

BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ KỸ THUẬT TP.HCM  
KHOA ĐIỆN - ĐIỆN TỬ



**HCMUTE**

**BÁO CÁO ĐỒ ÁN**  
**ỨNG DỤNG IOT VÀO ĐIỀU KHIỂN THIẾT BỊ QUA**  
**GOOGLE ASSISTANT KẾT HỢP THỊ GIÁC MÁY TÍNH**  
**GIÁM SÁT HỌC SINH QUA CAMERA TRONG PHÒNG HỌC**

GV hướng dẫn: TS. Huỳnh Thế Thiện

Nhóm sinh viên thực hiện

Họ và Tên	MSSV	Mã lớp
Huỳnh Nhất Vũ	22139077	22139A
Hoàng Ngọc Chiến	22139006	22139B

TP. Hồ Chí Minh, Tháng 11 năm 2025

# Mục lục

<b>1</b>	<b>Tổng quan</b>	<b>1</b>
1.1	Mục tiêu đề tài . . . . .	2
1.2	Giới hạn đề tài . . . . .	3
1.3	Bố cục báo cáo . . . . .	3
<b>2</b>	<b>Cơ sở lý thuyết</b>	<b>4</b>
2.1	Sơ lược về Internet of Things . . . . .	4
2.1.1	Định nghĩa IoT . . . . .	4
2.1.2	Ứng dụng của IoT . . . . .	4
2.1.3	Đặc điểm công nghệ của IoT . . . . .	5
2.2	Tổng quan về thị giác máy tính (Computer Vision) . . . . .	5
2.2.1	Định nghĩa thị giác máy tính . . . . .	6
2.2.2	Một số phương pháp phát hiện vật thể . . . . .	7
2.2.3	Triển khai mô hình học sâu với ONNX Runtime . . . . .	8
2.2.4	Ứng dụng đếm số lượng học sinh trong lớp học . . . . .	9
2.2.5	Ứng dụng nhận diện khuôn mặt để điểm danh học sinh . . . . .	10
2.2.6	Ứng dụng phân tích tư thế và phát hiện ngủ gật . . . . .	15
2.2.7	Tổng kết vai trò Computer Vision trong đề tài . . . . .	18
2.3	Phần cứng sử dụng trong hệ thống . . . . .	18
2.3.1	ESP32 DevKit V1 . . . . .	18
2.3.2	Raspberry Pi 4B . . . . .	20
2.3.3	Camera OV5647 . . . . .	21
2.3.4	Cảm biến nhiệt độ - độ ẩm DHT22 . . . . .	22
2.3.5	Cảm biến ánh sáng BH1750 . . . . .	23
2.3.6	Module relay điều khiển thiết bị . . . . .	24
2.3.7	Keypad 3x4 điều khiển thiết bị . . . . .	25
2.3.8	Nguồn cấp . . . . .	26
2.4	Các chuẩn giao tiếp . . . . .	29
2.4.1	Giao thức HTTPS (HTTP/REST) . . . . .	29

2.4.2	Giao thức MQTT . . . . .	30
2.4.3	Giao thức I <sup>2</sup> C . . . . .	31
2.4.4	Nền tảng Home Assistant . . . . .	33
2.4.5	Nền tảng Google Assistant . . . . .	34
<b>3</b>	<b>Thiết kế hệ thống</b>	<b>36</b>
3.1	Mô hình hoạt động của hệ thống . . . . .	36
3.2	Sơ đồ nguyên lý tổng quan phần cứng . . . . .	37
<b>4</b>	<b>Thi công hệ thống</b>	<b>39</b>
4.1	Lưu đồ thuật toán toàn hệ thống . . . . .	39
4.2	Thi công phần cứng . . . . .	40
4.2.1	Linh kiện sử dụng . . . . .	41
4.2.2	Tiến hành thi công phần cứng . . . . .	41
<b>5</b>	<b>Giới thiệu ứng dụng Flutter trên điện thoại</b>	<b>45</b>
5.1	Giới thiệu tổng quan ứng dụng Flutter . . . . .	45
5.2	Thiết lập và tích hợp hệ thống . . . . .	46
5.2.1	Kết nối MQTT thông qua Home Assistant . . . . .	46
5.2.2	Liên kết Home Assistant với Google Assistant . . . . .	48
5.2.3	Thiết lập server xử lý trên Raspberry Pi . . . . .	49
5.3	Chức năng của ứng dụng Flutter . . . . .	50
5.3.1	Điều khiển thiết bị và giám sát cảm biến . . . . .	50
5.3.2	Diểm danh học sinh bằng nhận diện khuôn mặt . . . . .	51
5.3.3	Quản lý khuôn mặt học sinh . . . . .	52
5.3.4	Xem lịch sử điểm danh . . . . .	53
5.3.5	Kiểm tra số lượng học sinh . . . . .	54
5.3.6	Phát hiện trạng thái ngủ gật . . . . .	54
5.3.7	Giám sát camera thời gian thực . . . . .	55
<b>6</b>	<b>Nhận xét và đánh giá</b>	<b>56</b>
6.1	Kết quả đạt được . . . . .	56
6.2	Dánh giá hệ thống . . . . .	56
6.2.1	Ưu điểm . . . . .	56

6.2.2	Hạn chế . . . . .	57
6.3	Hướng phát triển . . . . .	57
6.4	Kết luận chung . . . . .	58

## Danh mục hình ảnh

1	Mối quan hệ giữa thị giác máy tính và học máy . . . . .	6
2	Minh họa mô hình YOLOv8n-face phát hiện khuôn mặt và các điểm đặc trưng (facial landmarks) . . . . .	11
3	Kiến trúc mô hình MobileFaceNet dùng cho trích xuất đặc trưng khuôn mặt . .	13
4	Minh họa cơ chế ArcFace với ràng buộc góc trong không gian đặc trưng . . . .	14
5	Minh họa YOLOv8n-Pose phát hiện người và các điểm khớp cơ thể . . . . .	15
6	Minh họa landmarks vùng mắt và chỉ số EAR trong phát hiện trạng thái mắt .	17
7	Bo mạch ESP32 DevKit V1 sử dụng trong hệ thống . . . . .	19
8	Sơ đồ chân của ESP32 DevKit V1 . . . . .	19
9	Máy tính nhúng Raspberry Pi 4B . . . . .	20
10	Sơ đồ các cổng kết nối trên Raspberry Pi 4 Model B . . . . .	21
11	Module camera OV5647 dùng cho Raspberry Pi . . . . .	22
12	Cảm biến nhiệt độ – độ ẩm DHT22 . . . . .	23
13	Cảm biến cường độ ánh sáng BH1750 . . . . .	24
14	Module relay 4 kênh sử dụng để điều khiển đèn và quạt . . . . .	25
15	Keypad ma trận 3x4 sử dụng để điều khiển quạt, đèn và chuyển đổi chế độ tự động	26
16	Adapter 5 V–3 A cấp nguồn cho ESP32 và các module ngoại vi . . . . .	27
17	Bộ sạc Xiaomi 33 W cấp nguồn cho Raspberry Pi 4B qua cổng USB Type-C . .	28
18	Giao tiếp HTTPS giữa Raspberry Pi, Firebase Realtime Database và ESP32 DevKit V1 . . . . .	30
19	Minh họa giao tiếp I <sup>2</sup> C giữa thiết bị master và các slave . . . . .	33
20	Tổng quan Home Assistant . . . . .	34
21	Minh họa Google Assistant điều khiển thiết bị thông qua Home Assistant và MQTT	35
22	Sơ đồ hoạt động tổng thể của hệ thống lớp học thông minh . . . . .	36
23	Sơ đồ nguyên lý tổng quan phần cứng hệ thống lớp học thông minh . . . . .	37
24	Lưu đồ thuật toán toàn hệ thống lớp học thông minh . . . . .	40
25	Bố trí tổng thể các thiết bị trong mô hình lớp học thông minh . . . . .	42
26	Bố trí phần cứng điều khiển bên trong mô hình . . . . .	42
27	Hệ thống lớp học thông minh vận hành thực tế . . . . .	43

28	Sơ đồ kết nối các thành phần phần cứng trong mô hình lớp học thông minh . . . . .	44
29	Trang tổng quan của ứng dụng Flutter . . . . .	46
30	Luồng điều khiển thiết bị từ Flutter thông qua Home Assistant và MQTT . . . . .	47
31	Giao diện Dashboard của Home Assistant trong hệ thống lớp học thông minh . . . . .	48
32	Liên kết Google Assistant với Home Assistant để điều khiển thiết bị . . . . .	49
33	Server Flask xử lý thị giác máy tính được khởi chạy trên Raspberry Pi . . . . .	50
34	Giao diện điều khiển thiết bị và giám sát cảm biến trên ứng dụng Flutter . . . . .	51
35	Giao diện hiển thị kết quả điểm danh khuôn mặt trên ứng dụng . . . . .	52
36	Chức năng thêm và quản lý khuôn mặt học sinh . . . . .	53
37	Giao diện lịch sử điểm danh học sinh . . . . .	53
38	Giao diện kiểm tra số lượng học sinh trên ứng dụng Flutter . . . . .	54
39	Giao diện kiểm tra trạng thái ngủ gật của học sinh . . . . .	55
40	Giao diện giám sát camera thời gian thực trên ứng dụng Flutter . . . . .	55

## **Danh mục bảng**

1	So sánh các phương pháp phát hiện vật thể . . . . .	8
2	So sánh các phiên bản YOLOv8 theo độ chính xác và độ phức tạp mô hình . . .	10
3	Thống kê linh kiện sử dụng cho hệ thống . . . . .	41

# 1 Tổng quan

Trong thời đại ngày nay, sự bùng nổ của cách mạng khoa học công nghệ đã thúc đẩy sự phát triển mạnh mẽ của Internet of Things (IoT), mang lại những thay đổi tích cực cho cuộc sống hiện tại và định hình tương lai của chúng ta. Cùng với sự phát triển không ngừng, Internet of Things (IoT) được ứng dụng rộng rãi trong các lĩnh vực như điện thoại thông minh, máy tính bàn, thiết bị công nghiệp... IoT đang dần trở thành một xu hướng toàn cầu. Đây là mạng lưới các thiết bị và vật dụng được kết nối thông qua Internet, cho phép con người điều khiển và quản lý chúng một cách hiệu quả.

Thay vì chúng ta phải lúc nào cũng theo dõi và tự bật/tắt các thiết bị khi không dùng, hoặc trong không gian học tập thì học sinh phải di chuyển đến công tắc để bật đèn, quạt, máy lạnh, tivi,...Những điều này vừa bất tiện vừa mất thời gian. IoT đã đem lại phương pháp giải quyết bằng việc kết nối các thiết bị đó lại để có thể điều khiển từ xa. Việc chúng ta bước vào phòng học và hệ thống tự động bật đèn, quạt, máy lạnh,...Những điều tưởng chừng chỉ có trong tưởng tượng hay trên phim nay đã được đưa vào thực tế.

Để giải quyết những vấn đề trên, nhóm đã lựa chọn đề tài: “Ứng dụng IoT vào điều khiển thiết bị qua Google Assistant và Home Assistant kết hợp thị giác máy tính giám sát học sinh qua camera trong phòng học.”

Hệ thống này được thiết kế nhằm hỗ trợ bật/tắt thiết bị trong phòng học một cách linh hoạt và tiện lợi thông qua giọng nói. Ngoài tính năng điều khiển từ xa, hệ thống còn cung cấp khả năng giám sát trạng thái hoạt động của thiết bị thông qua ứng dụng điện thoại và Webserver.

Bên cạnh đó, hệ thống còn được trang bị các cảm biến môi trường như nhiệt độ - độ ẩm và cảm biến ánh sáng nhằm thu thập thông số trong phòng học. Các dữ liệu này giúp hệ thống tự động đưa ra quyết định điều khiển thiết bị.

Đặc biệt, đề tài còn được mở rộng bằng việc tích hợp các tính năng trí tuệ nhân tạo (AI) để hỗ trợ giám sát lớp học thông minh hơn. Cụ thể, hệ thống AI sử dụng camera kết hợp với các mô hình học sâu để:

Dếm số lượng học sinh trong phòng học theo thời gian thực, giúp quản lý số lượng, tự động hóa chế độ thiết bị. Nhận diện khuôn mặt để điểm danh tự động, giảm thao tác thủ công và tăng độ chính xác.

Phát hiện ngủ gật hoặc tư thế không tập trung, dựa trên mô hình nhận dạng keypoint cơ thể (pose estimation), hỗ trợ giáo viên theo dõi lớp học dễ dàng hơn.

Việc tích hợp IoT, cảm biến môi trường và AI thị giác máy tính không chỉ giúp tự động hóa việc điều khiển thiết bị, mà còn nâng cao hiệu quả quản lý lớp học, tối ưu năng lượng và mang đến một môi trường học tập hiện đại, tiện nghi và thông minh hơn.

## 1.1 Mục tiêu đề tài

Đề tài được thực hiện với mục tiêu xây dựng một hệ thống phòng học thông minh (Smart Classroom) tích hợp IoT và thị giác máy tính nhằm tự động hóa việc điều khiển thiết bị, giám sát trạng thái lớp học và hỗ trợ hoạt động giảng dạy. Các mục tiêu cụ thể như sau:

- Xây dựng hệ thống IoT điều khiển thiết bị trong phòng học
  - + Thiết kế và triển khai mô-đun điều khiển các thiết bị như đèn, quạt thông qua vi điều khiển ESP32.
  - + Cho phép điều khiển thiết bị bằng giọng nói thông qua Google Assistant và Home Assistant.
  - + Hỗ trợ điều khiển từ xa qua điện thoại hoặc Webserver theo thời gian thực.
  - + Hiển thị trạng thái thiết bị và đồng bộ dữ liệu với hệ thống cơ sở dữ liệu.
- Tích hợp cảm biến môi trường để tự động điều chỉnh thiết bị
  - + Thu thập dữ liệu nhiệt độ – độ ẩm (DHT22) và cường độ ánh sáng (BH1750).
  - + Tự động bật/tắt quạt khi nhiệt độ hoặc độ ẩm vượt ngưỡng cho phép.
  - + Tự động điều chỉnh đèn khi ánh sáng quá thấp.
  - + Tối ưu năng lượng bằng cách tắt thiết bị khi phòng không có người.
- Ứng dụng trí tuệ nhân tạo (AI) vào giám sát phòng học
  - + Phát triển hệ thống đếm số lượng học sinh theo thời gian thực bằng mô hình YOLO.
  - + Triển khai nhận diện khuôn mặt để điểm danh tự động, giảm thao tác thủ công.
  - + Phát hiện ngũ gật hoặc tư thế không tập trung dựa trên mô hình nhận dạng keypoint (pose estimation).
  - + Kết hợp dữ liệu AI với IoT để tự động điều chỉnh thiết bị (ví dụ: nhiều học sinh → tăng công suất quạt; ít người → giảm hoặc tắt quạt).
- Xây dựng giao diện điều khiển thân thiện và trực quan
  - + Phát triển giao diện Web/ứng dụng hiển thị thông tin môi trường, sĩ số lớp học và trạng thái thiết bị.
  - + Tối ưu trải nghiệm người dùng cho giáo viên và học sinh.
  - + Đồng bộ dữ liệu liên tục giữa hệ thống IoT và AI.

## **1.2 Giới hạn đề tài**

Mặc dù hệ thống phòng học thông minh (Smart Classroom) được xây dựng với nhiều chức năng kết hợp giữa IoT và trí tuệ nhân tạo (AI), đề tài vẫn còn tồn tại một số giới hạn nhất định do phạm vi nghiên cứu và điều kiện thực tế:

- Giới hạn về phần cứng: Hệ thống được triển khai trên Raspberry Pi 4B và ESP32, do đó tốc độ xử lý, bộ nhớ và khả năng mở rộng còn hạn chế so với các thiết bị chuyên dụng. Camera OV5647 chỉ hỗ trợ chất lượng nhất định và chưa phù hợp cho môi trường ánh sáng quá yếu hoặc quá mạnh.
- Giới hạn về mô hình AI: Các mô hình như YOLO và MobileFaceNet chạy trên Raspberry Pi nên tốc độ nhận diện chưa thể đạt mức tối ưu như trên máy tính hiệu năng cao. Hệ thống chưa áp dụng các kỹ thuật tăng tốc chuyên sâu như TensorRT hoặc GPU rời.
- Độ chính xác bị ảnh hưởng bởi môi trường: Quá trình đếm người, nhận diện khuôn mặt hoặc phát hiện ngủ gật có thể bị giảm độ chính xác khi:
  - + Ánh sáng phòng học thay đổi đột ngột,
  - + Học sinh che mặt, đeo khẩu trang hoặc xoay đầu,
  - + Camera đặt ở vị trí không tối ưu.
- Tự động hóa dựa trên cảm biến còn đơn giản: Quy tắc điều khiển dựa trên ngưỡng nhiệt độ, độ ẩm, ánh sáng hoặc số lượng người mới ở mức cơ bản, chưa áp dụng thuật toán học máy để tối ưu hóa theo thời gian thực.
- Giới hạn về cơ sở dữ liệu và giao diện: Hệ thống Webserver và cơ sở dữ liệu lưu trữ chỉ ở mức đáp ứng nhu cầu thử nghiệm, chưa tối ưu cho quy mô lớn hoặc trường học nhiều phòng.

## **1.3 Bô cục báo cáo**

## 2 Cơ sở lý thuyết

### 2.1 Sơ lược về Internet of Things

#### 2.1.1 Định nghĩa IoT

Internet of Things (IoT) là một hệ thống các thiết bị vật lý được kết nối với Internet, bao gồm các thiết bị, phương tiện, máy móc, tòa nhà và nhiều đối tượng khác được tích hợp vi mạch điện tử, phần mềm điều khiển, cảm biến và khả năng kết nối mạng. Những thiết bị này có khả năng thu thập dữ liệu, trao đổi thông tin và tương tác với nhau mà không cần sự can thiệp trực tiếp của con người.

Các thiết bị IoT có thể kết nối với nhau thông qua nhiều chuẩn truyền thông khác nhau như WiFi, Bluetooth, ZigBee, hồng ngoại, LoRa, BLE, RFID, LTE, v.v. Những thiết bị phổ biến trong hệ sinh thái IoT bao gồm điện thoại thông minh, đồng hồ thông minh, thiết bị gia dụng, cảm biến môi trường, ô tô thông minh, bóng đèn thông minh và nhiều thiết bị tự động hóa khác.

Về bản chất, Internet of Things tạo ra một mạng lưới liên kết giữa mọi vật thể xung quanh chúng ta, bao gồm cả con người, từ đó hình thành nên mối tương tác giữa *người – người*, *người – thiết bị* và *thiết bị – thiết bị*.

#### 2.1.2 Ứng dụng của IoT

IoT ngày nay được ứng dụng rộng rãi trong nhiều lĩnh vực:

**Lĩnh vực y tế:** Các thiết bị IoT như cảm biến đo nhịp tim, nhiệt độ cơ thể, cảm biến sinh học giúp theo dõi liên tục tình trạng sức khỏe của bệnh nhân. IoT còn hỗ trợ quản lý tài nguyên trong bệnh viện, theo dõi thiết bị và vật tư y tế.

**Lĩnh vực nhà thông minh (Smart Home):** IoT cho phép tự động hóa hệ thống chiếu sáng, điều hòa, an ninh và thiết bị gia dụng. Ví dụ: đèn thông minh tự điều chỉnh độ sáng theo ánh sáng tự nhiên; camera an ninh thông minh phát hiện chuyển động bất thường; máy rửa bát hoặc các thiết bị gia dụng khác có thể điều khiển từ xa qua Internet.

**Lĩnh vực nông nghiệp thông minh (Smart Farming):** IoT được ứng dụng trong hệ thống tưới tiêu tự động dựa trên cảm biến độ ẩm đất, giúp tiết kiệm nước và tối ưu quá

trình chăm sóc cây trồng. Trong chăn nuôi, cảm biến có thể gắn lên gia súc để theo dõi sức khỏe hoặc vị trí của chúng.

**Lĩnh vực thành phố thông minh (Smart City):** IoT hỗ trợ xây dựng hệ thống giao thông thông minh với khả năng điều chỉnh đèn tín hiệu dựa trên mật độ phương tiện. Ngoài ra, IoT còn ứng dụng trong quản lý năng lượng, rác thải, chiếu sáng công cộng, an ninh đô thị, v.v.

### 2.1.3 Đặc điểm công nghệ của IoT

Một trong những đặc điểm quan trọng của IoT là khả năng định danh và nhận dạng rõ ràng từng đối tượng trong hệ thống. Tất cả các đồ vật, thậm chí cả con người, đều có thể được gắn mã định danh (ID), cảm biến hoặc thẻ RFID để đảm bảo phân biệt chính xác giữa các đối tượng với nhau.

Nhờ khả năng định danh và kết nối này, con người có thể giám sát và điều khiển thiết bị thông qua máy tính, điện thoại thông minh hoặc hệ thống phần mềm từ xa. Đây chính là nền tảng quan trọng giúp IoT trở thành công nghệ chủ chốt cho các hệ thống tự động hóa hiện đại.

## 2.2 Tổng quan về thị giác máy tính (Computer Vision)

Thị giác máy tính (Computer Vision) là một lĩnh vực quan trọng của trí tuệ nhân tạo, cho phép máy tính phân tích, nhận dạng và hiểu nội dung trong hình ảnh hoặc video. Thông qua việc xử lý dữ liệu hình ảnh thu nhận từ camera, hệ thống thị giác máy tính có thể trích xuất thông tin, phát hiện đối tượng và suy luận về trạng thái của môi trường xung quanh.

Trong những năm gần đây, cùng với sự phát triển của học sâu (Deep Learning) và phần cứng tính toán, thị giác máy tính đã đạt được nhiều bước tiến vượt bậc và được ứng dụng rộng rãi trong các lĩnh vực như giám sát an ninh, giao thông thông minh, y tế, sản xuất công nghiệp và giáo dục. Các mô hình mạng nơ-ron tích chập (Convolutional Neural Network – CNN) đã góp phần nâng cao đáng kể độ chính xác và tốc độ xử lý của các hệ thống thị giác máy tính hiện đại.

Trong phạm vi đề tài này, thị giác máy tính được ứng dụng nhằm xây dựng một hệ thống lớp học thông minh với khả năng giám sát và hỗ trợ quản lý lớp học một cách tự động. Cụ thể, hệ thống sử dụng các thuật toán thị giác máy tính để đếm số lượng học sinh trong phòng học theo thời gian thực, nhận diện khuôn mặt để thực hiện điểm danh tự động, cũng như phát hiện các hành vi mất tập trung như ngủ gật thông qua việc phân tích tư thế và chuyển động của học sinh. Các kết quả phân tích từ hệ thống thị giác máy tính sẽ được kết hợp với hệ thống IoT để nâng cao hiệu quả quản lý và điều khiển thiết bị trong lớp học.

### 2.2.1 Định nghĩa thị giác máy tính

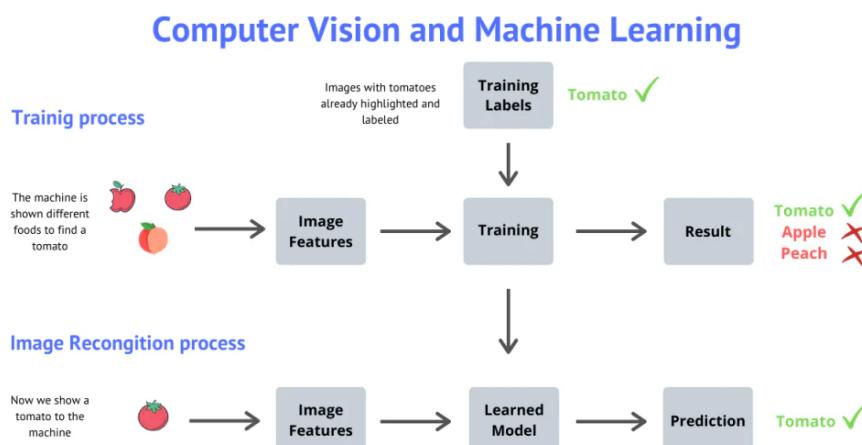
Thị giác máy tính (Computer Vision – CV) là công nghệ cho phép máy tính thu nhận hình ảnh từ các thiết bị như camera, sau đó xử lý và phân tích dữ liệu hình ảnh nhằm hiểu và suy luận về môi trường thực tế. Mục tiêu của thị giác máy tính không chỉ dừng lại ở việc “nhìn thấy” hình ảnh, mà còn bao gồm việc nhận dạng đối tượng, xác định mối quan hệ không gian, theo dõi chuyển động và đưa ra quyết định dựa trên ngữ cảnh.

Các hệ thống thị giác máy tính hiện đại thường kết hợp nhiều bước xử lý khác nhau như tiền xử lý ảnh, trích xuất đặc trưng và suy luận bằng các mô hình học máy (Machine Learning) hoặc học sâu (Deep Learning). Nhờ đó, máy tính có thể thực hiện các tác vụ phức tạp như phát hiện đối tượng, nhận diện khuôn mặt, phân tích tư thế cơ thể (pose estimation) và nhận dạng hành vi trong môi trường thực tế.

Về mối quan hệ với học máy, thị giác máy tính có thể được xem là một lĩnh vực ứng dụng chuyên sâu của Machine Learning. Trong khi Machine Learning tập trung vào việc xây dựng các mô hình học từ dữ liệu để dự đoán hoặc phân loại, thì thị giác máy tính tập trung vào dữ liệu đầu vào là hình ảnh và video. Các bài toán trong thị giác máy tính thường yêu cầu xử lý dữ liệu có chiều cao, chiều rộng và chiều sâu, mang tính không gian, do đó đòi hỏi các mô hình học sâu chuyên biệt như mạng nơ-ron tích chập (Convolutional Neural Network – CNN).

Nói cách khác, Machine Learning đóng vai trò là nền tảng thuật toán, còn thị giác máy tính là lĩnh vực ứng dụng các thuật toán đó để giải quyết các bài toán liên quan đến hình ảnh. Sự kết hợp giữa hai lĩnh vực này đã tạo ra bước tiến lớn trong khả năng “hiểu” hình ảnh của máy tính, giúp các hệ thống tự động đạt độ chính xác và độ tin cậy cao hơn so với các phương pháp xử lý ảnh truyền thống.

Trong phạm vi đề tài, thị giác máy tính đóng vai trò là thành phần cốt lõi giúp hệ thống hiểu được trạng thái lớp học thông qua dữ liệu hình ảnh thu nhận từ camera. Các thông tin như số lượng học sinh, danh tính khuôn mặt và tư thế học tập được trích xuất từ mô hình học sâu, sau đó cung cấp dữ liệu đầu vào quan trọng cho hệ thống IoT nhằm hỗ trợ việc điều khiển thiết bị và giám sát lớp học một cách thông minh và tự động.



Hình 1: Mối quan hệ giữa thị giác máy tính và học máy

### 2.2.2 Một số phương pháp phát hiện vật thể

Các thuật toán phát hiện đối tượng (Object Detection) đã trải qua nhiều giai đoạn phát triển, từ các phương pháp truyền thống dựa trên đặc trưng thủ công đến các mô hình học sâu hiện đại.

**Haar Cascade:** Là một trong những phương pháp phát hiện đối tượng sớm nhất, sử dụng các đặc trưng Haar và bộ phân loại AdaBoost. Ưu điểm của Haar Cascade là tốc độ xử lý nhanh và dễ triển khai, tuy nhiên độ chính xác không cao, nhạy cảm với điều kiện ánh sáng và góc nhìn, do đó chỉ phù hợp với các bài toán đơn giản như phát hiện khuôn mặt trong điều kiện cố định.

**HOG + SVM:** Phương pháp này trích xuất đặc trưng HOG (Histogram of Oriented Gradients) để mô tả hình dạng đối tượng, sau đó sử dụng bộ phân loại SVM để nhận dạng. HOG + SVM cho kết quả tốt hơn Haar Cascade trong bài toán phát hiện người, tuy nhiên khả năng mở rộng kém và hiệu suất giảm mạnh khi số lượng đối tượng cần phát hiện tăng lên.

**R-CNN, Fast R-CNN, Faster R-CNN:** Đây là nhóm mô hình phát hiện đối tượng dựa trên mạng nơ-ron tích chập (CNN). Các mô hình này hoạt động theo cơ chế hai giai đoạn (two-stage detector), trong đó giai đoạn đầu đề xuất các vùng quan tâm (Region Proposal), giai đoạn sau thực hiện phân loại và hồi quy bounding box. Nhóm phương pháp này có độ chính xác cao nhưng tốc độ xử lý chậm, yêu cầu tài nguyên tính toán lớn, không phù hợp cho các hệ thống thời gian thực.

**YOLO (You Only Look Once):** YOLO là mô hình phát hiện đối tượng một giai đoạn (one-stage detector), thực hiện đồng thời việc định vị và phân loại đối tượng trong một lần suy luận duy nhất. Nhờ kiến trúc tối ưu, YOLO có tốc độ xử lý nhanh, độ chính xác cao và đặc biệt phù hợp cho các ứng dụng thời gian thực như giám sát, đếm người và phân tích hành vi.

Qua quá trình phát triển, có thể nhận thấy xu hướng chung của các thuật toán phát hiện đối tượng là chuyển từ các phương pháp truyền thống dựa trên đặc trưng thủ công sang các mô hình học sâu với khả năng tự động trích xuất đặc trưng. Trong đó, các mô hình one-stage detector như YOLO ngày càng được ưu tiên nhờ sự cân bằng tốt giữa tốc độ và độ chính xác, đặc biệt phù hợp cho các hệ thống nhúng và ứng dụng thực tế.

Phương pháp	Đặc điểm chính	Ưu điểm	Hạn chế
Haar Cascade	Sử dụng đặc trưng Haar và bộ phân loại AdaBoost	Tốc độ xử lý nhanh, dễ triển khai	Dộ chính xác thấp, nhạy với ánh sáng và góc nhìn
HOG + SVM	Trích xuất đặc trưng HOG kết hợp bộ phân loại SVM	Phát hiện người tương đối tốt, ổn định hơn Haar	Không phù hợp dữ liệu lớn, khó mở rộng
R-CNN	Hai giai đoạn: đề xuất vùng và phân loại bằng CNN	Dộ chính xác cao	Tốc độ xử lý chậm, tiêu tốn tài nguyên
Fast R-CNN	Cải tiến R-CNN, dùng shared convolution	Nhanh hơn R-CNN, chính xác	Vẫn phụ thuộc vào Selective Search
Faster R-CNN	Sử dụng RPN để đề xuất vùng	Cân bằng tốt giữa tốc độ và độ chính xác	Chưa phù hợp thời gian thực trên hệ nhúng
YOLO	Phát hiện đối tượng trong một lần suy luận (one-stage)	Tốc độ nhanh, phù hợp thời gian thực	Dộ chính xác giảm nhẹ với vật thể rất nhỏ

Bảng 1: So sánh các phương pháp phát hiện vật thể

### 2.2.3 Triển khai mô hình học sâu với ONNX Runtime

Trong đồ án, các mô hình học sâu như YOLOv8n, YOLOv8n-face và MobileFaceNet được triển khai dưới định dạng ONNX (Open Neural Network Exchange). ONNX là một định dạng trung gian cho phép biểu diễn mô hình học sâu một cách độc lập với framework huấn luyện ban đầu, giúp tăng khả năng tương thích và tái sử dụng mô hình trên nhiều nền tảng khác nhau.

Việc chuyển các mô hình sang định dạng ONNX cho phép hệ thống triển khai trực tiếp trên Raspberry Pi mà không phụ thuộc vào các framework huấn luyện nặng như PyTorch hay TensorFlow. Điều này đặc biệt quan trọng đối với các thiết bị nhúng có tài nguyên phần cứng hạn chế.

ONNX Runtime được sử dụng làm công cụ suy luận (inference engine) chính cho hệ thống. Đây là một môi trường thực thi tối ưu cho các mô hình ONNX, hỗ trợ nhiều nền tảng phần cứng khác nhau, trong đó có kiến trúc CPU ARM của Raspberry Pi. ONNX Runtime thực hiện các tối ưu ở mức đồ thị tính toán và toán tử, giúp giảm thời gian suy luận và cải thiện hiệu năng xử lý.

Các ưu điểm chính của ONNX Runtime trong đồ án bao gồm:

Hỗ trợ hiệu quả kiến trúc CPU ARM, phù hợp với Raspberry Pi.

Tối ưu tốc độ suy luận, đáp ứng yêu cầu xử lý thời gian thực đối với video và hình ảnh.

Cho phép triển khai đồng thời nhiều mô hình học sâu trong cùng một hệ thống.

Giảm độ phức tạp trong quá trình cài đặt và triển khai so với việc sử dụng trực tiếp framework huấn luyện.

Nhờ sử dụng ONNX Runtime, hệ thống có thể chạy ổn định các mô hình thị giác máy tính như phát hiện người, phát hiện khuôn mặt, nhận diện khuôn mặt và phân tích tư thế trên Raspberry Pi. Điều này góp phần đảm bảo tính khả thi và hiệu quả của hệ thống lớp học thông minh trong môi trường thực tế.

#### 2.2.4 Ứng dụng đếm số lượng học sinh trong lớp học

Đếm số lượng học sinh trong lớp học là một trong những ứng dụng quan trọng của thị giác máy tính trong đề tài lớp học thông minh. Thông tin về sĩ số lớp học không chỉ phục vụ cho công tác quản lý và giám sát, mà còn đóng vai trò là dữ liệu đầu vào cho hệ thống IoT nhằm tự động điều chỉnh các thiết bị như đèn và quạt theo tình trạng thực tế của lớp học.

Bài toán đếm học sinh được tiếp cận dưới dạng bài toán phát hiện đối tượng (Object Detection), trong đó mỗi học sinh xuất hiện trong khung hình camera được xem là một đối tượng thuộc lớp *person*. Hệ thống cần xác định chính xác vị trí và số lượng các đối tượng này trong ảnh thu nhận từ camera theo thời gian thực. Do yêu cầu xử lý liên tục và độ trễ thấp, mô hình phát hiện đối tượng được lựa chọn phải đảm bảo cân bằng giữa độ chính xác và tốc độ suy luận.

Trong đề tài, mô hình YOLOv8n được lựa chọn để giải quyết bài toán đếm số lượng học sinh nhờ ưu điểm về tốc độ xử lý nhanh, kích thước mô hình nhỏ và khả năng triển khai hiệu quả trên nền tảng phần cứng hạn chế như Raspberry Pi.

#### Mô hình YOLOv8n

YOLOv8n là phiên bản *nano* của dòng mô hình YOLOv8, được thiết kế nhằm tối ưu cho các ứng dụng thời gian thực trên hệ thống nhúng. YOLOv8n thuộc nhóm mô hình phát hiện đối tượng theo hướng *one-stage detector*, trong đó toàn bộ quá trình phát hiện và phân loại đối tượng được thực hiện trong một lần suy luận duy nhất trên ảnh đầu vào.

So với các phiên bản lớn hơn như YOLOv8s, YOLOv8m hay YOLOv8l, YOLOv8n có số lượng tham số và kích thước mô hình nhỏ hơn đáng kể, giúp giảm yêu cầu về bộ nhớ và năng lực tính toán. Điều này đặc biệt quan trọng khi triển khai trên Raspberry Pi, nơi tài nguyên CPU và RAM bị giới hạn.

Về mặt kiến trúc, YOLOv8n bao gồm ba thành phần chính: backbone, neck và head. Backbone đảm nhiệm việc trích xuất các đặc trưng quan trọng từ ảnh đầu vào; neck kết hợp các đặc trưng ở nhiều mức độ phân giải khác nhau nhằm nâng cao khả năng phát hiện đối tượng có kích thước khác nhau; head thực hiện dự đoán trực tiếp vị trí bounding box, nhãn lớp và độ

tin cậy của từng đối tượng theo cơ chế anchor-free. Cách tiếp cận anchor-free giúp đơn giản hóa quá trình huấn luyện và cải thiện khả năng tổng quát hóa của mô hình.

Nhờ kiến trúc gọn nhẹ và cơ chế suy luận nhanh, YOLOv8n đáp ứng tốt yêu cầu phát hiện người trong môi trường lớp học với độ chính xác chấp nhận được, đồng thời đảm bảo tốc độ xử lý phù hợp cho các ứng dụng giám sát theo thời gian thực.

Mô hình	Kích thước ảnh	mAP <sub>val</sub> (50–95)	Số tham số (M)	FLOPs (B)
YOLOv8n	640	37.3	3.2	8.7
YOLOv8s	640	44.9	11.2	28.6
YOLOv8m	640	50.2	25.9	78.9
YOLOv8l	640	52.9	43.7	165.2
YOLOv8x	640	53.9	68.2	257.8

Bảng 2: So sánh các phiên bản YOLOv8 theo độ chính xác và độ phức tạp mô hình

### Ứng dụng YOLOv8n trong bài toán đếm học sinh

Trong đề tài, mô hình YOLOv8n được sử dụng để phát hiện học sinh thông qua hình ảnh thu nhận từ camera lắp đặt trong lớp học. Mô hình được cấu hình để tập trung phát hiện đối tượng thuộc lớp *person*, tương ứng với mỗi học sinh xuất hiện trong khung hình.

Quá trình đếm học sinh được thực hiện bằng cách xác định số lượng bounding box hợp lệ sau khi áp dụng thuật toán Non-Max Suppression nhằm loại bỏ các khung trùng lặp. Số lượng bounding box cuối cùng chính là số lượng học sinh có mặt trong lớp tại thời điểm quan sát.

Kết quả đếm số lượng học sinh được cập nhật liên tục theo thời gian thực và gửi về hệ thống trung tâm. Thông tin này được sử dụng để hỗ trợ giám sát sĩ số lớp học, đồng thời kết hợp với hệ thống IoT để đưa ra các quyết định điều khiển thiết bị như tăng hoặc giảm số lượng quat hoạt động, bật/tắt đèn nhằm tối ưu hóa năng lượng và nâng cao sự tiện nghi trong phòng học.

#### 2.2.5 Ứng dụng nhận diện khuôn mặt để điểm danh học sinh

Bên cạnh chức năng đếm số lượng học sinh, hệ thống lớp học thông minh còn cần xác định danh tính từng học sinh nhằm phục vụ công tác điểm danh tự động. Nhận diện khuôn mặt là một bài toán quan trọng của thị giác máy tính, cho phép hệ thống nhận biết một cá nhân cụ thể dựa trên đặc trưng sinh trắc học của khuôn mặt thu nhận từ camera.

Trong đồ án, chức năng nhận diện khuôn mặt được triển khai nhằm thay thế phương pháp điểm danh thủ công truyền thống, giúp giảm thời gian điểm danh, hạn chế sai sót và nâng cao mức độ tự động hóa trong quản lý lớp học.

## Mô hình YOLOv8n-face

YOLOv8n-face là phiên bản chuyên biệt của mô hình YOLOv8, được huấn luyện cho bài toán phát hiện khuôn mặt (Face Detection). Mô hình này kế thừa kiến trúc one-stage detector của YOLOv8, cho phép phát hiện đối tượng chỉ trong một lần suy luận, từ đó đạt tốc độ xử lý cao và phù hợp với các hệ thống thời gian thực.

Không giống các mô hình YOLO thông thường chỉ dự đoán hộp bao (bounding box), YOLOv8n-face còn dự đoán đồng thời 5 điểm đặc trưng (facial landmarks) trên khuôn mặt, bao gồm hai mắt, mũi và hai khóm miệng. Các điểm đặc trưng này cung cấp thông tin hình học quan trọng, phục vụ cho các bước xử lý tiếp theo như căn chỉnh khuôn mặt và nhận diện danh tính.

Về kiến trúc tổng thể, YOLOv8n-face gồm ba thành phần chính:

**Backbone:** mạng nơ-ron tích chập nhẹ (CSP-based) dùng để trích xuất đặc trưng hình ảnh.

**Neck:** cấu trúc FPN/PAN giúp kết hợp đặc trưng đa tỉ lệ, tăng khả năng phát hiện khuôn mặt ở nhiều kích thước khác nhau.

**Head (anchor-free):** dự đoán trực tiếp tọa độ bounding box, độ tin cậy và landmarks mà không sử dụng anchor box, giúp giảm độ phức tạp và tăng tốc độ suy luận.

Phiên bản **YOLOv8n (nano)** có số lượng tham số nhỏ, dung lượng mô hình thấp và thời gian suy luận nhanh, rất phù hợp để triển khai trên các thiết bị nhúng có tài nguyên hạn chế như Raspberry Pi 4B trong đồ án.



Hình 2: Minh họa mô hình YOLOv8n-face phát hiện khuôn mặt và các điểm đặc trưng (facial landmarks)

## Ứng dụng YOLOv8n-face trong bài toán đếm học sinh

Trong hệ thống của đề tài, YOLOv8n-face được sử dụng để phát hiện khuôn mặt học sinh từ luồng video thời gian thực thu nhận bởi camera OV5647. Mỗi khuôn mặt được phát hiện tương ứng với một học sinh trong lớp học. Do đó, số lượng bounding box hợp lệ sau khi xử lý sẽ phản

ánh trực tiếp số lượng học sinh có mặt trong lớp tại thời điểm đó.

Quy trình áp dụng YOLOv8n-face trong hệ thống được thực hiện như sau:

1. Ảnh đầu vào từ camera được tiền xử lý (resize, chuẩn hóa giá trị pixel) trước khi đưa vào mô hình.
2. YOLOv8n-face dự đoán các bounding box khuôn mặt kèm theo độ tin cậy và 5 điểm landmarks.
3. Thuật toán Non-Maximum Suppression (NMS) được áp dụng để loại bỏ các bounding box trùng lặp.
4. Số lượng bounding box còn lại sau NMS được sử dụng để xác định số lượng học sinh trong lớp học.

Ngoài chức năng đếm số lượng học sinh, các điểm landmarks do YOLOv8n-face cung cấp còn được sử dụng để căn chỉnh khuôn mặt trước khi đưa vào mô hình nhận diện khuôn mặt MobileFaceNet. Việc căn chỉnh giúp chuẩn hóa tư thế khuôn mặt, từ đó nâng cao độ chính xác cho quá trình nhận diện và điểm danh tự động.

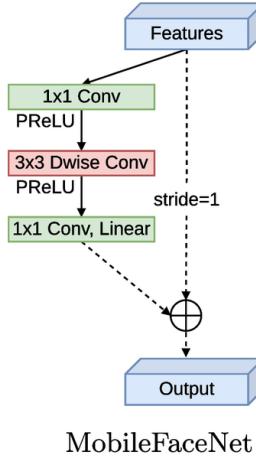
Nhờ việc sử dụng YOLOv8n-face, hệ thống đạt được sự cân bằng giữa độ chính xác và tốc độ xử lý, đáp ứng tốt yêu cầu triển khai thời gian thực trên nền tảng phần cứng nhúng, đồng thời tạo nền tảng cho các ứng dụng thị giác máy tính khác trong đề tài.

## Mô hình MobileFaceNet

Trong đồ án, mô hình MobileFaceNet được sử dụng để trích xuất đặc trưng khuôn mặt. Đây là một mạng nơ-ron tích chập nhẹ (lightweight CNN), được thiết kế tối ưu cho các thiết bị có tài nguyên hạn chế như Raspberry Pi.

MobileFaceNet không thực hiện phân loại trực tiếp, mà ánh xạ mỗi khuôn mặt đầu vào thành một vector đặc trưng (embedding) trong không gian nhiều chiều. Vector này chứa các đặc trưng quan trọng đại diện cho khuôn mặt và được sử dụng để so sánh với các khuôn mặt khác trong cơ sở dữ liệu.

Ưu điểm của MobileFaceNet là kích thước mô hình nhỏ, tốc độ suy luận nhanh và vẫn đảm bảo độ chính xác cao, rất phù hợp cho các hệ thống nhúng và ứng dụng thời gian thực.



Hình 3: Kiến trúc mô hình MobileFaceNet dùng cho trích xuất đặc trưng khuôn mặt

### ArcFace và không gian đặc trưng

Để nâng cao khả năng phân biệt giữa các khuôn mặt, mô hình MobileFaceNet trong đồ án sử dụng hàm mất mát ArcFace trong quá trình huấn luyện. ArcFace áp dụng ràng buộc góc (angular margin) trong không gian đặc trưng, giúp các vector đặc trưng của cùng một người nằm gần nhau hơn, đồng thời tăng khoảng cách giữa các vector của những người khác nhau.

Về mặt toán học, ArcFace chuẩn hóa vector đặc trưng  $\mathbf{x}$  và vector trọng số  $\mathbf{W}$  về cùng độ dài, sau đó biểu diễn xác suất phân loại dựa trên góc giữa hai vector này. Hàm mất mát ArcFace được định nghĩa như sau:

$$L = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cdot \cos(\theta_{y_i} + m)}}{e^{s \cdot \cos(\theta_{y_i} + m)} + \sum_{j \neq y_i} e^{s \cdot \cos(\theta_j)}} \quad (1)$$

Trong đó:

$N$  là số lượng mẫu trong một batch huấn luyện,

$\theta_{y_i}$  là góc giữa vector đặc trưng của mẫu thứ  $i$  và vector trọng số của lớp đúng,

$\theta_j$  là góc giữa vector đặc trưng và vector trọng số của lớp  $j$ ,

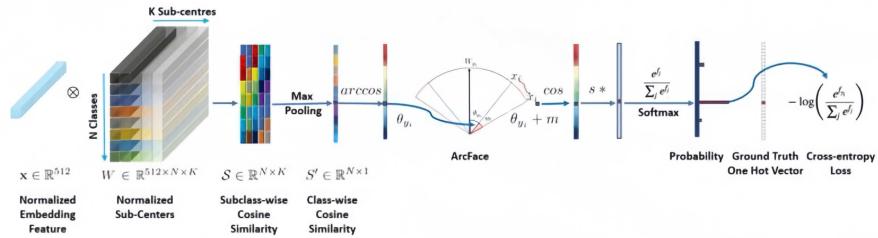
$m$  là hệ số biên góc (angular margin),

$s$  là hệ số tỉ lệ (scale factor) nhằm ổn định quá trình huấn luyện.

Việc cộng thêm biên góc  $m$  vào  $\theta_{y_i}$  buộc mô hình phải học các đặc trưng khuôn mặt có độ phân tách lớn hơn giữa các lớp khác nhau, từ đó cải thiện khả năng phân biệt trong không gian embedding. Nhờ cơ chế này, các khuôn mặt của cùng một học sinh sẽ được ánh xạ gần nhau hơn, trong khi các khuôn mặt của những học sinh khác nhau sẽ được đẩy xa nhau hơn trong không gian đặc trưng.

Nhờ ArcFace, hệ thống nhận diện đạt độ chính xác cao ngay cả khi số lượng học sinh trong

lớp lớn hoặc khi khuôn mặt có sự thay đổi nhỏ về biểu cảm, góc nhìn và điều kiện ánh sáng.



Hình 4: Minh họa cơ chế ArcFace với ràng buộc góc trong không gian đặc trưng

### Quy trình điểm danh bằng nhận diện khuôn mặt

Quy trình nhận diện khuôn mặt và điểm danh trong đồ án được thực hiện theo các bước sau:

1. Thu thập bộ dữ liệu hình ảnh khuôn mặt học sinh
2. Mô hình phát hiện khuôn mặt xác định vị trí các khuôn mặt trong ảnh.
3. Các vùng khuôn mặt được cắt, căn chỉnh và chuẩn hóa kích thước.
4. MobileFaceNet trích xuất vector đặc trưng cho từng khuôn mặt.
5. Vector đặc trưng được so sánh với cơ sở dữ liệu bằng độ đo cosine similarity.
6. Nếu độ tương đồng vượt ngưỡng xác định, khuôn mặt được gán đúng danh tính học sinh.

Để tăng độ tin cậy, học sinh chỉ được xác nhận là có mặt khi khuôn mặt được nhận diện liên tục trong một khoảng thời gian nhất định, nhằm tránh các trường hợp nhận diện nhầm do nhiễu.

### Ứng dụng trong hệ thống lớp học thông minh

Chức năng nhận diện khuôn mặt được tích hợp trực tiếp vào hệ thống lớp học thông minh và mang lại nhiều lợi ích thực tiễn:

Tự động điểm danh học sinh theo thời gian thực.

Lưu trữ lịch sử điểm danh vào cơ sở dữ liệu để phục vụ quản lý.

Kết hợp với chức năng đếm số lượng học sinh để kiểm tra tính nhất quán của sĩ số.

Cung cấp dữ liệu đầu vào cho hệ thống IoT và AI nhằm nâng cao khả năng giám sát lớp học.

### 2.2.6 Ứng dụng phân tích tư thế và phát hiện ngủ gật

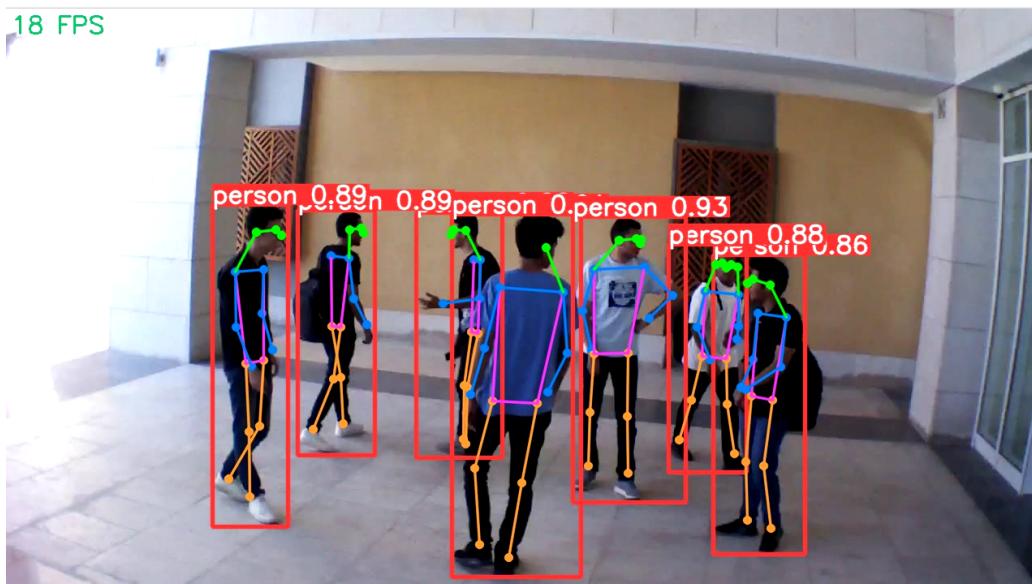
Trong môi trường lớp học, tình trạng học sinh ngủ gật hoặc mất tập trung có ảnh hưởng trực tiếp đến hiệu quả học tập. Do đó, bên cạnh các chức năng đếm số lượng học sinh và nhận diện khuôn mặt để điểm danh, đồ án còn triển khai ứng dụng phân tích tư thế và phát hiện ngủ gật dựa trên thị giác máy tính.

Ứng dụng này kết hợp giữa hai hướng tiếp cận: phân tích tư thế cơ thể thông qua các điểm khớp (pose estimation) và phân tích trạng thái mắt dựa trên landmarks khuôn mặt. Việc kết hợp nhiều đặc trưng giúp hệ thống tăng độ tin cậy và giảm các trường hợp phát hiện sai.

#### Mô hình YOLOv8n-Pose

YOLOv8n-Pose là phiên bản mở rộng của YOLOv8, cho phép mô hình không chỉ phát hiện đối tượng người mà còn dự đoán đồng thời các điểm khớp (keypoints) trên cơ thể. Mỗi người được biểu diễn bởi một bounding box và tập hợp các keypoint tương ứng như: mũi, mắt, vai, hông, đầu gối và mắt cá chân.

Trong đồ án, YOLOv8n-Pose được sử dụng để trích xuất các keypoint quan trọng phục vụ phân tích tư thế ngồi học của học sinh, đặc biệt tập trung vào các điểm: mũi, vai và hông. Đây là các điểm có ý nghĩa lớn trong việc xác định trạng thái cúi đầu hoặc gập người.



Hình 5: Minh họa YOLOv8n-Pose phát hiện người và các điểm khớp cơ thể

#### Phân tích tư thế cúi đầu

Dựa trên các keypoint do YOLOv8n-Pose cung cấp, hệ thống tính toán các đại lượng hình học nhằm đánh giá tư thế của học sinh, bao gồm:

**HDS (Head Down Score):** tỷ lệ khoảng cách theo trực dọc giữa mũi và vai so với khoảng cách vai–hông.

**NPA (Neck Pitch Angle):** góc giữa vector từ vai đến mũi và phương thẳng đứng.

**TA (Torso Angle):** góc nghiêng của thân người so với phương thẳng đứng.

Các chỉ số này được làm mượt theo thời gian bằng trung bình trượt nhằm giảm nhiễu do chuyển động ngắn hạn. Tuy nhiên, trong phiên bản cuối của hệ thống, các chỉ số tư thế chỉ đóng vai trò hỗ trợ, không phải tiêu chí quyết định trực tiếp trạng thái ngủ gật.

### Theo dõi và định danh học sinh theo thời gian

Dể đảm bảo việc phân tích được duy trì liên tục cho từng học sinh, hệ thống sử dụng một bộ theo dõi đơn giản (simple tracker) dựa trên khoảng cách giữa các điểm trung tâm cơ thể. Mỗi học sinh được gán một mã định danh (ID) và các chỉ số tư thế được theo dõi ổn định theo từng khung hình.

Cách tiếp cận này giúp hệ thống phân biệt được các học sinh khác nhau trong lớp học và tránh việc nhầm lẫn khi có nhiều người xuất hiện đồng thời trong khung hình.

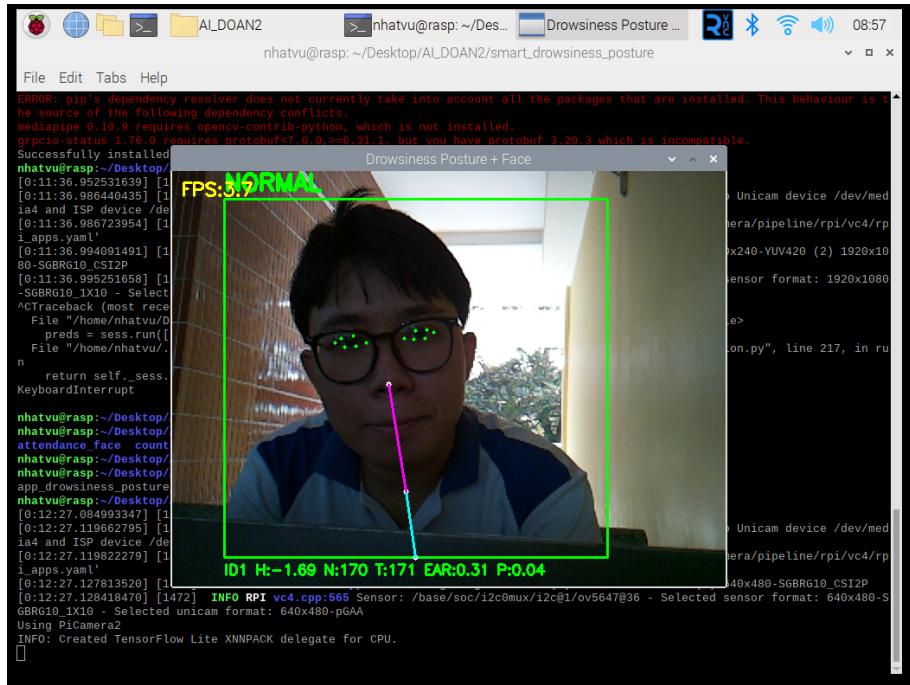
### Phân tích trạng thái mắt bằng Face Mesh

Bên cạnh phân tích tư thế, hệ thống còn sử dụng Face Mesh của MediaPipe để trích xuất các landmarks chi tiết trên khuôn mặt, đặc biệt là vùng mắt. Từ các landmarks này, hệ thống tính toán:

**EAR (Eye Aspect Ratio):** tỷ lệ hình học phản ánh mức độ mở của mắt.

**PERCLOS:** tỷ lệ thời gian mắt đóng trong một cửa sổ thời gian xác định.

EAR được sử dụng để xác định trạng thái mắt đóng/mở tại mỗi khung hình, trong khi PERCLOS phản ánh mức độ mệt mỏi hoặc buồn ngủ trong một khoảng thời gian liên tục.



Hình 6: Minh họa landmarks vùng mắt và chỉ số EAR trong phát hiện trạng thái mắt

### Nguyên lý phát hiện ngủ gật

Trong hệ thống của đồ án, trạng thái ngủ gật được xác định dựa trên **phân tích trạng thái mắt** thay vì chỉ dựa vào tư thế. Cụ thể, một học sinh được xem là có dấu hiệu ngủ gật khi:

Giá trị EAR nhỏ hơn ngưỡng xác định trong nhiều khung hình liên tiếp;

Chỉ số PERCLOS vượt quá ngưỡng cho phép trong cửa sổ thời gian quan sát.

Việc sử dụng trạng thái mắt làm tiêu chí quyết định giúp hệ thống phát hiện chính xác hơn các trường hợp ngủ gật thực sự, đồng thời giảm nhầm lẫn với các tư thế cúi đầu khi học sinh đang đọc bài hoặc viết bài.

### Ứng dụng trong hệ thống lớp học thông minh

Kết quả phát hiện ngủ gật được hiển thị trực tiếp trên giao diện giám sát và đồng thời ghi nhận vào tệp nhật ký để phục vụ thống kê. Dữ liệu này có thể được sử dụng để:

Hỗ trợ giáo viên giám sát mức độ tập trung của lớp học;

Phân tích hành vi học tập theo thời gian;

Kết hợp với hệ thống IoT để đưa ra các phản hồi thông minh như cảnh báo hoặc điều chỉnh môi trường học tập.

Nhờ việc kết hợp giữa YOLOv8n-Pose và phân tích landmarks khuôn mặt, hệ thống đạt được sự cân bằng giữa độ chính xác và khả năng triển khai thời gian thực trên nền tảng Raspberry Pi.

### **2.2.7 Tổng kết vai trò Computer Vision trong đề tài**

Thị giác máy tính giúp hệ thống lớp học thông minh thực hiện:

Dễm học sinh tự động.

Nhận diện khuôn mặt để điểm danh.

Phát hiện ngủ gật để hỗ trợ giám sát lớp học.

Hỗ trợ hệ thống IoT điều chỉnh thiết bị theo trạng thái lớp.

## **2.3 Phần cứng sử dụng trong hệ thống**

Trong hệ thống lớp học thông minh, phần cứng đóng vai trò nền tảng để thu thập dữ liệu môi trường, xử lý hình ảnh và điều khiển các thiết bị điện trong phòng học. Các thành phần chính bao gồm vi điều khiển ESP32, máy tính nhúng Raspberry Pi 4B, camera OV5647, cảm biến nhiệt độ – độ ẩm DHT22, cảm biến ánh sáng BH1750, các module relay và khối nguồn cấp.

### **2.3.1 ESP32 DevKit V1**

ESP32 DevKit V1 là bo mạch phát triển chính được sử dụng trong hệ thống IoT của đề tài. Đây là phiên bản phổ biến của dòng ESP32 do Espressif phát hành, cung cấp đầy đủ các chân GPIO và các giao thức giao tiếp cần thiết cho việc đọc cảm biến và điều khiển thiết bị điện trong mô hình phòng học thông minh.

Các đặc điểm chính:

Vi xử lý Xtensa® dual-core 32-bit, xung nhịp tối đa 240 MHz.

Tích hợp WiFi 2.4 GHz (802.11 b/g/n) và Bluetooth/BLE.

Hỗ trợ đầy đủ giao thức ngoại vi: UART, I<sup>2</sup>C, SPI, PWM, ADC, DAC, ...

Có 30–32 chân GPIO sử dụng được (tùy phiên bản), phù hợp cho việc kết nối nhiều cảm biến và module relay.

Tích hợp cổng micro-USB, mạch nạp và UART converter giúp lập trình dễ dàng thông qua Arduino IDE, PlatformIO hoặc ESP-IDF.

Trong đề tài, ESP32 DevKit V1 được sử dụng để:

Thu thập dữ liệu từ cảm biến DHT22 (nhiệt độ – độ ẩm) và BH1750 (ánh sáng).

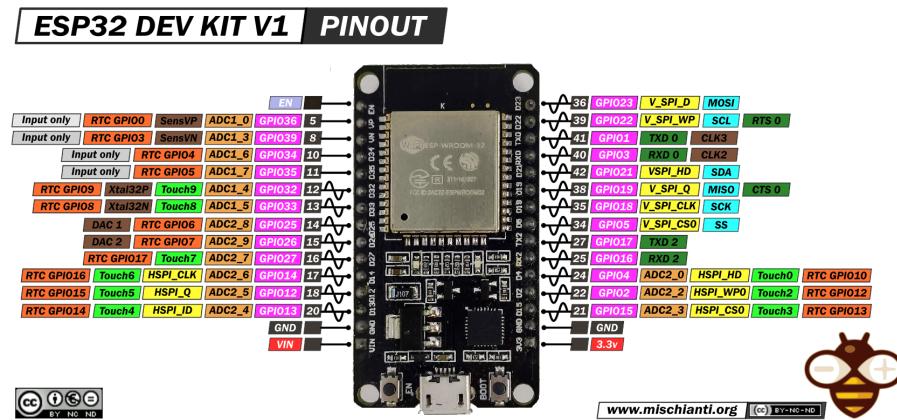
Điều khiển hai module relay (tổng cộng 6 kênh) để bật/tắt 3 đèn và 3 quạt trong mô hình phòng học.

Kết nối với Google Assistant, Home Assistant và Webserver để nhận lệnh điều khiển thiết bị.

Gửi dữ liệu cảm biến lên hệ thống giúp giám sát tình trạng phòng học theo thời gian thực.



Hình 7: Bo mạch ESP32 DevKit V1 sử dụng trong hệ thống



Hình 8: Sơ đồ chân của ESP32 DevKit V1

### 2.3.2 Raspberry Pi 4B

Raspberry Pi 4B được sử dụng làm máy tính nhúng để xử lý các mô hình trí tuệ nhân tạo trong hệ thống thị giác máy tính.

Một số thông số chính:

CPU ARM Cortex-A72, 4 nhân.

Dung lượng RAM tùy chọn (2 GB, 4 GB hoặc 8 GB). Trong dự án sử dụng bản 4GB RAM.

Cổng CSI dành cho camera.

Hệ điều hành Raspbian / Raspberry Pi OS hỗ trợ Python, OpenCV, ONNX Runtime.

Trong đề tài, Raspberry Pi 4B có nhiệm vụ:

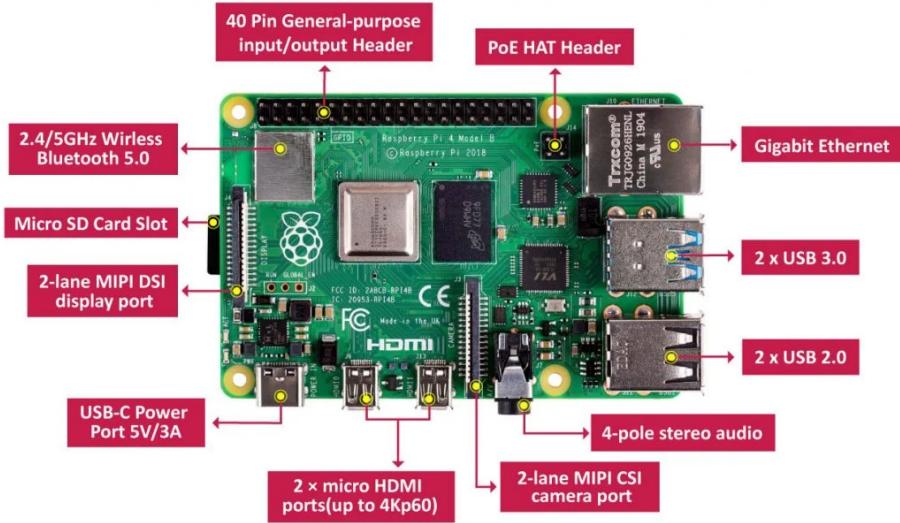
Thu nhận hình ảnh thời gian thực từ camera OV5647.

Chạy các mô hình YOLOv8 (đếm người), MobileFaceNet (nhận diện khuôn mặt) và YOLOv8-Pose (phát hiện tư thế, ngủ gật).

Gửi kết quả nhận dạng đến hệ thống IoT để phục vụ giám sát và điều khiển thiết bị.



Hình 9: Máy tính nhúng Raspberry Pi 4B



Hình 10: Sơ đồ các cổng kết nối trên Raspberry Pi 4 Model B

### 2.3.3 Camera OV5647

Camera OV5647 là module camera chính kết nối với Raspberry Pi thông qua cổng CSI, dùng để thu hình ảnh trong lớp học.

**Đặc điểm:**

Cảm biến CMOS độ phân giải 5 MP.

Hỗ trợ độ phân giải 1080p với tốc độ khung hình lên đến 30 fps.

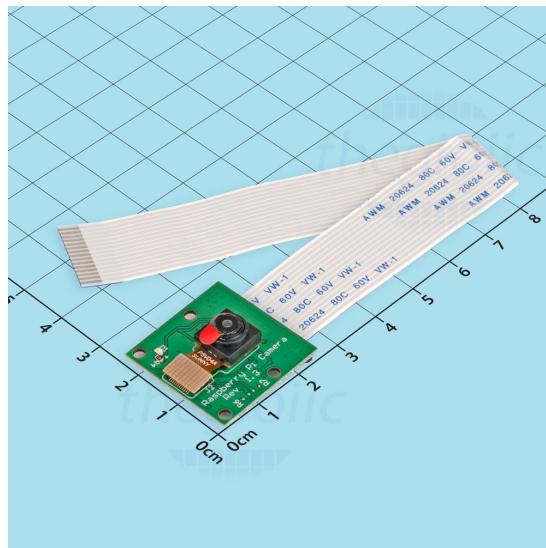
Tích hợp ống kính cố định, phù hợp giám sát không gian lớp học.

Camera được gắn tại vị trí có góc nhìn bao quát để:

Quan sát số lượng học sinh trong lớp.

Thu nhận khuôn mặt phục vụ điểm danh.

Theo dõi tư thế ngồi học và biểu hiện buồn ngủ.



Hình 11: Module camera OV5647 dùng cho Raspberry Pi

#### 2.3.4 Cảm biến nhiệt độ - độ ẩm DHT22

DHT22 là cảm biến dùng để đo nhiệt độ và độ ẩm môi trường trong phòng học.

Thông số cơ bản:

Dải đo nhiệt độ:  $-40^{\circ}\text{C}$  đến  $80^{\circ}\text{C}$ .

Dải đo độ ẩm: 0 % đến 100 % RH.

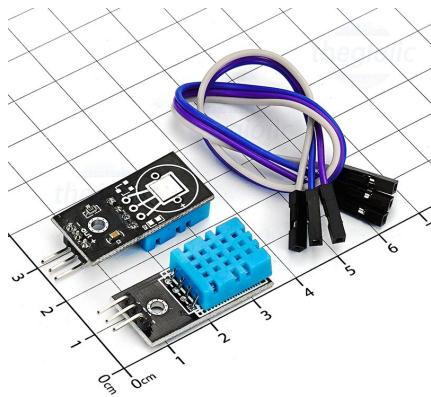
Giao tiếp một dây (1-wire), dễ dàng kết nối với ESP32.

Ứng dụng trong hệ thống:

Giám sát trạng thái nhiệt độ, độ ẩm trong lớp học.

Kích hoạt điều khiển quạt khi nhiệt độ vượt ngưỡng thiết lập.

Gửi dữ liệu lên Webserver/ứng dụng để người dùng theo dõi.



Hình 12: Cảm biến nhiệt độ – độ ẩm DHT22

### 2.3.5 Cảm biến ánh sáng BH1750

BH1750 là cảm biến đo cường độ ánh sáng môi trường, giao tiếp với vi điều khiển qua giao thức I<sup>2</sup>C.

**Đặc điểm:**

Do trực tiếp cường độ ánh sáng theo đơn vị Lux.

Giao tiếp I<sup>2</sup>C với độ phân giải cao.

Tiêu thụ công suất thấp, dễ tích hợp vào hệ thống IoT.

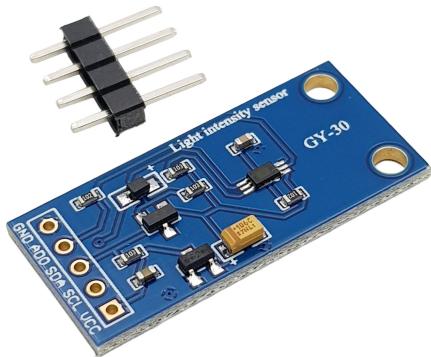
**Ứng dụng trong hệ thống:**

Do độ sáng trong phòng học.

Tự động bật đèn khi ánh sáng dưới ngưỡng cho phép.

Hỗ trợ tối ưu hóa năng lượng bằng việc tắt bớt đèn khi ánh sáng tự nhiên đủ.

Gửi dữ liệu lên Webserver/ứng dụng để người dùng theo dõi.



Hình 13: Cảm biến cường độ ánh sáng BH1750

### 2.3.6 Module relay điều khiển thiết bị

Module relay 4 kênh được sử dụng trong hệ thống để điều khiển các thiết bị điện xoay chiều (đèn, quạt, ...) thông qua tín hiệu điều khiển mức thấp từ vi điều khiển ESP32. Việc sử dụng module nhiều kênh giúp hệ thống có thể điều khiển đồng thời nhiều thiết bị trong phòng học một cách linh hoạt.

Đặc điểm:

Relay hoạt động ở mức 5 V, tiếp điểm chịu được tải AC 220 V.

Kích mức thấp (LOW-trigger), tích hợp optocoupler giúp cách ly tín hiệu và bảo vệ ESP32.

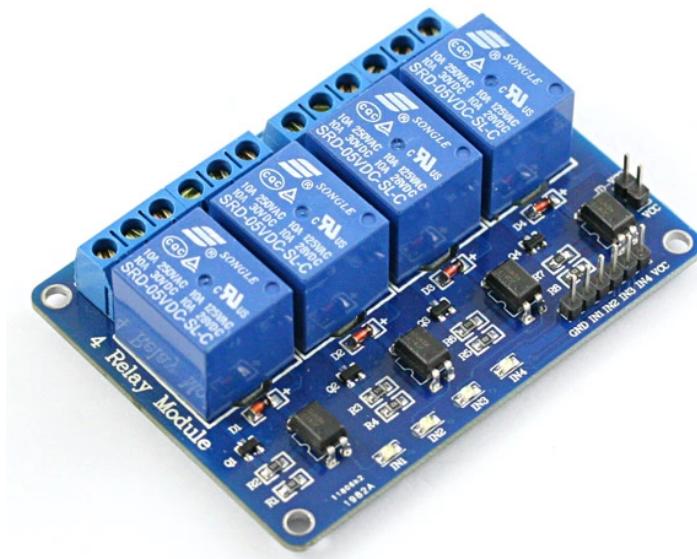
Gồm 4 kênh độc lập, phù hợp điều khiển tối đa 4 thiết bị điện.

Ứng dụng trong hệ thống:

Điều khiển bật/tắt 3 đèn trong mô hình phòng học.

Điều khiển bật/tắt 3 quạt theo điều kiện môi trường (nhiệt độ, độ ẩm) và dữ liệu từ hệ thống AI (số lượng học sinh).

Dễ dàng mở rộng bằng cách ghép thêm một module 4 kênh khác nếu cần điều khiển nhiều thiết bị hơn.



Hình 14: Module relay 4 kênh sử dụng để điều khiển đèn và quạt

### 2.3.7 Keypad 3x4 điều khiển thiết bị

Keypad 3x4 là thiết bị nhập liệu dạng ma trận gồm 12 phím bấm, được sử dụng trong hệ thống để cho phép người dùng điều khiển thủ công các thiết bị điện trong lớp học. Việc sử dụng keypad giúp hệ thống linh hoạt hơn, cho phép chuyển đổi giữa chế độ điều khiển tự động và điều khiển thủ công khi cần thiết.

Trong đồ án, keypad 3x4 được kết nối trực tiếp với vi điều khiển ESP32 thông qua các chân GPIO và được cấu hình theo dạng ma trận hàng–cột để giảm số lượng chân sử dụng.

#### Chức năng điều khiển thiết bị

Keypad 3x4 được sử dụng để điều khiển các thiết bị điện trong lớp học, bao gồm:

**3 quạt:** bật/tắt từng quạt riêng biệt.

**3 đèn:** bật/tắt từng đèn riêng biệt.

Mỗi phím trên keypad được gán một chức năng cụ thể tương ứng với từng thiết bị, giúp người dùng dễ dàng thao tác và kiểm soát trạng thái hoạt động của hệ thống.

#### Chuyển đổi chế độ điều khiển tự động và thủ công

Ngoài chức năng bật/tắt thiết bị, keypad còn được sử dụng để chuyển đổi chế độ điều khiển của hệ thống. Cụ thể, phím (\*) được dùng để bật hoặc tắt chế độ điều khiển tự động.

Khi chế độ tự động được kích hoạt, hệ thống sẽ:

Điều khiển **quạt** dựa trên dữ liệu từ cảm biến nhiệt độ và độ ẩm (DHT22).

Điều khiển **đèn** dựa trên cường độ ánh sáng môi trường đo được từ cảm biến ánh sáng (BH1750).

Ngược lại, khi chế độ tự động bị tắt, người dùng có thể sử dụng keypad để điều khiển thủ công trạng thái bật/tắt của từng quạt và đèn theo nhu cầu.

### Vai trò của keypad trong hệ thống

Việc tích hợp keypad 3x4 mang lại các lợi ích sau:

Tăng tính linh hoạt trong điều khiển hệ thống.

Cho phép vận hành hệ thống ngay cả khi không có kết nối mạng.

Hỗ trợ chuyển đổi nhanh giữa điều khiển tự động và điều khiển thủ công.

Keypad đóng vai trò là giao diện tương tác trực tiếp giữa người dùng và hệ thống lớp học thông minh, giúp nâng cao tính tiện dụng và khả năng ứng dụng thực tế của đề tài.



Hình 15: Keypad ma trận 3x4 sử dụng để điều khiển quạt, đèn và chuyển đổi chế độ tự động

### 2.3.8 Nguồn cấp

Nguồn cấp là thành phần quan trọng nhằm đảm bảo hệ thống hoạt động ổn định, đặc biệt khi các thiết bị xử lý và cảm biến hoạt động liên tục. Trong đồ án, nguồn được lựa chọn dựa trên yêu cầu điện áp, dòng điện và công suất tiêu thụ của từng khối phần cứng.

## Nguồn cấp cho ESP32 và các module ngoại vi

ESP32, các cảm biến (DHT22, BH1750), keypad 3x4 và module relay đều sử dụng nguồn một chiều 5 V. Do đó, hệ thống sử dụng adapter 5 V–3 A để cấp nguồn cho toàn bộ khối điều khiển và cảm biến.

Thông số chính của adapter:

Điện áp đầu ra: 5 V DC.

Dòng điện tối đa: 3 A.

Công suất tối đa: 15 W.

Mức dòng 3 A đảm bảo đủ công suất khi ESP32 hoạt động ở chế độ WiFi, đồng thời điều khiển nhiều thiết bị ngoại vi và relay mà không gây sụt áp.



Hình 16: Adapter 5 V–3 A cấp nguồn cho ESP32 và các module ngoại vi

## Nguồn cấp cho Raspberry Pi 4B

Raspberry Pi 4B yêu cầu nguồn 5 V với dòng điện lớn, đặc biệt khi xử lý các tác vụ thị giác máy tính và chạy mô hình AI. Trong đồ án, Raspberry Pi 4B được cấp nguồn thông qua cổng USB Type-C bằng bộ sạc Xiaomi 33 W.

Thông số cấp nguồn cho Raspberry Pi:

Điện áp đầu ra: 5 V DC (qua USB Type-C).

Dòng điện tối đa: lên đến 3 A.

Công suất danh định: 15 W (từ bộ sạc 33 W).

Việc sử dụng bộ sạc công suất lớn giúp Raspberry Pi hoạt động ổn định, tránh hiện tượng sụt áp hoặc cảnh báo thiếu nguồn khi chạy các mô hình YOLO và xử lý video thời gian thực.



Hình 17: Bộ sạc Xiaomi 33 W cấp nguồn cho Raspberry Pi 4B qua cổng USB Type-C

### Phân tách nguồn trong hệ thống

Hệ thống sử dụng nguồn riêng cho ESP32 và Raspberry Pi nhằm:

Giảm nhiễu và sụt áp giữa các khối phần cứng.

Đảm bảo độ ổn định khi Raspberry Pi xử lý tải nặng.

Nâng cao độ tin cậy và an toàn cho hệ thống.

Cách cấp nguồn này phù hợp với hệ thống lớp học thông minh có nhiều thiết bị hoạt động đồng thời và yêu cầu tính ổn định cao.

Sau khi quá trình xác thực hoàn tất, lớp bảo mật thứ hai được triển khai để bảo vệ kênh truyền dữ liệu thực tế. VoWiFi sử dụng bộ giao thức IPSec, đặc biệt là giao thức ESP (Encapsulating Security Payload) ở chế độ đường hầm (tunnel mode). Ở chế độ này, giao thức ESP sẽ lấy toàn bộ gói tin IP gốc của người dùng (bao gồm cả báo hiệu SIP và dữ liệu thoại RTP), sau đó thực hiện hai hành động bảo mật cốt lõi. Thứ nhất, nó mã hóa toàn bộ gói tin gốc bằng khóa mã hóa đã được thỏa thuận, đảm bảo tính bí mật (confidentiality). Thứ hai, nó tính toán một mã xác thực thông điệp trên gói tin đã mã hóa bằng khóa toàn vẹn, đảm bảo tính toàn vẹn (integrity). Gói tin được bảo vệ này sau đó được đặt vào payload của một gói tin IP mới, có địa chỉ IP nguồn/đích là của UE và ePDG, rồi mới được gửi đi trên Internet. Điều này đảm bảo rằng ngay cả khi bị chặn bắt, kẻ tấn công cũng không thể đọc hay thay đổi nội dung của cuộc gọi.

## 2.4 Các chuẩn giao tiếp

### 2.4.1 Giao thức HTTPS (HTTP/REST)

#### **Khái niệm**

HTTPS (HyperText Transfer Protocol Secure) là phiên bản bảo mật của giao thức HTTP, sử dụng cơ chế mã hóa SSL/TLS để đảm bảo an toàn dữ liệu trong quá trình truyền tải. HTTPS thường được sử dụng kết hợp với kiến trúc REST để trao đổi dữ liệu giữa các thiết bị và máy chủ thông qua các yêu cầu HTTP tiêu chuẩn.

#### **Thông số kỹ thuật**

Tầng giao thức: Tầng ứng dụng (Application Layer)

Giao thức nền: TCP/IP

Cổng mặc định: 443

Dịnh dạng dữ liệu: JSON

Phương thức hỗ trợ: GET, POST, PUT, PATCH

#### **Vai trò trong hệ thống**

Giao thức HTTPS (REST API) được sử dụng để giao tiếp giữa Raspberry Pi, ESP32 DevKit V1 và Firebase Realtime Database. Raspberry Pi gửi dữ liệu kết quả xử lý thị giác máy tính lên Firebase, trong khi ESP32 truy xuất trạng thái điều khiển từ Firebase để điều khiển các thiết bị vật lý trong lớp học.

#### **Chức năng chính**

Truyền dữ liệu phân tích từ Raspberry Pi lên nền tảng đám mây.

Đồng bộ trạng thái thiết bị theo thời gian thực thông qua Firebase.

Cho phép ESP32 đọc dữ liệu điều khiển và điều khiển các thiết bị như quạt và đèn.

#### **Ví dụ dữ liệu truyền**

#### **Ưu điểm**

Bảo mật cao nhờ cơ chế mã hóa SSL/TLS.

Dễ triển khai và tương thích tốt với Firebase.

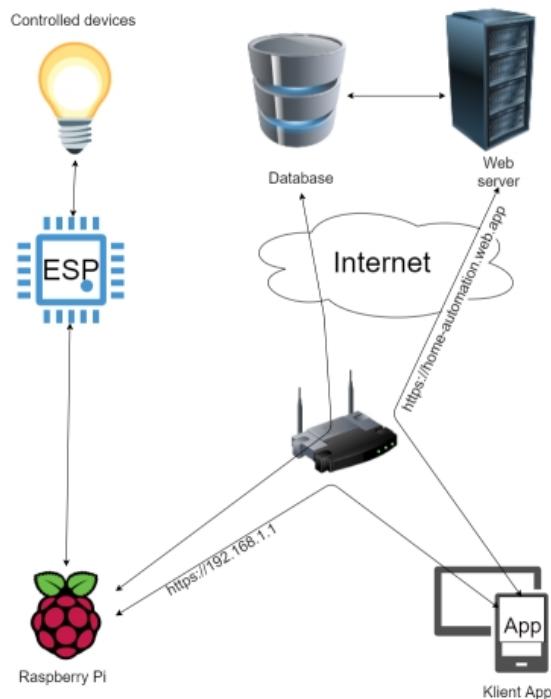
Phù hợp cho hệ thống IoT kết hợp nền tảng đám mây.

## Nhược điểm

Dộ trễ cao hơn so với các giao thức IoT nhẹ như MQTT.

Không tối ưu cho truyền dữ liệu liên tục với tần suất cao.

Phụ thuộc vào kết nối Internet ổn định.



Hình 18: Giao tiếp HTTPS giữa Raspberry Pi, Firebase Realtime Database và ESP32 DevKit V1

Việc sử dụng giao thức HTTPS (REST API) mang lại nhiều ưu điểm cho hệ thống như tính bảo mật cao, khả năng triển khai đơn giản và khả năng mở rộng tốt. Đồng thời, chuẩn giao tiếp này cho phép tích hợp hiệu quả giữa các mô-đun thị giác máy tính, nền tảng đám mây và các thiết bị IoT, góp phần đảm bảo hệ thống lớp học thông minh hoạt động ổn định và đáng tin cậy trong môi trường thực tế.

### 2.4.2 Giao thức MQTT

#### Khái niệm

MQTT (Message Queuing Telemetry Transport) là giao thức truyền thông nhẹ, hoạt động theo mô hình publish/subscribe, được thiết kế dành cho các thiết bị IoT có tài nguyên hạn chế và kết nối mạng không ổn định.

#### Thông số kỹ thuật

Tầng giao thức: Tầng ứng dụng (Application Layer)

Giao thức nền: TCP/IP

Cổng mặc định: 1883 (không mã hóa), 8883 (TLS)

Mô hình truyền thông: Publish / Subscribe

Dịnh dạng dữ liệu: JSON hoặc chuỗi ký tự

### **Vai trò trong hệ thống**

Trong hệ thống lớp học thông minh, MQTT được sử dụng để truyền lệnh điều khiển và trạng thái thiết bị giữa Raspberry Pi và ESP32 DevKit V1 thông qua MQTT Broker. Raspberry Pi đóng vai trò publisher gửi lệnh, trong khi ESP32 đóng vai trò subscriber nhận lệnh và điều khiển thiết bị.

### **Chức năng chính**

Truyền lệnh điều khiển thiết bị với độ trễ thấp.

Giảm tải cho Raspberry Pi khi điều khiển nhiều thiết bị.

Phù hợp cho giao tiếp IoT thời gian thực.

### **Ưu điểm**

Giao thức nhẹ, tiết kiệm băng thông.

Độ trễ thấp, phù hợp điều khiển thời gian thực.

Hoạt động ổn định trên thiết bị tài nguyên thấp như ESP32.

### **Nhược điểm**

Cần triển khai MQTT Broker riêng.

Không phù hợp cho lưu trữ dữ liệu dài hạn.

## **2.4.3 Giao thức I<sup>2</sup>C**

### **Khái niệm**

I<sup>2</sup>C (Inter-Integrated Circuit) là giao thức truyền thông nối tiếp đồng bộ, được phát triển bởi Philips, cho phép nhiều thiết bị tích hợp giao tiếp với nhau chỉ thông qua hai đường dây tín hiệu. Giao thức này được thiết kế nhằm giảm số lượng dây kết nối và đơn giản hóa việc giao tiếp giữa vi điều khiển và các linh kiện ngoại vi.

Hai đường tín hiệu chính của I<sup>2</sup>C bao gồm:

**SDA (Serial Data):** đường truyền dữ liệu hai chiều.

**SCL (Serial Clock):** đường xung nhịp do thiết bị master tạo ra.

I<sup>2</sup>C hỗ trợ mô hình truyền thông *master-slave*, trong đó thiết bị master chịu trách nhiệm khởi tạo giao tiếp và điều khiển xung nhịp, còn các thiết bị slave phản hồi theo địa chỉ được gán sẵn.

### Đặc điểm kỹ thuật

Số dây kết nối: 2 dây (SDA, SCL).

Kiểu truyền dữ liệu: Nối tiếp, đồng bộ.

Hỗ trợ nhiều slave trên cùng một bus thông qua địa chỉ.

Tốc độ truyền phổ biến: 100 kHz (Standard Mode), 400 kHz (Fast Mode).

Khoảng cách truyền ngắn, phù hợp giao tiếp trên bo mạch.

### Cơ chế kết nối Master – Slave

Giao thức I<sup>2</sup>C cho phép kết nối nhiều thiết bị theo mô hình *multi-drop bus*, trong đó các thiết bị cùng chia sẻ chung hai đường tín hiệu SDA và SCL. Tùy theo cấu hình hệ thống, I<sup>2</sup>C hỗ trợ hai cơ chế kết nối chính:

**Một master – nhiều slave (Single Master):** Đây là cơ chế phổ biến nhất, trong đó chỉ có một thiết bị master điều khiển bus và nhiều thiết bị slave phản hồi theo địa chỉ. Trong đồ án, ESP32 đóng vai trò master, còn cảm biến BH1750 là slave.

**Nhiều master – nhiều slave (Multi-Master):** I<sup>2</sup>C cho phép nhiều thiết bị master cùng tồn tại trên bus. Các master sẽ thực hiện cơ chế phân xử (arbitration) để tránh xung đột khi truyền dữ liệu. Tuy nhiên, cơ chế này ít được sử dụng trong các hệ thống nhúng đơn giản.

### Vai trò trong hệ thống

Trong đồ án lớp học thông minh, giao thức I<sup>2</sup>C được sử dụng để giao tiếp giữa vi điều khiển ESP32 và cảm biến ánh sáng BH1750. ESP32 đóng vai trò là thiết bị master, thực hiện việc khởi tạo giao tiếp, gửi yêu cầu đo và nhận dữ liệu ánh sáng từ cảm biến.

Việc sử dụng I<sup>2</sup>C giúp giảm số lượng chân kết nối, đơn giản hóa sơ đồ phần cứng và cho phép dễ dàng mở rộng thêm các cảm biến khác trên cùng bus trong tương lai.

### Ưu điểm

Giảm số lượng dây kết nối so với các giao thức song song.

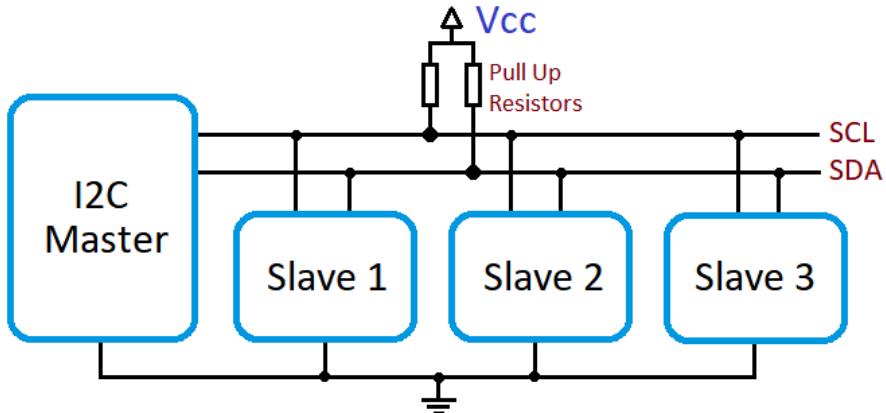
Dễ triển khai, phổ biến trong các hệ thống nhúng.

Phù hợp với các cảm biến môi trường và thiết bị tiêu thụ năng lượng thấp.

### Hạn chế

Tốc độ truyền thấp hơn so với SPI.

Khoảng cách truyền hạn chế, dễ bị nhiễu khi dây kết nối dài.



Hình 19: Minh họa giao tiếp I<sup>2</sup>C giữa thiết bị master và các slave

#### 2.4.4 Nền tảng Home Assistant

Home Assistant là một nền tảng mã nguồn mở dùng để xây dựng và quản lý các hệ thống tự động hóa thông minh. Nền tảng này cho phép giám sát, điều khiển và tự động hóa các thiết bị IoT thông qua một giao diện tập trung, có thể hoạt động cục bộ hoặc kết hợp với các dịch vụ đám mây.

Home Assistant hỗ trợ nhiều giao thức truyền thông phổ biến như MQTT, HTTP/REST, WebSocket và dễ dàng tích hợp với các nền tảng điều khiển bằng giọng nói như Google Assistant. Nhờ khả năng mở rộng cao và hoạt động ổn định trên Raspberry Pi, Home Assistant thường được sử dụng như trung tâm điều khiển trong các hệ thống IoT quy mô nhỏ và trung bình.

#### Vai trò của Home Assistant trong hệ thống

Trong hệ thống lớp học thông minh của đề tài, Home Assistant đóng vai trò là trung tâm điều khiển trung gian, thực hiện các chức năng chính sau:

Nhận lệnh điều khiển từ người dùng thông qua giao diện Web hoặc giọng nói.

Kết nối với Google Assistant để hỗ trợ điều khiển thiết bị bằng giọng nói.

Giao tiếp với ESP32 thông qua giao thức MQTT để điều khiển các thiết bị như đèn và quạt.

Hỗ trợ điều khiển thủ công song song với chế độ tự động của hệ thống.

Qua Home Assistant, người dùng có thể quản lý toàn bộ hệ thống lớp học thông minh một cách trực quan, linh hoạt và thuận tiện.



## Google Asystent vs. Google Home vs. Home Assistant

Hình 20: Tổng quan Home Asisstant

### Tổng kết

Home Assistant giúp kết nối hiệu quả giữa người dùng, nền tảng điều khiển giọng nói và các thiết bị IoT trong hệ thống. Việc sử dụng Home Assistant góp phần nâng cao mức độ tự động hóa, tăng tính linh hoạt trong điều khiển và đảm bảo hệ thống lớp học thông minh hoạt động ổn định trong môi trường thực tế.

#### 2.4.5 Nền tảng Google Assistant

Google Assistant là trợ lý ảo do Google phát triển, cho phép người dùng tương tác với hệ thống thông qua giọng nói tự nhiên. Google Assistant hỗ trợ nhận dạng giọng nói (Speech-to-Text), xử lý ngôn ngữ tự nhiên (Natural Language Processing) và chuyển đổi lệnh nói thành các hành động điều khiển cụ thể.

Trong các hệ thống IoT, Google Assistant thường được sử dụng như một giao diện điều khiển thân thiện, giúp người dùng điều khiển thiết bị mà không cần thao tác trực tiếp trên ứng dụng hoặc phần cứng.

#### Vai trò của Google Assistant trong hệ thống

Trong hệ thống lớp học thông minh của đề tài, Google Assistant được tích hợp nhằm hỗ trợ điều khiển thiết bị bằng giọng nói. Quy trình hoạt động cơ bản như sau:

Người dùng phát lệnh bằng giọng nói (ví dụ: bật đèn, tắt quạt).

Google Assistant nhận dạng và xử lý nội dung lệnh.

Lệnh được chuyển tiếp đến Home Assistant.

Home Assistant gửi lệnh điều khiển xuống ESP32 thông qua giao thức MQTT.

ESP32 thực hiện bật/tắt các thiết bị tương ứng.

Cách tiếp cận này giúp hệ thống vận hành thuận tiện, giảm thao tác thủ công và nâng cao mức độ tự động hóa trong phòng học.

### **Ưu điểm khi sử dụng Google Assistant**

Điều khiển thiết bị bằng giọng nói, thao tác nhanh và trực quan.

Không cần tiếp xúc trực tiếp với công tắc hoặc giao diện điều khiển.

Dễ dàng tích hợp với Home Assistant và các hệ thống IoT hiện có.

### **Hạn chế**

Phụ thuộc vào kết nối Internet.

Độ chính xác nhận dạng giọng nói có thể bị ảnh hưởng bởi môi trường ồn.



Hình 21: Minh họa Google Assistant điều khiển thiết bị thông qua Home Assistant và MQTT

### **Tổng kết**

Việc tích hợp Google Assistant giúp hệ thống lớp học thông minh có thêm một phương thức điều khiển hiện đại, tiện lợi và phù hợp với xu hướng nhà thông minh. Kết hợp với Home Assistant và MQTT, Google Assistant góp phần nâng cao trải nghiệm người dùng và tăng tính ứng dụng thực tế của đề tài.

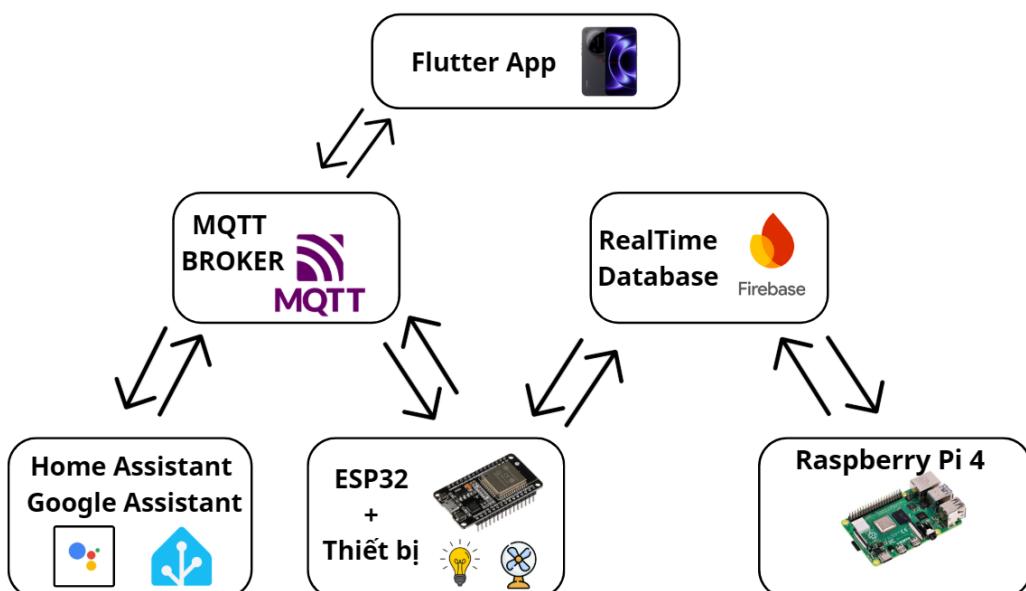
### 3 Thiết kế hệ thống

#### 3.1 Mô hình hoạt động của hệ thống

Sơ đồ hoạt động của hệ thống lớp học thông minh thể hiện kiến trúc tổng thể và luồng trao đổi dữ liệu giữa các thành phần phần cứng, phần mềm và dịch vụ đám mây trong quá trình vận hành. Hệ thống được thiết kế theo mô hình phân tán, trong đó Raspberry Pi 4 đóng vai trò trung tâm xử lý, thực hiện các tác vụ thị giác máy tính như đếm số lượng học sinh, nhận diện khuôn mặt để điểm danh và phân tích tư thế nhằm phát hiện trạng thái ngủ gật. Các kết quả xử lý được cập nhật lên cơ sở dữ liệu thời gian thực để phục vụ giám sát và lưu trữ.

ESP32 đảm nhiệm vai trò điều khiển trực tiếp các thiết bị trong lớp học như đèn và quạt thông qua module relay. ESP32 nhận lệnh điều khiển từ MQTT Broker theo mô hình publish/subscribe, sau đó xử lý và thực thi các hành động tương ứng trên phần cứng. Đồng thời, trạng thái thiết bị và dữ liệu cảm biến có thể được gửi ngược lại hệ thống nhằm đảm bảo tính đồng bộ.

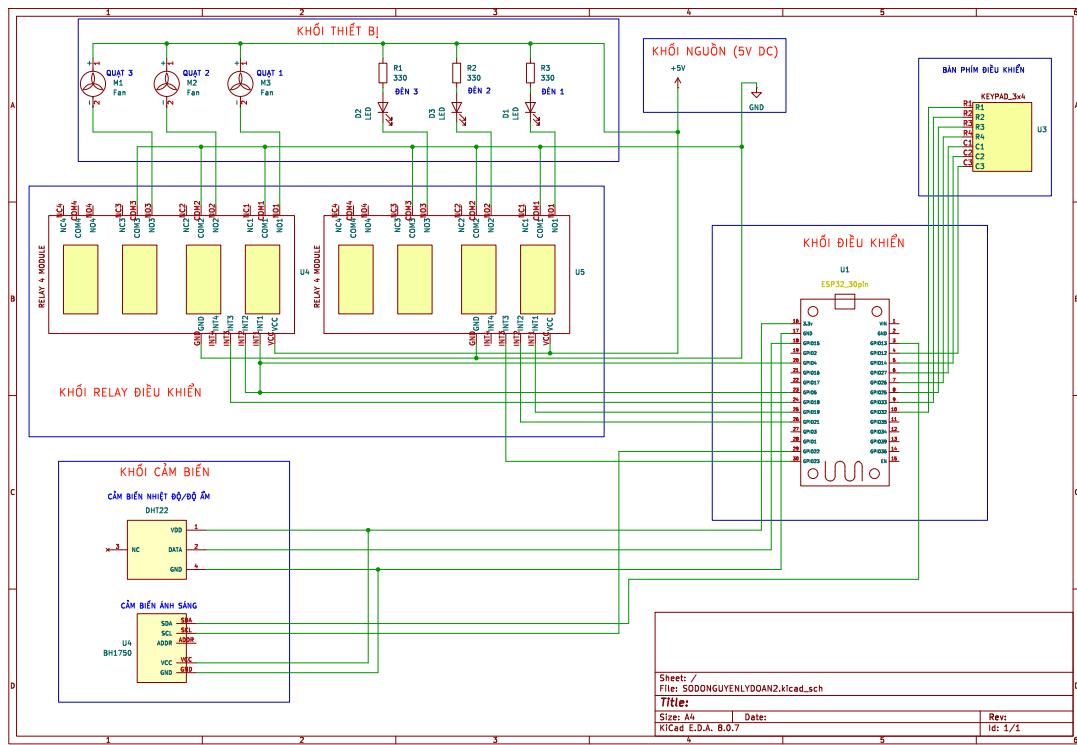
MQTT Broker đóng vai trò trung gian truyền thông, giúp kết nối các thành phần trong hệ thống một cách linh hoạt và giảm sự phụ thuộc trực tiếp giữa các thiết bị. Ứng dụng Flutter và các nền tảng như Home Assistant, Google Assistant có thể gửi lệnh điều khiển thông qua MQTT, trong khi Firebase Realtime Database được sử dụng để đồng bộ dữ liệu giữa Raspberry Pi và ứng dụng người dùng theo thời gian thực. Nhờ kiến trúc này, hệ thống đảm bảo khả năng hoạt động ổn định, mở rộng linh hoạt và đáp ứng tốt yêu cầu của một lớp học thông minh trong môi trường thực tế.



Hình 22: Sơ đồ hoạt động tổng thể của hệ thống lớp học thông minh

### 3.2 Sơ đồ nguyên lý tổng quan phần cứng

Sơ đồ nguyên lý tổng quan phần cứng mô tả cách kết nối và tương tác giữa các khối phần cứng chính trong hệ thống lớp học thông minh, bao gồm khối điều khiển trung tâm, khối cảm biến, khối chấp hành và khối giao diện người dùng. Sơ đồ này tập trung vào phần hệ thống IoT do vi điều khiển ESP32 đảm nhiệm, trong khi Raspberry Pi được xem là một khối xử lý độc lập cho thị giác máy tính và không tham gia trực tiếp vào mạch điều khiển phần cứng mức thấp.



Hình 23: Sơ đồ nguyên lý tổng quan phần cứng hệ thống lớp học thông minh

#### Khối điều khiển trung tâm (ESP32)

ESP32 DevKit V1 đóng vai trò là bộ điều khiển trung tâm của hệ thống IoT. Vi điều khiển này chịu trách nhiệm:

Đọc dữ liệu từ các cảm biến môi trường.

Xử lý lệnh điều khiển nhận được từ MQTT Broker hoặc từ keypad.

Xuất tín hiệu điều khiển tới các module relay để bật/tắt đèn và quạt.

Các chân GPIO của ESP32 được phân bổ để kết nối với keypad 3x4 (theo cấu trúc ma trận hàng–cột), cảm biến DHT22, cảm biến BH1750 (qua giao thức I<sup>2</sup>C) và các chân điều khiển relay. Việc bố trí này giúp tối ưu số lượng chân sử dụng và đảm bảo khả năng mở rộng trong tương

lai.

## **Khối cảm biến**

Khối cảm biến bao gồm:

**Cảm biến DHT22:** kết nối với ESP32 thông qua một chân GPIO, cung cấp dữ liệu nhiệt độ và độ ẩm môi trường.

**Cảm biến ánh sáng BH1750:** giao tiếp với ESP32 qua bus I<sup>2</sup>C (SDA, SCL), cho phép đo cường độ ánh sáng theo đơn vị Lux.

Dữ liệu từ các cảm biến này được ESP32 sử dụng để điều khiển thiết bị ở chế độ tự động, đồng thời gửi lên hệ thống giám sát thông qua mạng.

## **Khối relay và thiết bị chấp hành**

Hệ thống sử dụng hai module relay 4 kênh để điều khiển các thiết bị điện trong lớp học, bao gồm:

3 quạt điện (Quạt 1, Quạt 2, Quạt 3).

3 đèn chiếu sáng (Đèn 1, Đèn 2, Đèn 3).

Các relay được kích hoạt bằng tín hiệu mức thấp từ ESP32, cho phép cách ly an toàn giữa mạch điều khiển điện áp thấp và tải xoay chiều 220 V. Việc chia tải qua nhiều relay giúp hệ thống điều khiển linh hoạt từng thiết bị riêng biệt.

## **Khối giao diện người dùng (Keypad 3x4)**

Keypad 3x4 được kết nối trực tiếp với ESP32 và hoạt động như một giao diện điều khiển cục bộ. Người dùng có thể:

Bật/tắt thủ công từng đèn và quạt.

Sử dụng phím (\*) để chuyển đổi giữa chế độ điều khiển tự động và điều khiển thủ công.

Việc tích hợp keypad giúp hệ thống vẫn có thể vận hành ngay cả khi mất kết nối mạng hoặc không sử dụng ứng dụng điều khiển từ xa.

## **Khối nguồn**

Toàn bộ khối ESP32, cảm biến, keypad và relay được cấp nguồn từ adapter 5 V–3 A. Nguồn được phân phối song song tới các module, đảm bảo cung cấp đủ dòng cho relay hoạt động ổn định mà không gây sụt áp cho vi điều khiển.

## Vai trò của Raspberry Pi trong hệ thống

Mặc dù không được thể hiện trong sơ đồ nguyên lý phần cứng, Raspberry Pi 4B giữ vai trò quan trọng trong hệ thống tổng thể. Raspberry Pi được kết nối với camera OV5647 và đảm nhiệm toàn bộ các tác vụ thị giác máy tính như:

Dếm số lượng học sinh trong lớp.

Nhận diện khuôn mặt để điểm danh.

Phân tích tư thế và phát hiện trạng thái ngủ gật.

Các kết quả xử lý từ Raspberry Pi không đi qua mạch phần cứng ESP32 mà được truyền tới hệ thống IoT thông qua HTTPS (Firebase), từ đó ảnh hưởng gián tiếp đến quyết định điều khiển thiết bị trong lớp học.

## 4 Thi công hệ thống

Chương này trình bày quá trình thi công và triển khai hệ thống lớp học thông minh, bao gồm việc lựa chọn linh kiện, lắp ráp phần cứng và chuẩn bị cho quá trình vận hành thực tế. Nội dung chương tập trung vào thi công phần cứng của hệ thống IoT và thị giác máy tính đã được thiết kế ở chương trước.

### 4.1 Lưu đồ thuật toán toàn hệ thống

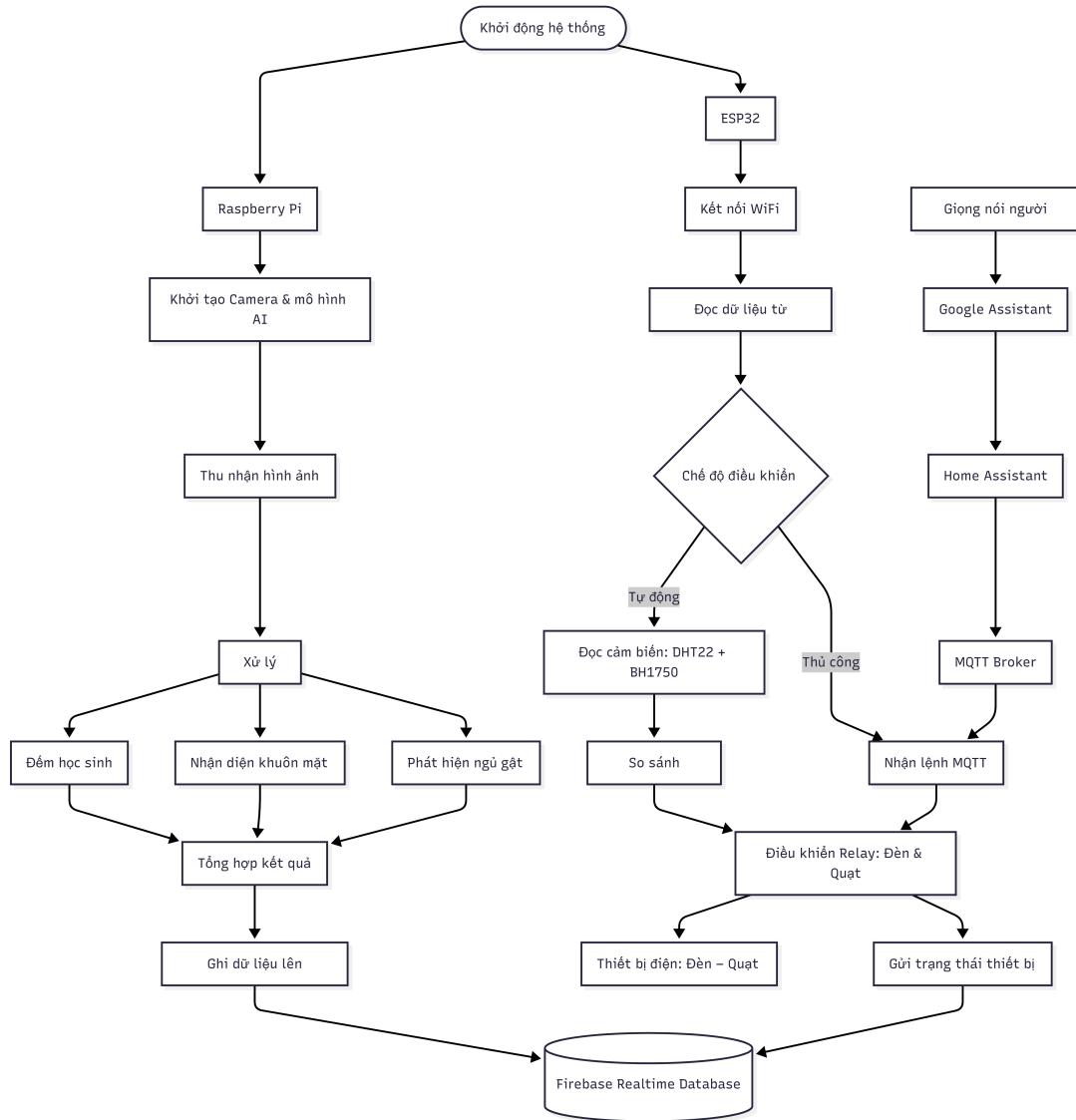
Lưu đồ thuật toán toàn hệ thống mô tả trình tự hoạt động và luồng xử lý dữ liệu giữa các thành phần chính của hệ thống lớp học thông minh, bao gồm Raspberry Pi, ESP32, hệ thống cảm biến, nền tảng đám mây và các thiết bị điều khiển.

Khi hệ thống khởi động, Raspberry Pi tiến hành khởi tạo camera và các mô hình trí tuệ nhân tạo. Hình ảnh nhận từ camera được xử lý để thực hiện các tác vụ như đếm số lượng học sinh, nhận diện khuôn mặt và phát hiện trạng thái ngủ gật. Các kết quả này được tổng hợp và gửi lên Firebase Realtime Database để lưu trữ và đồng bộ dữ liệu.

Song song đó, ESP32 kết nối WiFi và đọc dữ liệu từ Firebase để xác định chế độ điều khiển hiện tại của hệ thống. Trong chế độ tự động, ESP32 thu thập dữ liệu từ các cảm biến môi trường (DHT22 và BH1750), so sánh với các ngưỡng cài đặt và điều khiển đèn, quạt thông qua module relay. Trong chế độ thủ công, ESP32 nhận lệnh điều khiển từ người dùng thông qua giao thức MQTT.

Lệnh điều khiển thủ công được tạo từ giọng nói người dùng, xử lý bởi Google Assistant, chuyển tiếp qua Home Assistant và MQTT Broker trước khi được gửi đến ESP32. Sau khi thiết bị được điều khiển, trạng thái hoạt động sẽ được cập nhật ngược lại lên Firebase nhằm đảm bảo tính đồng bộ của toàn hệ thống.

Lưu đồ thuật toán thể hiện rõ sự phối hợp giữa các khối xử lý, điều khiển và giao tiếp, giúp hệ thống hoạt động ổn định, linh hoạt và đáp ứng yêu cầu của một lớp học thông minh.



Hình 24: Lưu đồ thuật toán toàn hệ thống lớp học thông minh

## 4.2 Thi công phần cứng

Phần thi công phần cứng nhằm hiện thực hóa sơ đồ nguyên lý và thiết kế hệ thống đã trình bày ở Chương 3. Quá trình này bao gồm việc lựa chọn linh kiện phù hợp, kết nối các module và bố trí phần cứng đảm bảo hệ thống hoạt động ổn định, an toàn và dễ mở rộng.

#### 4.2.1 Linh kiện sử dụng

Bảng 3 liệt kê các linh kiện chính được sử dụng trong quá trình thi công phần cứng của hệ thống lớp học thông minh.

STT	Tên thiết bị	Điện áp	Số lượng
1	ESP32 DevKit V1	5 V	1
2	Raspberry Pi 4B (4GB RAM)	5 V	1
3	Camera OV5647	3.3 V	1
4	Module relay 4 kênh	5 V	2
5	Cảm biến nhiệt độ – độ ẩm DHT22	5 V	1
6	Cảm biến ánh sáng BH1750 (I <sup>2</sup> C)	3.3 V	1
7	Keypad ma trận 3x4	3.3 V	1
8	Breadboard		2
9	Adapter nguồn 5 V – 3 A	5 V	1
10	Bộ sạc USB Type-C Xiaomi 33 W	5 V	1
11	Dây kết nối, jack nguồn	5 V	Nhiều
12	Quạt 4010	5 V	3
13	LED Trắng 10mm Đức	1.8-2.8 V	3

Bảng 3: Thống kê linh kiện sử dụng cho hệ thống

#### 4.2.2 Tiến hành thi công phần cứng

Quá trình thi công phần cứng được thực hiện dựa trên sơ đồ nguyên lý và thiết kế hệ thống đã trình bày ở chương trước. Mục tiêu của quá trình này là hiện thực hóa mô hình lớp học thông minh với đầy đủ các khái niệm IoT và thị giác máy tính, đảm bảo hệ thống hoạt động ổn định, an toàn và dễ dàng quan sát trong quá trình vận hành thử nghiệm.

Toàn bộ phần cứng được bố trí trong một mô hình phòng học thu nhỏ. Raspberry Pi 4B được đặt tại vị trí trung tâm phía trên để thuận tiện cho việc kết nối camera OV5647 và xử lý dữ liệu thị giác máy tính. Camera được lắp cố định hướng xuống khu vực trung tâm mô hình nhằm đảm bảo góc quan sát bao quát cho các tác vụ đếm học sinh, nhận diện khuôn mặt và phân tích tư thế.

Các thiết bị chấp hành như đèn chiếu sáng và quạt được bố trí dọc theo các mặt bên của mô hình, mô phỏng đúng cấu trúc của một phòng học thực tế. Keypad ma trận 3x4 được lắp đặt ở mặt trước mô hình, giúp người dùng dễ dàng thao tác điều khiển thủ công và chuyển đổi giữa chế độ tự động và thủ công khi cần thiết.



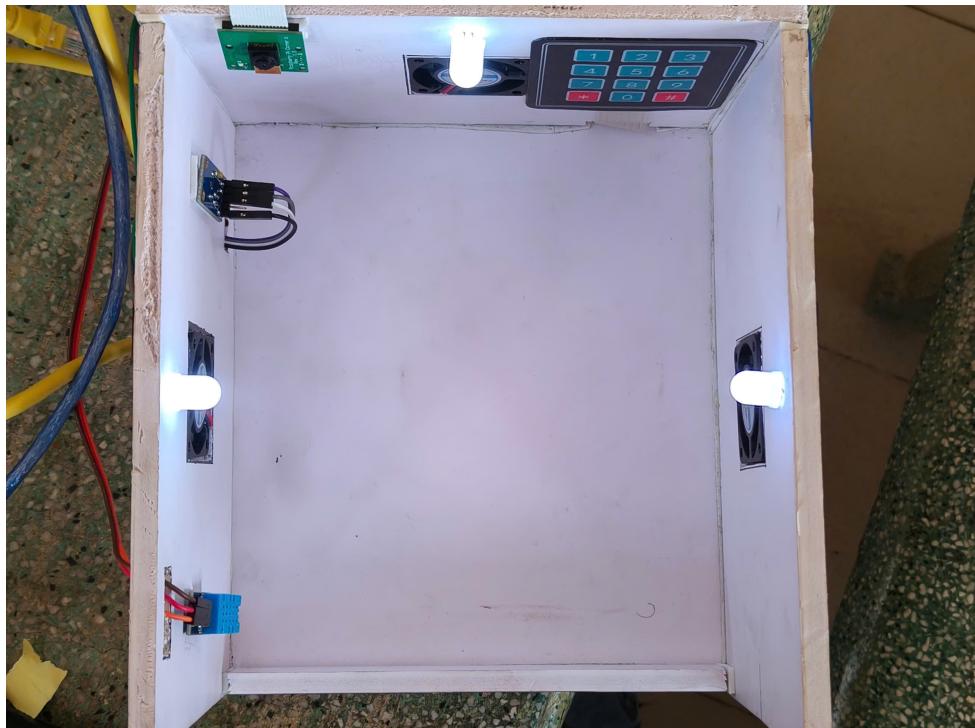
Hình 25: Bố trí tổng thể các thiết bị trong mô hình lớp học thông minh

Phần cứng điều khiển được bố trí bên trong mô hình, bao gồm ESP32 DevKit V1, breadboard và các module relay. ESP32 đóng vai trò là bộ điều khiển trung tâm của hệ thống IoT, thực hiện thu thập dữ liệu từ các cảm biến môi trường và điều khiển các thiết bị thông qua relay. Các dây kết nối nguồn và tín hiệu được đi gọn gàng, phân tách rõ ràng giữa khối điều khiển điện áp thấp và khối tải, nhằm giảm nhiễu điện và tăng độ an toàn trong quá trình vận hành.



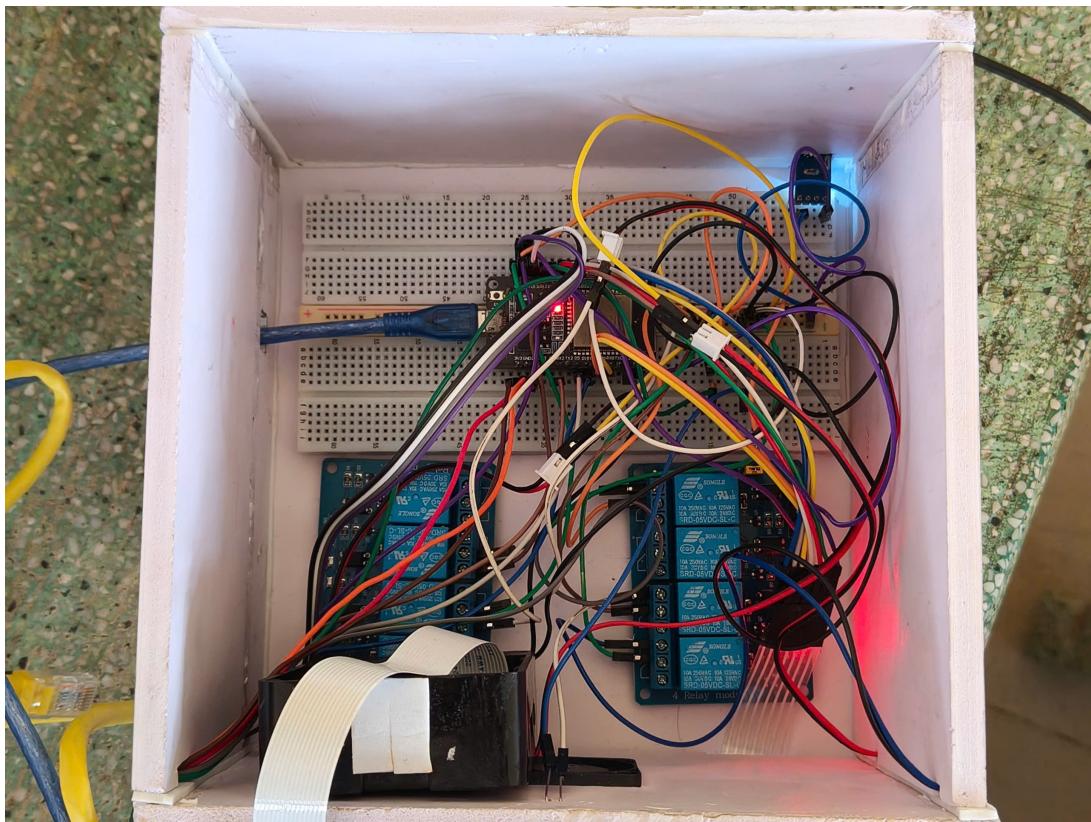
Hình 26: Bố trí phần cứng điều khiển bên trong mô hình

Sau khi hoàn tất lắp ráp, hệ thống được tiến hành vận hành thử nghiệm. Trong quá trình hoạt động, ESP32 nhận lệnh điều khiển từ MQTT Broker hoặc dữ liệu từ các cảm biến môi trường để bật/tắt đèn và quạt thông qua module relay. Đồng thời, Raspberry Pi xử lý hình ảnh thu nhận từ camera và thực hiện các tác vụ thị giác máy tính. Trạng thái hoạt động của các thiết bị được cập nhật liên tục, đảm bảo tính đồng bộ giữa phần cứng và phần mềm của hệ thống.



Hình 27: Hệ thống lớp học thông minh vận hành thực tế

Sơ đồ kết nối phần cứng của mô hình thể hiện mối liên hệ giữa các thành phần chính trong hệ thống, bao gồm ESP32, các cảm biến môi trường, keypad, module relay và các thiết bị chấp hành như đèn và quạt. Sơ đồ giúp minh họa rõ ràng luồng kết nối tín hiệu và nguồn, đồng thời hỗ trợ quá trình kiểm tra, sửa lỗi và mở rộng hệ thống trong tương lai.



Hình 28: Sơ đồ kết nối các thành phần phần cứng trong mô hình lớp học thông minh

Qua quá trình thi công và kiểm tra thực tế, hệ thống phần cứng đáp ứng tốt các yêu cầu về tính ổn định, khả năng điều khiển linh hoạt và khả năng tích hợp với hệ thống thị giác máy tính và nền tảng IoT. Đây là nền tảng quan trọng để triển khai các chức năng thông minh của hệ thống trong môi trường lớp học thực tế.

## 5 Giới thiệu ứng dụng Flutter trên điện thoại

Trong hệ thống lớp học thông minh, bên cạnh khối xử lý IoT và thị giác máy tính, ứng dụng di động đóng vai trò là giao diện tương tác trực tiếp giữa người dùng và toàn bộ hệ thống. Ứng dụng Flutter được phát triển nhằm mục đích giám sát trạng thái thiết bị, theo dõi dữ liệu môi trường và hỗ trợ điều khiển hệ thống một cách thuận tiện thông qua điện thoại thông minh.

Ứng dụng Flutter hoạt động như một thành phần giao diện (frontend), kết nối với các nền tảng trung gian như Firebase Realtime Database, MQTT Broker và Home Assistant để thực hiện trao đổi dữ liệu và điều khiển thiết bị theo thời gian thực.

### 5.1 Giới thiệu tổng quan ứng dụng Flutter

Trong phạm vi đề tài này, ứng dụng Flutter hiện tại mới được triển khai và kiểm thử trên nền tảng Android. Flutter được lựa chọn làm nền tảng phát triển ứng dụng nhờ các ưu điểm sau:

Giao diện trực quan, dễ tùy biến và thân thiện với người dùng.

Hiệu năng cao, đáp ứng tốt các yêu cầu hiển thị dữ liệu thời gian thực.

Dễ dàng tích hợp với các nền tảng IoT như Firebase, MQTT và REST API.

Phù hợp cho các hệ thống Smart Home và Smart Classroom.

Ứng dụng Flutter trong đề tài được thiết kế với các mục tiêu chính:

Hiển thị trạng thái bật/tắt của các thiết bị trong lớp học.

Giám sát dữ liệu cảm biến môi trường.

Đồng bộ dữ liệu với hệ thống IoT và AI.

Làm nền tảng mở rộng cho các chức năng điều khiển nâng cao.



Hình 29: Trang tổng quan của ứng dụng Flutter

## 5.2 Thiết lập và tích hợp hệ thống

Để ứng dụng Flutter có thể hoạt động đồng bộ với hệ thống lớp học thông minh, các thành phần xử lý và điều khiển được thiết lập và liên kết thông qua các nền tảng trung gian.

### 5.2.1 Kết nối MQTT thông qua Home Assistant

MQTT được sử dụng trong hệ thống nhằm truyền lệnh điều khiển với độ trễ thấp. Tuy nhiên, ứng dụng Flutter không kết nối trực tiếp đến MQTT Broker mà thông qua Home Assistant để đảm bảo tính bảo mật, khả năng quản lý tập trung và dễ mở rộng hệ thống.

Trong đề tài, Home Assistant được triển khai trên môi trường máy ảo (Virtual Machine). Việc sử dụng máy ảo giúp hệ thống hoạt động ổn định, tách biệt với phần cứng IoT, đồng thời thuận tiện cho việc cấu hình, sao lưu và mở rộng trong tương lai.

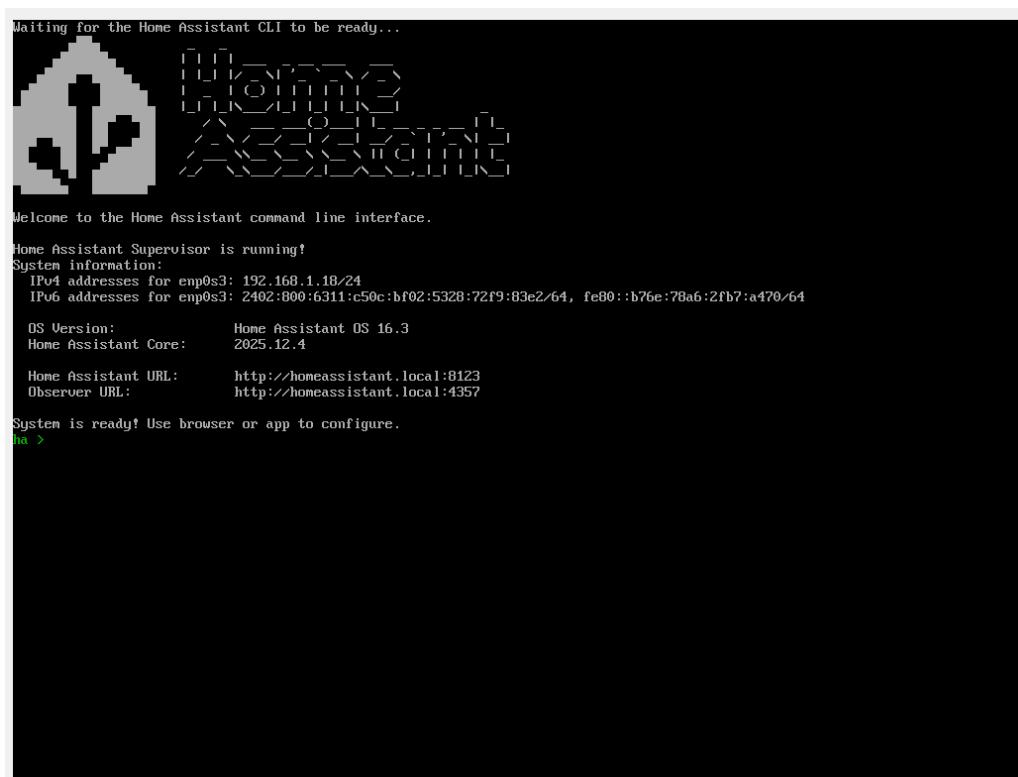
Luồng hoạt động của hệ thống được mô tả như sau:

1. Người dùng thao tác điều khiển trên ứng dụng Flutter.
2. Lệnh điều khiển được gửi tới Home Assistant.
3. Home Assistant publish lệnh lên MQTT Broker.
4. ESP32 subscribe và thực thi lệnh điều khiển các thiết bị.

Cách tiếp cận này giúp tăng tính bảo mật, giảm độ phức tạp trong ứng dụng Flutter và tận dụng khả năng tự động hóa mạnh mẽ của Home Assistant.

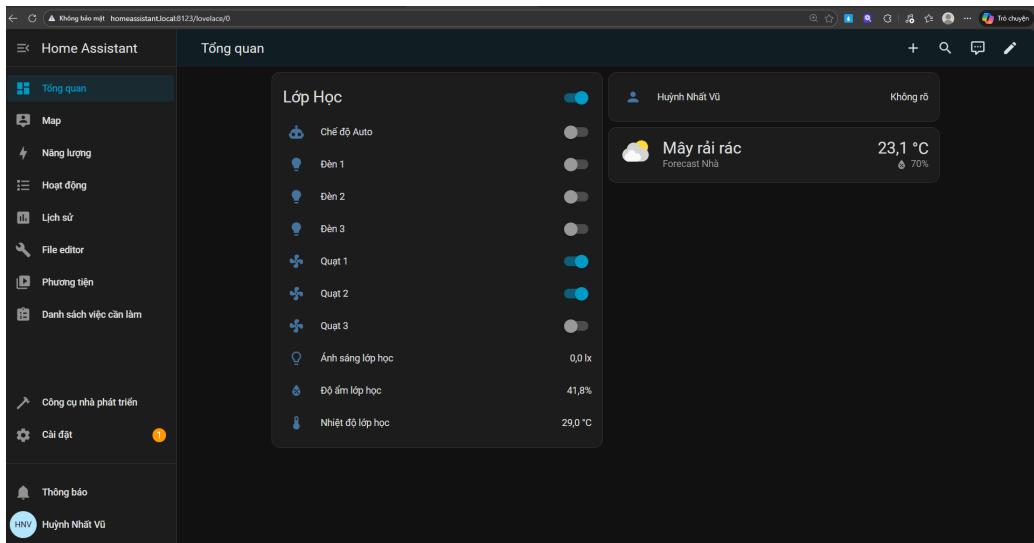
Trong hệ thống của đề tài, Home Assistant được triển khai trên môi trường máy ảo (Virtual Machine) nhằm đóng vai trò là trung tâm điều khiển và quản lý các thiết bị IoT. Việc sử dụng máy ảo giúp hệ thống hoạt động ổn định, tách biệt với phần cứng xử lý IoT và thị giác máy tính, đồng thời thuận tiện cho quá trình cài đặt, cấu hình và bảo trì.

Sau khi khởi động máy ảo, Home Assistant OS được nạp và chạy tự động, cung cấp giao diện dòng lệnh (CLI) để giám sát trạng thái hệ thống. Khi hệ thống sẵn sàng, người dùng có thể truy cập giao diện Web của Home Assistant thông qua địa chỉ IP hoặc tên miền cục bộ để tiến hành cấu hình và quản lý thiết bị.



Hình 30: Luồng điều khiển thiết bị từ Flutter thông qua Home Assistant và MQTT

Giao diện Dashboard của Home Assistant cho phép người dùng giám sát trạng thái thiết bị, theo dõi dữ liệu và điều khiển hệ thống một cách trực quan thông qua trình duyệt web hoặc ứng dụng di động.



Hình 31: Giao diện Dashboard của Home Assistant trong hệ thống lớp học thông minh

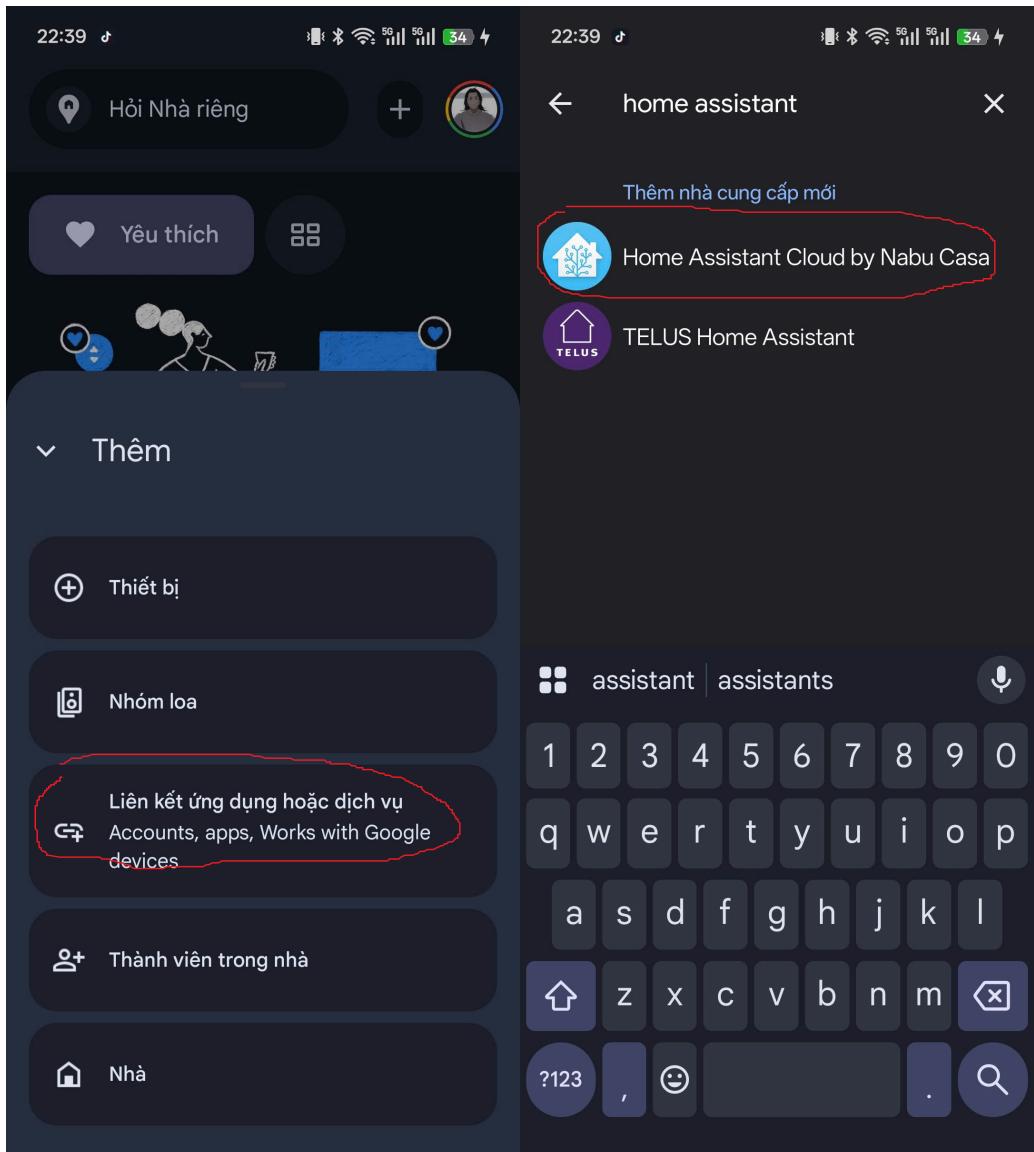
### 5.2.2 Liên kết Home Assistant với Google Assistant

Để hỗ trợ điều khiển thiết bị bằng giọng nói, hệ thống sử dụng Home Assistant làm cầu nối giữa Google Assistant và các thiết bị IoT.

Quy trình liên kết bao gồm các bước chính:

1. Cấu hình Home Assistant Cloud hoặc Google Actions.
2. Đăng nhập tài khoản Google.
3. Liên kết Home Assistant với Google Home.
4. Đồng bộ các thiết bị điều khiển.

Sau khi liên kết thành công, người dùng có thể sử dụng giọng nói để điều khiển các thiết bị trong lớp học.



Hình 32: Liên kết Google Assistant với Home Assistant để điều khiển thiết bị

### 5.2.3 Thiết lập server xử lý trên Raspberry Pi

Raspberry Pi đóng vai trò là máy chủ trung tâm đảm nhiệm các tác vụ xử lý thị giác máy tính và cung cấp dữ liệu cho hệ thống lớp học thông minh. Trên Raspberry Pi, một server Python được xây dựng và triển khai dựa trên framework Flask nhằm tiếp nhận dữ liệu từ camera và cung cấp dịch vụ xử lý ảnh theo thời gian thực.

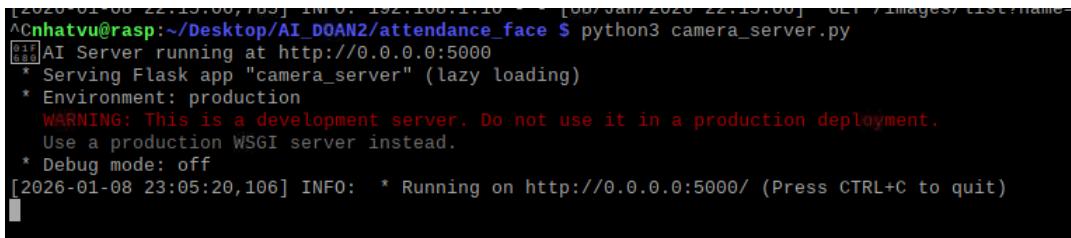
Server có các chức năng chính:

Thu nhận luồng hình ảnh từ camera OV5647 kết nối với Raspberry Pi.

Thực hiện suy luận các mô hình trí tuệ nhân tạo như YOLO (đếm người), MobileFaceNet (nhận diện khuôn mặt) và mô hình phân tích tư thế.

Cung cấp API nội bộ để xử lý dữ liệu hình ảnh và gửi kết quả lên Firebase Realtime Database.

Server Flask được khởi chạy trực tiếp trên Raspberry Pi thông qua môi trường Python và lắng nghe tại cổng xác định (port 5000). Khi server hoạt động, hệ thống sẵn sàng tiếp nhận yêu cầu xử lý và cung cấp dữ liệu cho các thành phần khác trong hệ thống.



```
[2026-01-08 22:15:00,105] INFO: [807 Jan/2026 22:15:00] 0.0.0.0:5000 - /images/list?name=^Cnhatvu@rasp:~/Desktop/AI_DONAN2/attendance_face $ python3 camera_server.py
[0:1F]AI Server running at http://0.0.0.0:5000
[0:6F]* Serving Flask app "camera_server" (lazy loading)
[0:6F]* Environment: production
[0:6F]WARNING: This is a development server. Do not use it in a production deployment.
[0:6F]Use a production WSGI server instead.
[0:6F]* Debug mode: off
[0:6F][2026-01-08 23:05:20,106] INFO: * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

Hình 33: Server Flask xử lý thị giác máy tính được khởi chạy trên Raspberry Pi

### 5.3 Chức năng của ứng dụng Flutter

Ứng dụng Flutter được phát triển nhằm cung cấp giao diện trực quan để người dùng giám sát và điều khiển hệ thống lớp học thông minh. Các chức năng chính của ứng dụng bao gồm điều khiển thiết bị IoT, điểm danh bằng khuôn mặt, giám sát camera và theo dõi lịch sử hoạt động.

#### 5.3.1 Điều khiển thiết bị và giám sát cảm biến

Ứng dụng cho phép người dùng theo dõi dữ liệu cảm biến môi trường trong lớp học bao gồm nhiệt độ, độ ẩm và cường độ ánh sáng. Đồng thời, người dùng có thể điều khiển trạng thái bật/tắt của các thiết bị như quạt và đèn thông qua giao diện công tắc.

Ngoài chế độ điều khiển thủ công, ứng dụng còn hỗ trợ bật/tắt chế độ điều khiển tự động dựa trên dữ liệu cảm biến, giúp tối ưu hóa môi trường học tập và tiết kiệm năng lượng.

Ngoài việc điều khiển bằng nút bấm và cảm biến người dùng có thể điều khiển bằng giọng nói bằng cách nói "OK Google!, Bật/Tắt (Tên thiết bị)"

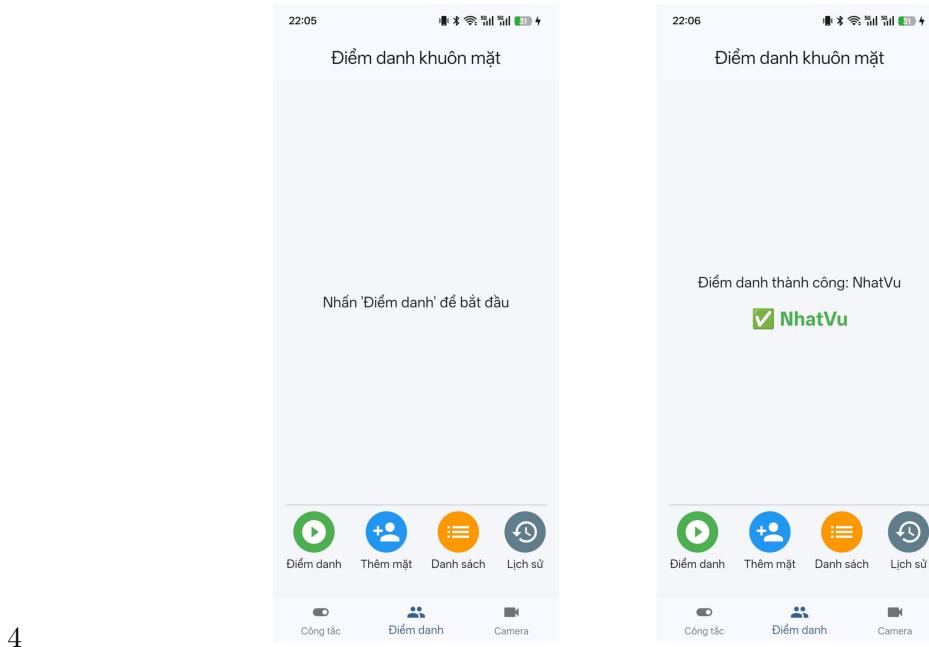


Hình 34: Giao diện điều khiển thiết bị và giám sát cảm biến trên ứng dụng Flutter

### 5.3.2 Điểm danh học sinh bằng nhận diện khuôn mặt

Ứng dụng Flutter tích hợp chức năng điểm danh học sinh dựa trên kết quả nhận diện khuôn mặt từ hệ thống thị giác máy tính chạy trên Raspberry Pi. Khi hệ thống nhận diện thành công khuôn mặt học sinh, thông tin điểm danh sẽ được hiển thị trực tiếp trên ứng dụng.

Chức năng này giúp giảm thời gian điểm danh thủ công, đồng thời nâng cao độ chính xác và tính tự động trong quản lý lớp học.

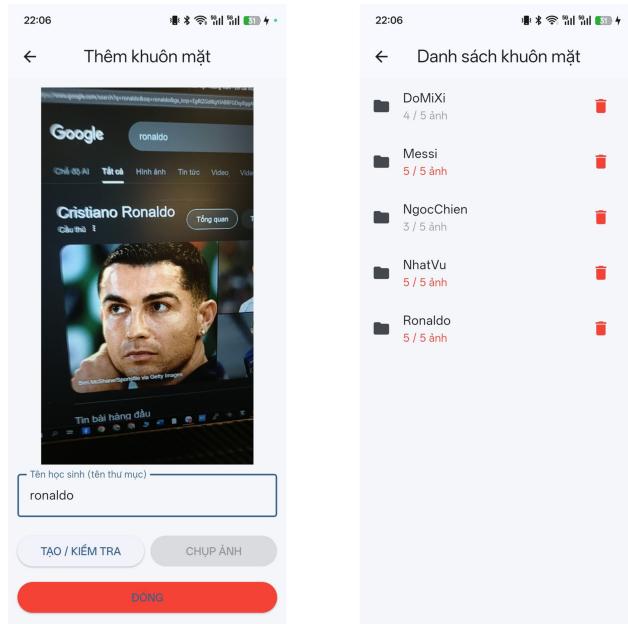


Hình 35: Giao diện hiển thị kết quả điểm danh khuôn mặt trên ứng dụng

### 5.3.3 Quản lý khuôn mặt học sinh

Ứng dụng hỗ trợ chức năng thêm mới và quản lý danh sách khuôn mặt học sinh. Người dùng có thể chụp ảnh khuôn mặt, đặt tên và lưu trữ dữ liệu để phục vụ cho quá trình nhận diện và điểm danh tự động.

Danh sách khuôn mặt đã lưu được hiển thị rõ ràng, cho phép xóa hoặc cập nhật khi cần thiết.



Hình 36: Chức năng thêm và quản lý khuôn mặt học sinh

#### 5.3.4 Xem lịch sử điểm danh

Ứng dụng cung cấp chức năng xem lịch sử điểm danh theo từng ngày, bao gồm thông tin học sinh, thời gian điểm danh và trạng thái có mặt. Dữ liệu lịch sử được lưu vào file.xlsx và có thể xuất ra file để phục vụ thống kê và quản lý.

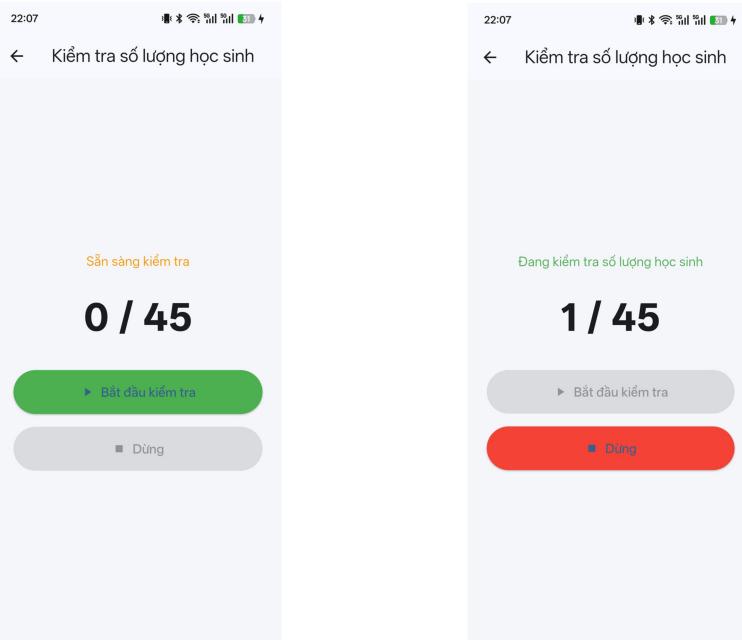
STT	Tên	Giờ vào
1	NhatVu	21:14:56

Hình 37: Giao diện lịch sử điểm danh học sinh

### 5.3.5 Kiểm tra số lượng học sinh

Ứng dụng cho phép người dùng khởi động chức năng kiểm tra số lượng học sinh trong lớp học. Dữ liệu số lượng học sinh được lấy từ kết quả phát hiện người của mô hình YOLO chạy trên Raspberry Pi và cập nhật theo thời gian thực.

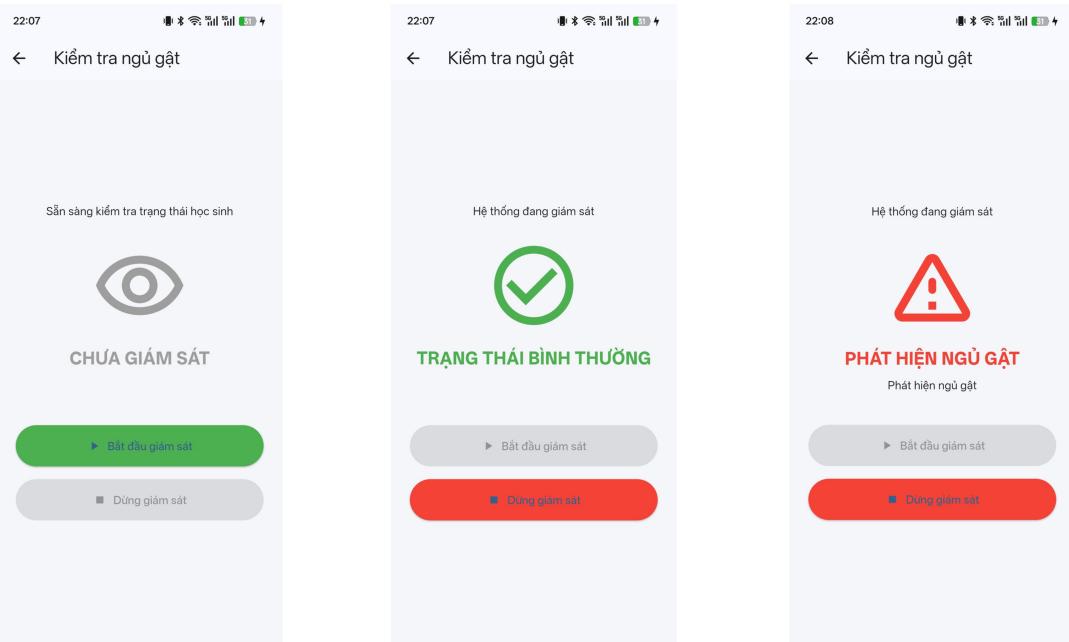
Chức năng này hỗ trợ giáo viên nhanh chóng nắm bắt số lượng học sinh tại từng thời điểm.



Hình 38: Giao diện kiểm tra số lượng học sinh trên ứng dụng Flutter

### 5.3.6 Phát hiện trạng thái ngủ gật

Ứng dụng Flutter hiển thị kết quả phân tích trạng thái ngủ gật của học sinh dựa trên dữ liệu từ hệ thống phân tích trạng thái mắt và tư thế. Khi phát hiện học sinh có dấu hiệu ngủ gật, hệ thống sẽ cập nhật trạng thái tương ứng để hỗ trợ công tác giám sát lớp học.

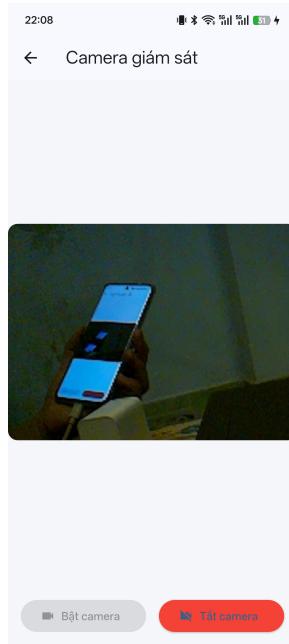


Hình 39: Giao diện kiểm tra trạng thái ngủ gật của học sinh

### 5.3.7 Giám sát camera thời gian thực

Ứng dụng Flutter cho phép người dùng xem trực tiếp hình ảnh từ camera giám sát được kết nối với Raspberry Pi. Người dùng có thể bật hoặc tắt luồng camera từ xa thông qua ứng dụng.

Chức năng này giúp giáo viên quan sát tình trạng lớp học và hỗ trợ các tác vụ giám sát thông minh.



Hình 40: Giao diện giám sát camera thời gian thực trên ứng dụng Flutter

## **6 Nhận xét và đánh giá**

### **6.1 Kết quả đạt được**

Sau quá trình nghiên cứu, thiết kế và thi công, đề tài đã xây dựng thành công một hệ thống lớp học thông minh tích hợp giữa Internet of Things (IoT) và thị giác máy tính. Hệ thống hoạt động ổn định trong mô hình thử nghiệm và đáp ứng được các mục tiêu đề ra ban đầu.

Cụ thể, hệ thống đã thực hiện được các chức năng chính sau:

Điều khiển bật/tắt các thiết bị trong lớp học (đèn, quạt) thông qua ESP32 bằng nhiều phương thức: điều khiển thủ công, điều khiển tự động theo cảm biến và điều khiển bằng giọng nói thông qua Google Assistant.

Thu thập và giám sát dữ liệu môi trường như nhiệt độ, độ ẩm và cường độ ánh sáng, từ đó hỗ trợ điều khiển thiết bị một cách hợp lý và tiết kiệm năng lượng.

Triển khai thành công các mô hình thị giác máy tính trên Raspberry Pi để đếm số lượng học sinh, nhận diện khuôn mặt phục vụ điểm danh và phát hiện trạng thái ngủ gật.

Xây dựng ứng dụng Flutter trên nền tảng Android giúp người dùng giám sát hệ thống, điều khiển thiết bị, xem lịch sử điểm danh và theo dõi camera theo thời gian thực.

Kết quả thực nghiệm cho thấy hệ thống có khả năng vận hành ổn định, dữ liệu giữa các thành phần được đồng bộ tương đối tốt và đáp ứng được yêu cầu của một mô hình lớp học thông minh ở quy mô nhỏ.

### **6.2 Đánh giá hệ thống**

#### **6.2.1 Ưu điểm**

Hệ thống lớp học thông minh trong đề tài có một số ưu điểm nổi bật:

Kiến trúc hệ thống rõ ràng, phân tách hợp lý giữa các khía cạnh IoT, thị giác máy tính và giao diện người dùng.

Sử dụng các nền tảng và công nghệ phổ biến như ESP32, Raspberry Pi, MQTT, Firebase, Home Assistant và Flutter, giúp hệ thống dễ triển khai và mở rộng.

Các mô hình trí tuệ nhân tạo được lựa chọn phù hợp với khả năng xử lý của Raspberry Pi, đảm bảo cân bằng giữa độ chính xác và tốc độ suy luận.

Ứng dụng Flutter có giao diện trực quan, dễ sử dụng, hỗ trợ nhiều chức năng quản lý lớp học trong cùng một nền tảng.

Hệ thống có tính ứng dụng thực tế cao, phù hợp để triển khai thử nghiệm trong môi trường lớp học hoặc phòng học thông minh.

### 6.2.2 Hạn chế

Bên cạnh những kết quả đạt được, hệ thống vẫn còn tồn tại một số hạn chế:

Khả năng xử lý của Raspberry Pi còn hạn chế, khiến tốc độ suy luận của các mô hình AI giảm khi số lượng học sinh trong khung hình tăng cao.

Độ chính xác của các chức năng thị giác máy tính phụ thuộc nhiều vào điều kiện ánh sáng và vị trí lắp đặt camera.

Các thuật toán điều khiển tự động dựa trên cảm biến và dữ liệu AI hiện vẫn sử dụng ngưỡng cố định, chưa có khả năng tự học hoặc thích nghi theo thời gian.

Hệ thống mới được kiểm thử trên quy mô mô hình và chưa được triển khai thực tế trong môi trường lớp học lớn.

Ứng dụng Flutter hiện tại mới được phát triển và kiểm thử trên nền tảng Android, chưa hỗ trợ đa nền tảng.

## 6.3 Hướng phát triển

Trong tương lai, đề tài có thể được mở rộng và phát triển theo các hướng sau:

Nâng cấp phần cứng xử lý, sử dụng các thiết bị có hiệu năng cao hơn hoặc tích hợp tăng tốc phần cứng (GPU, TPU) để cải thiện tốc độ suy luận AI.

Áp dụng các kỹ thuật tối ưu mô hình như quantization, pruning hoặc TensorRT nhằm nâng cao hiệu suất trên hệ thống nhúng.

Phát triển các thuật toán điều khiển thông minh hơn dựa trên học máy để hệ thống có khả năng tự điều chỉnh theo thói quen sử dụng và điều kiện môi trường.

Mở rộng hệ thống cho nhiều phòng học, kết hợp quản lý tập trung thông qua nền tảng Web.

Hoàn thiện và triển khai ứng dụng Flutter trên các nền tảng khác như iOS và Web.

#### **6.4 Kết luận chung**

Thông qua đề tài, nhóm đã tiếp cận và vận dụng hiệu quả các kiến thức về IoT, thị giác máy tính, trí tuệ nhân tạo và phát triển ứng dụng di động vào việc xây dựng một hệ thống lớp học thông minh. Mặc dù vẫn còn một số hạn chế nhất định, hệ thống đã chứng minh được tính khả thi và tiềm năng ứng dụng trong thực tế. Đây là nền tảng quan trọng để tiếp tục nghiên cứu, hoàn thiện và phát triển các giải pháp giáo dục thông minh trong tương lai.

## Tài liệu

- [1] David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton, Jerome Henry, *IoT Fundamentals: Networking Technologies, Protocols, and Use Cases for the Internet of Things*, Cisco Press, 2017.
- [2] Andy Stanford-Clark and Arlen Nipper, *MQTT Version 3.1.1 Specification*, OASIS Standard, 2014. Truy cập tại: <https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html>
- [3] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, *You Only Look Once: Unified, Real-Time Object Detection*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. Truy cập tại: <https://arxiv.org/abs/1506.02640>
- [4] Jiankang Deng, Jia Guo, Niannan Xue, Stefanos Zafeiriou, *ArcFace: Additive Angular Margin Loss for Deep Face Recognition*, IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019. Truy cập tại: <https://arxiv.org/abs/1801.07698>
- [5] Home Assistant Documentation, *Home Assistant – Open Source Home Automation Platform*, 2024. Truy cập tại: <https://www.home-assistant.io/docs/>
- [6] Google, *Flutter Documentation*, 2024. Truy cập tại: <https://docs.flutter.dev/>