**Serial.begin(9600);**

**pinMode(LED1,OUTPUT);**

**pinMode(LED2,OUTPUT);**

**pinMode(BT1,INPUT\_PULLUP);**

**digitalWrite(LED1,LOW);**

**digitalWrite(LED2,LOW);**

**setup\_wifi();**

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

**client.setCallback(callback);**

**dht.begin();**

**}**

**/\*----Loop-----\*/**

**void loop(){**

**// Kết nối wifi**

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

**reconnect();**

**}**

**client.loop();**

**// Khi nhấn nút sẽ bật đèn, đồng thời publish giá trị trạng thái của đèn lên server ThingsBoard**

**StaticJsonBuffer<200> jsonBuffer;//memory allocated on the stack and has a fixed size**

**JsonObject& data = jsonBuffer.createObject();//Allocates an empty JsonObject**

**char mesage[256];// mang ky tu**

**buttonStatus = digitalRead(BT1);**

**if(buttonStatus != oldButton){**

**oldButton = buttonStatus;**

**delay(50);**

**if(buttonStatus == 1){**

**ledStatus = !ledStatus; //đảo trạng thái led**

**if(ledStatus == 1){**

**Ctl\_Led(LED2,1);**

**data["R2"] = true;**

**data.printTo(mesage, sizeof(mesage));**

**client.publish(mqtt\_topic\_led\_response,mesage);**

**}**

**else{**

**Ctl\_Led(LED2,0);**

**data["R2"] = false;**

**data.printTo(mesage, sizeof(mesage));**

**client.publish(mqtt\_topic\_led\_response,mesage);**

**}**

**}**

**}**

**now = millis();**

**// cu 2s gui du lieu cảm biến 1 lần**

**if (now - lastMeasure > 2000){**

**lastMeasure = now;**

**Update\_data();**

**}**

**}**

**/\*\***

**\* @brief: fucntion callback of mqtt**

**\* @param topic: topic mqtt client subcrice to**

**\* mesage: message**

**\* length: len of message**

**\* @retval None**

**\*\*/**

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

**// Hien thi len Serial topic va message nhan duoc**

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

**String Topic = topic;**

**Serial.print(Topic);**

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

**String messageTemp;**

**Serial.println(messageTemp);**

**// gop message nhan duoc tu kieu byte sang string**

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

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

**}**

**Serial.println(messageTemp);**

**// convert message tu kieu String sang Objetct**

**StaticJsonBuffer<200> jsonBuffer;**

**JsonObject &data = jsonBuffer.parseObject(messageTemp);**

**// Kiem tra qua trinh convert co loi hay khong**

**if (!data.success()){**

**Serial.println("parseObject() failed");**

**return;**

**}**

**// Du lieu nhan duoc khi nhan nut tren giao dien ThingsBoard co dang {“method”:”x”,”params”:y} nen can so sanh method va params**

**String methodName = String((const char\*)data["method"]); //convert lai string de so sanh method**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Den 1\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**if(methodName.equals("R1")){**

**if(data["params"]==true){**

**Ctl\_Led(LED1,1);**

**}**

**else{**

**Ctl\_Led(LED1,0);**

**}**

**}**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Den 2\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**if(methodName.equals("R2")){**

**if(data["params"]==true){**

**Ctl\_Led(LED2,1);**

**}**

**else{**

**Ctl\_Led(LED2,0);**

**}**

**}**

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**}**

**/\*\***

**\* @brief: Control LEDx**

**\* @param: led is led to cotrol**

**\* where is LED1,LED2**

**\* @param: Status:**

**\* 1: LED ON**

**\* 0: LED OFF**

**\* @retval None**

**\*\*/**

**void Ctl\_Led(int led,int Status){**

**digitalWrite(led,Status == 1 ? HIGH : LOW);**

**}**

**/\*\***

**\* @brief: Setup wifi for ESP and print IP address.**

**\* @param: None**

**\* @retval None**

**\*\*/**

**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.println("WiFi connected");**

**Serial.println("IP address: ");**

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

**}**

**/\*\***

**\* @brief: Connect to MQTT broker with ID, user and pass**

**\* and pushlish init value of LEDs**

**\* @param: None**

**\* @retval None**

**\*\*/**

**void reconnect(){**

**// Chờ tới khi kết nối**

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

**Serial.print("Attempting MQTT connection...");// Thực hiện kết nối với mqtt user và pass**

**if (client.connect(DeviceID,mqtt\_user, mqtt\_pwd)) {**

**Serial.println("connected");// Khi kết nối sẽ publish thông báo**

**// client.publish(mqtt\_topic\_sensor,"OFF1");**

**// client.publish(mqtt\_topic\_sensor,"OFF2");**

**// client.publish(mqtt\_topic\_sensor,"OFF3");**

**delay(50);**

**// subcrice đến topic của mỗi đèn để nhận tín hiệu điều khiển**

**client.subscribe(mqtt\_topic\_led\_request);**

**}**

**else{**

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

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

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

**delay(5000);// Đợi 5s**

**}**

**}**

**}**

**/\*\***

**\* @brief: Read data from sensors. Temperature,Humidity and light**

**\* after convert to JSON and publish.**

**\* @param: None**

**\* @retval None**

**\*\*/**

**void Update\_data(){**

**StaticJsonBuffer<200> jsonBuffer;//memory allocated on the stack and has a fixed size**

**JsonObject& data = jsonBuffer.createObject();//Allocates an empty JsonObject**

**char mesage[256];// mang ky tu**

**int l = analogRead(quangtro);**

**if(l > 1000){**

**Ctl\_Led(LED1,1);**

**data["R1"] = true;**

**data.printTo(mesage, sizeof(mesage));**

**client.publish(mqtt\_topic\_led\_response,mesage);**

**Serial.println("Đèn sáng");**

**}**

**else if(l < 300 ){**

**Ctl\_Led(LED1,0);**

**data["R1"] = false;**

**data.printTo(mesage, sizeof(mesage));**

**client.publish(mqtt\_topic\_led\_response,mesage);**

**Serial.println("Đèn tắt");**

**}**

**float h = dht.readHumidity();**

**float t = dht.readTemperature();**

**if (isnan(l)){ l = 0;}**

**if (isnan(h)){ h = 0; }**

**if (isnan(t)){ t = 0; }**

**// gan du lieu vao chuoi Json**

**data["ID"] = DeviceID;**

**data["Temperature"] = t ;**

**data["Humidity"] = h;**

**data["Light"] = l;**

**data.printTo(mesage, sizeof(mesage));// lưu thanh chuoi JSON**

**// gui du lieu len ThingsBoard**

**Serial.print("Format data: ");**

**Serial.println(mesage);**

**client.publish(mqtt\_topic\_sensor,mesage);**

**}**