

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH



Hệ điều hành (CO2017)

Báo cáo đề tài nghiên cứu
TÌM HIỂU VỀ THINGSBOARD.IO
VÀ TRIỂN KHAI THINGSBOARD TRÊN NÚT EDGE/FOG
TRONG EDGE COMPUTING DÙNG RASPBERRYPI

GVHD: Nguyễn Quang Hùng
SV: Tạ Nguyễn Tiến Dũng - 2110965
Phạm Thanh Phong - 2037031
Huỳnh Nguyễn Hiếu Nhân - 2013961
Võ Hoàng Nam - 2013835
Nguyễn Huy Tùng - 2112616

TP. HỒ CHÍ MINH, THÁNG 11/2022

Mục lục

1	Đặt vấn đề	3
1.1	Tính cấp thiết của đề tài.	3
1.2	Mô hình nông nghiệp thông minh sử dụng Edge computing.	3
1.2.1	Nông nghiệp thông minh.	3
1.2.2	Mô hình nông nghiệp thông minh triển khai edge computing.	3
2	Lý thuyết nền tảng	4
2.1	Truyền nhận dữ liệu IoTs.	4
2.1.1	Giao thức MQTT.	4
2.1.2	Publish & Subscribe:	6
2.2	ThingsBoard.io	8
2.2.1	ThingBoard là gì?	8
2.2.2	Các tính năng của ThingBoard	9
2.2.3	Một số hạn chế và thách thức khi sử dụng ThingBoard nói riêng và IoT nói chung	9
2.3	Raspberry Pi	10
2.3.1	Raspberry Pi là gì?	10
2.3.2	Raspberry Pi dùng để làm gì?	10
2.3.3	Hệ điều hành và phần mềm Raspberry Pi.	11
2.3.4	Ưu điểm và nhược điểm của Raspberry Pi.	13
2.4	Edge computing	13
2.4.1	Khái niệm về Edge Computing.	13
2.4.2	Các thành phần cơ bản trong Edge Computing.	14
2.4.3	Cách thức các Edge Computing hoạt động.	14
2.4.4	Ưu điểm của Edge Computing	15
2.4.5	Nhược điểm của Edge Computing.	15
3	Hiện thực hệ thống	16
3.1	Kịch bản cho hệ thống.	16
3.2	Kế hoạch tưới tiêu cho cây đậu tương	17
3.3	Thu thập dữ liệu từ cảm biến	17
3.4	Xử lý dữ liệu đã thu thập	19
3.5	Trực quan hóa các dữ liệu bằng dashboard	22
3.6	Kích hoạt hệ thống máy tưới nước	24
4	Demo và kiểm thử hệ thống	26
4.1	Code python cho việc kiểm thử	26
4.2	Kết quả	29
4.2.1	Kiểm tra dữ liệu gửi từ Edge lên Cloud	29
4.2.2	Kiểm tra xử lý của rule chain	30
4.2.3	Kiểm tra thay đổi thời kì sinh trưởng	31
4.3	Kiểm tra dashboard	33



5	Mức độ hiệu quả của hệ thống.	34
5.1	Ưu điểm	34
5.2	Nhược điểm.	34
5.3	Khả năng mở rộng hệ thống.	34
6	Tổng kết	35

1 Đặt vấn đề

1.1 Tính cấp thiết của đề tài.

- Điện toán biên cho phép các thiết bị ở các vị trí từ xa xử lý dữ liệu ở "biên" của mạng, bằng thiết bị hoặc máy chủ cục bộ. Và khi dữ liệu cần được xử lý trong trung tâm dữ liệu trung tâm, chỉ dữ liệu quan trọng nhất được truyền đi, do đó giảm thiểu độ trễ.
- Sử dụng điện toán biên để cải thiện thời gian phản hồi của các thiết bị từ xa, kịp thời hơn từ dữ liệu thiết bị. Điện toán biên giúp khả thi điện toán thời gian thực ở những vị trí mà thông thường không khả thi và giảm tắc nghẽn trên các mạng và trung tâm dữ liệu hỗ trợ các thiết bị biên.
- Nếu không có điện toán biên, khối lượng dữ liệu khổng lồ do các thiết bị biên tạo ra sẽ chiếm hầu hết các mạng kinh doanh ngày nay, mọi hoạt động trên mạng bị ảnh hưởng và cản trở. Chi phí CNTT có thể tăng vọt. Khách hàng không hài lòng có thể đưa doanh nghiệp của họ đi nơi khác. Máy móc có giá trị có thể bị hư hỏng hoặc đơn giản là kém hiệu quả hơn. Nhưng quan trọng nhất, sự an toàn của người lao động có thể bị tổn hại trong các ngành dựa vào các cảm biến thông minh để giữ an toàn cho họ.

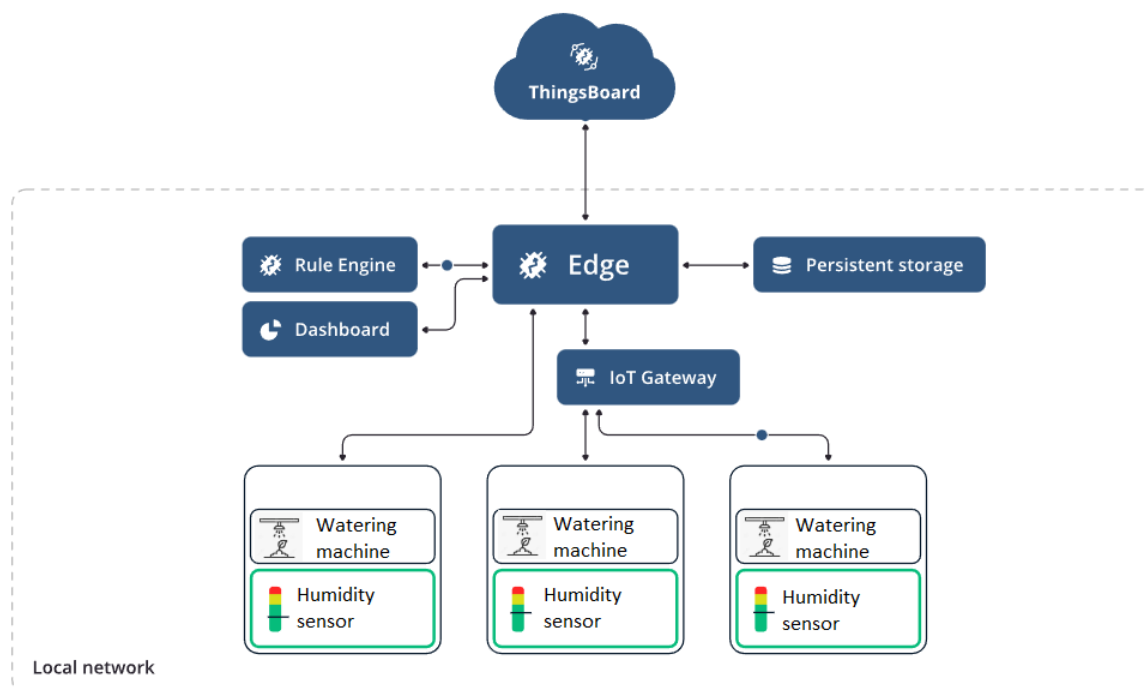
1.2 Mô hình nông nghiệp thông minh sử dụng Edge computing.

1.2.1 Nông nghiệp thông minh.

- Cùng với sự phát triển của thời đại công nghệ 4.0, mô hình nông nghiệp thông minh đang trở thành xu thế mới, thay thế nền nông nghiệp lạc hậu chủ yếu phụ thuộc vào sức người.
- Khi được ứng dụng trong thực tế, nông nghiệp thông minh giúp người nông dân nâng cao năng suất tiếp nhận thông tin, nâng cao khả năng quản lý, giảm nhân công lao động, ứng phó hiệu quả với rủi ro, từ đó nâng cao năng suất, hiệu quả công việc. Bên cạnh đó, số lao động tham gia vào lĩnh vực nông nghiệp ở nước ta đang có xu hướng giảm, vì vậy các hệ thống nông nghiệp thông minh và tự động hóa là giải pháp hàng đầu cho vấn đề này.

1.2.2 Mô hình nông nghiệp thông minh triển khai edge computing.

- IoT Gateway: là 1 thiết bị raspberry pi được cài sẵn nền tảng thingsboard. Có nhiệm vụ tiếp nhận dữ liệu từ các cảm biến độ ẩm, từ đó xử lý dữ liệu và gửi các rpc call để điều khiển các thiết bị tưới nước.
- Rule engine, edge, Dashboard, Persistent storage: là những tính năng có trong nền tảng thingsboard.
- Humidity sensor: cảm biến độ ẩm. Có nhiệm vụ gửi dữ liệu độ ẩm đến IoT Gateway.
- Watering machine: thiết bị tưới nước. Có nhiệm vụ tiếp nhận các rpc call từ IoT Gateway và thực hiện tưới nước cho cây
- ThingsBoard cloud: là một cloud server, nếu không có kết nối mạng Edge sẽ đưa dữ liệu thu thập được vào một hàng đợi, khi có kết nối mạng, Edge sẽ đưa dữ liệu lên cloud sau một khoảng thời gian nhất định.



Hình 1: Mô hình nông nghiệp thông minh triển khai edge computing.

2 Lý thuyết nền tảng

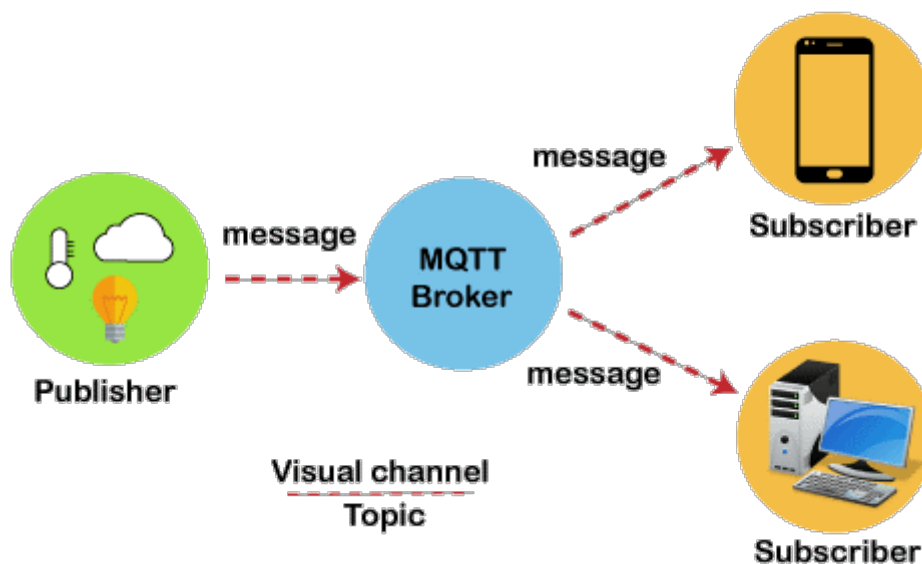
2.1 Truyền nhận dữ liệu IoTs.

2.1.1 Giao thức MQTT.

2.1.1.1 Giao thức MQTT là gì:

MQTT (Message Queuing Telemetry Transport) .Đây là giao thức truyền thông điệp (message) theo mô hình publish/subscribe (cung cấp / thuê bao), được sử dụng cho các thiết bị IoT với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định. Nó dựa trên một Broker (tạm dịch là “Máy chủ môi giới”) “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 và dễ cài đặt.

MQTT Architecture



MQTT là lựa chọn lý tưởng trong các môi trường như:

- Những nơi mà giá mạng viễn thông đắt đỏ hoặc băng thông thấp hay thiếu tin cậy.
- Khi chạy trên thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ.
- Bởi vì giao thức này sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng M2M (Machine to Machine).

MQTT cũng là giao thức được sử dụng trong Facebook Messenger.

2.1.1.2 Một số tính năng và điểm nổi bật:

Với những tính năng trên, MQTT mang lại nhiều lợi ích nhất là trong hệ thống SCADA (Supervisory Control And Data Acquisition) khi truy cập dữ liệu IoT.

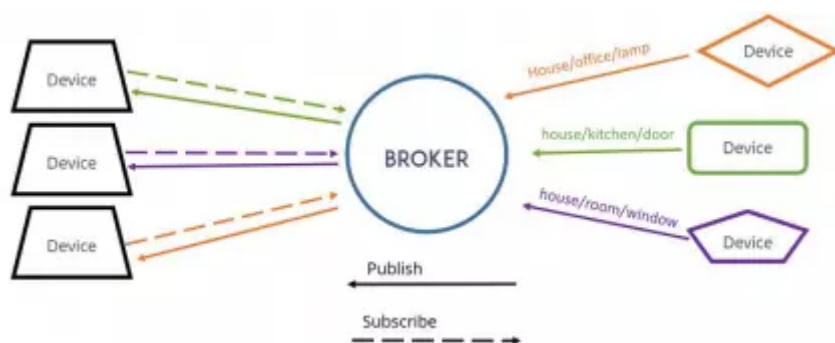
- Truyền thông tin hiệu quả hơn.
- Tăng khả năng mở rộng.
- Giảm đáng kể tiêu thụ băng thông mạng.
- Rất phù hợp cho điều khiển và đo lường.
- Tối đa hóa băng thông có sẵn.
- Chi phí thấp.
- Rất an toàn, bảo mật.
- Được sử dụng trong các ngành công nghiệp dầu khí, các công ty lớn như Amazon, Facebook,...

- Tiết kiệm thời gian phát triển.
- Giao thức publish/subscribe thu thập nhiều dữ liệu hơn và tốn ít băng thông hơn so với giao thức cũ.

2.1.2 Publish & Subscribe:

2.1.2.1 Thành phần của mô hình Publish & Subscribe:

1. Client
2. Publisher - Nơi gửi thông điệp
3. Subscriber - Nơi nhận thông điệp
4. Broker - Máy chủ môi giới



Trong đó Broker được coi như trung tâm, nó là điểm giao của tất cả các kết nối đến từ Client (Publisher/Subscriber). Nhiệm vụ chính của Broker là nhận thông điệp (message) từ Publisher, xếp vào hàng đợi rồi chuyển đến một địa điểm 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 sẽ được chia thành hai nhóm là Publisher và Subscriber. Client chỉ làm ít nhất một trong 2 việc là publish các thông điệp (message) lên một/nhiều topic cụ thể hoặc subscribe một/nhiều 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, Android, iOS,...

Ưu điểm:

- Kết nối riêng rẽ, độc lập.
- Khả năng mở rộng.
- Thời gian tách biệt (Time decoupling).
- Đồng bộ riêng rẽ (Synchronization decoupling).

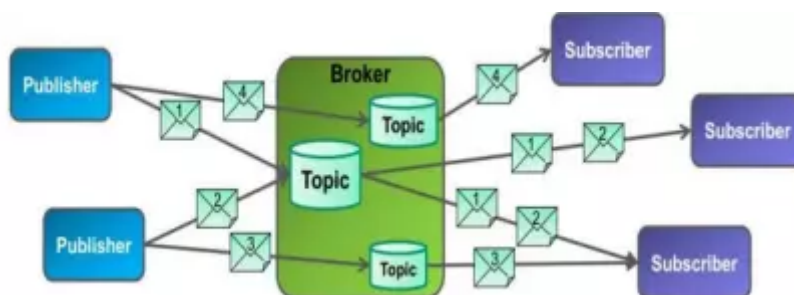
Nhược điểm:

- Máy chủ môi giới (Broker) không cần thông báo về trạng thái gửi thông điệp. Do đó không có cách nào để phát hiện xem thông điệp đã gửi đúng hay chưa.
- Publisher không hề biết gì về trạng thái của subscribe và ngược lại. Vậy làm sao chúng ta có thể đảm bảo mọi thứ đều ổn.
- Những kẻ xấu (Malicious Publisher) có thể gửi những thông điệp xấu, và các Subscriber sẽ truy cập vào những thứ mà họ không nên nhận.

2.1.2.2 Cơ chế hoạt động theo mô hình Publish & Subscribe:

Mô hình Publish & Subscribe mang một số tính chất riêng và đặc điểm như:

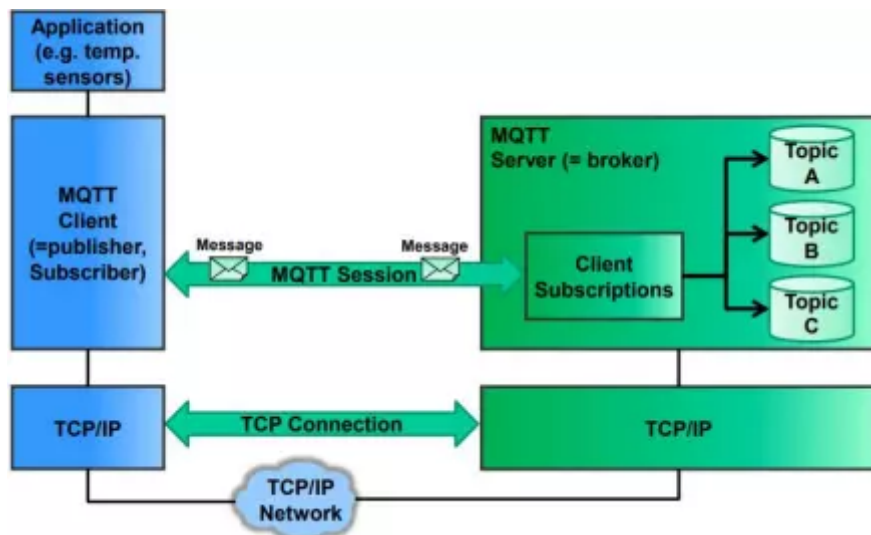
- Space decoupling (Không gian tách biệt)
- Time decoupling (Thời gian tách biệt)
- Synchronization decoupling (Sự đồng bộ riêng rẽ)
- MQTT sử dụng cơ chế lọc thông điệp dựa vào tiêu đề (subject-based)
- MQTT có một tầng gọi là chất lượng dịch vụ (Quality of Services – QoS). Nó giúp cho dễ dàng nhận biết được là message có được truyền thành công hay không.



MQTT hoạt động theo cơ chế client/server, nơi mà mỗi cảm biến là một khách hàng (client) và kết nối đến một máy chủ, có thể hiểu như một Máy chủ môi giới (broker), thông qua giao thức TCP (Transmission Control Protocol). Broker chịu trách nhiệm điều phối tất cả các thông điệp giữa phía gửi đến đúng phía nhận.

MQTT là giao thức định hướng bản tin. Mỗi bản tin là một đoạn rời rạc của tín hiệu và broker không thể nhìn thấy. Mỗi bản tin được publish một địa chỉ, có thể hiểu như một kênh (Topic). Client đăng ký vào một vài kênh để nhận/gửi dữ liệu, gọi là subscribe. Client có thể subscribe vào nhiều kênh. Mỗi client sẽ nhận được dữ liệu khi bất kỳ trạm nào khác gửi dữ liệu vào kênh đã đăng ký. Khi một client gửi một bản tin đến một kênh nào đó gọi là publish.

2.1.2.3 Kiến trúc thành phần:



1. Thành phần chính của MQTT là Client (Publisher/Subscriber), Server (Broker), Sessions (tạm dịch là Phiên làm việc), Subscriptions và Topics.
2. MQTT Client (Publisher/Subscriber): Clients sẽ subscribe một hoặc nhiều topics để gửi và nhận thông điệp từ những topic tương ứng.
3. MQTT Server (Broker): Broker nhận những thông tin subscribe (Subscriptions) từ client, nhận thông điệp, chuyển những thông điệp đến các Subscriber tương ứng dựa trên Subscriptions từ client.
4. Topic: Có thể coi Topic là một hàng đợi các thông điệp, và có sẵn khuôn mẫu dành cho Subscriber hoặc Publisher. Một cách logic thì các topic cho phép Client trao đổi thông tin với những ngữ nghĩa đã được định nghĩa sẵn. Ví dụ: Dữ liệu cảm biến nhiệt độ của một tòa nhà.
5. Session: Một session được định nghĩa là kết nối từ client đến server. Tất cả các giao tiếp giữa client và server đều là 1 phần của session.
6. Subscription: Không giống như session, subscription về mặt logic là kết nối từ client đến topic. Khi đã subscribe một topic, Client có thể nhận/gửi thông điệp (message) với topic đó

2.2 ThingsBoard.io

2.2.1 ThingBoard là gì?

- ThingsBoard là một nền tảng IoT mã nguồn mở để giám sát, xử lý dữ liệu, trực quan hóa dữ liệu cùng với quản lý thiết bị.
- Nó hỗ trợ các giao thức IoT tiêu chuẩn công nghiệp - MQTT, CoAP và HTTP. ThingsBoard kết hợp khả năng mở rộng, khả năng chịu lỗi và hiệu suất để thu thập dữ liệu thiết bị để xử lý và giám sát. Nó cung cấp gateway server có thể giao tiếp với các thiết bị đính kèm.

- Xây dựng dựa trên Netty Framework do nó cung cấp hỗ trợ cho nhiều giao thức và ứng dụng. Có thể thêm các giao thức phần cứng mới bằng cách thêm các trình xử lý kênh Inbound và Outbound cho các giao thức mới với netty framework.

2.2.2 Các tính năng của ThingBoard

- **Bảo mật:** Hỗ trợ cung cấp và quản lý các thiết bị có quản lý thông tin đăng nhập. Các quy tắc bảo mật tùy chỉnh cho từng giao thức có thể được áp dụng.
- **Trang tổng quan và trực quan hóa Dữ liệu:** Hỗ trợ tích hợp cho hơn 100 thành phần widget. Thu thập và trực quan hóa dữ liệu từ các thiết bị và nội dung trong trang tổng quan bằng các tiện ích con.
- **Phép đo từ xa:** Phân tích phép đo từ xa đến và kích hoạt cảnh báo với xử lý sự kiện phức tạp. Hỗ trợ API lưu trữ sự kiện để nắm bắt các chỉ số đo từ xa.
- **Hỗ trợ Rest API và RPC:** Quy trình làm việc dữ liệu có thể được thiết kế bằng cách sử dụng Rest API và RPC request.
- **Đẩy dữ liệu thiết bị:** Hỗ trợ đẩy dữ liệu thiết bị sang các hệ thống khác theo thời gian thực.
- **Tích hợp với các hàng đợi tin nhắn khác nhau:** Các trình kết nối khác nhau có sẵn để kết nối với các triển khai khác nhau của hàng đợi tin nhắn. Hỗ trợ triển khai nhiều hàng đợi tin nhắn: Kafka, RabbitMQ, AWS SQS, Azure Service Bus và Google Pub / Sub.
- **Hỗ trợ cơ sở dữ liệu kết hợp:** Lưu trữ tất cả các thực thể trong cơ sở dữ liệu SQL và dữ liệu đo từ xa trong cơ sở dữ liệu NoSQL.
- **Công cụ quy tắc:** Công cụ quy tắc tích hợp, để định cấu hình các quy tắc cho trạng thái thiết bị và các thông báo hoặc cảnh báo khác nhau có thể được tạo dựa trên phép đo từ xa của thiết bị. Các quy tắc xử lý dữ liệu có thể được thay đổi trong thời gian chạy dựa trên trạng thái thiết bị.
- **Chế độ triển khai, tiêu chuẩn và cụm:** Triển khai tại chỗ và đám mây được hỗ trợ cùng với chế độ tiêu chuẩn và cụm được hỗ trợ
- **Quản lý cảnh báo:** Tạo và quản lý các cảnh báo liên quan đến thiết bị, nội dung và khách hàng,... Chính sách quy tắc có thể được áp dụng để tăng cảnh báo khi thay đổi trạng thái thiết bị.

2.2.3 Một số hạn chế và thách thức khi sử dụng ThingBoard nói riêng và IoT nói chung

Chính vì IoT là một nền tảng IoT cho nên ThingBoard sẽ có một số khó khăn và thách thức khi chúng ta triển khai thực hiện hệ thống này.

- Vấn đề tài chính
 - Hạn chế về tài chính là một trong những yếu tố khó áp dụng IoT trong IA((Information Agencies) vì IA cần mua thiết bị và thiết bị mới cũng như cần đào tạo đội ngũ nhân viên sử dụng công nghệ IoT. Đây là bởi vì ban lãnh đạo cao nhất không coi IA là cơ quan có thể tạo ra lợi nhuận cho tổ chức. Họ nghĩ thông tin đó không thể mang lại

bất kỳ điều gì tốt cho họ vì họ cho rằng thông tin có thể ổn chỉ bằng làm theo cách truyền thống. Ví dụ: chỉ bằng cách hỏi những người có kinh nghiệm về vấn đề họ đang gặp phải đối mặt. Nhưng trong thực tế, hiện nay, hầu hết thông tin cần phải có bằng chứng, cũng cần được hiển thị cho đúng người vào đúng thời điểm và đến đúng nơi.

- An ninh và quyền riêng tư
 - Trong việc áp dụng IoT, có một vấn đề về bảo mật và quyền riêng tư. Đây có thể coi là vấn đề nghiêm trọng bởi vì thông tin có thể bị cho sai người. Ví dụ, nếu sai người có thể nhận được thông tin cá nhân liên quan đến cuộc sống cá nhân của mình, họ có thể bị tống tiền hoặc có thể đe dọa trong tương lai. Với tình huống này, nó sẽ mang lại rất nhiều căng thẳng cho người đó cho đến một ngày có thể người đó thấy tự tử sẽ là cách giải thoát duy nhất cho cuộc sống. Có 3 vấn đề chính của bảo mật trên Internet là malware, Trojan Horse và phần mềm tường lửa.
- Thiếu kiến thức chuyên môn về công nghệ
 - Trong thời đại công nghệ hiện nay, hầu hết mọi người đều biết đến công nghệ nhưng rất ít người theo dõi sự phát triển của công nghệ. Hầu hết các nhân viên trong IA cần bắt kịp với công nghệ để có được thông tin mới nhất và tính năng từ công nghệ mới nhất. Trong IA, thiếu chuyên môn về công nghệ IoT mang lại hầu hết thách thức. Điều này là do khi ở đó không có nhân viên chuyên gia về IoT trong IA, sẽ không có hướng dẫn về cách sử dụng và cũng không có chức năng của công nghệ IoT đó khi có điều gì đó xảy ra với công nghệ. Cơ quan cần cử một số nhân viên kỹ thuật đi đào tạo để đảm bảo rằng có chuyên môn trong lĩnh vực này. Theo bài báo của Cisco, 75 percents dự án IoT không thành công là do thiếu chuyên môn. Kiến thức chuyên môn của IoT dựa trên các yếu tố là kỹ năng kinh doanh và chuyên môn kỹ thuật. Kỹ năng kinh doanh là cần thiết vì cần phải giao tiếp giữa CNTT và doanh nghiệp. Kỹ năng này cũng cần thiết vì cần phải làm cho người dùng hiểu về cách thức hoạt động của IoT hoặc công nghệ đó và cũng cần lập tài liệu tham khảo cho tương lai. Kỹ năng kỹ thuật là cần thiết vì cần hiểu thêm về chính công nghệ IoT. Nếu bất cứ điều gì xảy ra với công nghệ, những kỹ năng này là bắt buộc.

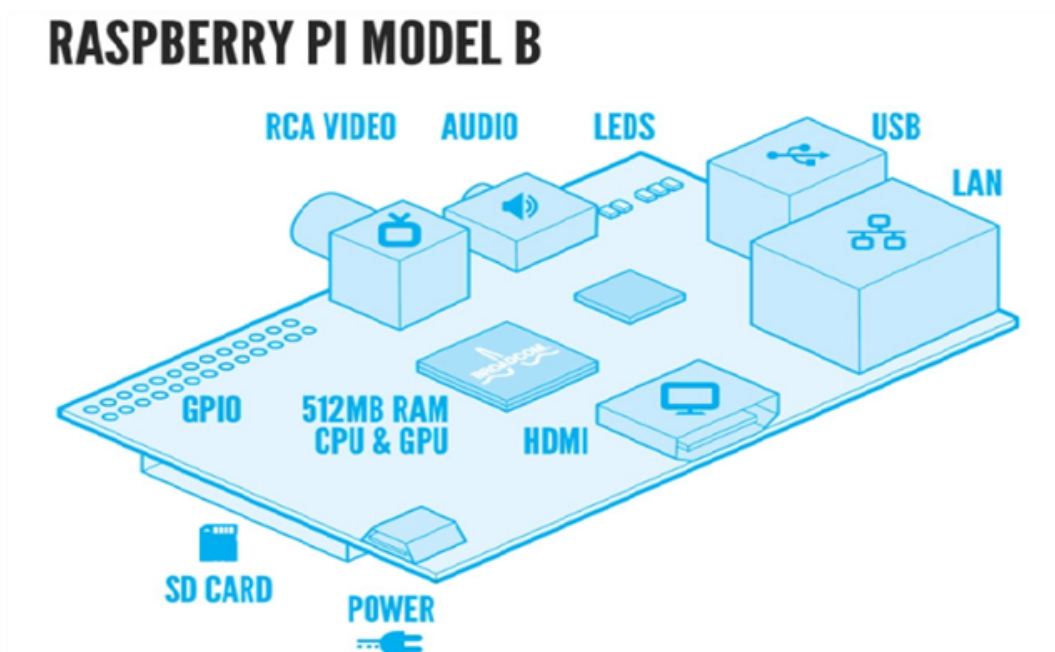
2.3 Raspberry Pi

2.3.1 Raspberry Pi là gì?

- Raspberry Pi là một máy vi tính rất nhỏ gọn, kích thước hai cạnh chỉ cỡ một cái thẻ ATM, trong đó đã tích hợp mọi thứ cần thiết để bạn sử dụng như một máy vi tính. Bộ xử lý SoC Broadcom BCM2835 của nó bao gồm CPU, GPU, RAM, khe cắm thẻ microSD, WiFi, Bluetooth và 4 cổng USB 2.0.

2.3.2 Raspberry Pi dùng để làm gì?

- Với khả năng tùy biến cao, Raspberry Pi có thể biến thành rất nhiều thiết bị từ phần cứng đến phần mềm, có thể kể đến một số công dụng như sau
 - Đầu coi phim HD giống như TV Box, hỗ trợ KODI đầy đủ.
 - Máy chơi game cầm tay, console, game thùng. Chơi như máy điện tử bằng ngày xưa, giả lập được nhiều hệ máy.
 - Cắm máy tải Torrent 24/24.



Hình 2: Raspberry Pi.

- 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 WiFi. . .
 - Làm thiết bị điều khiển SmartHome, điều khiển mọi thiết bị điện tử trong nhà
 - Điều khiển robot, máy in không dây từ xa, AirPlay. . .
- Trên đây chỉ là một vài ứng dụng dễ thấy của Raspberry Pi, chúng có thể được sử dụng cho vô vàn những mục đích khác nhau.

2.3.3 Hệ điều hành và phần mềm Raspberry Pi.

- Raspberry Pi hoạt động trong hệ sinh thái mã nguồn mở: Nó chạy Linux (nhiều loại bản phân phối) và hệ điều hành được hỗ trợ chính, Raspbian, là mã nguồn mở và chạy một bộ phần mềm mã nguồn mở.
 - Raspbian: Đây là bản build Linux dựa trên nền Debian (Gần giống ubuntu) với giao diện LXDE (thay vì GNOME). Có đầy đủ web browser, media player, tools, etc . . . Nói chung HĐH này dành cho những người muốn dùng Raspberry Pi như một cái PC.



Hình 3: Raspbian.



Hình 4: Raspbmc.

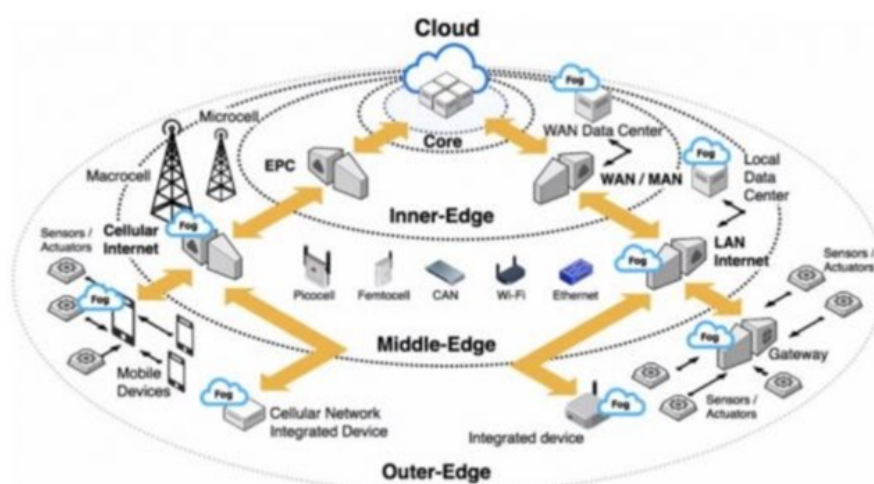
- Raspbmc: được coi là bản Raspbian lược bỏ đi LXDE để sử dụng XBMC. Với Raspberry Pi cài Raspbmc bạn có thể biến chiếc TV bình thường thành một chiếc TV thông minh có khả năng truy cập mạng Internet để tìm kiếm và xem phim có độ phân giải 1080p trực tiếp trên TV.

2.3.4 Ưu điểm và nhược điểm của Raspberry Pi.

- Giá rẻ, nhỏ gọn, tiết kiệm điện, GPU mạnh, phục vụ cho nhiều mục đích, khả năng hoạt động liên tục 24/7.
- CPU cấu hình thấp, không có tích hợp WiFi, yêu cầu phải có kiến thức cơ bản về Linux, điện tử.

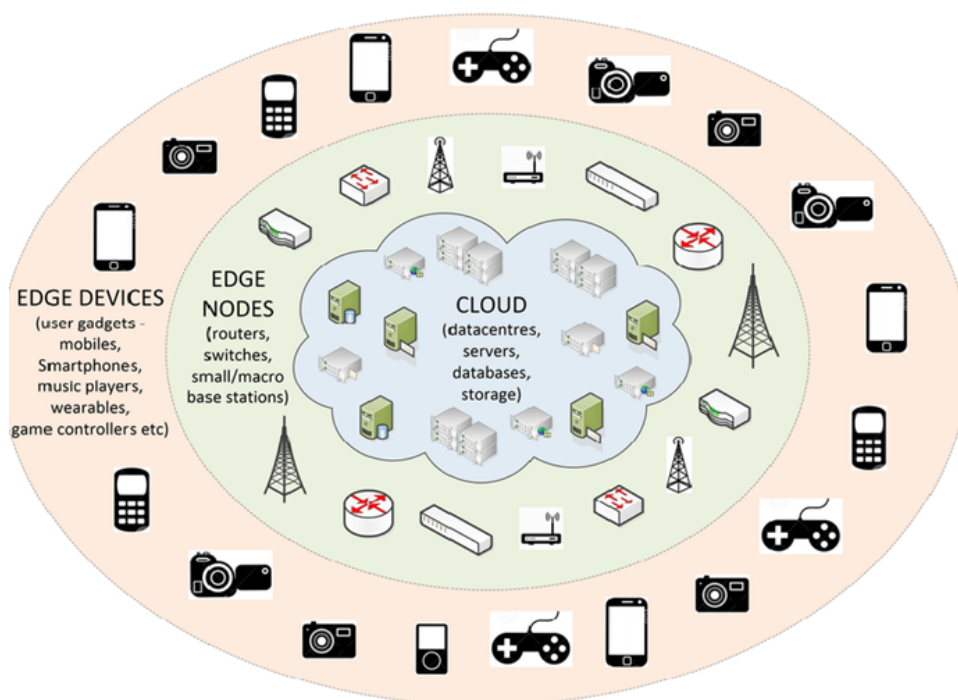
2.4 Edge computing

2.4.1 Khái niệm về Edge Computing.



Hình 5: Edge computing.

- Edge Computing, hay Điện toán biên là một mạng lưới các trung tâm xử lý và lưu trữ dữ liệu cục bộ trước khi chúng được gửi tới Trung tâm dữ liệu hay chuyển lên các đám mây điện toán. là phương pháp tối ưu hoá hệ thống điện toán đám mây, hoạt động bằng cách nắm bắt và xử lý thông tin càng gần nguồn dữ liệu hoặc sự kiện mong muốn càng tốt. Nó dựa vào cảm biến, thiết bị điện toán và máy móc để thu thập dữ liệu và cung cấp dữ liệu cho các máy chủ biên hoặc đám mây. Tùy thuộc vào nhiệm vụ và kết quả mong muốn, dữ liệu này có thể cung cấp cho hệ thống phân tích và học máy, cung cấp khả năng tự động hóa hoặc cung cấp khả năng hiển thị về trạng thái hiện tại của thiết bị, hệ thống hoặc sản phẩm.



Hình 6: Các thành phần cơ bản trong Edge Computing.

2.4.2 Các thành phần cơ bản trong Edge Computing.

- Cloud Server: Đây có thể là một đám mây công cộng hoặc riêng tư, hoặc có thể là một trung tâm dữ liệu
- Edge device: là thiết bị được tích hợp khả năng tính toán như máy ATM, máy ảnh số hoặc ô tô. Các thiết bị biên thường có năng lực tính toán hạn chế, chỉ xử lý các yêu cầu tức thời cần độ trễ thấp.
- Edge node: là một cách gọi chung để chỉ bất kỳ edge device, edge server hoặc edge gateway nào mà tính toán biên có thể được thực hiện.
- Edge server: là một máy tính có mục đích chung được đặt trong một cơ sở hoạt động từ xa như nhà máy, cửa hàng bán lẻ, khách sạn, trung tâm phân phối hoặc chi nhánh ngân hàng.
- Edge gateway: một edge gateway thường là một máy chủ biên, ngoài việc xử lý khối lượng công việc của ứng dụng doanh nghiệp còn thực hiện các chức năng mạng như biên dịch giao thức, bảo vệ tường lửa hoặc kết nối không dây.

2.4.3 Cách thức các Edge Computing hoạt động.

- Các kiến trúc sư CNTT đã chuyển trọng tâm từ trung tâm dữ liệu trung tâm sang biên hợp lý của cơ sở hạ tầng - lấy tài nguyên lưu trữ và tính toán từ trung tâm dữ liệu và chuyển các tài nguyên đó đến điểm dữ liệu được tạo ra. Nguyên tắc rất đơn giản: Nếu bạn

không thể đưa dữ liệu đến gần trung tâm dữ liệu hơn, hãy đưa trung tâm dữ liệu đến gần dữ liệu hơn. Khái niệm về edge computing không phải là mới và nó bắt nguồn từ những ý tưởng hàng thập kỷ về máy tính từ xa - chẳng hạn như văn phòng từ xa và văn phòng chi nhánh - nơi đáng tin cậy và hiệu quả hơn khi đặt tài nguyên máy tính ở vị trí mong muốn hơn là dựa vào một vị trí trung tâm duy nhất.

2.4.4 Ưu điểm của Edge Computing

- Giới hạn về tốc độ xử lý Cloud Computing: Cloud server có thể xử lý những tác vụ rất lớn, nhưng do chúng thường được đặt ở các vị trí rất xa nên độ trễ đường truyền qua Internet có thể tính bằng hàng trăm mili giây. Ngược lại, các edge device có thể yếu hơn rất nhiều nhưng với lượng dữ liệu không quá lớn từ các thiết bị IoT, chúng có thể cung cấp tốc độ phản hồi ở mức micro giây từ khoảng cách ngắn. Thử nghĩ về những chiếc xe tự lái, mỗi mili giây độ trễ có thể đánh đổi bằng an toàn của con người.
- Đảm bảo đường truyền dữ liệu: Các edge server ở gần hay thậm chí trong cùng mạng cục bộ luôn có thể đảm bảo tốc độ và sự ổn định khi truyền dữ liệu. Một ví dụ dễ thấy là mỗi khi cáp quang gặp sự cố, ảnh hưởng đến kết nối Internet trong nước là không đáng kể trong khi băng thông quốc tế luôn sụt giảm đến mức khó chịu.
- Bảo mật an toàn cao: Vấn đề nổi bật của điện toán đám mây, nhất là đám mây công cộng chính là tính riêng tư và bảo mật dữ liệu. Có thể nói edge computing phần nào đó khá giống với đám mây lai hybrid cloud, khi mà những xử lý cục bộ - tại thiết bị biên luôn cho sự an tâm tốt hơn về bảo mật. Chỉ các dữ liệu không quan trọng mới được đẩy lên máy chủ công cộng.
- Giảm tải băng thông của Cloud Computing: Băng thông đến các Cloud server đặt ở xa là nguồn tài nguyên hạn chế mà ai cũng muốn tiết kiệm. Những tải công việc được xử lý ngay tại biên không chỉ giúp giảm lượng dữ liệu phải truyền qua Internet đến Cloud Server mà còn cắt giảm chi phí đầu tư cho năng lực xử lý đám mây.

2.4.5 Nhược điểm của Edge Computing.

- Mặc dù Edge Computing có tiềm năng và cung cấp nhiều lợi ích hấp dẫn trên vô số trường hợp sử dụng, nhưng công nghệ này vẫn còn xa vời. Ngoài các vấn đề truyền thống về giới hạn mạng, thì vẫn còn một số cân nhắc khi sử dụng Edge Computing như:
 - Thiếu cơ sở vật chất tại chỗ: Edge Computing thực hiện xử lý ở vùng biên, nơi đặt gần máy chủ nhất. Tuy nhiên, có những vùng kém phát triển gây khó khăn khi triển khai cơ sở hạ tầng điện toán biên, khiến cho mô hình này không phát triển được tối đa khả năng.
 - Bảo mật: Các thiết bị IoT nổi tiếng là không an toàn, vì vậy điều quan trọng cần thiết kế Edge Computing chú trọng đến việc quản lý thiết bị phù hợp, chẳng hạn như thực thi cấu hình theo hướng chính sách, cũng như bảo mật trong tài nguyên máy tính và lưu trữ, bao gồm các yếu tố như patch và update phần mềm. Đặc biệt, quan tâm đến việc mã hóa dữ liệu khi ở chế độ nghỉ và trong chuyển bay. Các dịch vụ IoT từ các nhà cung cấp đám mây lớn bao gồm truyền thông an toàn, nhưng điều này không tự động khi xây dựng một trang web tiên tiến từ đầu.
 - Data lifecycle (Vòng đời dữ liệu): Ngày nay, tình trạng thừa dữ liệu không cần thiết ngày càng phổ biến. Hầu hết dữ liệu liên quan đến phân tích thời gian thực là dữ liệu

ngắn hạn không được lưu giữ lâu dài. Doanh nghiệp cần phải quyết định dữ liệu nào cần giữ lại và dữ liệu nào cần loại bỏ sau khi thực hiện phân tích. Và dữ liệu được lưu giữ phải được bảo vệ theo các chính sách quản lý và kinh doanh.

- Chi phí: Mặc dù Edge Computing mang đến nhiều lợi ích và khả năng tiết kiệm băng thông cũng như năng lực xử lý đám mây. Tuy nhiên, chi phí ban đầu bỏ ra cho Edge Computing là không hề nhỏ.

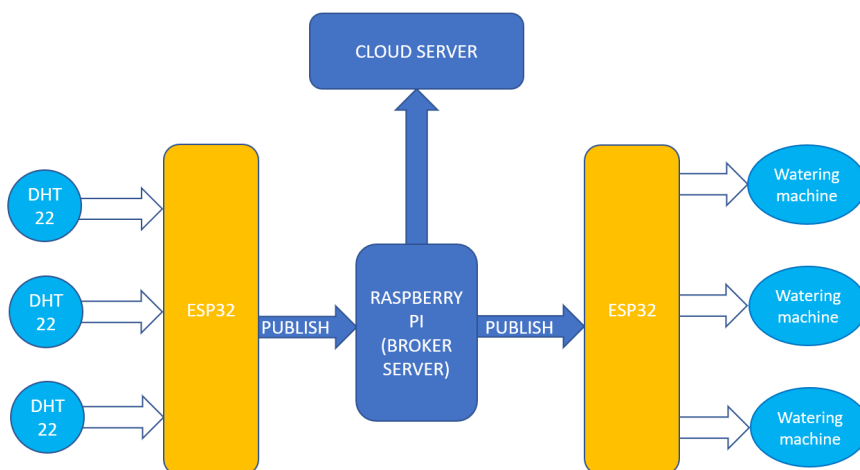
3 Hiện thực hệ thống

3.1 Kịch bản cho hệ thống.

Nhóm sẽ thực hiện một hệ thống tưới cây đơn giản theo thời kì sinh trưởng (growth period) của cây dựa vào độ ẩm và theo dõi nhiệt độ. Ta sẽ dùng một Raspberry Pi làm một thiết bị biên (edge node) sẽ tiếp nhận dữ liệu nhiệt độ(temperature) và độ ẩm (humidity) từ cảm biến được gửi bằng vi điều khiển ESP32 giao thức MQTT với Raspberry đóng vai trò như một broker server. Các thiết bị cảm biến sẽ là nhóm Publisher và máy tưới sẽ là Subscriber. Ngoài ra còn có 1 thuộc tính(attribute) được thêm vào chính là "growth period" để nhận biết thời kì sinh trưởng của cây

Nhóm sẽ sử dụng dịch vụ Thingsboard Edge trên Raspberry với Cloud là Demo Thingsboard (miễn phí). Các dữ liệu từ cảm biến sẽ được gửi (publish) đến 1 topic "v1/devices/me/telemetry" của Raspberry và sẽ được các rule chain xử lý dữ liệu vừa nhận được (đưa dữ liệu lên cloud, gửi cảnh báo và hiển thị lên dashboard). Nếu dữ liệu hệ thống độ ẩm nằm ngoài ngưỡng cho phép thì edge sẽ gửi 1 message đến topic "v1/devices/me/rpc/request/+" và được các máy tưới nhận được (subscribe).

Để có thể cài đặt được Thingsboard Edge nhóm đã sử dụng docker để đơn giản hóa nhiều cách thiết lập khá là phức tạp chỉ với một vài dòng lệnh đơn giản từ docker.



Hình 7: Sơ đồ hệ thống tưới nước

3.2 Kế hoạch tưới tiêu cho cây đậu tương

Bảng dưới đây là nhiệt độ và độ ẩm để cây đậu tương sinh trưởng và phát triển tốt nhất theo từng thời kì.

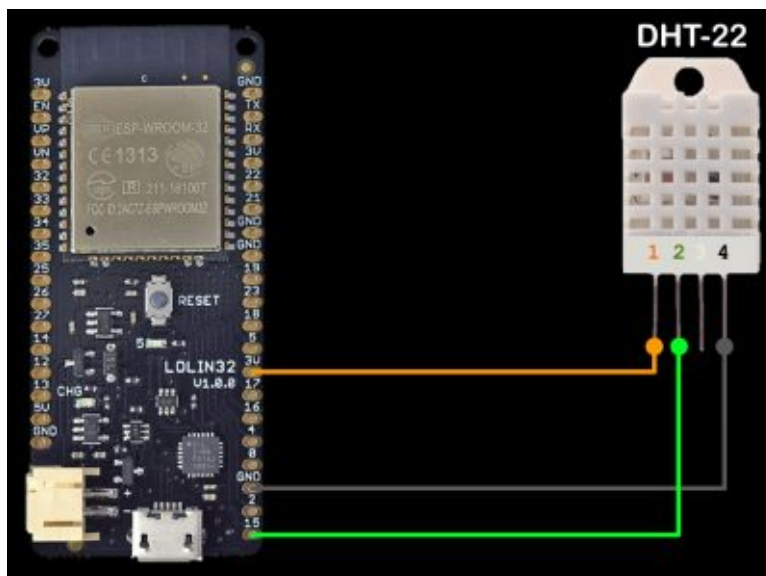
Thời kì sinh trưởng	Nhiệt độ($^{\circ}C$)	Độ ẩm(%)
Mọc(Nascent)	18 - 22	70 - 85
Sinh trưởng cành lá (Leaf growth)	20 - 23	60 - 65
Ra hoa(Flowering)	22 - 25	60 - 65
Hình thành quả hạt(Seed formation)	21 - 23	60 - 75
Hạt chín(Seed ripening)	19 - 20	55 - 65

Hệ thống của nhóm sẽ kích hoạt máy tưới nếu độ ẩm vượt dưới ngưỡng của mỗi thời kì sinh trưởng tương ứng. Sau đó khi đạt tới độ ẩm cần thiết sẽ tự động tắt. Nhiệt độ sẽ tạo 1 cảnh báo khi nhiệt độ môi trường nằm ngoài khoảng nhiệt độ ứng với từng thời kì sinh trưởng.

3.3 Thu thập dữ liệu từ cảm biến

ta sẽ dùng cảm biến DHT22 để thu nhận nhiệt độ và cảm biến và dùng vi điều khiển ESP32 có tích hợp wifi để kết nối với raspberry pi với một khoảng cách khá xa. ESP32 sẽ được nạp code được viết bằng arduino hoặc micropython để gửi dữ liệu lên raspberry pi (edge node).

Thời gian gửi dữ liệu sẽ được chọn là 15 phút vì không nên gửi dữ liệu liên tục do các ESP32 sẽ được dùng pin và do kết nối wifi nên sẽ tốn pin hơn so với kết nối mạng có dây. Vì vậy thời gian lặp lại việc gửi dữ liệu không được quá ngắn mà nên ở một mức chấp nhận được.



Hình 8: Sơ đồ mắc dây ESP32 và DHT22



Code nạp cho ESP32, lưu ý rằng ta có thể dùng lệnh : "\$hostname -I" để lấy IP address của raspberry pi dùng làm sever.

```
1 from time import sleep
2 from umqtt.simple import MQTTClient
3 from machine import Pin
4 from dht import DHT22
5 import json
6 import network
7
8 # MQTT
9 SERVER = '127.0.0.1' # MQTT Server Address (Change to the IP address of your Pi)
10 CLIENT_ID = 'LB4oqgfzi0Ulc9ingWun' # access token
11 TOPIC_TELEMETRY = b'v1/devices/me/telemetry'
12 #wifi SSID and password
13 WIFI_SSID = "NHAN"
14 WIFI_PASSWORD = "0123456789"
15
16
17
18
19 # connect to the wifi network
20 def connectWIFI():
21     wlan = network.WLAN(network.STA_IF)
22     wlan.active(True)
23     wlan.connect(WIFI_SSID,WIFI_PASSWORD)
24     while not wlan.isconnected():
25         pass
26
27     print ("wifi connected")
28     print(wlan.ifconfig())
29
30
31
32
33 client = MQTTClient(CLIENT_ID, SERVER)
34 client.connect() # Connect to MQTT broker
35 sensor = DHT22(Pin(15, Pin.IN, Pin.PULL_UP)) # DHT-22 on GPIO 15 (input with internal
    pull-up resistor)
36 while True:
37     try:
38         connectWIFI()
39         sensor.measure() # Poll sensor
40         temp = sensor.temperature()
41         humi = sensor.humidity()
42         if isinstance(temp, float) and isinstance(humi, float): # Confirm sensor results
            are numeric
43             msg = collect_data = {'temperature': temp, 'humidity': humi}
44             client.publish(TOPIC_TELEMETRY, msg) # Publish sensor data to MQTT topic
45             print(msg)
46         else:
47             print('Invalid sensor readings.')
```

```

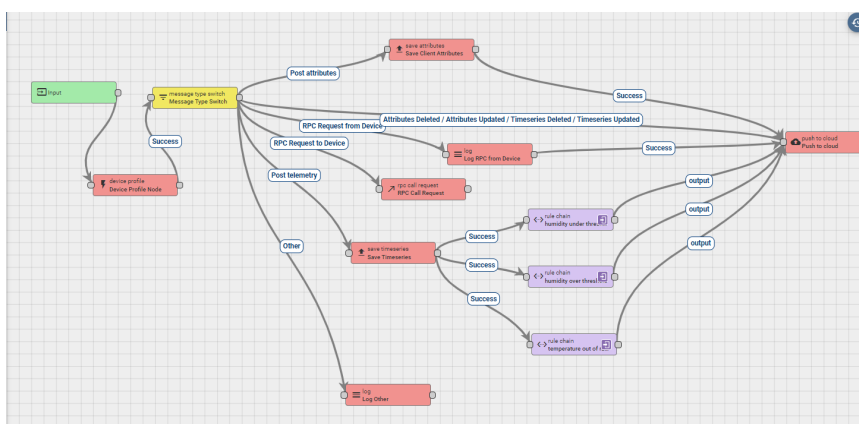
48 except OSError:
49     print('Failed to read sensor.')
50     sleep(15*60)

```

3.4 Xử lý dữ liệu đã thu thập

Khi sử dụng dịch vụ thingsboard edge, cũng giống như khi dùng cloud sever. Thingsboard cung cấp cho ta hệ thống rule chain để xử lý dữ liệu đầu vào. Chỉ khác nhau là thingsboard edge sẽ thêm thao tác gửi dữ liệu lên cloud.

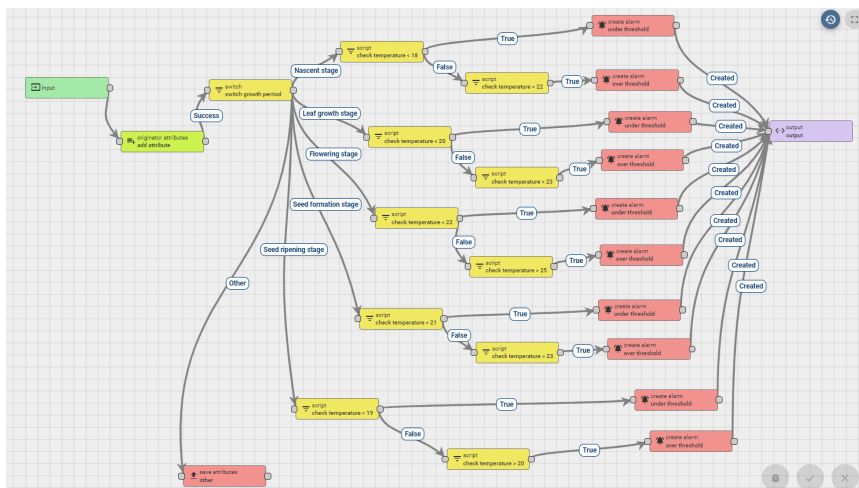
Root rule chain trên thingsboard edge:



Hình 9: Hình ảnh về root rule chain

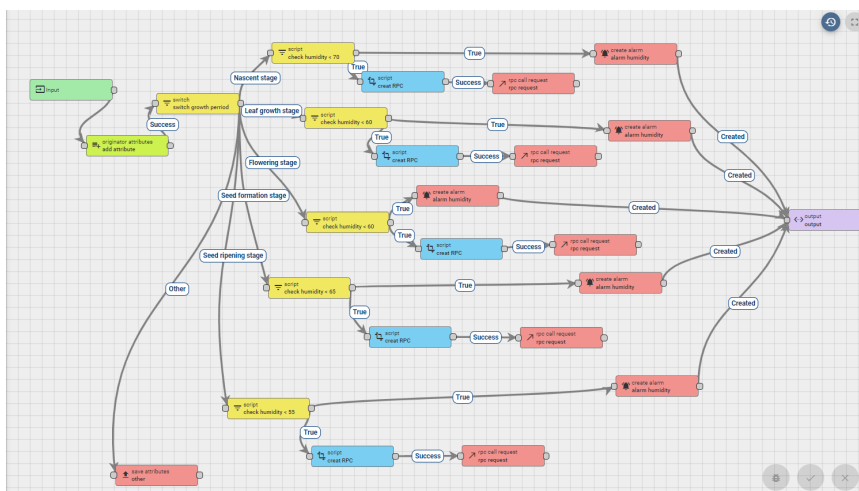
Ta sẽ thấy 3 nhánh rule chain khác lần lượt là "process temperature data out of range", "process humidity data under threshold" và "process humidity data over threshold". Nhánh "process temperature data out of range" sẽ gửi alarm cảnh báo khi nhiệt độ nằm ngoài khoảng mà ta đã thấy ở bảng kế hoạch tưới tiêu. "process humidity data under threshold" và "process humidity data over threshold" giống nhau chỉ khác là là một nhánh để cảnh báo và gửi RPC để bật máy tưới khi độ thấp dưới ngưỡng đã cho, nhánh còn lại dùng để tắt máy tưới khi độ ẩm đạt đủ mức cho phép.

Rule chain "process temperature data out of range":



Hình 10: Hình ảnh về "process temperature data out of range" rule chain

Rule chain "process humidity data under threshold":

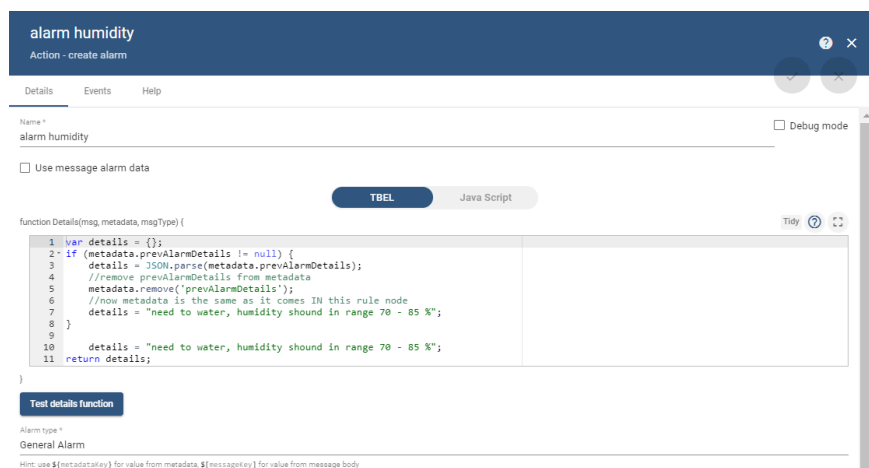


Hình 11: Hình ảnh về "process humidity data under threshold" rule chain

Node originator attribute để thêm vào thuộc tính Growth_Period của thiết bị vào message để có thể dùng cho node switch nhằm phân luồng dữ liệu để có thể xử lý trên các nhánh với nhiệt độ thích hợp. Mỗi nhánh với các thời kì sinh trưởng có cách xử lý giống nhau chỉ khác các giá trị độ ẩm(nhiệt độ) ta so sánh. Node check humidity bé hơn một ngưỡng nhiệt độ dùng để phân luồng các giá trị độ ẩm nằm ngoài khoảng mong muốn, khi độ ẩm thấp thì tạo một message chứa RPC kích hoạt máy bơm và tạo cảnh báo gửi thông báo về thiết bị.



Hình 12: Hình ảnh về node creat RPC

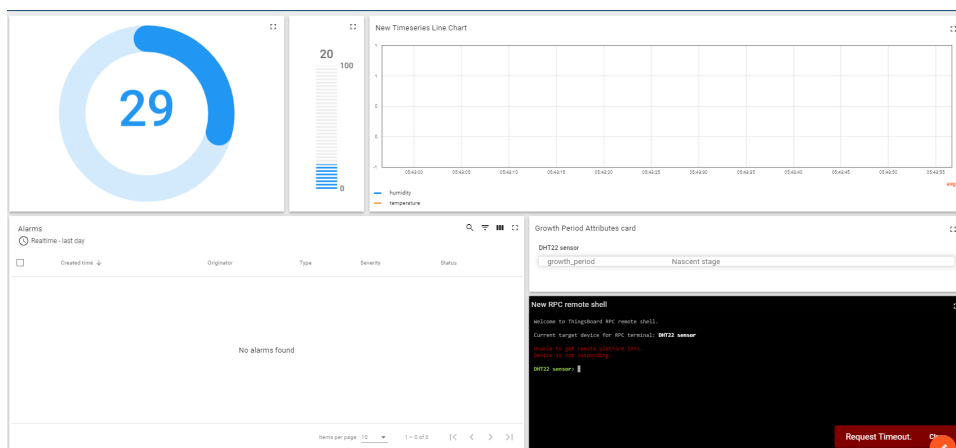


Hình 13: Hình ảnh về node creat alarm

3.5 Trực quan hóa các dữ liệu bằng dashboard

Một trong những điểm mạnh của Thingsboard là cho phép người dùng tạo ra các dashboard giúp cho ta có thể dễ dàng quan sát và giám sát hệ thống tốt hơn.

Hình ảnh dưới đây là hình ảnh thiết kế của giao diện dashboard mà nhóm thiết kế:



Hình 14: Dashboard

Dashboard trên sẽ bao gồm một đồng hồ hiển thị nhiệt độ, một cột hiển thị độ ẩm, một biểu đồ hiển thị độ ẩm và nhiệt độ theo thời gian. Ngoài ra còn một bảng hiển thị các alarm được kích hoạt. Một bảng hiển thị thời kì sinh trưởng hiện tại của cây và 1 terminal dùng để nhập vào thời kì sinh trưởng trong 5 thời kì ở trên thì sẽ có 1 rpc gửi đến để thực hiện việc cập nhật thuộc tính trên . Do đó chính raspberry vẫn sẽ thực thi lệnh nhằm giám xác các thuộc tính trên server cụ thể ở đây ta có thuộc tính là thời kì sinh trưởng.

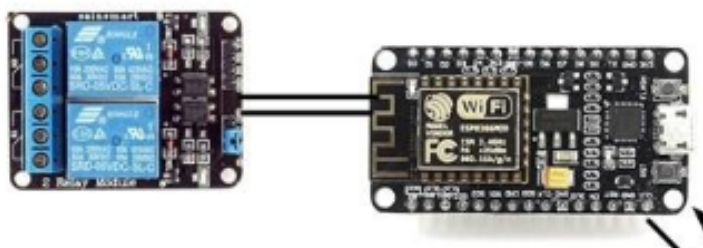
```
1 import random
2 import paho.mqtt.client as mqttclient
3 import time
4 import json
5
6 BROKER_ADDRESS = "localhost"
7 PORT = 1883
8 THINGS_BOARD_ACCESS_TOKEN = "LB4oqgfzi0Ulc9ingWun"
9
10
11 def set_growth_period(temp_data):
12     if temp_data['growth_period'] in ["Nascent stage", "Leaf growth stage", "Flowering stage", "Seed formation stage", "Seed ripening stage"]:
13         print("Growth period now is " + temp_data['growth_period'] )
14         client.publish('v1/devices/me/attributes', json.dumps(temp_data), 1)
15     else:
16         print("Entered growth period doesnot exist \n")
17
18 #code for mqtt connection
19
```



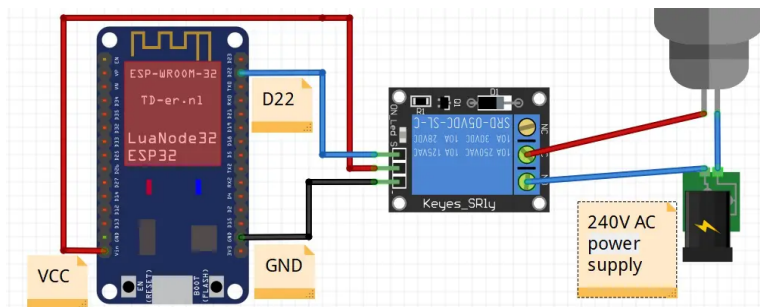
```
20 def subscribed(client, userdata, mid, granted_qos):
21     print("Subscribed...")
22
23
24 def on_message(client, userdata, message):
25     print("Received: ", message.payload.decode("utf-8"))
26     try:
27         json_obj = json.loads(message.payload)
28         if json_obj['method'] == "sendCommand":
29             temp_data = {'growth_period': True}
30             temp_data['growth_period'] = json_obj['params']['command']
31             set_growth_period(temp_data)
32             client.publish(message.topic.replace('request', 'response'))
33
34     except:
35         pass
36
37 def connected(client, userdata, flags, rc):
38     if rc == 0:
39         print("Thingsboard connected successfully!!")
40         client.subscribe("v1/devices/me/rpc/request/+")
41     else:
42         print("Connection is failed")
43
44
45 client = mqttclient.Client("Gateway_Thingsboard")
46 client.username_pw_set(THINGS_BOARD_ACCESS_TOKEN)
47
48 client.on_connect = connected
49 client.connect(BROKER_ADDRESS, 1883)
50 client.loop_start()
51
52 client.on_subscribe = subscribed
53 client.on_message = on_message
54
55
56 client.publish('v1/devices/me/attributes', json.dumps({'growth_period': 'Nascent
    stage'}), 1)
57 try:
58     while(True):
59         time.sleep(15*60)
60 except KeyboardInterrupt:
61     pass
```

3.6 Kích hoạt hệ thống máy tưới nước

Cũng giống như lúc đọc dữ liệu, ta vẫn sẽ dùng ESP32 thông qua giao thức MQTT để nhận message chứa RPC để bật hoạt tắt relay điều khiển nguồn của máy bơm. Ở đây ta sẽ dùng relay đóng ở mức thấp(logic = 0). Nghĩa là khi có thực thi lệnh `active_water_pump()` nhận được khi đọc message chứa RPC được gửi từ raspberry pi, chân GPIO mắc với relay của ESP32 sẽ được đặt về 0, là cho chân điều khiển nguồn của relay tích cực và dẫn điện nguồn của máy tưới. Ngược lại nếu nhận được RPC là `unactive_water_pump()` thì chân GPIO bằng 1 làm cho nguồn của máy tưới bị ngắt.



Hình 15: Dashboard



Hình 16: Dashboard

Hình trên cho thấy được nguyên lý và cách mắc của relay. Ở đây ta sẽ thấy được relay điều khiển nguồn điện phía bên phải nhờ việc đóng ngắt điện của nguồn. Việc đóng ngắt ấy sẽ được điều khiển bằng 1 tín hiệu của vi điều khiển mà cụ thể ở đây là vi điều khiển ESP32.



Code nạp cho mạch ESP32 điều khiển relay:

```
1 import network
2 from umqtt.simple import MQTTClient
3 from machine import Pin
4 from time import sleep
5 import json
6
7 # MQTT
8 SERVER = '127.0.0.1' # MQTT Server Address (Change to the IP address of your Pi)
9 CLIENT_ID = 'LB4oqgfzi0Ulc9ingWun' # access token
10 TOPIC_REQUEST = 'v1/devices/me/telemetry'
11 #wifi SSID and password
12 WIFI_SSID = 'NHAN'
13 WIFI_PASSWORD = '0123456789'
14
15 relay = Pin(2, Pin.OUT)
16
17 # function active pump or unactive
18 def unactive_water_pump():
19     relay.value(1)
20
21 def active_water_pump():
22     relay.value(0)
23
24
25 # WiFi connectivity function.
26 def connectWIFI():
27     wlan = network.WLAN(network.STA_IF)
28     wlan.active(True)
29     wlan.connect(WIFI_SSID, WIFI_PASSWORD)
30     while not wlan.isconnected():
31         pass
32     print("wifi connected")
33     print(wlan.ifconfig())
34
35
36 #MQTT callback service function
37 def sub_cb(topic, message):
38     print("Received: ", message.payload.decode("utf-8"))
39     try:
40         json_obj = json.loads(message.payload)
41         if json_obj['method'] == "active_water_pump":
42             unactive_water_pump()
43             # add response
44         elif json_obj['method'] == "unactive_water_pump":
45             unactive_water_pump()
46         client.publish(message.topic.replace('request', 'response'), message)
47     except:
48         pass
49
50
```

```
51 try:
52     connectWIFI()
53     client = MQTTClient(CLIENT_ID, SERVER)
54     client.connect() # Connect to MQTT broker
55     client.set_callback(sub_cb)
56     client.subscribe(TOPIC_REQUEST)
57     print("MQTT connected and ready to receive message")
58     while True:
59         client.wait_msg()
60 finally:
61     client.disconnect()
```

4 Demo và kiểm thử hệ thống

Việc kiểm thử ở đây chủ yếu sẽ là kiểm tra xem hệ thống thingsboard edge có thực hoạt động tốt không do đó ta sẽ viết và thực thi script python để kiểm thử ở cả hai phía nhận dữ liệu và kích hệ thống tưới nước. Do nhóm thực sự không có đủ phần cứng để có thể thực hiện như thực tế nên các script trên sẽ được chạy trên máy host(raspberry) với các giá trị nhiệt độ và độ ẩm được ta cho sẵn.

4.1 Code python cho việc kiểm thử

Dưới đây là 2 đoạn code python cho việc kiểm tra Thingsboard Edge có thực sự hoạt động hay không:

Code python cho việc kiểm tra gửi dữ liệu từ cảm biến

```
1
2 print("Xin chào ThingsBoard")
3 import random
4 import paho.mqtt.client as mqttclient
5 import time
6 import json
7
8 BROKER_ADDRESS = "127.0.0.1"
9 PORT = 1883
10 THINGS_BOARD_ACCESS_TOKEN = "LB4oqgfzi0Ulc9ingWun"
11
12
13 def read_input():
14     temp = 15 #random.randint(40, 50)
15     humi = 10#random.randint(20, 80)
16     return temp,humi
17
18
19 def subscribed(client, userdata, mid, granted_qos):
20     #print("Subscribed...")
21     return
22
```



```
23
24 def connected(client, usedata, flags, rc):
25     if rc == 0:
26         #print("Thingsboard connected successfully!!")
27         client.subscribe("v1/devices/me/rpc/request/+")
28     else:
29         print("Connection is failed")
30
31
32
33
34
35 client = mqttclient.Client("Gateway_Thingsboard")
36 client.username_pw_set(THINGS_BOARD_ACCESS_TOKEN)
37
38 client.on_connect = connected
39 client.connect(BROKER_ADDRESS, 1883)
40 client.loop_start()
41
42 client.on_subscribe = subscribed
43 temp = 45
44 humi = 50
45 client.publish('v1/devices/me/attributes', json.dumps({'growth_period': 'Nascent
    stage'}), 1)
46 try:
47     while(True):
48         temp,humi = read_input()
49         print("Publish: temperature :"+ str (temp)+ ", humidity :"+ str(humi))
50         collect_data = {'temperature': temp,'humidity': humi}
51         client.publish('v1/devices/me/telemetry', json.dumps(collect_data), 1)
52         time.sleep(30)
53 except KeyboardInterrupt:
54     pass
```

Code python cho việc kiểm tra nhận dữ liệu đến máy tưới

```
1 print("Xin cho ThingsBoard")
2 import random
3 import paho.mqtt.client as mqttclient
4 import time
5 import json
6
7 BROKER_ADDRESS = "127.0.0.1"
8 PORT = 1883
9 THINGS_BOARD_ACCESS_TOKEN = "LB4oqgfzi0Ulc9ingWun"
10
11
12
13 def set_growth_period(temp_data):
14     if temp_data['growth_period'] in ["Nascent stage","Leaf growth stage","Flowering
        stage","Seed formation stage","Seed ripening stage"]:
15         print("Growth period now is " + temp_data['growth_period'] )
```

```
16         client.publish('v1/devices/me/attributes', json.dumps(temp_data), 1)
17     else:
18         print("Entered growth period doesnot exist \n")
19
20 def active_water_pump():
21     print("watering...")
22
23 def unactive_water_pump():
24     print("Stop watering!")
25
26 def subscribed(client, userdata, mid, granted_qos):
27     #print("Subscribed...")
28     return
29
30
31 def on_message(client, userdata, message):
32     print("Received: ", message.payload.decode("utf-8"))
33     try:
34         json_obj = json.loads(message.payload)
35         if json_obj['method'] == "active_water_pump":
36             client.publish(message.topic.replace('request', 'response'),
37                             active_water_pump(), 1)
38         elif json_obj['method'] == "unactive_water_pump":
39             client.publish(message.topic.replace('request', 'response'),
40                             unactive_water_pump(), 1)
41         elif json_obj['method'] == "sendCommand":
42             temp_data = {'growth_period': True}
43             temp_data['growth_period'] = json_obj['params']['command']
44             client.publish(message.topic.replace('request',
45             'response'),set_growth_period(temp_data), 1)
46
47     except:
48         pass
49     '''
50     temp_data = {'value': True}
51     try:
52         jsonobj = json.loads(message.payload)
53         if jsonobj['method'] == "setValue":
54             temp_data['value'] = jsonobj['params']
55             client.publish('v1/devices/me/attributes', json.dumps(temp_data), 1)
56     except:
57         pass
58     '''
59
60 def connected(client, userdata, flags, rc):
61     if rc == 0:
62         #print("Thingsboard connected successfully!!")
63         client.subscribe("v1/devices/me/rpc/request/+")
64     else:
65         print("Connection is failed")
66
67 client = mqttclient.Client("Gateway_Thingsboard")
```

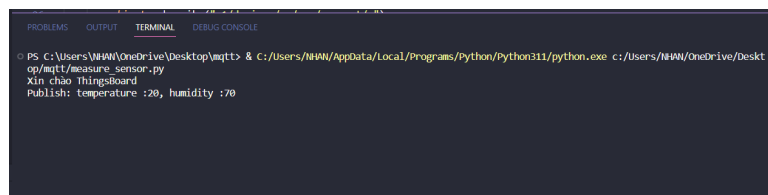


```
66 client.username_pw_set(THINGS_BOARD_ACCESS_TOKEN)
67
68 client.on_connect = connected
69 client.connect(BROKER_ADDRESS, 1883)
70 client.loop_start()
71
72 client.on_subscribe = subscribed
73 client.on_message = on_message
74
75
76 try:
77     while(True):
78         time.sleep(10)
79 except KeyboardInterrupt:
80     pass
```

4.2 Kết quả

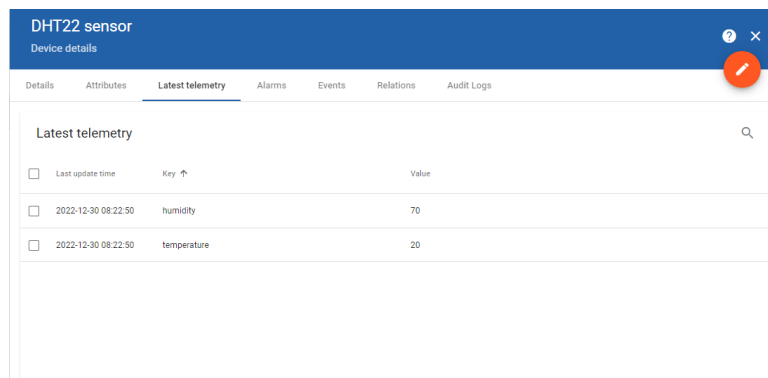
4.2.1 Kiểm tra dữ liệu gửi từ Edge lên Cloud

Đầu tiên ta sẽ cho temperature = 20, humidity = 70 và ban đầu thời kì sinh trưởng của cây là giai đoạn mọc (Nascent stage).

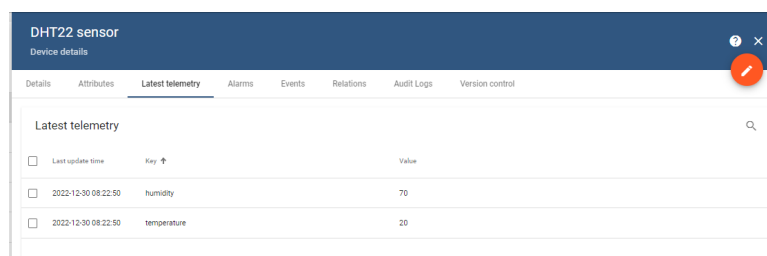


Hình 17: Ảnh trên terminal cho thấy data đã được gửi

Dữ liệu sẽ được nhận ở Edge Node và sau đó sẽ được gửi lên lại cloud.



Hình 18: Device nhận được data gửi đến trên Edge



DHT22 sensor		
Device details		
Latest telemetry		
Key	Value	
humidity	70	
temperature	20	

Hình 19: Device nhận được data gửi đến trên cloud

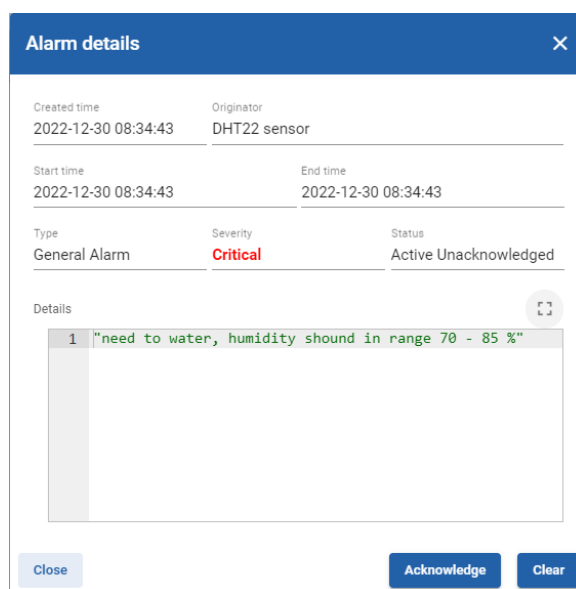
4.2.2 Kiểm tra xử lý của rule chain

Ta sẽ cho temperature = 20, humidity = 30 và ban đầu thời kì sinh trưởng của cây là giai đoạn mọc (Nascent stage) để kiểm tra có kích hoạt máy bơm và tạo alarm hay không.

Ở trường hợp này, bên phía máy tưới trên terminal ta sẽ nhận được RPC và một alarm được thông báo đến DHT22 trên Thingsboard Edge.

```
PS C:\Users\NHN> & C:\Users\NHN\AppData\Local\Programs\Python\Python311\python.exe c:\Users\NHN\OneDrive\Desktop\mqtt_send_rpc.py
Xin chào ThingsBoard
Received: {"method":"active_water_pump","params":{"growth_period":"Nascent stage"}}
watering...
```

Hình 20: Hệ thống máy tưới nhận và thực thi lệnh bật



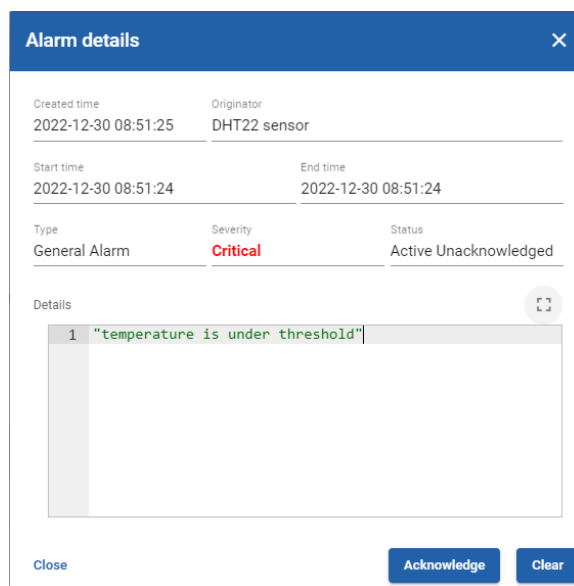
Alarm details		
Created time	2022-12-30 08:34:43	
Originator	DHT22 sensor	
Start time	2022-12-30 08:34:43	
End time	2022-12-30 08:34:43	
Type	Severity	Status
General Alarm	Critical	Active Unacknowledged
Details		
1 "need to water, humidity should in range 70 - 85 %"		
Close Acknowledge Clear		

Hình 21: alarm khi độ ẩm dưới ngưỡng đã cho

ta sẽ xem tiếp một trường hợp khác là $\text{temperature} = 16$, $\text{humidity} = 90$ vẫn ở $\text{growth_period} = \text{"Nascent stage"}$, với trường hợp này thì ở thiết bị máy tưới sẽ nhận lệnh tắt máy tưới và gửi về alarm.

```
PS C:\Users\NHNAN & C:/Users/NHNAN/AppData/Local/Programs/Python/Python311/python.exe c:/Users/NHNAN/OneDrive/Desktop/mqtt/send_rpc.py
Xin chào ThingsBoard
Received: {"method":"active_water_pump","params":{"growth_period":"Nascent stage"}}
watering...
Received: {"method":"unactive_water_pump","params":{"growth_period":"Nascent stage"}}
Stop watering!
```

Hình 22: Hệ thống máy tưới nhận và thực thi lệnh tắt



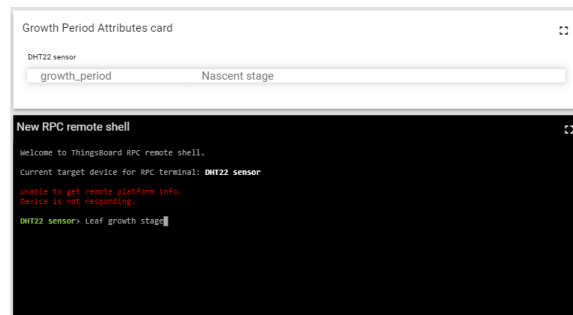
Alarm details		
Created time	Originator	
2022-12-30 08:51:25	DHT22 sensor	
Start time	End time	
2022-12-30 08:51:24	2022-12-30 08:51:24	
Type	Severity	Status
General Alarm	Critical	Active Unacknowledged
Details		
1 "temperature is under threshold"		
<div>Close Acknowledge Clear</div>		

Hình 23: alarm khi nhiệt độ ngoài ngưỡng đã cho

4.2.3 Kiểm tra thay đổi thời kì sinh trưởng

Để thay đổi thuộc tính thời kì sinh trưởng(growth_period) ta sẽ nhập vào terminal ở trên dashboard với đúng trong 5 thời kì sinh trưởng mà nhóm đã đề ra thì thuộc tính sẽ được cập nhập, nếu khác 5 giá trị trên thì terminal sẽ 1 ra 1 dòng thông báo ko hợp lệ.

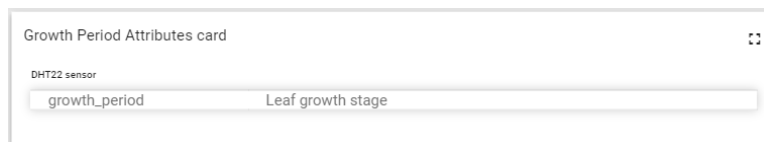
Khi ta nhập vào terminal đó là "Leaf growth stage", với trạng thái hiện tại là "Nascent stage"



Ta sẽ thu được kết quả trên terminal

```
Received: {"method":"getTermInfo","params":null}
Received: {"method":"unactive_water_pump","params":{"growth_period":"Nascent stage"}}
Stop watering!
Received: {"method":"getTermInfo","params":null}
Received: {"method":"unactive_water_pump","params":{"growth_period":"Nascent stage"}}
Stop watering!
Received: {"method":"unactive_water_pump","params":{"growth_period":"Nascent stage"}}
Stop watering!
Received: {"method":"getTermInfo","params":null}
Received: {"method":"sendCommand","params":{"command":"Leaf growth stage"}}
Growth period now is Leaf growth stage
```

Bảng hiển thị trên dashboard cũng sẽ thay đổi.



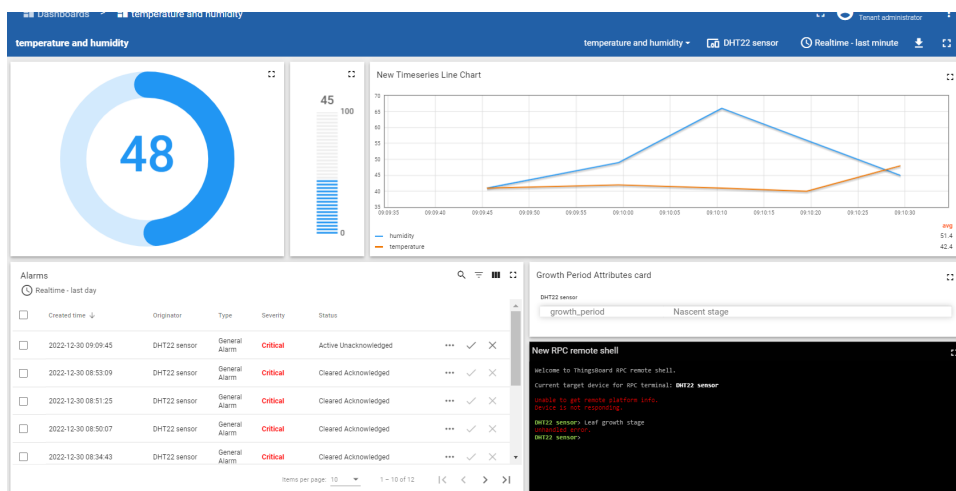
Kết quả khi nhập vào 1 giá trị khác với 5 thời kì đã được đề cập.

```
PS C:\Users\NHN> & C:\Users\NHN\AppData\Local\Programs\Python\Python311\python.exe c:/Users/NHN/OneDrive/Desktop/mqtt/send_rpc.py
Xin chào ThingsBoard
Received: {"method":"sendCommand","params":{"command":"ahfadsf"}}
Entered growth period doesnot exist
```



4.3 Kiểm tra dashboard

Khi gửi dữ liệu được 1 khoảng thời gian liên tục thì ta có thể có 1 biểu đồ để quan sát nhiệt độ được gửi đến.



5 Mức độ hiệu quả của hệ thống.

5.1 Ưu điểm

- Có thể sử dụng một máy ảo hoặc một bản giả lập của Raspberry Pi để thực hiện thành công mô hình này. Sử dụng Docker giúp cho công việc cài đặt triển khai Edge dễ dàng hơn.
- Edge trên Raspberry Pi đã có thể tự xử lý và sàng lọc dữ liệu gửi lên server khi có kết nối Internet.
- Thử nghiệm được một chương trình sinh dữ liệu trực tiếp làm đầu vào dữ liệu cho mô hình.
- Tín hiệu gửi về sau khi được Edge xử lý hiện thực bằng cách nâng mức điện thế cho thiết bị nhận phản hồi bắt đầu chạy có thể áp dụng để chạy nhiều thiết bị khác nhau, từ đó áp dụng được cho nhiều mô hình IoT có chức năng khác nhau. Mô hình này là khởi đầu và giúp ta hiểu được sơ bộ về việc áp dụng Edge computing trên các hệ thống lớn hơn và phức tạp hơn.

5.2 Nhược điểm.

- Vẫn chưa sử dụng được sensor thực gửi về dữ liệu thực theo thời gian, còn thiếu thốn nhiều phần cứng, chủ yếu sử dụng giả lập hoặc phần mềm.
- Máy ảo hay giả lập Raspberry hiện tại đều thiếu ổn định, còn thiết bị Raspberry thật thì khá dễ hỏng, mô hình triển khai Edge trên Raspberry này sẽ không được ưa chuộng nhiều.
- Sau khi cập nhật thay đổi trên Server thì mất khá lâu để Edge nhận được và đồng bộ thay đổi.

5.3 Khả năng mở rộng hệ thống.

- Hệ thống có thể mở rộng bằng cách sử dụng thiết bị Raspberry Pi thật thay vì máy ảo và giả lập. Raspberry Pi có hệ thống Ram riêng sẽ hoạt động độc lập tốt hơn và triển khai hoàn thiện hơn, đôi khi là nhanh hơn. Việc có thiết bị thật tách riêng độc lập giúp chúng ta có thể thao tác cài đặt các ứng dụng cần thiết dễ dàng hơn trên thiết bị.
- Sử dụng Internet làm môi trường truyền nhận thông tin và tín hiệu giúp tăng độ linh hoạt: các phần của mô hình như sensor, edge và server có thể hoạt động mặc dù ở những vị trí cách xa nhau.
- Có thể triển khai Edge trên các thiết bị khác với Raspberry Pi để phục vụ cho những chức năng khác nhau hoặc để ứng dụng các lợi thế về tốc độ dữ liệu, khả năng bảo mật,... của phần cứng triển khai.
- Phần Server có thể triển khai trên máy tính lớn và hiện đại hơn, các cụm máy tính server tối ưu hơn giúp kết nối và điều khiển được nhiều Edge hơn, tốc độ nhanh hơn và bảo quản, lưu trữ được nhiều thông tin hơn.
- Phần sensor gửi dữ liệu có thể triển khai trên nhiều loại sensor và dữ liệu khác nhau nếu chuyển được dữ liệu sang dạng số.

6 Tổng kết

- Edge computing đã xuất hiện như một giải pháp trong thời đại Internet vạn vật mà có hàng tỉ thiết bị được kết nối internet, có thể xử lý một cách hiệu quả với lượng dữ liệu khổng lồ mà các thiết bị truyền tải.
- Edge computing là một trong những cải tiến công nghệ hứa hẹn trong tương lai và đang nhanh chóng trở thành một cơ sở hạ tầng quan trọng. Đó là vì tiềm năng của Edge computing được kết hợp với các công nghệ khác như AI, đám mây và mạng 5G. Mô hình nông nghiệp thông minh đã bắt đầu áp dụng Edge computing vì những lợi ích của nó mang lại cho việc quản lý trồng trọt, chăn nuôi một cách hiệu quả. Không chỉ dừng ở lĩnh vực nông nghiệp mà còn vô vàn các lĩnh vực khác mà Edge computing rất có tiềm năng để phát triển như ngân hàng, quản lý đô thị, tự động hóa công nghiệp,...
- Nhờ các ưu việt và mạnh mẽ trong bước đầu phát triển ở nhiều lĩnh vực, đặc biệt trên nền tảng kỹ thuật số, có thể nói rằng điện toán biên là xu hướng tất yếu cho sự phát triển các dịch vụ về công nghệ thông tin cho các ngân hàng và tổ chức tài chính.