**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**--- VIỆN ĐIỆN ---**

\*\*\*

Ảnh có chứa văn bản, ký hiệu

Mô tả được tạo tự động

**BÁO CÁO BÀI TẬP LỚN**

**Bài tập lớn cuối kỳ môn Mạng và các giao thức IoT**

**Thiết bị Phát hiện và Thống kê thời gian dừng lại của khách hàng trước sản phẩm**

**Họ và Tên : Trần Quang Minh MSSV : 20181659**

**Họ và Tên : Chu Quốc Đạt MSSV : 20181383**

**Họ và Tên : Đồng Xuân Luân MSSV : 20181634**

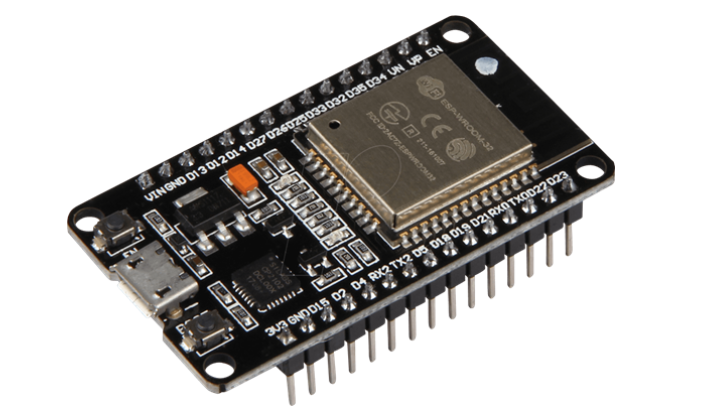
**Lớp học: EE4266 – 133265**

**Giảng viên hướng dẫn: TS. Nguyễn Quốc Cường**

# **Giới Thiệu**

Bài tập lớp của nhóm 15 môn Giao thức Mạng và IoT có mục đích thiết kế hệ thống cảm biến chuyển động của người phục vụ cho bài toán khảo sát thời gian khách hàng đứng lại trước sản phẩm. Hệ thống của nhóm sử dụng module ESP32, cảm biến chuyển động PIR HC-SR501, giao thức MQTT với Broker là Mosquitto chạy Local-host và phần mềm thiết kế app điện thoại MIT App Inventor. Công việc của nhóm bao gồm thiết kế hệ thống truyền nhận dữ liệu giữa các thiết bị node (ESP và cảm biến) với Server (MQTT Broker) và thiết bị người dùng (App Mobile).

# **Tổng quan về Module ESP32**



*Hình 1 ESP32-WROOM-32 DEVBOARD*

## **Thông tin cơ bản về Module ESP32**

* 30 chân GPIO.
* Điện áp logic là 2,7 đến 3,3 V.
* 1 Enable Button (Chân reset) và 1 User Button(GPIO 0).
* Led báo nguồn và User Led (GPIO 2).
* 3 UART: Serial Debug mặc định là UART 0.
* Clock: 240MHz



Hình 2 ESP32 Board Pinout

**Input Only Pins:**

GPIO 34 đến 39 – input only pins. Các chân này không có nội trở kéo lên hoặc kéo xuống. Chúng không thể được sử dụng làm output, vì vậy chỉ sử dụng các chân này làm input

### **Analog to Digital Converter (ADC):**

ESP32 có 18 kênh input ADC và độ phân giải 12 bit (trong khi ESP8266 chỉ có 1 kênh ADC độ phân giải 10 bit). Đây là các GPIO có thể được sử dụng làm ADC và các kênh tương ứng:

* ADC1\_CH0 (GPIO 36)
* ADC1\_CH1 (GPIO 37)
* ADC1\_CH2 (GPIO 38)
* ADC1\_CH3 (GPIO 39)
* ADC1\_CH4 (GPIO 32)
* ADC1\_CH5 (GPIO 33)
* ADC1\_CH6 (GPIO 34)
* ADC1\_CH7 (GPIO 35)
* ADC2\_CH0 (GPIO 4)
* ADC2\_CH1 (GPIO 0)
* ADC2\_CH2 (GPIO 2)
* ADC2\_CH3 (GPIO 15)
* ADC2\_CH4 (GPIO 13)
* ADC2\_CH5 (GPIO 12)
* ADC2\_CH6 (GPIO 14)
* ADC2\_CH7 (GPIO 27)
* ADC2\_CH8 (GPIO 25)
* ADC2\_CH9 (GPIO 26)

### **Digital to Analog Converter (DAC):**

* DAC1 (GPIO25)
* DAC2 (GPIO26)

### **PWM:**

Bộ điều khiển ESP32 LED PWM có 16 kênh độc lập có thể được cấu hình để tạo tín hiệu PWM với các đặc tính khác nhau. Tất cả các chân có thể hoạt động như output đều có thể được sử dụng làm chân PWM (GPIO từ 34 đến 39 không thể tạo PWM). Các thông số cấu hình:

* Tần số tín hiệu.
* Duty cycle.
* Kênh PWM.
* Chân GPIO xuất tín hiệu.

### **I2C**

ESP32 có hai kênh I2C và bất kì chân nào cũng có thể cấu hình làm chân SDA và SCL. Khi sử dụng ESP32 với Arduino IDE, chân I2C mặc định là:

* GPIO 21 (SDA)
* GPIO 22 (SCL)

### **SPI:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SPI** | **MOSI** | **MISO** | **CLK** | **CS** |
| **VSPI** | GPIO 23 | GPIO 19 | GPIO 18 | GPIO 5 |
| **HSPI** | GPIO 13 | GPIO 12 | GPIO 14 | GPIO 15 |

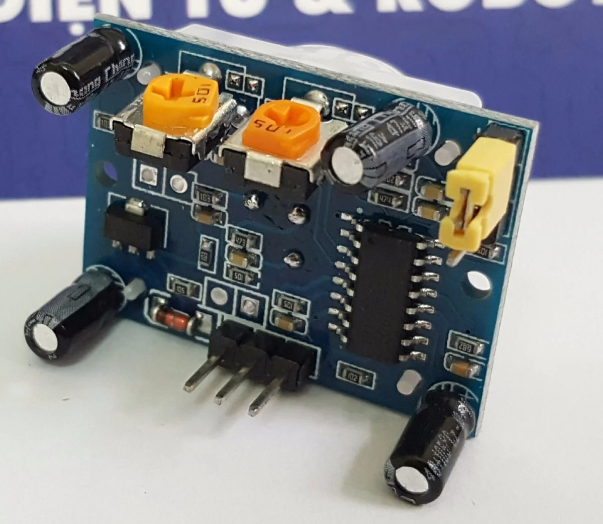
Bảng 1 ESP32 board SPI pinout

* ESP32 có module giao tiếp SPI với thiết bị ngoại vi, gọi là SPI0, SPI1, HSPI, VSPI. SPI0 chỉ dành riêng để kết nối bộ nhớ flash của ESP32 với các thiết bị bố nhớ flash khác bên ngoài.
* SPI1 được kết nối cũng tương tự như SPI0 nhưng nó dùng để ghi dữ liệu cho bộ nhớ Flash của chip.
* HSPI và VSPI và sử dụng tự do. SPI1 và HSPI và VSPI đều có 3 cổng kết với chip, giúp chúng ta dễ dàng kết nối đồng thời với 3 slave bằng giao tiếp SPI mà ESP32 sẽ là thiết bị master.

### **Interrupts**

Tất cả các GPIO có thể được cấu hình như ngắt.

# **Tổng quan cảm biến PIR HC-SR501**



Hình 3 *Cảm biến PIR HC-SR501*

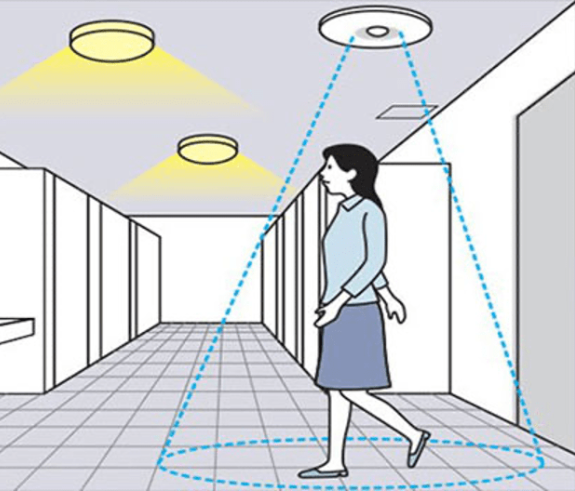
Cảm biến thân nhiệt chuyển động PIR (Passive infrared sensor) HC-SR501 được sử dụng để phát hiện chuyển động của các vật thể phát ra bức xạ hồng ngoại (con người, con vật, các vật phát nhiệt,...), cảm biến có thể chỉnh được độ nhạy để giới hạn khoảng cách bắt xa gần cũng như cường độ bức xạ của vật thể mong muốn, ngoài ra cảm biến còn có thể điều chỉnh thời gian kích trễ (giữ tín hiệu bao lâu sau khi kích hoạt) qua biến trở tích hợp sẵn.

## **Thông số kỹ thuật:**

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

## **Nguyên lý hoạt động:**

Nguyên lý cảm biến vật cản hồng ngoại thụ động (PIR) hoạt động là sử dụng một cặp cảm biến nhiệt điện để phát hiện năng lượng nhiệt từ sóng hồng ngoại trong môi trường xung quanh. Hai cảm biến này được thiết kế nằm cạnh nhau và khi có sự chênh lệch về tín hiệu giữa cả hai cảm biến này (ví dụ: nếu một người vào phòng), cảm biến vật cản hồng ngoại sẽ hoạt động. Thiết bị sẽ kích hoạt báo động, thông báo cho cơ quan chức năng (có thể là loa báo, đèn báo…) hoặc có thể tự động bật tắt đèn, quạt. Bức xạ hồng ngoại tập trung vào mỗi trong số hai cảm biến nhiệt điện bằng cách sử dụng một loạt thấu kính được gọi là thấu kính Fresnel được cấu tạo như vỏ của cảm biến. Những thấu kính này giúp mở rộng vùng cảm nhận của cảm biến có thể thấy hầu hết ở vị trí mắt cảm biến hồng ngoại.



Hình 4 Minh họa tầm hoạt động của cảm biến PIR

# **Tổng quan về giao thức MQTT**

## **Định nghĩa:**

MQTT (Message Queuing Telemetry Transport) 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 là lựa chọn lý tưởng trong các môi trường như:

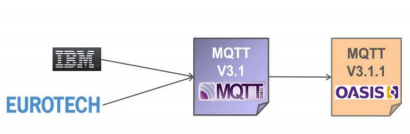
* Những nơi mà giá mạng viễn thông đắt đỏ hoặc băng thông thấp hay thiếu tin cậy.
* Khi chạy trên thiết bị nhúng bị giới hạn về tài nguyên tốc độ và bộ nhớ.
* Bởi vì giao thức này sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng M2M (Machine to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

## **Lịch sử hình thành:**

MQTT được phát minh bởi Andy Stanford - Clark (IBM) và Arlen Nipper (EUROTECH) cuối năm 1999 khi mà nhiệm vụ của họ là tạo ra một giao thức sao cho sự hao phí năng lượng và băng thông là thấp nhất để kết nối đến đường ống dẫn dầu thông qua sự kết nối của vệ tinh.

Năm 2011, IBM và Eurotech đã trao lại MQTT cho một dự án của Eclipse có tên là Paho.

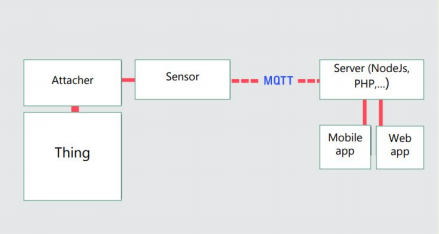
Năm 2013 MQTT đã được đệ trình lên OASIS (Organization for the Advancement of Structured Information Standards) để chuẩn hóa.



Hình 5 Lịch sử của MQTT

## **Vị trí của MQTT trong mô hình IoT**

Một số ưu điểm nổi bật của MQTT như: băng thông thấp, độ tin cậy cao và có thể sử dụng ngay cả khi hệ thống mạng không ổn định, tốn rất ít byte cho việc kết nối với server và connection có thể giữ trạng thái open xuyên suốt, có thể kết nối nhiều thiết bị (MQTT client) thông qua một MQTT server (broker). 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 IoT.



Hình 6 Mô tả chức năng của MQTT trong mô hình IoT

## **Tính năng, đặc điểm nổi bật**

* Dạng truyền thông điệp theo mô hình Pub/Sub cung cấp việc truyền tin phân tán một chiều, tách biệt với phần ứng dụng.
* Việc truyền thông điệp là ngay lập tức, không quan tâm đến nội dung được truyền.
* Sử dụng TCP/IP là giao thức nền.
* Tồn tại ba mức độ tin cậy cho việc truyền dữ liệu (QoS: Quality of service)
* QoS 0: Broker/client sẽ gửi dữ liệu đúng một lần, quá trình gửi được xác nhận bởi chỉ giao thức TCP/IP.
* QoS 1: Broker/client sẽ gửi dữ liệu với ít nhất một lần xác nhận từ đầu kia, nghĩa là có thể có nhiều hơn 1 lần xác nhận đã nhận được dữ liệu.
* QoS 2: Broker/client đảm bảo khi gửi dữ liệu thì phía nhận chỉ nhận được đúng một lần, quá trình này phải trải qua 4 bước bắt tay.
* Phần bao bọc dữ liệu truyền nhỏ và được giảm đến mức tối thiểu để giảm tải cho đường truyền.

## **Ưu điểm của MQTT**

Với những tính năng, đặc điểm nổi bật 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 do thám.
* 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.

## **Mô hình Pub/Sub và Cơ chế hoạt động của MQTT**

1. Thành phần

Client

* Publisher - Nơi gửi thông điệp
* Subscriber - Nơi nhận thông điệp

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 thì 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.

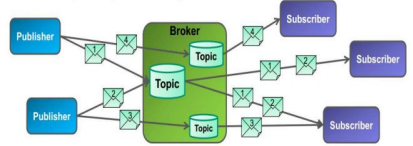
1. Ưu điểm

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

1. 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.

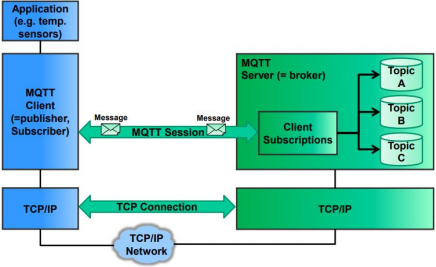
## **Cơ chế tổng quan**



Hình 7 Tổng quan cách hoạt động của MQTT

* 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 kí 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.

## **Kiến trúc thành phần**



Hình 8 Kiến trúc của MQTT

* Thành phần chính của MQTT là Client (Publisher/Subscriber), Server (Broker), Sessions (tạm dịch là Phiên làm việc), Subscriptions và Topics.
* MQTT Client (Publisher/Subscriber): Clients sẽ subscribe một hoặc nhiều topics để gửi và nhận thông điệp từ những topic tương ứng.
* 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.
* 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.
* 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.
* 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.

# **Tổng quan về Mosquitto**

Mosquitto là một MQTT Broker mã nguồn mở cho phép thiết bị truyền nhận dữ liệu theo giao thức MQTT version 5.0, 3.1.1 và 3.1 – Một giao thức nhanh, nhẹ theo mô hình publish/subscribe được sử dụng rất nhiều trong lĩnh vực Internet of Things. Mosquitto cung cấp một thư viện viết bằng ngôn ngữ C để triển khai các MQTT Client và có thể dễ dàng sử dụng bằng dòng lệnh: “mosquitto\_pub” và “mosquitto\_sub”.

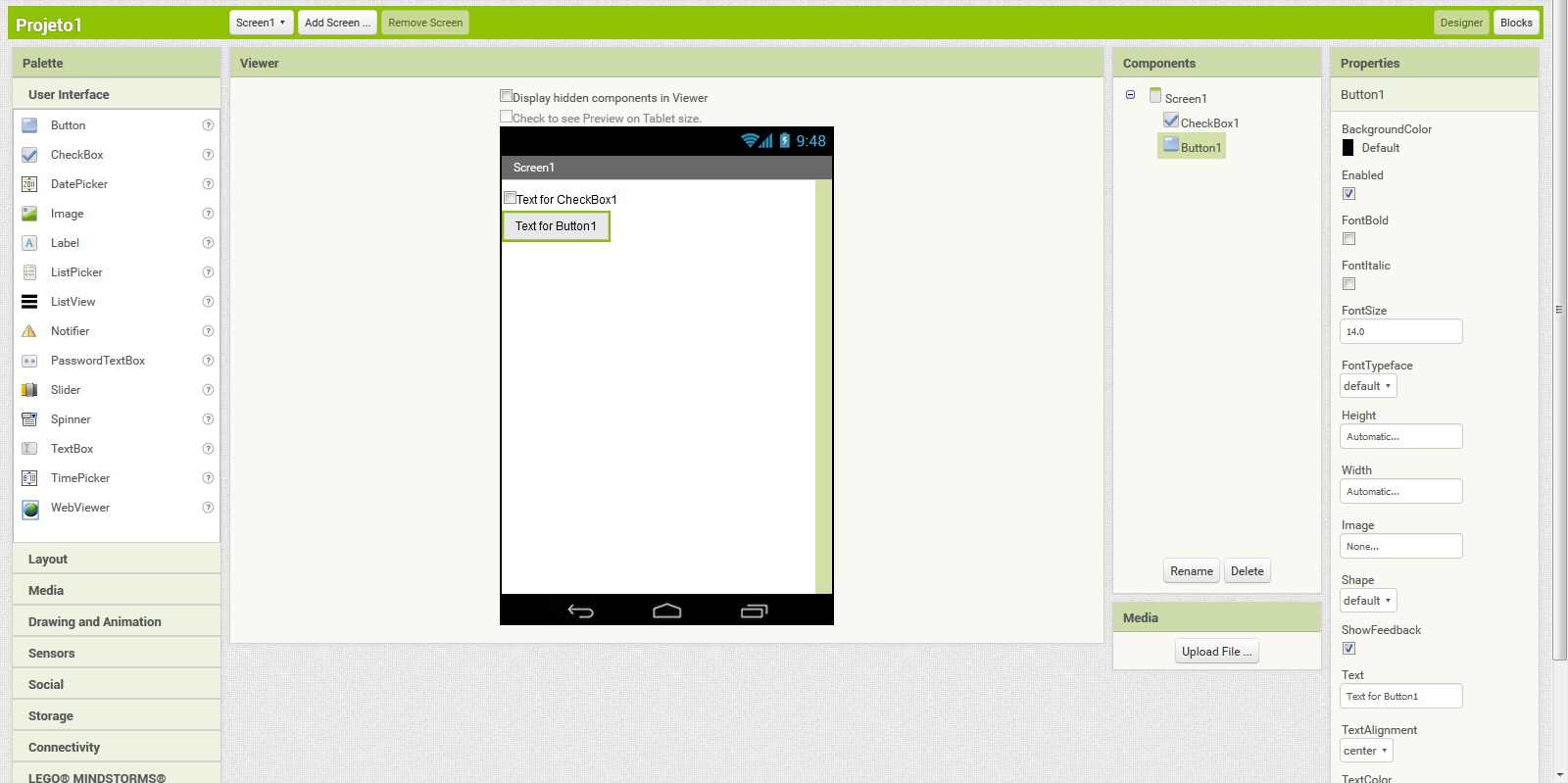
* Ưu điểm:
  + Ưu điểm nổi bật của Mosquitto là tốc độ truyền nhận và xử lí dữ liệu nhanh, độ ổn định cao, được sử dụng rộng rãi và phù hợp với những ứng dụng embedded.
  + Mosquitto rất nhẹ và phù hợp để sử dụng trên tất cả các thiết bị.
  + Ngoài ra, Mosquitto cũng được hỗ trợ các giao thức TLS/SSL (các giao thức nhằm xác thực server và client, mã hóa các message để bảo mật dữ liệu).
* Nhược điểm:
  + Một số nhược điểm của mosquitto là khó thiết kế khi làm những ứng dụng lớn và ít phương thức xác thực thiết bị nên khả năng bảo mật vẫn chưa tối ưu.

1. **Tổng quan về MIT App Inventor**

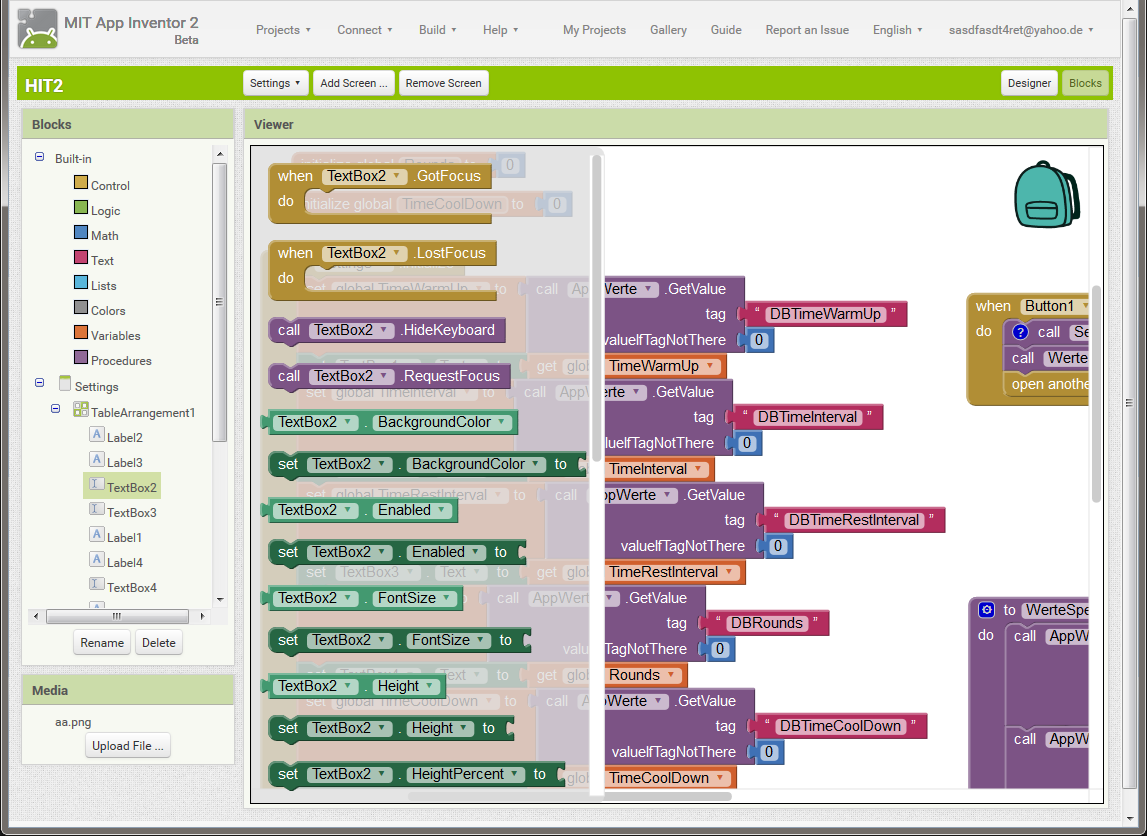
MIT App Inventor là một trang web tích hợp môi trường phát triển được cung cấp bởi Google và hiện nay được cung cấp bởi Massachusetts Institute of Technology (MIT). Nó cho phép những lập trình viên mới bắt đầu lập trình có thể dễ dàng thiết kế được những ứng dụng trên điện thoại Android.

Giao diện phát triển của MIT sử dụng giao diện đồ họa người dùng dạng kéo thả thân thiện và dễ sử dụng.

MIT App Inventor có đường dẫn là <https://appinventor.mit.edu>

****

Hình Giao diện thiết kế của MIT App Inventor

****

Hình Giao diện lập trình kéo thả của MIT App Inventor

# **Yêu cầu thiết kế**

1. Yêu cầu phần cứng

Thiết bị sử dụng ESP32 và cảm biến PIR là thiết bị cảm biến chuyển động, thiết bị trong hệ thống gọi là node.

Mỗi node sẽ có điện áp hoạt động là 5V. Các node sẽ phải kết nối với cùng mạng WiFi với MQTT Broker để có thể kết nối được với Server.

Broker MQTT sẽ phải được host trên một máy tính.

1. Yêu cầu phần mềm thiết bị

Thiết bị node có khả năng phát hiện người (khách hàng) khi họ đi vào vùng chưng bày sản phẩm. Khi khách hàng vẫn còn trong vùng chưng bày, thiết bị tiến hành đếm thời gian theo đơn vị nhỏ nhất là giây. Khi khách hàng đi ra khỏi vùng chưng bày, dữ liệu thời gian này sẽ được đưa lên Server và hiển thị lên App.

Thiết bị gửi bản tin MQTT lên kênh đăng ký đã được định sẵn, gói tin gửi lên phải chức thông tin của node (mã của node), thông tin về trạng thái khách hàng (còn trong vùng chưng bày hay không), thời gian khách hành đứng trong vùng sản phẩm. Bản tin sẽ được gửi khi có khách đi ra hoặc đi vào qua node cảm biến, và liên tục với tần suất 1 giây một bản tin khi có khách hành trong vùng chưng bày sản phẩm.

Thiết bị phải có khả năng tự động kết nối lại WiFi và kết nối với MQTT Broker trong trường hợp mất mạng hoặc mất kết nối với Broker hoặc cả 2.

1. Yêu cầu phần mềm ứng dụng

Phần mềm trên giao diện người dùng cuối cần phải được thiết kế thân thiện và dễ dàng thao tác. Quá trình cài đặt ứng dụng phải dễ dàng và thuận tiện nhất.

Phần mềm phải hiện thị được toàn bộ trạng thái của từng node và thời gian khách đi vào vùng sản phẩm của mỗi node, bao gồm thời gian hiện tại hoặc lần gần nhất và thời gian tổng với mỗi node. Ngoài ra, giao diện cần phải có phần cài đặt địa chỉ MQTT Broker và kênh để lắng nghe dữ liệu truyền lên.

# **Nhật ký cuộc họp nhóm**

**Tuần 1: ngày 19 tháng 06 năm 2022**

**Địa điểm:** MS Team

**Tham dự:** Trần Quang Minh, Chu Quốc Đạt (Đồng Xuân Luân ốm không tham dự họp được)

**Nội dung:** Xác định rõ yêu cầu của bài toán, phân tích bài toán, phân tích thành phần, mô tả lại hệ thống. Một vài đầu mục được thống nhất như sau:

**Các vấn đề thảo luận và đã thống nhất trong cuộc họp:**

1. **Chốt phương án 2 thiết bị (node) sử dụng ESP32 code bằng Arduino**
2. **Server là người đếm thời gian**
3. **Thống nhất bản tin:**

Tên topic: "Human\_Present"

Cấu trúc bản tin:

***{<số thứ tự node cảm biến>,<trạng thái khách>}***

*Trong đó:*

*trạng thái 1 là khách đến, trạng thái 0 là khách đi*

1. **Mô tả hiển thị dữ liệu trên app PC và app điện thoại:**

[15:45:35] - Quầy 1, có người đến

[15:46:00] - Quầy 1, có người rời đi, thời gian khách hàng dừng lại 25s

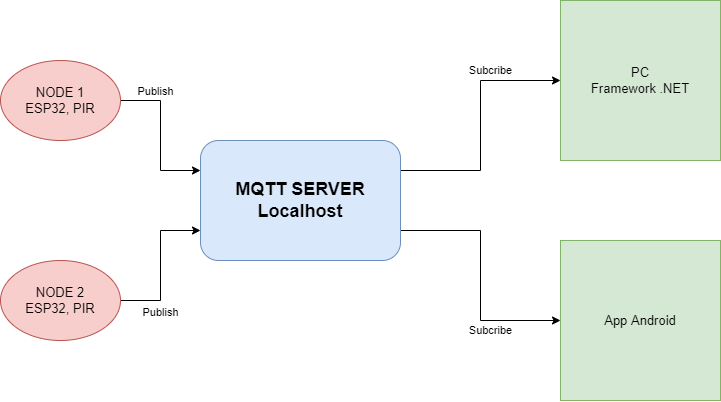
1. **Bản tin mẫu:**

{1,1}

{2,0}

{1,0}

1. **Sơ đồ tổng qua hệ thống phải xây dựng:**



Hình 11 Sơ đồ tổng quan hệ thống

**Nhiệm vụ và báo cáo tuần 1:**

* Tìm hiểu cách truy cập mqtt từ framework .net (app PC)
  + *Chưa xong*
* Thử nghiệm viết app Android bằng MIT (app điện thoại)
  + *Luân đã làm 1 lần*
* Thử nghiệm Localhost MQTT (Đạt)
  + *Đã chạy được MQTT Server trên máy tính cá nhân, thử nghiệm pub/sub thành công trên máy tính khác trong cùng một mạng qua ứng dụng MQTT.FX*
* Mua cảm biến PIR thử nghiệm (Ven)
  + *Đã hoàn thành thử nghiệm tín hiệu của cảm biến trả về khi có chuyển động*

**Tuần 2: ngày 28 tháng 06 năm 2022**

**Địa điểm:** MS Team

**Tham dự:** Trần Quang Minh, Chu Quốc Đạt, Đồng Xuân Luân.

**Nội dung:** Báo cáo tiến độ của các nhiệm vụ đã giao ở tuần 1, tiến hành triển khai các kế hoạch thiết kế, thử nghiệm dựa trên các kết quả đã có. Triển khai file báo cáo chung theo format.

**Các vấn đề thảo luận và đã thống nhất trong cuộc họp:**

1. **Chỉnh lại cấu trúc bản tin:**

Tên topic: "Human\_Present"

Cấu trúc bản tin:

***{<số thứ tự node cảm biến>,<trạng thái khách>,<thời gian>}***

*Trong đó:*

*Trạng thái 1 là khách đến, trạng thái 0 là khách đi*

*Thời gian là thời gian khách đứng lại*

1. **Bản tin mẫu:**

{1,1,0}

{2,0,30}

{1,0,45}

1. **Bỏ chức năng hiển thị dữ liệu lên máy tính, chỉnh lại sơ đồ khối tổng quát hệ thống:**

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 12 Sơ đồ khối tổng quát hệ thống

**Nhiệm vụ và báo cáo tuần 2:**

* Thiết kế bản thử nghiệm của thiết bị cảm biến (2 nodes), thử kết nối với MQTT Server LocalHost và trao đổi bản tin.
  + *Thiết bị đã kết nối WiFi, MQTT Server (chưa kết nối với MQTT LocalHost, chưa có Authen, triển khai bằng Arduino), đã triển khai cấu trúc bản tin theo dạng file .json, thử nghiệm cảm biến PIR, demo thiết bị (node)*

**Ảnh có chứa văn bản

Mô tả được tạo tự động**

Hình 13 Dữ liệu từ thiết bị đã gửi lên được MQTT Broker thử nghiệm

* Thiết kế thử nghiệm ứng dụng App điện thoại, thử kết nối với MQTT Server LocalHost và trao đổi bản tin.
  + *Chưa triển khai được*

**Tuần 3: ngày 06 tháng 07 năm 2022**

**Địa điểm:** MS Team

**Tham dự:** Trần Quang Minh, Chu Quốc Đạt, Đồng Xuân Luân (bận).

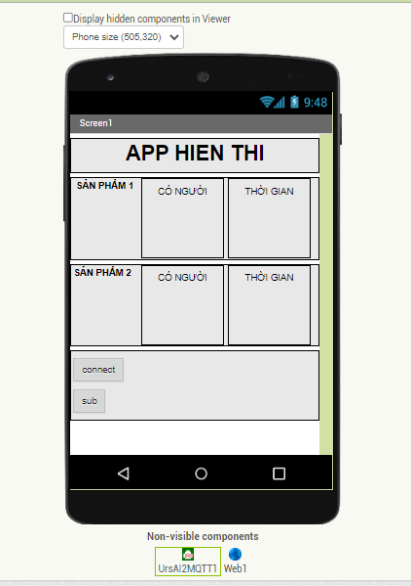
**Nội dung:** Báo cáo tiến độ của các nhiệm vụ đã giao ở tuần 2, tiến hành triển khai các kế hoạch thiết kế, thử nghiệm dựa trên các kết quả đã có.

**Các vấn đề thảo luận và đã thống nhất trong cuộc họp:**

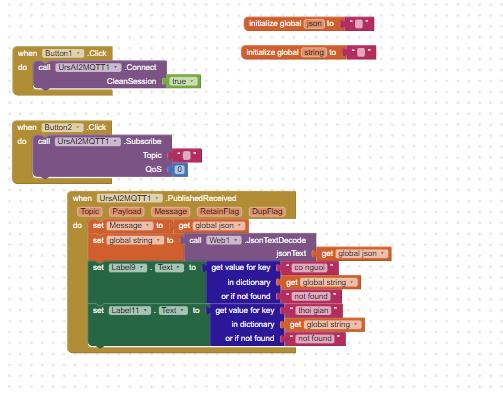
1. **Thống nhất bỏ hộp dữ liệu text trên giao diện app**
2. **Thêm phần tổng thời gian khách đứng trong vùng chưng bày cho mỗi node**

**Nhiệm vụ và báo cáo tuần 3:**

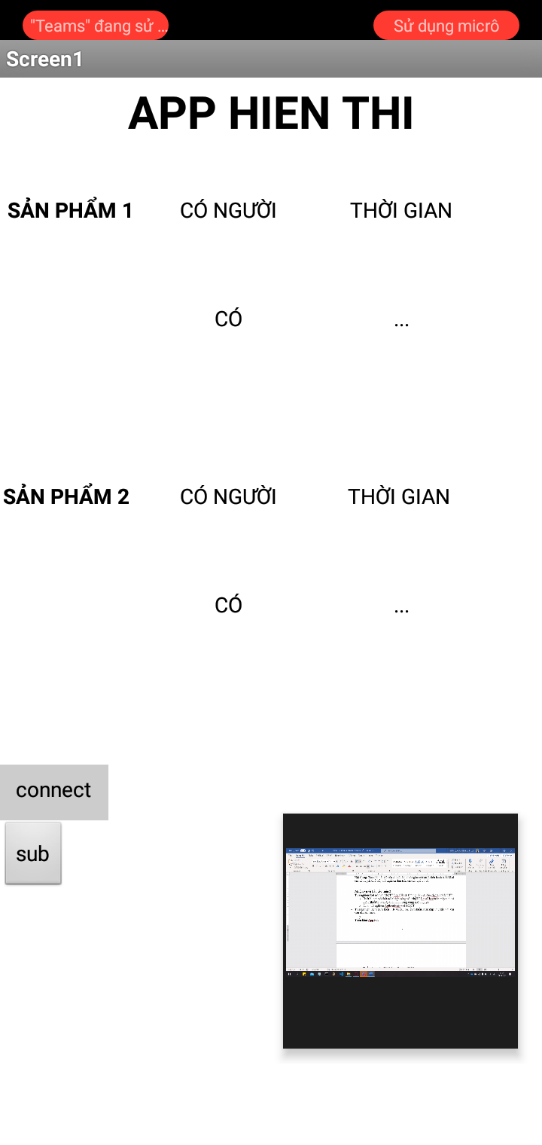
* Thử nghiệm kết nối với MQTT LocalHost, thử nghiệm Authenticate với MQTT.
  + *Thiết bị node đã kết nối thành công với MQTT LocalHost trên máy tính cá nhân (thiết bị và máy tính nằm trong cùng một mạng).*
  + *Chưa thử nghiệm Authenticate với MQTT.*
* Thử nghiệm thêm cảm biến PIR và các loại cảm biến khác đáp ứng tốt hơn với yêu cầu bài toán.
  + *Đã lập trình cho node cảm biến hoạt động bình thường và đã có thử nghiệm với người đi qua. Cảm biến trả xung cạnh lên khi có chuyển động trong phạm vi hình nón có độ sâu 4m. Không cần thiết phải sử dụng cảm biến khác.*
* Triển khai App điện thoại.
  + *Đã có khung của App điện thoại*



Hình 14 Giao diện app trên phần mềm thiết kế MIT



Hình 15 Code Block ứng dụng trên phần mềm thiết kế MIT



Hình 16 App thực tế trên điện thoại Android

**Tuần 4: ngày 20 tháng 7 năm 2022**

**Địa điểm:** MS Team

**Tham dự:** Trần Quang Minh, Chu Quốc Đạt, Đồng Xuân Luân.

**Nội dung:** Báo cáo tiến độ của các nhiệm vụ đã giao ở tuần 3, tiến hành triển khai các kế hoạch thiết kế, thử nghiệm dựa trên các kết quả đã có.

**Các vấn đề thảo luận và đã thống nhất trong cuộc họp:**

1. **Thống nhất bản tin gửi từ thiết bị lên khi có khách trong khu vực với chu kỳ 1 giây, cả Server và thiết bị đếm thời gian**
2. **Thống nhất giao diện App (front-end)**
3. **Thống nhất lịch họp cuối cùng để ghép các thành phần và demo sản phẩm**
4. **Thống nhất lại bản tin:**

Tên topic: "Human\_Present"

Cấu trúc bản tin:

***{***

***“DeviceID: uint8\_t <ID của thiết bị>”,***

***“State: uint8\_t <trạng thái của khách>”,***

***“Period: uint32\_t <thời gian khách đứng lại>”***

***}***

*Trong đó:*

*<trạng thái của khách> có giá trị là 0 và 1 (có hoặc không có)*

*<thời gian khách đứng lại> là thời gian tính bằng giây (s)*

*<ID của thiết bị> là ID được gán cho thiết bị trong lúc sản xuất*

1. **Bản tin mẫu:**

{

“DeviceID”: 121,

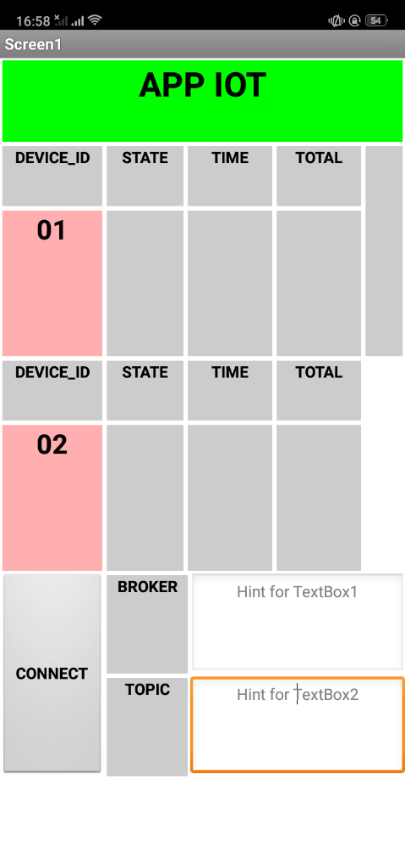
“State”: 1,

“Period”: 32

}

**Nhiệm vụ và báo cáo tuần 4:**

* Thử nghiệm bảo mật với MQTT.
  + *Chưa thể tích hợp bảo mật vào hệ thống*
* Hoàn thành phần cứng của thiết bị (chế tạo node 2).
  + *Đã chế tạo xong*
* Xử lý back-end của App và tiến hành kết nối với các thiết bị.
  + *Đã hoàn thiện App điện thoại*



Hình 17 Giao diện cuối cùng của App

**Tuần 5: ngày 26 tháng 07 năm 2022**

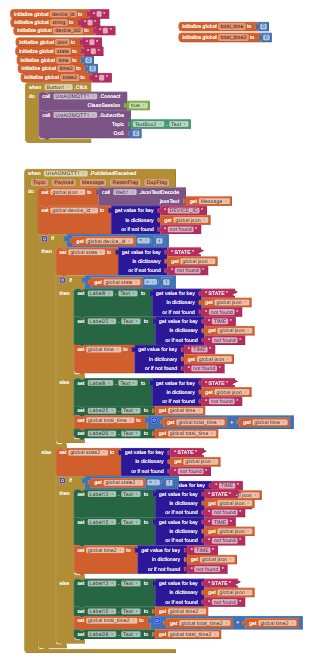
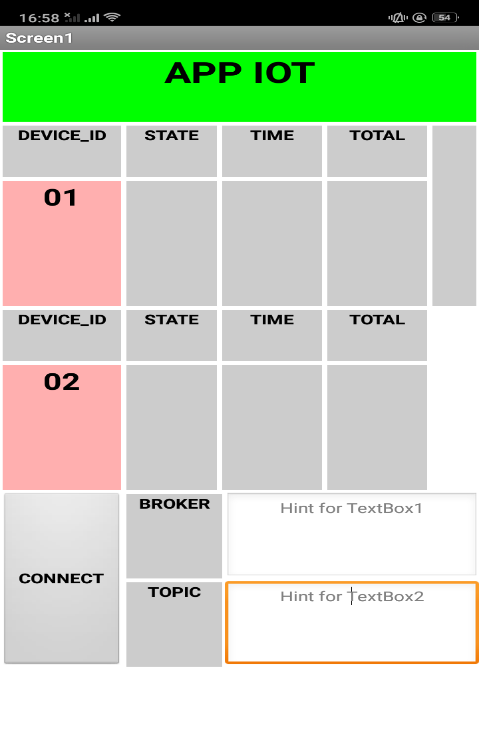
**Địa điểm:** Offline tại phòng MANDevices Lab

**Tham dự:** Trần Quang Minh, Chu Quốc Đạt, Đồng Xuân Luân.

**Nội dung:** Thử nghiệm lần cuối, ghép nối các thành phần đầu cuối, quay video demo, chỉnh sửa báo cáo và hoàn thiện dự án

**Các vấn đề thảo luận và đã thống nhất trong cuộc họp:**

1. **Hoàn thành quá trình thử nghiệm cuối cùng và quay video demo**
2. **Hoàn thành báo cáo**



Hình 18 Giao diện và Code Block ứng dụng

# **Thiết kế hệ thống**

Từ yêu cầu thiết kế đã nên trên và các quyết định được đưa ra trong buổi họp, mô hình mô tả tổng quan hệ thống đã được thiết kế và thống nhất như sau:

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình 19 Mô hình tổng quan hệ thống đã thống nhất

Từ mô hình này, hệ thống được chia ra làm 3 phần:

* Thiết bị node, có nhiệm vụ ghi nhận sự kiện người qua lại, gửi bản tin (Publish) lên Server MQTT để cập nhập trạng thái.
* MQTT Server được xây dựng trên một máy tính có vai trò là nơi trung chuyển dữ liệu từ thiết bị node sang ứng dụng.
* Ứng dụng điện thoại, được thiết lập lắng nghe ở kênh và MQTT Server nhất định đã thống nhất trước, có nhiệm vụ hiển thị trạng thái, thời gian và các thông tin liên quan đến hệ thống trên giao diện.

Mô hình hệ thống phải được thiết kế tin giản, gói tin phải được chuyển tiếp nhanh bảo đảo tính thời gian thực. Các cảm biến của thiết bị node phải được hiệu chỉnh và thử nghiệm để đảm bảo tính chính xác của các trạng thái khách qua lại.

# **Phân công nhiệm vụ**

**Phân công nhiệm vụ tổng:**

Tất cả mọi thành viên đóng góp trong quá trình viết báo cáo hằng tuần

1. Trần Quang Minh: Thiết kế thiết bị cảm biến (MQTT Client), làm sản phẩm thực tế.
2. Chu Quốc Đạt: Triển khai Broker (MQTT Server) trên Local-Host (trên máy tính)
3. Đồng Xuân Luân: Thiết kế ứng dụng (App Android) trên điện thoại (MQTT Client)

# **Tính toán thiết kế**

Bài toán yêu cầu các thiết bị phải có khả năng kết nối mạng, mạng được kết nối phải có chung mạng với Server và ứng dụng quản lý. Do vậy nên lựa chọn dòng vi điều khiển ESP với tính năng WiFi được tích hợp sẵn trong chip. Ngoài ra, dòng ESP32 còn là một dòng chip mạnh và phổ biến trên thị trường hiện nay.

Thiết bị cảm biến chuyển động có thể sử dụng nhiều cách như cảm biến siêu âm, cảm biến lazer, cảm biến hồng ngoại hoặc PIR,… Nhưng để đơn giản và phù hợp nhất với bài toán, chúng em lựa chọn loại cảm biến PIR. Cảm biến phải được hiệu chuẩn để có thể hoạt động trong phạm vi 2-3m.

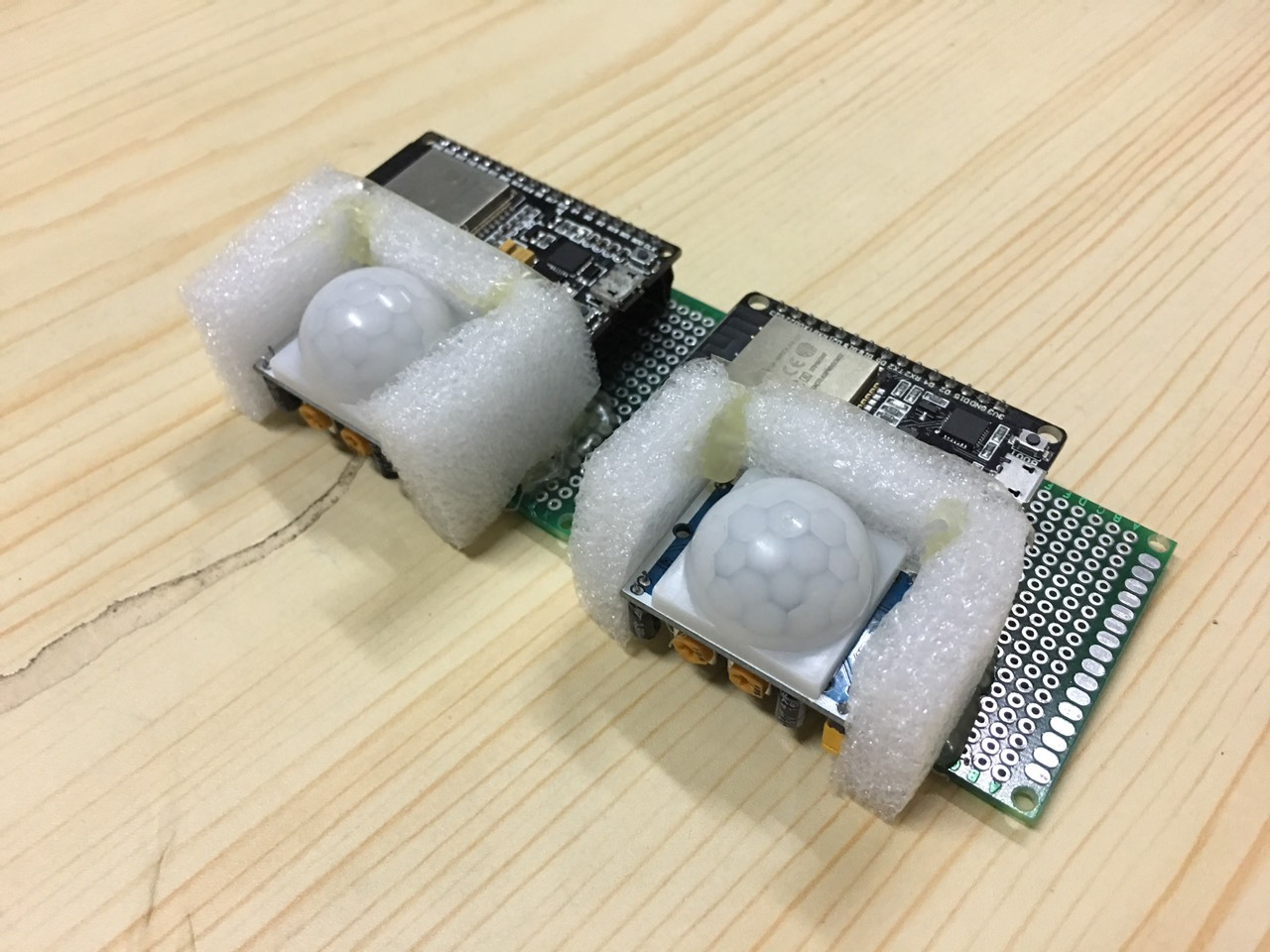
Server và ứng dụng phải được tính toán thiết kế cho nhiều thiết bị cùng kết nối và đẩy dữ liệu. Các gói dữ liệu gửi lên từ nhiều node phải được xử lý đồng thời và nhanh chóng.

# **Triển khai lắp ráp mạch thực**

Hình ảnh thực tế của thiết bị:



Hình 20 Hình ảnh 2 thiết bị từ hướng trên xuống



Hình 21 Hình ảnh 2 thiết bị từ hướng chéo

# **Thử nghiệm đánh giá**

Để đảm bảo tính chính xác của cảm biến, cần phải có quá trình thử nghiệm để có thể đánh giá và điều chỉnh cảm biến. Đối với loại cảm biến PIR HC-SR501, có 3 tham số có thể được điều chỉnh. Các tham số đó là:

* Chế độ hoạt động: có thể là chế độ 1 lần hoặc liên tục
* Thời gian giữa 2 lần kích hoạt
* Khoảng cách (hay phạm vi) hoạt động

Chế độ được thiết lập cho thiết bị cảm biến là chế độ xung 1 lần để có thể đếm chính xác lượng người qua lại (1 người là 1 xung duy nhất). Vì thiết bị cảm biến sẽ được đặt ở khu vực cửa của khu trưng bày hàng nên tầm hoạt động của cảm biến sẽ phải rơi vào tầm 2-3m. Thời gian đáp ứng của cảm biến phải được đẩy nhanh nhất có thể. Qua nhiều lần thử nghiệm và kiểm tra dạng xung đầu ra của cảm biến bằng osciloscope, thời gian đáp ứng ổn định và nhanh nhất là tầm 2s, khoảng cách mà cảm biến có thể cảm nhận được chuyển động là 3m. Bọn em có video thử nghiệm được gửi cùng với dự án.



Hình 22 Setup để thử nghiệm

# **Kết luận**

Bản báo cáo trên đã nêu được đầy đủ yêu cầu, cách xử lý bài toán, quá trình xây dựng hệ thống bao gồm việc thiết kế, thử nghiệm và triển khai của bài tập lớn môn Mạng và các giao thức IoT. Hệ thống thiết kế đã được đưa và thử nghiệm và triển khai chạy ổn định theo đúng như thiết kế ban đầu. Qua bài tập lớn này, nhóm 15 bọn em đã học được rất nhiều thứ liên quan đến việc cùng nhau triển khai một dự án thật theo một nhóm, và xử lý dự án từ đề bài cho đến sản phẩm cuối. Trong quá trình thực hiện, còn có nhiều thiết sót, chúng em xin tiếp thu và học tập để có thể trau dồi bản thân thêm nữa trong các dự án tương lai. Video demo và thử nghiệm sản phẩm đã được bọn em đính kèm cùng báo cáo.

# **Tài liệu tham khảo**

|  |  |
| --- | --- |
| [1] | <https://mqtt.org> đường dẫn tới trang chủ các tiêu chuẩn giao thức mqtt |
| [2] | <https://mosquitto.org/> đường dẫn tới trang chủ mosquito broker |
| [3] | <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html> đường dẫn tới thông tin lập trình ESP32 |
|  |  |
|  |  |