ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: KỸ THUẬT ĐIỀU KHIỂN VÀ TỰ ĐỘNG HÓA

ĐỀ TÀI:

GIÁM SÁT VÀ ĐIỀU KHIỂN HỆ THỐNG ĐIỆN TRONG NHÀ THÔNG MINH

Người hướng dẫn: **TS. NGUYỄN THỊ THANH QUỲNH**
Sinh viên thực hiện:
1. PHAN NGUYỄN BẢO QUỐC – MSSV: 105200379 – LỚP: 20TDHCLC1
2. HOÀNG CÔNG MINH – MSSV: 105200371 – LỚP: 20TDHCLC1

Đà Nẵng, 6/2025

TÓM TẮT

Tên đề tài: Giám sát và điều khiển hệ thống điện trong nhà thông minh

Sinh viên thực hiện 1: Phan Nguyễn Bảo Quốc

Số thẻ SV: 105200379 Lớp: 20TĐHCLC1

Sinh viên thực hiện 2: Hoàng Công Minh

Số thẻ SV: 105200371 Lớp: 20TĐHCLC1

Trong bối cảnh cuộc cách mạng công nghiệp 4.0 phát triển mạnh mẽ, các công nghệ như Internet of Thing (IoT) và Trí tuệ nhân tạo (AI) đã và đang được ứng dụng rộng rãi vào đời sống. Đồ án này tập trung nghiên cứu, thiết kế và triển khai một hệ thống nhà thông minh đa hệ sinh thái, có khả năng tích hợp các thiết bị từ nhiều hãng khác nhau như Xiaomi, Yeelight, Tuya... vào một nền tảng điều khiển thống nhất dựa trên Home Assistant và Flask Server. Hệ thống bao gồm các tính năng nổi bật như:

- Điều khiển thiết bị thông minh từ nhiều hệ sinh thái thông qua giao diện web than thiện.
- Giám sát môi trường sống với cảm biến khí gas, nhiệt độ, độ ẩm
- Hệ thống bảo mật thông minh bằng cách nhận diện khuôn mặt (Sử dụng mô hình MTCNN và FaceNet)
- Tương tác bằng cử chỉ tay (sử dụng thư viện MeadiaPipe)
- Lưu trữ lịch sử và hiển thị dữ liệu thời gian thực

Đồ án không chỉ giải quyết bài toán phân mảng giữa các hệ sinh thái thiết bị mà còn hướng đến giải pháp mở, chi phí hợp lý, phù hợp với người dùng phổ thông tại Việt Nam. Sau khi hoàn thành đồ án, em đã tiếp thu được nhiều kiến thức về cách xây dựng và quản lý nhà thông minh, cách kết nối hệ thống, lập trình điều khiển thiết bị. Bên cạnh đó em cũng có thêm kĩ năng tự đọc hiểu tài liệu, kĩ năng làm việc nhóm và kĩ năng sử dụng các phần mềm. Từ đó giúp em có thêm nhiều hiển biết và nhận thức về chuyên ngành, có cái nhìn tốt hơn về ngành bản thân đang theo học tập, tư tin hơn sau khi tốt nghiệp.

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN

CỘNG HÒA XÃ HÔI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

TT	Họ tên sinh viên	Số thẻ SV	Lớp	Ngành
1	Phan Nguyễn Bảo Quốc	105200379	20TĐHCLC1	Kĩ Thuật Điều Khiển và Tự Động Hoá
2	Hoàng Công Minh	105200371	20TĐHCLC1	Kĩ Thuật Điều Khiển và Tự Động Hoá

1. *Tên đề tài đồ án:*

Giám sát và điều khiển hệ thống điện trong nhà thông minh

2. *Đề tài thuộc diện: ☐Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện*
3. *Các số liệu và dữ liệu ban đầu:*

……..……..…
…....……
……..…….…..……..……
……

4. *Nội dung các phần thuyết minh và tính toán:*

a. Phần chung:

TT	Họ tên sinh viên	Nội dung
1	Phan Nguyễn Bảo Quốc	- Tìm hiểu đề tài - Tìm hiểu các xu hướng phát triển nhà thông minh hiện nay. - Đọc các tài liệu, tìm hiểu giao thức - Lên lý tưởng thiết kế hệ thống và các chức năng có trong hệ thống - Viết thuyết minh đồ án
2	Hoàng Công Minh	

b. Phần riêng:

TT	Họ tên sinh viên	Nội dung
1	Phan Nguyễn Bảo Quốc	- Lập trình ESP32 giao tiếp với cảm biến - Kết nối ESP32 với server - Xây dựng server, API chức năng, MySQL

		- Lựa chọn giao thức giao tiếp giữa các phần, đẩy dashboard public
2	Hoàng Công Minh	- Triển khai hệ thống nhận diện khuôn mặt - Tích hợp chức năng nhận diện cử chỉ tay điều khiển thiết bị. - Xây dựng giao diện và các chức năng của dashboard - Lựa chọn phần cứng.

5. *Họ tên người hướng dẫn:*

Người hướng dẫn	Phần/ Nội dung
TS. Nguyễn Thị Thanh Quỳnh	Hướng dẫn quy trình thiết kế dự án cho doanh nghiệp Đề xuất các phương án, các tính năng cần thiết Hướng dẫn thuyết minh báo cáo Đánh giá thuyết minh báo cáo và sản phẩm

6. *Ngày giao nhiệm vụ đồ án:* *... .../... .../2025*
7. *Ngày hoàn thành đồ án:* *... .../... .../2025*

Đà Nẵng, ngày tháng năm 2025

Trưởng Bộ môn Tự động hóa **Người hướng dẫn**

TS. Giáp Quang Huy

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

PHIẾU KIỂM SOÁT TIẾN ĐỘ LÀM ĐỒ ÁN TỐT NGHIỆP

(Phiếu dành cho người hướng dẫn/sinh viên)

Họ tên sinh viên 1: Phan Nguyễn Bảo Quốc Số thẻ SV : 105200379

Họ tên sinh viên 2: Hoàng Công Minh Số thẻ SV : 105200371

Tên đề tài ĐATN:

Giám sát và điều khiển hệ thống điện trong nhà thông minh

Họ tên người HD: TS.Nguyễn Thị Thanh Quỳnh Đơn vị: Khoa Điện

Tuần	**Ngày**	**Khối lượng**		**GVHD ký tên**
		đã thực hiện (%)	**tiếp tục thực hiện (%)**	
1	10/3-16/3	- Nhận đề tài	- Tìm hiểu tình hình chung của sự phát triển nhà thông minh. - Xu hướng phát triển nhà thông minh	
2	17/3-23/3	- Tìm hiểu đề tài, công nghệ nền tảng (IoT, AI, HomeAssistant...) - Xác định yêu cầu của 1 hệ thống cơ bản, lập đề cương chi tiết	- Tiếp tục nghiên cứu, thực hiện dự án.	
3	24/3-30/3	- Thiết kế hệ thống tổng thể, sơ đồ kết nối các thiết bị. - Cài đặt HomeAssitant, nghiên cứu các tích hợp	- Vẽ sơ đồ hệ thống. - Viết báo cáo - Lựa chọn giao thức kết nối	
4	1/4-2/4	Duyệt lần 1: Đánh giá khối lượng hoàn thành _____ % : Được tiếp tục làm ĐATN □ Không tiếp tục thực hiện ĐATN □		

5	3/4-13/4	- Hoàn thành báo cáo chương 1. - Lập trình điều khiển thiết bị ESP32, giao tiếp với cảm biến, servo, camera	- Nghiên cứu về giao thức kết nối giữa ESP32 với server. - Xây dựng server backend Flask	
6	14/4-20/4	- Hoàn thành xây dựng server, cơ sở dữ liệu MySQL, API điều khiển thiết bị.	- Nghiên cứu về AI nhận diện khuôn mặt, xử lý ảnh từ camera. - Tìm hiểu về nhận diện cử chỉ tay	
7	21/4-27/4	- Triển khai hệ thống nhận diện khuôn mặt với MTCNN + FaceNet. - Kết hợp với ESP32 để chạy quy trình	- Tích hợp vào chương trình chính - Tiếp tục hoàn thành báo cáo chương 1, 2, 3	
8	28/4-29/5	Duyệt lần 2: Đánh giá khối lượng hoàn thành _____ % : Được tiếp tục làm ĐATN □ Không tiếp tục thực hiện ĐATN □		
9	30/4-6/5	- Tích hợp nhận diện cử chỉ tay bằng MediaPipe - Thiết kế giao diện dashboard hiển thị dữ liệu, điều khiển thiết bị từ xa	- Thiết lập cử chỉ bật tắt thiết bị, kiểm nghiệm tính chính xác. - Viết báo cáo chương 4	
10	7/5-13/5	- Tích hợp nhận diện cử chỉ vào hệ thống, kết nối đến Flask API điều khiển thiết bị. - Xây dựng hoàn chỉnh phần cứng	- Chỉnh sửa giao diện dashboard kết hợp lấy API từ HomeAssistant.	
11	14/5-20/5	- Thử nghiệm toàn hệ thống, kiểm tra các giao tiếp	- Kiểm tra tính thời gian thực, độ trễ, các cảnh báo - Viết báo cáo chương 5	
12	21/5-22/5	Duyệt lần 3: Đánh giá khối lượng hoàn thành _____ % : Được tiếp tục làm ĐATN □ Không tiếp tục thực hiện ĐATN □		

13	23/5-29/5	- Tối ưu hoá mã nguồn, xử lý lỗi, cải thiện giao diện người dùng	- Đưa trang web public cho phép người dùng truy cập từ xa.	
14	30/5-5/6	- Cho phép truy cập trang Web từ xa với quy trình đăng nhập token bảo mật. - Viết báo cáo hoàn chỉnh, làm video demo.	- Hoàn thiện slide thuyết trình	
15	6/6-15/6	- Hoàn thành đồ án		

LỜI NÓI ĐẦU VÀ CẢM ƠN

Là sinh viên năm cuối đang theo học tại trường Đại học Bách Khoa Đà Nẵng. Với đồ án tốt nghiệp này, là cơ hội để chúng em có thêm kinh nghiệm bên cạnh những kiến thức đã được học tập. Mặc dù chưa phải sát với thực tế, nhưng nó đã giúp chúng em phần nào có thêm kinh nghiệm và củng cố lại kiến thức đã học để có thêm hành trang trên bước đường đời và nghề nghiệp của mình

Chúng em xin bày tỏ long biết ơn chân thành và sâu sắc đến Ban giám hiệu Nhà trường, Khoa Điện, cùng các thầy cô giảng viên đã tận tình giảng dạy, truyền thụ cho chúng em những kiến thức quý giá và tạo điều kiện thuận lợi cho em trong suốt quá trình học tập tại trường.

Đặt biệt, chúng em xin gửi lời cảm ơn trận trọng đến cô Nguyễn Thị Thanh Quỳnh, người đã tận tình hướng dẫn, định hướng và hỗ trợ chúng em xuyên suốt quá trình thực hiện đồ án tốt nghiệp. Những góp ý quý báu từ cô là kim chỉ nam giúp chúng em hoàn thiện đề tài một cách khoa học và gần với thực tế nhất. Kính chúc cô thật nhiều sức khoẻ và ngày càng thành công hơn nữa trên con đường sự nghiệp của mình.

Chúng em cũng xin gửi lời cảm ơn đến các bạn lớp 20TĐHCLC1 đã chia sẻ, trao đổi kiến thức và kinh nghiệm của bản thân trong thời gian thực hiện đồ án. Cảm ơn đến các anh chị, bạn bè và gia đình – những người luôn động viên, hỗ trợ chúng em về tinh thần lẫn kiến thức trong suốt quá trình thực hiện đề tài.

Dù đã nỗ lực hết sức, nhưng do thời gian và kiến thức còn hạn chế, điều kiện kinh tế không cho phép, đồ án không thể tránh khỏi những thiếu sót. Chúng em kính mong nhận được sự góp ý từ quý thầy cô để em có thể học hỏi và và hoàn thiện bản thân cũng như đề tài này.

Xin chân thành cảm ơn !

LỜI CAM ĐOAN LIÊM CHÍNH HỌC THUẬT

Chúng em xin cam đoan đồ án này là công trình học tập và nghiên cứu của chúng em. Các kết quả số liệu, hình ảnh, nội dung trình bày trong báo cáo đều trung thực và được thực hiện dưới sự hướng dẫn của giảng viên hướng dẫn. Những thông tin, tài liệu được tham khảo từ khác nguồn khác đều đã được trích dẫn rõ rang trong tài liệu. Chúng em xin hoàn toàn chịu trách nhiệm về tính trung thực và liêm chính học thuật của đồ án này. Nếu có hành vi sao chép, gian lận hay vi phạm về liêm chính học thuật, chúng em xin hoàn toàn chịu trách nhiệm trước Hội đồng và quy định của nhà trường.

Sinh viên thực hiện 1	Sinh viên thực hiện 2
{Chữ ký, họ và tên sinh viên}	{Chữ ký, họ và tên sinh viên}

MỤC LỤC

DANH MỤC CÁC BẢNG, HÌNH ẢNH

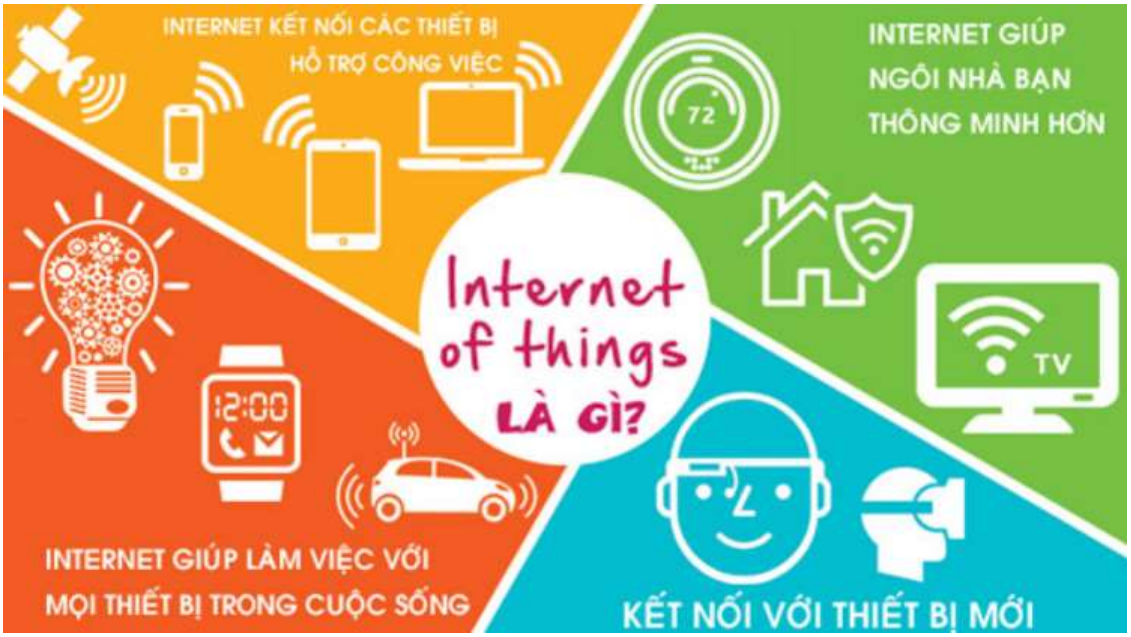
DANH MỤC CÁC KÝ HIỆU, CHỮ VIẾT TẮT

STT	Viết tắt	Tiếng Anh	Tiếng Việt
1	IoT	Internet of Things	Internet vạn vật
2	AI	Artificial Intelligence	Trí tuệ nhân tạo
3	ML	Machine Learning	Học máy
4	API	Application Programming Interface	Giao diện lập trình ứng dụng
5	HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
6	HTTPS	Hypertext Transfer Protocol Secure	Giao thức truyền tải siêu văn bản bảo mật
7	MQTT	Message Queuing Telemetry Transport	Giao thức truyền tải thông điệp hàng đợi đo lường từ xa
8	JSON	JavaScript Object Notation	Định dạng dữ liệu dựa trên JavaScript
9	IDE	Integrated Development Environment	Môi trường phát triển tích hợp
10	ESP32	Espressif Systems Platform 32-bit	Nền tảng vi điều khiển 32-bit của Espressif Systems
11	Home Assistant	Home Assistant	Nền tảng nhà thông minh Home Assistant (được nhắc đến như một nền tảng)
12	BLE	Bluetooth Low Energy	Bluetooth năng lượng thấp
13	CPU	Central Processing Unit	Bộ xử lý trung tâm
14	RAM	Random Access Memory	Bộ nhớ truy cập ngẫu nhiên
15	ROM	Read-Only Memory	Bộ nhớ chỉ đọc
16	Wi-Fi	Wireless Fidelity	Kết nối không dây
17	LAN	Local Area Network	Mạng cục bộ
18	WAN	Wide Area Network	Mạng diện rộng
19	IP	Internet Protocol	Giao thức Internet
20	URL	Uniform Resource Locator	Định vị tài nguyên thống nhất
21	USB	Universal Serial Bus	Chuẩn giao tiếp nối tiếp vạn năng
22	HDMI	High-Definition Multimedia Interface	Giao diện đa phương tiện độ nét cao
23	VGA	Video Graphics Array	Mảng đồ họa video
24	LED	Light-Emitting Diode	Đi-ốt phát quang
25	LCD	Liquid Crystal Display	Màn hình tinh thể lỏng
26	OLED	Organic Light-Emitting Diode	Đi-ốt phát quang hữu cơ

MỞ ĐẦU

Lý do chọn đề tài

Trong thời kì cách mạng công nghiệp 4.0, các công nghệ như Internet of Things (IoT) và Trí tuệ nhân tạo (AI) đang ngày càng phát triển và ứng dụng rộng rãi trong đời sống hằng ngày. Một trong những lĩnh vực nổi bật của xu thế này là nhà thông minh (Smart Home) – nơi các thiết bị và hệ thống được kết nối và điều khiển tự động nhằm phục vụ nhu cầu tiện nghi, an ninh, tiết kiệm năng lượng và nâng cao chất lượng cuộc sống.



Hình 1: Internet of Thing

Tuy nhiên, một vấn đề thực tế đang phát sinh và ngày càng rõ rệt trong các mô hình hiện nay là sự phân mảng giữa các hệ sinh thái thiết bị. Trong một ngôi nhà, có thể cùng lúc tồn tại nhiều thiết bị thông minh từ các hãng khác nhau như Xiaomi, Philips, Imou, Tuya, v.v…

Mỗi hệ sinh thái thường có ứng dụng điều khiển riêng biệt, giao thức khác nhau, và thiếu khả năng tương tác chéo. Một số thiết bị chỉ hỗ trợ các nền tảng kết nối riêng biệt, chẳng hạn như hai nền tảng lớn là Google Home và Apple HomeKit hoạt động độc lập, dẫn đến các hạn chế như:

- Khả năng tương tác kém: Mỗi hệ sinh thái sử dụng giao thức riêng, khiến việc tích hợp nhiều thiết bị từ các nhà sản xuất khác nhau trở nên phức tạp. Ví dụ, đèn Philips Hue có thể hoạt động tốt với Google Home nhưng lại hạn chế trên Apple HomeKit.

- Chi phí cao: Để đảm bảo tính đồng bộ, người dung phải đầu tư vào một hệ sinh thái duy nhất, dẫn đến tốn kém và thiếu linh hoạt.

Bên cạnh đó, mỗi thiết bị của một hãng sẽ có ưu điểm và nhược điểm riêng, nhu cầu cá nhân về sử dụng chức năng của mỗi thiết bị của hãng cũng sẽ khác nhau. Mỗi người sẽ có xu hướng lựa chọn các thiết bị từ nhiều hãng để tối ưu chi phí, và phù hợp với căn nhà của họ. Vì vậy, việc nghiên cứu và xây dựng một hệ thống điều khiển nhà thông minh đa hệ sinh thái, có khả năng tích hợp nhiều thiết bị thuộc các nền tảng khác nhau vào một giao diện điều khiển thống nhất, đồng thời áp dụng AI để tang cường tính thông minh và tự động hoá – là một hướng đi vừa thực tiễn, vừa cấp thiết.

Đề tài này không chỉ đáp ứng xu hướng công nghệ hiện đại mà còn hướng tới khả năng mở rộng linh hoạt, chi phí hợp lý, đồng thời tạo nền tảng cho việc phát triển các giải pháp nhà thông minh thực sự thân thiện với người dùng phổ thông tại Việt Nam.

Mục tiêu và yêu cầu của đề tài

Mục tiêu của đề tài là xây dựng một hệ thống quản lý thông minh đa nền tảng, có tính mở rộng, ứng dụng Machine Learning và tối ưu trải nghiệm người dùng với các yêu cầu cụ thể như :

- Tích hợp khả năng nhận diện khuôn mặt và cử chỉ tay để điều khiển thiết bị trong nhà.
- Sử dụng cảm biến để giám sát môi trường và cảnh báo khi có bất thường (ví dụ: phát hiện gas, nhiệt độ cao)
- Thiết kế giao diện web trực quan để người dùng theo dõi trạng thái thiết bị, điều khiển từ xa, và xem lịch sử dữ liệu cảm biến.
- Đảm bảo khả năng hoạt động thời gian thực, độ chính xác cao và tính bảo mật trong các hoạt động xác thực.
- Hệ thông cần có khả năng mở rộng và tích hợp thêm các thiết bị, chức năng mới trong tương lai.
- Đặt biệt hỗ trợ tích hợp nhiều thiết bị từ các hệ sinh thái khác nhau vào một nền tảng điều khiển thống nhất, có thể hoạt động độc lập hoặc phối hợp cùng nhau.

Các tính năng bổ sung hướng đến:

- Tự học hành vi người dùng trong tương lai.
- Hỗ trợ điều khiển bằng giọng nói
- Tối ưu hoá năng lượng tiêu thụ thông qua phan tích dữ liệu.

Phạm vi và phương pháp thực hiện

Đề tài tập trung phát triển một hệ thống nguyên mẫu (prototype) tích hợp cả phần cứng và phần mềm để chứng minh tính khả thi của mô hình nhà thông minh tích hợp model Machine Learning.

Phạm vi thực hiện:

Phần cứng: sử dụng chip vi xử lý ESP32 làm bộ điều khiển kết nối với các cảm biến khí gas, cảm biến siêu âm, servo điều khiển cửa, v.v.

Phần mềm: sử dụng nền tảng Flask làm backend, lưu trữ dữ liệu với MySQL, tích hợp Machine Learning nhận diện khuôn mặt bằng MTCNN + FaceNet, và nhận diện cử chỉ tay bằng MediaPipe.

Giao tiếp: sử dụng giao thức HTTP và giao thức WebSocket để trao đổi dữ liệu giữa chip vi xử lý ESP32, Flask và giao diện người dùng.

Tích hợp thiết bị đa hệ sinh thái: kết nối với Home Assistant để có thể điều khiển các thiết bị từ nhiều hãng khác nhau thông qua MQTT, RESTful API hoặc các tích hợp mở rộng.

Phương pháp thực hiện:

Phân tích và thiết kế hệ thống: xác định kiến trúc, luồng dữ liệu, phân chia chức năng giữa các thành phần.

Lập trình và tích hợp: xây dựng phần mềm backend, các thuật toán, giao diện web và tích hợp với phần cứng.

Kiểm thử và đánh giá: kiểm tra tính chính xác của nhận diện, khả năng điều khiển thiết bị và độ trễ thời gian thực.

Viết báo cáo và rút kinh nghiệm: tổng hợp kết quả, đánh giá ưu nhược điểm và đề xuất hướng phát triển tiếp theo.

Cấu trúc của báo cáo đồ án:

Với đề tài này, đồ án được xây dựng và trình bày chi tiết thông qua 5 chương bao gồm:

- Chương I: Trình bày cơ sở lý thuyết và tổng quan các hệ thống nhà thông minh hiện nay.
- Chương II: Mô tả chi tiết kiến trúc hệ thống, thiết kế phần cứng, phần mềm và các giao tiếp chính.
- Chương III: Đi sâu vào hệ thống nhận diện khuôn mặt bằng Machine Learning, cách triển khai mô hình và xử lý dữ liệu.

- Chương IV: Trình bày giải pháp điều khiển thiết bị bằng cử chỉ tay theo thời gian thực.
- Chương V: Đưa ra đánh giá kết quả thực nghiệm, những khó khan gặp phải và định hướng phát triển trong tương lai.

CHƯƠNG I: TỔNG QUAN VỀ NHÀ THÔNG MINH

1.1 Khái niệm và đặc điểm nhà thông minh

1.1.1 Khái niệm nhà thông minh

Nhà thông minh (Home automation, domotics, smart home) là tên gọi dùng để gọi tên các ngôi nhà, căn hộ, công trình xây dựng được trang bị, được cài đặt sử dụng các thiết bị thông minh nhằm mục đích giúp cho ngôi nhà trở nên thông minh. Có thể đáp ứng theo các ngữ cảnh thông minh một cách có chủ định theo thiết lập người dùng, có thể hoạt động một cách tự động hoặc bán tự động, và có thể thay thế con người thực hiện một số thao tác quản lý, điều khiển nhất định. Về mặt bản chất, nhà thông minh là sự kết nối có hệ thống của các thiết bị điện thông minh. Giúp ngôi nhà trở nên thông minh hơn, có thể đáp ứng được các chức năng tự động hoặc bán tự động theo ý của người dùng. Hệ thống điện tử được kết nối quan mạng không dây hoặc có dây, cho phép giao tiếp với người dùng thông qua bản điện tử đặt trong nhà, ứng dụng trên điện thoại di động, máy tính bảng hoặc một giao diện web, thậm chí có thể ra lệnh điều khiển các thiết bị bằng giọng nói.

Giải pháp nhà thông minh sẽ biến những món đồ điện tử bình thường trong ngôi nhà trở nên thông minh và gần gũi với người dùng hơn, chúng được kiểm soát thông qua các thiết bị truyền thông như điều khiển từ xa, điện thoại di động. Hệ thống nhà thông minh không chỉ đơn thuần là bật/ tắt thiết bị, mà còn có khả năng tự động hoá theo ngữ cảnh (automation), học hỏi hành vi người dùng, đưa ra các kịch bản thông minh như: bật đèn khi phát hiện người, tự động tưới cập vào sáng sớm, hoặc gửi cảnh báo khi phát hiện rò rỉ gas.

Hình 1.1 Smart Home

Gần đây, công nghệ nhà thông minh đang ngày càng hiện đại, đang cho phép kết nối kiểm soát hoàn toàn nhà thông minh từ bất kì đâu để ý tưởng về "Internet of Things" trở thành hiện thực. Nhà thông minh sẽ đem lại sự tiện nghi và thoải mái hơn đến với con người, vừa có thể tận hưởng nằm trên sofa để coi tivi vừa có thể kiểm soát được hệ thống các thiết bị trong nhà chỉ với một chiếc điện thoại hay máy tính bảng. Trong xu hướng tương lai, việc ứng dụng AI vào trong ngôi nhà sẽ được vận hành một cách trơn tru hơn, AI sẽ theo dõi thói quen người dùng và đưa ra các thao tác hợp lý và tạo cho chúng ta có cảm giác như một quản gia hỗ trợ thực thụ.

Hình 1.2 Smart Home Bkav

1.1.2 Đặc điểm và lợi ích một nhà thông minh

a) Tự động hoá thiết bị trong gia đình

Một trong những đặc điểm nổi bật và cốt lõi của nhà thông minh là khả năng tự động hóa các thiết bị điện trong gia đình. Người dùng có thể lập trình để các thiết bị hoạt động theo lịch trình, hoặc phản hồi theo các điều kiện thực tế như cảm biến chuyển động, cảm biến ánh sáng, nhiệt độ môi trường...

- Ví dụ: Hệ thống có thể tự động tắt đèn khi không có ai trong phòng, hoặc bật máy lạnh khi nhiệt độ vượt quá ngưỡng cài đặt.
- Các kịch bản tự động hoá giúp tiết kiệm thời gian, hạn chế thao tác lặp lại, nâng cao trải nghiệm sống tiện nghi.

b) Điều khiển từ xa qua Internet

Hệ thống nhà thông minh cho phép người dùng giám sát và điều khiển các thiết bị từ xa thông qua kết nối Internet. Bằng cách sử dụng smartphone, máy tính bảng hoặc máy tính cá nhân, người dùng có thể:

- Bật/tắt thiết bị ở bất cứ đâu, bất cứ lúc nào.
- Kiểm tra trạng thái hoạt động của hệ thống
- Nhận thông báo thời gian thực khi có sử cố hoặc bất thường xảy ra.

c) An ninh và giám sát thông minh

An toàn là một trong những yếu tố ưu tiên hang đầu trong nhà thông minh. Nhờ tích hợp các cảm biến và thiết bị giám sát, hệ thống có thể đảm bảo an ninh hiệu quả hơn so với giải pháp truyền thống:

- Camera an ninh IP có thể ghi hình liên tục, truyền tải dữ liệu về điện thoại hoặc lưu trữ đám mây.
- Cảm biến cửa, cảm biến chuyển động phát hiện khi có người xâm nhập trái phép, lập tức gửi ảnh báo cho chủ nhà.
- Hệ thống cảnh báo cháy, rò rỉ khí gas, nước hoạt động liên tục, giúp phòng tránh các tai nạn nghiêm trọng.

d) Giám sát và điều chỉnh môi trường sống

Nhà thông minh không chỉ giúp người dùng điều khiển thiết bị, mà còn có khả năng giám sát các chỉ số môi trường trong nhà, bao gồm:

- Nhiệt độ: Hệ thống điều hoà có thể được điều chỉnh tự động dựa trên nhiệt độ hiện tại.
- Độ ẩm: Cảm biến độ ẩm giúp kiểm soát không khí, hỗ trợ hệ thống máy hút ẩm hoặc tạo ẩm hoạt động hiệu quả.
- Chất lượng không khí: Một số hệ thống cao cấp tích hợp cảm biến đo bụi mịn (PM2.5), CO_2 để bật/tắt máy lọc không khí.

e) Cá nhân hoá và học thói quen người dùng

Nhà thông minh ngày nay không chỉ hoạt động theo lập trình cố định, mà còn có khả năng phân tích hành vi và học hỏi thói quen người dùng để đưa ra các đề xuất hoặc hành động phù hợp”

- Ví dụ: Nếu người dùng thường bật đèn phòng khách lúc 7h sáng, hệ thống có thể tự động thực hiện hành động này mỗi ngày, ngay cả khi chưa ra lệnh.
- Các kịch bản “Chào buổi sáng”, “đi ngủ”, “ra khỏi nhà”, có thể thiết lập theo thời điểm, vị trí và ngữ cảnh cụ thể.

Tính năng cá nhân hoá giúp hệ thống nhà thông minh trở nên linh hoạt và gần gũi hơn với người dùng, tạo cảm giác như một người quản gia chăm lo cho từng cá nhân trong gia đình.

f) Tính mở rộng và tương thích cao

Ngôi nhà thông minh được đánh gia cao và khác hẳn những ngôi nhà bình thường là do nó được trang bị một hệ thống mạng điều khiển và toàn bộ các thay đổi và điều khiển tự động trong ngôi nhà được xử lý đồng nhất thông qua hệ thống mạng và xử lý trung tâm. Hệ thống nhà thông minh hiện đại thường được xây dựng theo kiến trúc

mở, cho phép dễ dàng kết nối và mở rộng với nhiều loại thiết bị từ các hãng khác nhau:

- Hỗ trợ các chuẩn giao tiếp phổ biến như Zigbee, Z-ware, Wifi, Bluetook, MQTT..
- Dễ dàng tích hợp them thiết bị mới như cảm biến, công tắc thông minh, rèm cửa, robot hút bụi, v.v.
- Khả năng đồng bộ hoá với nền tảng điều khiển trung tâm (hub) hoặc nền tảng mã nguồn mở như Home Assistant, v.v.

Nhờ đó, người dùng có thể bắt đầu từ một hệ thống nhỏ và mở rộng dần theo nhu cầu sử dụng thực tế.

1.2 Quản lý đa ứng dụng

Một điểm nổi bật trong hệ thống của nhóm là khả năng thu thập và đồng bộ nhiều loại ứng dụng – như điều khiển thiết bị, giám sát cảm biến, cảnh báo an ninh, và thống kê tiêu thụ điện – vào cùng một nền tảng giao diện duy nhất do nhóm thiết kế.,

1.2.1 Nền tảng quản lý nhà thông minh Home Assistant

Một trong những thành phần trung tâm giúp hệ thống của nhóm hoạt động linh hoạt là nền tảng Home Assistant. Đây là một nền tảng quản lý nhà thông minh được lập trình bằng ngôn ngữ Python. Phần mềm này là một mã nguồn mở, hoạt động thông qua hầu hết các nền tảng hệ điều hành và quản lý nhà thông minh, giúp người dùng điều khiển qua giao diện web hoặc ứng dụng điện thoại. Nổi bật với khả năng hỗ trợ hàng trăm giao thức và thiết bị khác nhau (Tuya, Xiaomi, Philips Hue, MQTT, Zigbee, v.v.). Home Assistant cho phép nhóm tích hợp các thiết bị từ nhiều hệ sinh thái vào một hệ điều phối chung, thông qua đó có thể đồng bộ trạng thái thiết bị, điều khiển và nhận dữ liệu một cách thống nhất.

Hình 1.3 Giải pháp Home Assistant

Home Assistant đóng vai trò như Hub tổng, là một nền tảng tự động hoá và quản lý nhà thông minh. Phần mềm này cung cấp một loạt các tính năng để kiểm soát các thiết bị và tác vụ thông qua giao diện web và ứng dụng di động. Home Assistant được coi là cầu nối cho các thiết bị nhà thông minh sử dụng các công nghệ IoT khác nhau.

Home Assistant chạy cục bộ trong mạng, không phụ thuộc vào đám mây. Điều này đảm bảo dữ liệu của bạn lun nằm trong tầm kiểm soát, giảm thiểu rủi ro về bảo mật và quyền riêng tư. Với cộng đồng lớn và năng động, Home Assistant luôn được cập nhật và phát triển nhanh chóng. Người dùng có thể dễ dàng tìm kiếm sự hỗ trợ và chia sẽ kinh nghiệm.

Cơ chế Discovery và Handshake của Home Assistant trong mạng LAN

Home Assistant sử dụng cơ chế tự động phát hiện thiết bị (discovery) để nhận diện và kết nối với các thiết bị thông minh trong cùng mạng LAN. Quá trình này được thực hiện thông qua các giao thức phổ biến như:

- mDNS (Multicast DNS) và Zeroconf: Phát hiện các thiết bị hỗ trợ Bonjour hoặc Avahi (ví dụ: ESPHome, Chromecast).
- SSDP (Simple Service Discovery Protocol) và UPnP: Phát hiện các thiết bị như camera IP, TV thông minh, loa...
- Bluetooth/BLE: Phát hiện các thiết bị gần như cảm biến nhiệt độ, đèn BLE.
- DHCP Discovery: Một số thiết bị gửi metadata qua DHCP để Home Assistant nhận diện.
- MQTT Discovery: các thiết bị như Arduino, NodeMCU, hoặc Tasmota.

Quá trình handshake giữa Home Assistant và thiết bị bao gồm các bước:

- Bước 1: Khám phá thiết bị: Home Assistant gửi các gói tin broadcast trong mạng để tìm kiếm thiết bị tương thích.
- Bước 2: Xác thực: Một số thiết bị yêu cầu mã PIN, token hoặc tài khoản cloud để xác thực quyền điều khiển.
- Bước 3: Tạo thực thể (entities): Sau khi xác thực thành công, Home Assistant sẽ tạo các thực thể tương ứng (đèn, cảm biến, công tắc...) để người dùng có thể điều khiển và giám sát.
- Bước 4: Cấu hình tự động: Với các thiết bị hỗ trợ tốt như ESPHome, MQTT, Tuya..., Home Assistant có thể tự động cấu hình giao diện điều khiển và cập nhật trạng thái thiết bị theo thời gian thực.
- Bước 5: Giao tiếp định kỳ: Sau khi kết nối, Home Assistant duy trì liên lạc với thiết bị thông qua các giao thức như MQTT, RESTful API hoặc WebSocket để đảm bảo dữ liệu luôn được đồng bộ.

Tuy vậy, Home Assistant cũng đặt ra nhiều thách thức đối với người dùng. So với các giải pháp thương mại, Home Assistant có thể khó tiếp cận hơn đối với người dùng không có kiến thức kỹ thuật. Việc cài đặt và cấu hình ban đầu có thể đòi hỏi một chút học hỏi. Để chạy Home Assistant, bạn cần có một thiết bị chuyên dụng như Raspberry Pi hoặc một máy chủ luôn bật. Điều này có thể tạo ra chi phí ban đầu và tiêu thụ điện năng liên tục.

1.2.2 Thu thập dữ liệu từ nhiều ứng dụng và thiết bị

Một trong những điểm nổi bật của hệ thống là khả năng tích hợp và đồng bộ hoá thông tin từ nhiều nguồn dữ liệu khác nhau trên cùng một nền tảng điều khiển. Thay vì người dùng phải truy cập từng ứng dụng riêng lẻ cho từng loại thiết bị, toàn bộ dữ liệu và chức năng điều khiển được gom về một nơi duy nhất.

Quá trình thu thập dữ liệu được thực hiện thông qua nhiều phương thức:

- Dữ liệu từ cảm biến (nhiệt độ, độ ẩm, khí gas) được chip vi xử lý ESP32 (vi điều khiển lõi tích hợp Wi-Fi và Bluetooth) gửi lên server thông qua HTTP hoặc WebSocket.
- Dữ liệu an ninh: hình ảnh và thông tin khuôn mặt từ hệ thống Machine LEarning nhận diện, cảnh báo gửi về hệ thống Flask.
- Các thiết bị thuộc hệ sinh thái khác (như Tuya, Yeelight...) được kết nối thông qua nền tảng Home Assistant, sau đó được lấy dữ liệu định kỳ bằng API.
- Dữ liệu tiêu thụ năng lượng: Thu thập kết quả từ EVN để kết nối với Home Assistant, từ đó có thể theo dõi mức tiêu thụ điện năng theo ngày/tháng.

- Mọi dữ liệu được lưu trữ trong cơ sở dữ liệu MySQL với hai bảng chính: sensor_history (lịch sử môi trường, năng lượng) và alert_history (cảnh báo người lạ và khí gas).

1.2.3 Thiết kế giao diện hiển thị đa chức năng

Thiết kế hiển thị được nhóm thực hiện thông qua dashboard web tương tác, gồm nhiều phân vùng chức năng rõ ràng:

- Trang tổng quan (Overview): hiển thụ trực quan các chỉ số môi trường, tình trạng kết nối thiết bị và hình ảnh từ camera giám sát.
- Giám sát năng lượng (Energy Monitor): Thể hiện mức tiêu thụ điện theo ngày và tháng, kèm theo kiểu đồ cột và chi phí ước tính.
- Lịch sử cảnh báo (Alert History): thống kê các cảnh báo phát hiện người lạ, khí gas, biểu đồ điện năng.
- Điều khiển thiết bị (Device Control): cung cấp nút điều khiển công tắc, đèn, rèm cửa,.. ngay trên giao diện mà công cần chuyển sang ứng dụng khác.

Tất cả đều được đồng bộ thời gian thực bằng giao thức WebSocket và truy vấn API định kỳ. Người dùng không cần chuyển ứng dụng mà vẫn có thể theo dõi và điều khiển toàn bộ hệ thống từ một nền tảng duy nhất – vừa dễ sử dụng, vừa có khả năng mở rộng cho nhiều ứng dụng trong tương lai. Điều này không chỉ nâng cao trải nghiệm sử dụng mà còn giúp dễ dàng mở rộng cho các ứng dụng khác trong tương lai như điều khiển giọng nói, nhận diện hành vi tiêu dùng, phân tích năng lượng chuyên sâu.

1.3 Các giải pháp nhà thông minh hiện nay

Hiện nay, trên thị trường tồn tại nhiều hệ thống nhà thông minh khác nhau, từ các nền tảng quốc tế như Google Home, Apple HomeKit, Home Assistant, cho đến các hệ thống nội địa như FPT Smart Home và Lumi. Mỗi hệ thống có định hướng phát triển riêng, với ưu nhược điểm phù hợp với từng đối tượng người dùng và môi trường triển khai cụ thể.

1.3.1 Nền tảng quốc tế

- Google Home là nền tảng nhà thông minh phổ biến toàn cầu, được phát triển bởi Google. Hệ thống sử dụng Google Assistant làm trung tâm điều khiển bằng giọng nói, tích hợp sâu với các thiết bị Android và hệ sinh thái Google.
  - Ưu điểm: hỗ trợ đa thiết bị, dễ sử dụng, điều khiển bằng giọng nói tiếng Việt, tương thích rộng.
  - Nhược điểm: tùy biến hạn chế, phụ thuộc Internet, khó tích hợp thiết bị nội địa không chính thức.

- Apple HomeKit là nền tảng quản lý thiết bị nhà thông minh dành riêng cho hệ sinh thái Apple như iPhone, iPad, Siri. Hệ thống chú trọng tính riêng tư và bảo mật.
  - Ưu điểm: bảo mật cao, giao diện mượt mà, hoạt động ổn định trong hệ Apple.
  - Nhược điểm: ít thiết bị hỗ trợ, chi phí cao, hạn chế tính tùy chỉnh.
- Home Assistant (HA) là nền tảng mã nguồn mở mạnh mẽ, chuyên dụng cho những người dùng kỹ thuật cao muốn xây dựng hệ thống tùy biến và kiểm soát toàn bộ thiết bị tại nhà.
  - Ưu điểm: hỗ trợ hàng nghìn thiết bị và giao thức, hoạt động cục bộ (không cần Internet), tùy biến cực cao, hỗ trợ AI và tự động hóa phức tạp.
  - Nhược điểm: khó tiếp cận với người dùng phổ thông, cần cài đặt thủ công và cấu hình phức tạp.

Hình 1.4 Mô hình Smarthome của công ty Compro Technology

1.3.2 Nền tảng nội địa

- FPT Smart Home là hệ sinh thái nhà thông minh nội địa do Tập đoàn FPT phát triển, với ưu thế là tích hợp thiết bị đồng bộ, hỗ trợ tiếng Việt và được triển khai lắp đặt chuyên nghiệp.
  - Ưu điểm: giao diện tiếng Việt, đội ngũ hỗ trợ trong nước, thiết bị dễ sử dụng.
  - Nhược điểm: tính mở rộng hạn chế, khó tích hợp thiết bị từ bên thứ ba, tùy biến thấp.
- Lumi là một trong những thương hiệu nhà thông minh Việt Nam tiên phong và có thị phần lớn trong nước. Hệ thống Lumi chú trọng thiết kế đẹp, giao diện tiếng Việt và trải nghiệm người dùng.

- Ưu điểm: thiết kế phù hợp thẩm mỹ người Việt, hỗ trợ giọng nói tiếng Việt, tương thích Google Assistant/Alexa.
- Nhược điểm: cần đội ngũ kỹ thuật lắp đặt, khó tùy biến cao, hạn chế mở rộng với thiết bị bên ngoài hệ sinh thái Lumi.

1.3.3 So sánh tổng quan các hệ thống

Bảng 1.1 So sánh các giải pháp nhà thông minh

Tiêu chí	Google Home	Apple HomeKit	Home Assistant	FPT Smart Home	Lumi
Hỗ trợ thiết bị	Rộng rãi	Trung bình	Rất rộng (mở)	Trung bình	Rộng (nội bộ)
Điều khiển giọng nói	Có (Google Assitant)	Có (Siri)	Có thể tích hợp	Có (Tiếng Việt	Có (Tiếng Việt)
Tuỳ biến và mở rộng	Trung Bình	Thấp	Rất cao	Thấp	Trung bình
Dễ sử dụng	Cao	Cao	Trung bình – thấp	Rất cao	Rất cao
Giao diện tiếng Việt	Có	Không	Tuỳ biến	Có	Có
Cộng đồng hỗ trợ	Toàn cầu	Giới Hạn	Mạnh (open source)	Chủ yếu nội địa	Nội địa

1.3.4 Cơ hội và thách thức khi phát triển hệ thống riêng

Cơ hội:

- Có thể tích hợp thiết bị từ nhiều hệ sinh thái khác nhau vào một nền tảng duy nhất.
- Chủ động xây dựng các tính năng theo nhu cầu cụ thể như AI, tự động hóa, điều khiển bằng cử chỉ.
- Tối ưu chi phí phần cứng, không phụ thuộc vào nền tảng nước ngoài.

Thách thức:

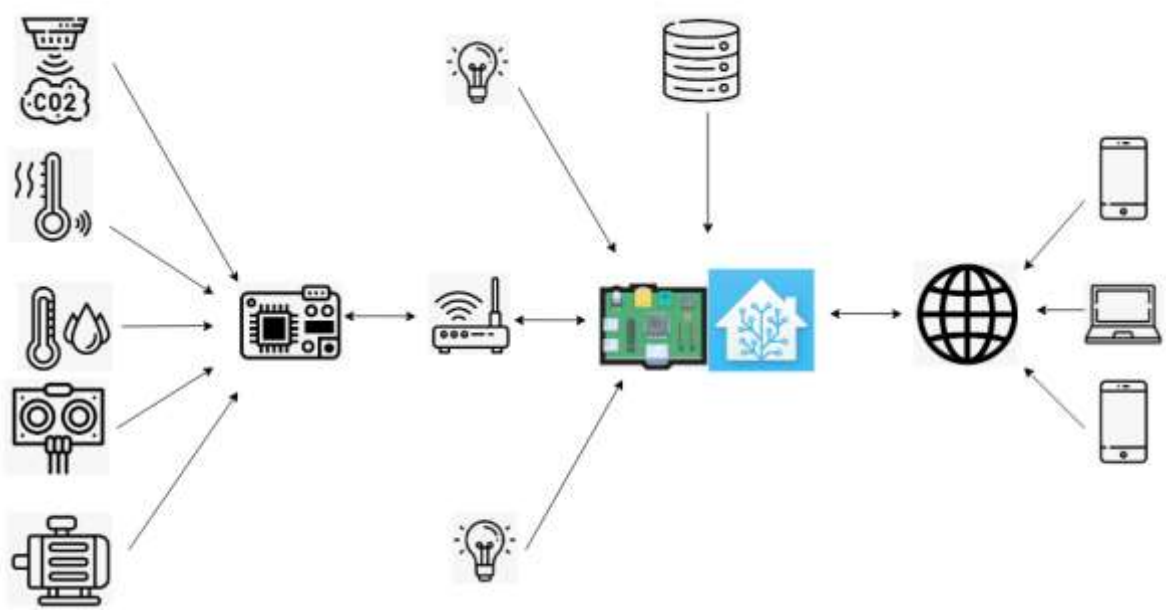
- Cần kiến thức kỹ thuật để cấu hình và quản lý hệ thống.
- Tính tương thích thiết bị có thể gặp hạn chế nếu không đồng bộ chuẩn giao tiếp.
- Giao diện và trải nghiệm người dùng cần đầu tư để canh tranh với các giải pháp thương mai.

CHƯƠNG II: THIẾT KẾ HỆ THỐNG QUẢN LÝ HỆ SINH THÁI NHÀ THÔNG MINH

2.1 Kiến trúc tổng thể hệ thống

2.1.1 Mô hình client-server

Hệ thống quản lý nhà thông minh được xây dựng theo mô hình client-server, bao gồm các thành phần chính ở các tầng các nhau. Có thể phân chia thành tầng thiết bị, tầng sử lý trung tâm và tầng giao diện. Các thiết bị phần cứng như chip vi xử lý ESP32 đóng vai trò phía client, chịu trách nhiệm thu thập dữ liệu từ các cảm biến và thực thi các lệnh điều khiển. Máy chủ trung tâm là ứng dụng Flask chịu trách nhiệm xử lý logic điều khiển và điều phối giao tiếp giữa các thành phần. Giao diện điều khiển dashboard trên nền web hoạt động như một client khác, cung cấp khả năng hiển thị dữ liệu thu thập được và cho phép người dùng tương tác với hệ thống. Bên cạnh đó, hệ thống tích hợp các thành phần model Machine Learning như nhận diện khuôn mặt và cử chỉ để nâng cao khả năng tương tác thông minh. Dữ liệu thu thập được lưu trữ trong cơ sở dữ liệu MySQL, trong khi việc trao đổi thông tin giữa các thành phần diễn ra theo thời gian thực qua giao thức WebSocket hoặc giao thức HTTP.



Hình 2.1 Cấu trúc tổng thể hệ thống

Chip vi xử lý ESP32 đảm nhận thu thập dữ liệu cảm biến và điều khiển một số thiết bị chấp hành, đồng thời truyền các giá trị này đến server thông qua kết nối Wi-Fi. Máy chủ Flask đóng vai trò trung tâm xử lý trong hệ thống. Khi nhận dữ liệu từ chip vi xử lý ESP32 hoặc từ Home Assistant, Flask sẽ phân tích và xử lý các giá trị này (ví dụ chạy thuật toán nhận diện khuôn mặt hoặc cử chỉ). Kết quả xử lý được

ghi vào cơ sở dữ liệu MySQL để lưu trữ lịch sử, đồng thời xử lý logic điều khiển thiết bị.

Dashboard web là giao diện đồ hoạ (chạy trên trình duyệt) hoạt động như một client khác. Giao diện này được phát triển trên nền tảng web và kết nối với máy chủ Flask qua giao thức Websocket/HTTP, cho phép nhận dữ liệu cập nhật theo thời gian thực. Nó sẽ gửi các yêu cầu đến server để lấy dữ liệu cảm biến và điều khiển hệ thống, đồng thời hiển thị trang thái thiết bị cho người dùng.

Hình 2.2 Giao diện web

2.1.2 Giới thiệu về Flask và Flask Server

Trong hệ thống nhà thông minh đa hệ sinh thái, việc xây dựng một máy chủ trung tâm (server) để tiếp nhận xử lý và điều khiển thiết bị là yêu tố then chốt. Để đáp ứng yêu cầu này, nhóm đã sử dụng Flask – một framework web mã mã nguồn mở, nhẹ và linh hoạt được viết bằng ngôn ngữ lập trình Python.

Vậy Flask là gì ?

Flask là một khuôn khổ web vi mô được viết bằng Python. Nó được phân loại là một microframework vì nó không yêu cầu các công cụ hoặc thư viện cụ thể. Được thiết kế để giúp lập trình viên xây dựng nhanh các ứng dụng web và hệ thống API. Flask không ràng buộc cấu trúc cố định như framework lớn (như Django), nhờ đó người dùng có thể tuỳ chỉnh linh hoạt các thành phần trong hệ thống theo đúng nhu cầu của mình.

Với mã cấu trúc đơn giản, dễ hiểu và tài liệu hướng dẫn phong phú, Flask là một lựa chọn lý tưởng cho đồ án thích hợp phần cứng và phần mềm như hệ thống IoT, điều khiển thiết bị từ xa, xử lý dữ liệu cảm biến, v.v.

Flask Server trong hệ thống:

Flask Server trong đồ án này là chương trình chạy trên máy tính hoặc thiết bị trung tâm, đảm nhiệm vai trò:

- Làm trung gian giao tiếp giữa các thiết bị phần cứng (chip vi xử lý ESP32, cảm biến, camera) và giao diện web người dùng (Dashboard)
- Xử lý dữ liệu cảm biến từ chip vi xử lý ESP32 gửi lên và lưu vào cơ sở dữ liệu MySQL.
- Nhận ảnh từ camera, phân tích hình ảnh bằng các model Machine Learning như MTCNN (phát hiện khuôn mặt) và FaceNet (nhận dạng khuôn mặt) hoặc MediaPice (nhận diện cử chỉ tay)
- Phản hồi lệnh điều khiển từ người dùng thông qua dashboard web, rồi gửi lệnh tương ứng đến HomeAssistant để bật tắt thiết bị hoặc gửi lệnh về chip xử lý ESP32 để mở cửa, kích hoạt cảnh báo...

Toàn bộ quá trình giao tiếp giữa Flask Server với các thành phần khác diễn ra qua các giao thức mạng:

- HTTP (RESTful API): để gửi và nhận dữ liệu dạng yêu cầu đơn lẻ
- WebSocket: để truyền dữ liệu liên tục theo thời gian thực giữa Flask và dashboard.

2.1.3 Tích hợp Machine Learning, cơ sở dữ liệu, giao tiếp thời gian thực

Hệ thống sử dụng Machine Learning để thực hiện nhận diện khuôn mặt và cử chỉ, kết hợp với cơ sở dữ liệu MySQL để lưu trữ thông tin. Trong quy trình nhận diện khuôn mặt, đầu tiên sử dụng mô hình MTCNN (Multi-task Cascaded Convolutional Networks) để phát hiện vị trí khuôn mặt và điểm chuẩn trên khuôn mặt từ ảnh đầu vào. Sau đó, ảnh khuôn mặt đã cắt truyền vào mô hình FaceNet để trích xuất các vector đặc trưng của khuôn mặt. Vector này được so sánh với cơ sở dữ liệu mặt đã được lưu trong file để nhận dạng hoặc xác thực danh tính người dùng. Nếu nhận diện được khuôn mặt hợp lệ của thành viên trong gia đình thì tự động mở cửa, nếu phát hiện khuôn mặt lạ thì kích hoạt cảnh báo. Nhờ vậy hệ thống vừa có thể phản hồi tức thời với các sự kiện an ninh thông minh, vừa ghi nhận và lưu trữ các dữ liệu liên quan.

Đối với nhận diện cử chỉ tay, hệ thống sử dụng thư viện MediaPipe của Google. MediaPipe cung cấp mô-đun Gesture Recognizer hoạt động theo thời gian thực, cho phép phát hiện và phân loại các cử chỉ bàn tay từ luồng ảnh; kết quả bao gồm các nhãn cử chỉ cùng tọa độ các điểm mốc bàn tay (hand landmarks).

Hệ thống kết hợp đồng thời giao tiếp qua giao thức HTTP và WebSocket để trao đổi dữ liệu cảm biến, trạng thái thiết bị và lệnh điều khiển. Giao thức HTTP (qua các API RESTful của Flask) được dùng cho các yêu cầu lấy dữ liệu hoặc gửi lệnh đơn lẻ (ví dụ như cập nhật cấu hình ban đầu, hoặc đọc giá trị sensor khi cần). Đối với trao đổi liên tục và thời gian thực, sử dụng WebSocket để thiết lập một kênh liên lạc

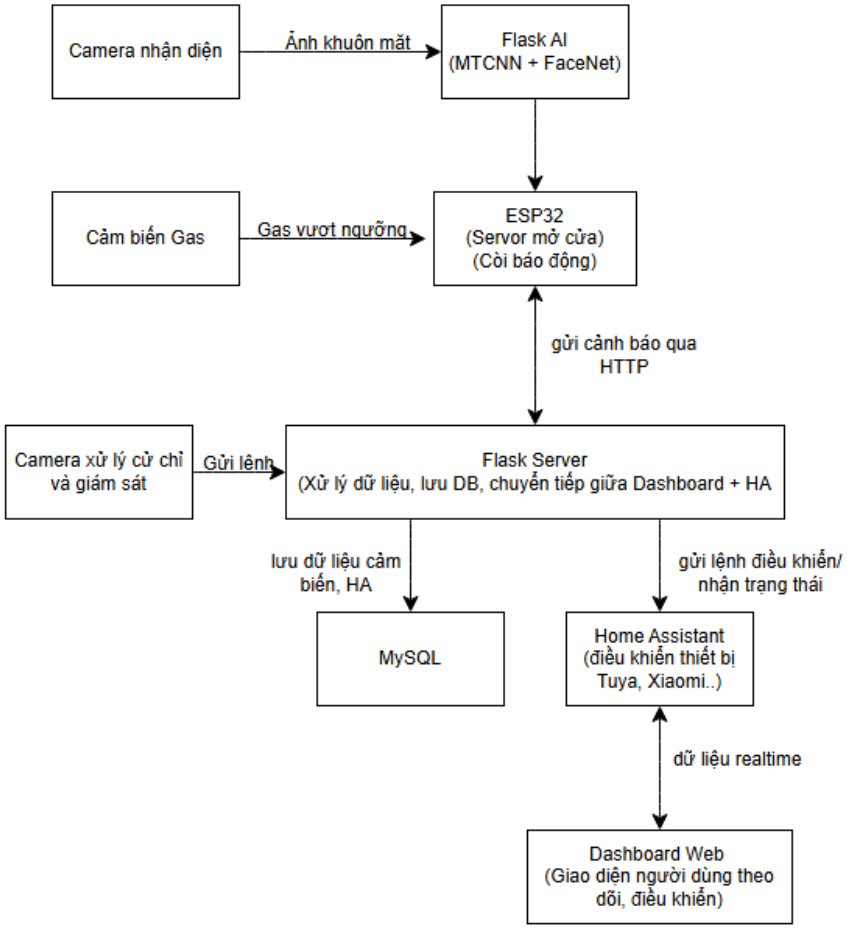
hai chiều liên tục giữa server và client (Dashboard, Home Assistant và chip vi xử lý ESP32). WebSocket cho phép server và client gửi dữ liệu qua lại bất cứ lúc nào mà không cần phải yêu cầu mới (không bị giới hạn bởi cơ chế request-response của HTTP). Nhờ vậy, khi chip vi xử lý ESP32 có giá trị cảm biến mới, Server sẽ nhanh chóng đẩy update đến Dashboard ngay lập tức. Và khi cảm biến xác nhận có người đứng trước cửa thì chip vi xử lý ESP32 sẽ lập tức gửi thông tin đến Server và thực hiện thao tác bật camera để xác thực gương mặt. Sau đó Server sẽ gửi ngược lại thông tin cho chip vi xử lý ESP32 và Dashboard để thực thi các lệnh trong thời gian thực. Cơ chế này đảm bảo Dashboard luôn hiển thị trạng thái thiết bị và giá trị cảm biến mới nhất, đồng thời lệnh điều khiển từ người dùng được triển khai tức thời.

2.1.4 Sơ đồ luồng dữ liệu và thành phần

Trong hệ thống của nhóm, các thành phần chính bao gồm chip vi xử lý ESP32, Flask server, Model Machine Learning xử lý hình ảnh, Dashboard web, Home Assistant, và cơ sở dữ liệu MySQL. Luồng dữ liệu chi tiết gồm:

- Camera gửi ảnh khuôn mặt đến Server → Server xử lý để nhận diện.
- Nếu đúng khuôn mặt → gửi lệnh mở cửa xuống chip vi xử lý ESP32 (servo hoạt động).
- Nếu sai khuôn mặt → gửi lệnh cảnh báo xuống chip vi xử lý ESP32 kích hoạt còi, gửi cảnh báo đến Flask.
- Cảm biến khí gas→ nếu vượt ngưỡng→ chip vi xử lý ESP32 gửi cảnh báo đến Server và kích hoạt còi.
- Server ghi lại cảnh báo vào MySQL.
- Server tiếp nhận dữ liệu cảm biến, cảnh báo, lưu vào MySQL.
- Người dùng điều khiển thiết bị qua Dashboard → Server gửi lệnh cho Home Assistant.
- Server đẩy dữ liệu cập nhật liên tục đến Dashboard bằng giao thức Websocket.

Để trực quan hoá luồng dữ liệu giữa các thành phần trong hệ thống. Hình 2.3 dưới dây minh hoạ rõ quy trình giao tiếp và xử lý các thiết bị cảm biến, server Flask, thành phần Machine Learning và giao diện người dùng.



Hình 2.3 Luồng dữ liệu chính

2.2 Thiết kế phần cứng của hệ thống

Để đáp ứng các chức năng chính của hệ thống nhà thông minh đa nền tảng như: nhận diện khuôn mặt để mở cửa, điều khiển thiết bị từ xa qua web, giám sát môi trường (khí gas, nhiệt độ, độ ẩm), phát hiện người trước cửa, và đồng bộ trạng thái thiết bị với nền tảng Home Assistant, việc lựa chọn các thiết bị phần cứng cần đảm bảo tiêu chí: tính tương thích cao, khả năng giao tiếp linh hoạt, và chi phí hợp lý.

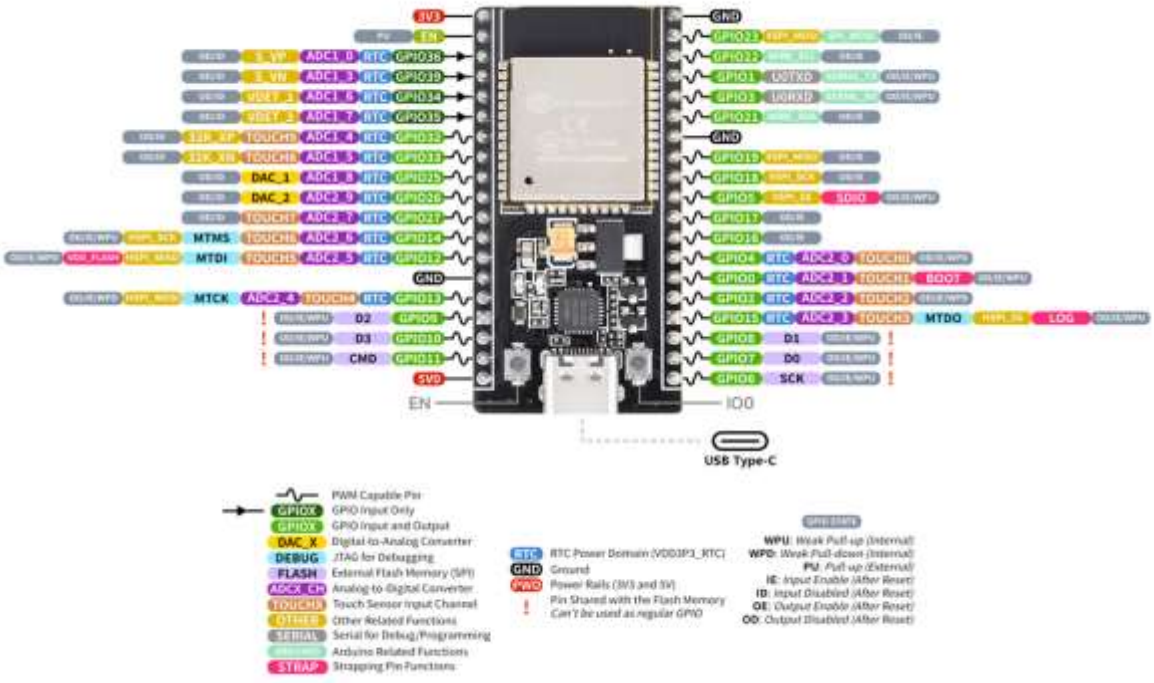
2.2.1 Bo mạch NodeMCU-32S

Bo mạch NodeMCU-32S là một bo mạch phát triển giàu tính năng, dựa trên module ESP32-WROOM-32. Với kết nối Wi-Fi 802.11 b/g/n và Bluetooth 4.2 (BR/EDR + BLE) tích hợp sẵn, đây là bo mạch hoàn toàn phù hợp cho các dự án IoT yêu cầu khả năng kết nối không dây mạnh mẽ.

Với hơn 30 chân GPIO đa năng hỗ trợ ADC, DAC, PWM, I²C, SPI, UART, cảm ứng điện dung và nhiều chức năng khác, NodeMCU-32S cho phép dễ dàng giao tiếp với nhiều loại cảm biến, màn hình và cơ cấu chấp hành.

Hình 2.4 Bo mạch phát triển NodeMCU-32S

Sơ đồ chân (pinout) của bo mạch NodeMCU-32S mang đến hầu hết các tính năng của module ESP32-WROOM-32 trong một định dạng thân thiện với breadboard. Các đường cấp nguồn chính—VIN (5 V), 3V3 và nhiều chân GND—cho phép bạn cấp nguồn cho bo mạch hoặc cung cấp điện cho cảm biến và cơ cấu chấp hành bên ngoài một cách dễ dàng.



Hình 2.5 Sơ đồ chân của bo mạch NodeMCU-32S

Về giao tiếp nối tiếp, bo mạch có sẵn RX0 và TX0 (UART0, kết nối với chip CP2102 để lập trình qua USB), cùng một cặp UART phụ RX2 (GPIO16) và TX2 (GPIO17) cho UART phần cứng thứ hai. Các chân I²C mặc định là SDA (GPIO21) và SCL (GPIO22), trong khi cổng SPI mặc định (VSPI) được ánh xạ đến MOSI (GPIO23), MISO (GPIO19), SCK (GPIO18) và CS (GPIO5). Ngoài ra còn có bus HSPI thay thế (GPIO12 – GPIO15) khi bạn cần kết nối nhiều thiết bị SPI.

Về tín hiệu analog, các chân GPIO32 đến GPIO39 là các kênh ADC1, còn ADC2 xuất hiện ở GPIO0, GPIO2, GPIO4, GPIO12–15 và GPIO25–27 (Lưu ý: ADC2 không hoạt động khi Wi-Fi đang bật). Hai ngõ ra DAC 8-bit nằm ở GPIO25 và GPIO26 — rất phù hợp để phát âm thanh hoặc tạo điện áp tham chiếu. Nhiều chân cũng hỗ trợ cảm ứng điện dung, cho phép tạo giao diện điều khiển không nút bấm hiện đại.

Tất cả các chân GPIO đều hỗ trợ kĩ thuật điều chế độ rộng xung PWM, dùng để điều chỉnh độ sáng LED hoặc điều khiển động cơ. Các chân chức năng đặc biệt như EN (reset) và BOOT (IO0) giúp việc nạp firmware dễ dàng. Lưu ý các chân bootstrapping như GPIO0, GPIO2 và GPIO15 cần được giữ ở trạng thái phù hợp (thường là HIGH) trong quá trình reset để đảm bảo khởi động bình thường.

Với số lượng lớn chân I/O số, khả năng xử lý tín hiệu analog đa dạng, và hỗ trợ nhiều bus tốc độ cao, sơ đồ chân vi điều khiển NodeMCU-32S cung cấp tính linh hoạt cần thiết cho mọi thứ - từ các nút cảm biến đơn giản đến thiết bị kết nối phức tạp - mà vẫn đủ nhỏ gọn cho quá trình thử nghiệm nhanh chóng.

2.2.2 Cảm biến khí gas MQ2

MQ2 là cảm biến khí, dùng để phát hiện các khí có thể gây cháy. Được cấu tạo từ chất bán dẫn SnO2. Chất này có độ nhạy thấp với không khí sạch, nhưng khi trong môi trường có chất dễ cháy, độ dẫn của nó rất nhạy. Chính nhờ đặc điểm này người ta đã ứng dụng và thiết kế ra cảm biến này.

Hình 2.6 Cảm biến khí gas MQ2

Khi môi trường không khí sạch, điện áp đầu ra của cảm biến thấp, giá trị điện áp đầu ra tỉ lệ thuận với nồng độ khí gas. Cảm biến khí gas MQ2 hoạt động tốt trong môi trường khí hoá lỏng LPG, H2 và các khí gây cháy khác. Nó được sử dụng rộng rãi trong công nghiệp và dân dụng vì đơn giản và chi phí thấp.

2.2.3 Cảm biến siêu âm SR-04

Cảm biến siêu âm HC-SR04 sử dụng nguyên lý phản xạ sóng siêu âm. Cảm biến gồm 2 module.1 module phát ra sóng siêu âm và 1 module thu sóng siêu âm phản xạ về. Đầu tiên cảm biến sẽ phát ra 1 sóng siêu âm với tần số 40khz. Nếu có chướng ngại vật trên đường đi, sóng siêu âm sẽ phản xạ lại và tác động lên module nhận sóng. Bằng cách đo thời gian từ lúc phát đến lúc nhận sóng ta sẽ tính được khoảng cách từ cảm biến đến chướng ngại vật.

Hình 2.7 Cảm biến siêu âm SR-04

Cảm biến siêu âm HC-SR04 được sử dụng rất phổ biến để xác định khoảng cách vì rẻ và chính xác. Cảm biến siêu âm HC-SR04 sử dụng sóng siêu âm và có thể đo khoảng cách trong khoảng từ 2 -> 300cm, với độ chính xác gần như chỉ phụ thuộc vào cách lập trình.

2.2.4 Thiết bị đo nhiệt độ, độ ẩm

Thiết bị đo nhiệt độ, độ ẩm Xiaomi Mi Temperature and Humidity Monitor 2 LYWSD03MMC thiết kế nhỏ gọn, màn hình LCD 1.5inch, chế tạo từ vật liệu ABS bảo vệ môi trường, sử dụng cảm biến Swiss Sensirion chính xác, cài đặt dễ dàng qua App Mihome. Với phạm vi đo nhiệt độ là 0°C - 60°C, độ chính xác nhiệt độ 0.1°C, phạm vi đô độ ẩm là 0 - 99% RH cùng độ chính xác độ ẩm 1%.

Hình 2.8 Thiết bị đo nhiệt độ, độ ẩm Xiaomi

Với việc sử dụng nhiệt ẩm kế điện tử Xiaomi LYWSD03MMC, có thể dễ dàng thu thập dữ liệu nhiệt độ và độ ẩm thông qua Bluetooth Low Energy (BLE) và tích hợp trực tiếp vào Home Assistant thông qua chip vi xử lý ESP32. Thiết bị này tương thích tốt với các nền tảng mã nguồn mở, cho phép cập nhật dữ liệu theo thời gian thực và xây dựng các automation như tự động bật quạt, điều hòa hoặc thông báo khi vượt ngưỡng nhiệt độ/độ ẩm thiết lập sẵn.

2.2.5 Bóng đèn thông minh Yeelight

Bóng đèn LED dây tóc Yeelight sử dụng dây tóc cao cấp có hiệu suất cao. Cụ thể hiệu suất ánh sáng là 117lm/w, tăng khả năng chiếu sáng nhưng vẫn tiết kiệm điện hơn so với các bóng đèn LED truyền thống. Ngoài ra các hạt đèn màu cung cấp nhiệt độ màu lên đến 3000K cho nguồn sáng ấm áp như ánh hoàng hôn, giúp không gian nhà bạn trở nên ấm áp hơn. Đặc biệt nguồn sàn vẫn vô cùng dịu nhẹ và không làm mỏi mắt người dùng.

Hình 2.9 Bóng đèn thông minh Yeelight

Thông số kỹ thuật:

- Model: YLDP12YL
- Công suất: 6W
- Nhiệt độ màu: 1700-3500K
- Cường độ sáng: 700 Lumen
- Kết nối: Wifi 2.4GHz

Bóng đèn dây tóc Yeelight YLDP12YL có khả năng kết nối với app trên smartphone. Hỗ trợ người dùng điều khiển tắt/ mở đèn và điều chỉnh độ sáng từ xa. Dù người dùng đang ở công ty vẫn có thể kiểm tra đèn hoặc mở/ tắt đèn tại nhà. Bên cạnh đó đèn LED Yeelight hỗ trợ kết nối với các thiết bị Homekit của Apple. Người dùng có thể điều khiển tắt/ mở đèn bằng giọng nói khi kết hợp loa thông minh có trợ lý ảo Google Assistant, Alexa hoặc Siri.

2.2.6 Bóng đèn thông minh Ezviz

Tương tự như bóng đèn của hãng Yeelight, hãng Ezviz cũng cung cấp giải pháp bóng đèn thông minh với khả năng bật/tắt thiết bị từ xa.

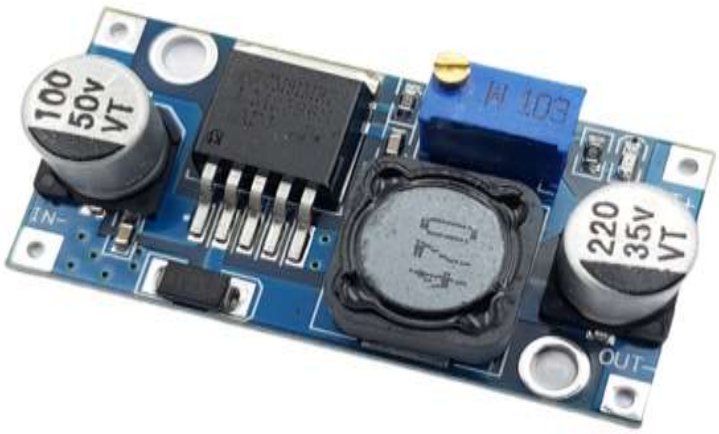
Hình 2.10 Bóng đèn thông minh Ezviz

Thông số kỹ thuật:

- Model: CS-HAL-LB1-LCAW
- Công suất: 8W
- Nhiệt độ màu: 2700K
- Cường độ sáng: 806 Lumen
- Kết nối: Wifi 2.4GHz

2.2.7 Module giảm áp LM2596

Mạch giảm áp DC LM2596 3A nhỏ gọn có khả năng giảm áp từ 30V xuống 1.5V mà vẫn đạt hiệu suất cao (92%) . Thích hợp cho các ứng dụng chia nguồn, hạ áp, cấp cho các thiết bị như camera, motor, robot,…



Hình 2.11 Module giảm áp LM2596

Thông số kỹ thuật

- Điện áp đầu vào: Từ 3V đến 30V.
- Điện áp đầu ra: Điều chỉnh được trong khoảng 1.5V đến 30V.
- Dòng đáp ứng tối đa là 3A.
- Hiệu suất: 92%
- Công suất: 15W
- Kích thước: 45 (dài) * 20 (rộng) * 14 (cao) mm

2.2.8 Thiết kế bo mạch in

Trong quá trình phát triển hệ thống, để đảm bảo mạch điện được sắp xếp gọn gàng, chắc chắn và dễ dàng triển khai thực tế, nhóm tiến hành thiết kế bo mạch in (PCB) dành cho các thành phần chính của hệ thống như: chip vi xử lý ESP32, cảm biến MQ2, servo điều khiển cửa, và các linh kiện hỗ trợ khác.

a. Yêu cầu thiết kế

- Kích thước nhỏ gọn, phù hợp với không gian lắp đặt trong hộp điện hoặc vỏ thiết bị.
- Tối ưu đường đi tín hiệu, giảm nhiễu giữa các mạch tín hiệu số và mạch điều khiển công suất.
- Cổng kết nối rõ ràng để dễ dàng lắp ráp, bảo trì và thay thế linh kiện.

- Khả năng mở rộng, ví dụ như chừa sẵn các chân header cho module cảm biến hoặc giao tiếp UART, I²C.

b. Phần mềm thiết kế

Nhóm sử dụng phần mềm Altium Designer để thiết kế sơ đồ nguyên lý và layout PCB. Các bước chính bao gồm:

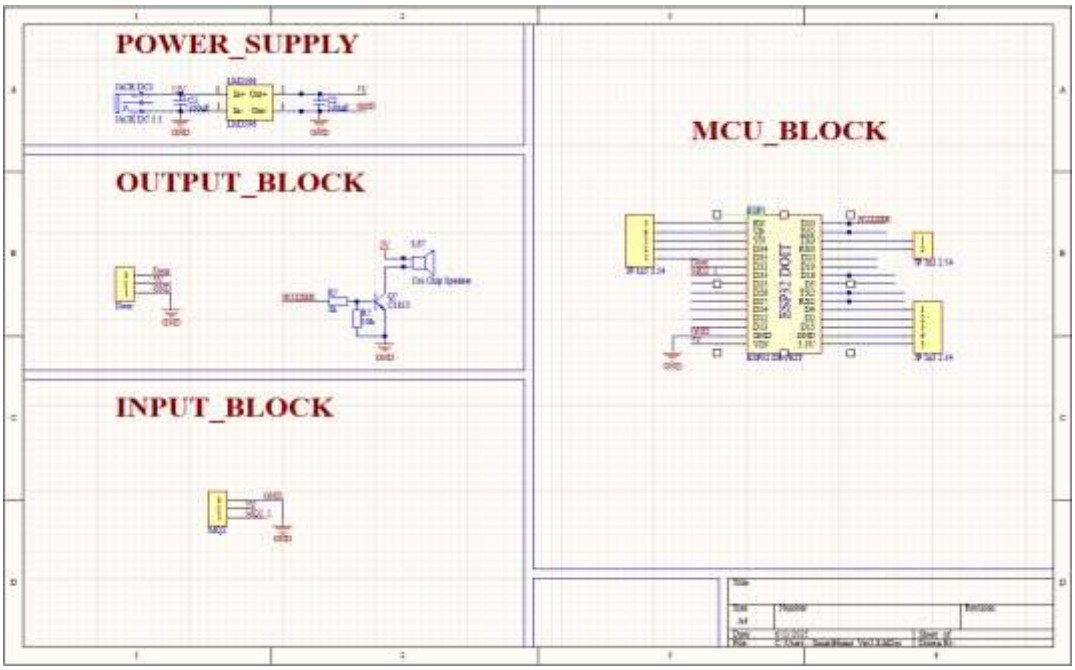
- Vẽ sơ đồ nguyên lý đầy đủ cho toàn bộ hệ thống.
- Chuyển sang môi trường PCB layout để bố trí linh kiện và đi dây.
- Tối ưu hóa bố trí để giảm nhiễu, sử dụng lưới ground, và tránh xung đột tín hiệu.
- Xuất file Gerber để sản xuất mạch in.

c. Các thành phần trên bo mạch

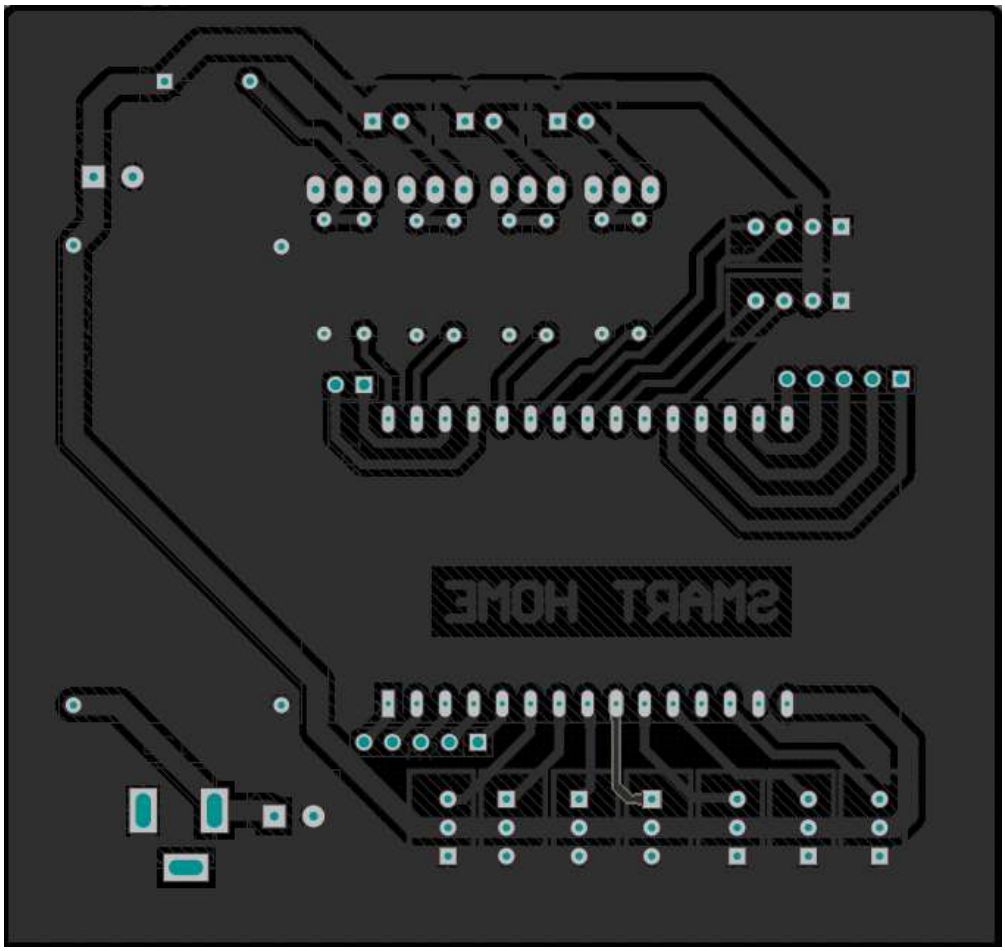
Bo mạch bao gồm các thành phần chính sau:

- ESP32 NodeMCU-32S.
- Nguồn 5V/3.3V ổn định từ LM2596.
- Header kết nối servo, cảm biến siêu âm HC-SR04.
- LED chỉ thị trạng thái, nút reset và boot.

d. Hình ảnh của bo mạch



Hình 2.12 Sơ đồ nguyên lý



Hình 2.13 Sơ đồ mạch in

2.3 Giao tiếp giữa các thành phần

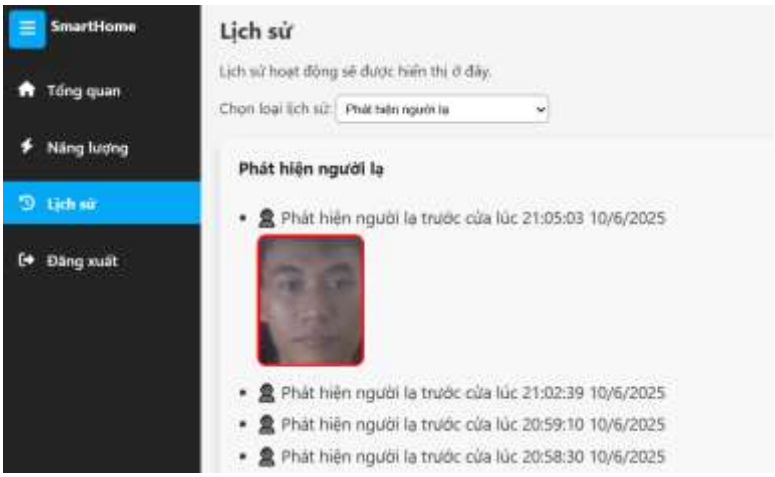
Hệ thống nhà thông minh được thiết kế với nhiều thành phần phần cứng và phần mềm liên kết chặt chẽ thông qua các giao thức truyền thông mạng. Quá trình giao tiếp giữa các thành phần chính bao gồm: chip vi xử lý ESP32, Flask Server, Dashboard web, Home Assistant, và cơ sở dữ liệu MySQL, mỗi thành phần đảm nhiệm một vai trò cụ thể và tương tác thông qua giao thức HTTP, WebSocket, hoặc API nội bộ.

2.3.1 Giao tiếp giữa vi điều khiển ESP32 và Flask Server

ESP32 là vi điều khiển chịu trách nhiệm tiếp nhận dữ liệu từ cảm biến khí gas và thực hiện các tác vụ điều khiển như kích hoạt còi báo động hoặc mở cửa qua servo. Khi phát hiện nồng độ khí gas vượt ngưỡng an toàn, chip vi xử lý ESP32 gửi cảnh báo đến Server Flask thông qua giao thức HTTP. Gói tin HTTP POST chứa giá trị đo từ cảm biến sẽ được Server xử lý và lưu trữ vào cơ sở dữ liệu MySQL. Trong trường hợp Server nhận diện được khuôn mặt hợp lệ từ hệ thống, nó sẽ gửi yêu cầu điều khiển trở lại cho chip vi xử lý ESP32 (mở cửa) thông qua HTTP hoặc WebSocket, tùy theo thiết lập thời gian thực.

2.3.2 Giao tiếp giữa Server và Machine Learning

Server tích hợp các model Machine Learning như MTCNN và FaceNet để xử lý hình ảnh từ camera và thực hiện nhận diện khuôn mặt. Hình ảnh từ camera được truyền về Server, tại đây MTCNN sẽ trích xuất khuôn mặt, và FaceNet sinh ra vector đặc trưng. Server sau đó đối chiếu vector này với dữ liệu trong file đã lưu để xác định danh tính. Nếu xác thực thành công, Server gửi lệnh điều khiển cho ESP32 mở cửa. Nếu thất bại, Server ghi nhận và lưu ảnh người lạ vào hệ thống cảnh báo.



Hình 2.14 Lưu ảnh người lạ vào hệ thống cảnh báo

2.3.3 Giao tiếp giữa Server và Home Assistant

Các thiết bị thông minh như đèn, quạt, ổ cắm thông minh,... được điều khiển gián tiếp thông qua nền tảng Home Assistant. Khi người dùng thao tác trên Dashboard web (ví dụ: bật/tắt thiết bị), yêu cầu điều khiển sẽ được gửi từ Dashboard đến Server. Sau đó, Server sẽ sử dụng API RESTful để chuyển tiếp lệnh đến Home Assistant. Home Assistant thực hiện điều khiển thực tế thiết bị và phản hồi lại trạng thái mới cho Server, từ đó Server cập nhật lại giao diện người dùng.

2.3.4 Giao tiếp giữa Server và giao diện người dùng

Giao diện người dùng là nơi người dùng theo dõi trạng thái hệ thống và gửi lệnh điều khiển. Giao tiếp giữa Server và giao diện chủ yếu sử dụng giao thức WebSocket để đảm bảo dữ liệu được cập nhật theo thời gian thực. Khi có sự kiện xảy ra như cảnh báo khí gas, phát hiện người lạ, hay thiết bị được bật/tắt, Server sẽ đẩy thông tin tức thời đến dashboard. Ngược lại, khi người dùng gửi lệnh điều khiển (ví dụ mở cửa, tắt quạt), yêu cầu này cũng được chuyển đến Server để xử lý ngay lập tức. Cơ chế này đảm bảo độ trễ thấp và phản hồi nhanh.

Hình 2.15 Cảnh báo khi phát hiện khí Gas – giao tiếp giữa Server và Dashboard

2.3.5 Giao tiếp với cơ sở dữ liệu MySQL

Tất cả các thông tin từ cảm biến, cảnh báo, và lệnh điều khiển đều được Server ghi lại vào cơ sở dữ liệu MySQL. Các bảng chính bao gồm:

- sensor_history: lưu các giá trị đo từ cảm biến môi trường (nhiệt độ, độ ẩm, gas,...).
- alert_history: lưu các sự kiện bất thường như cảnh báo khí gas hoặc nhận diện khuôn mặt lạ.

Giao diện sẽ truy xuất dữ liệu từ cơ sở dữ liệu để hiển thị các biểu đồ lịch sử và trạng thái hệ thống, giúp người dùng dễ dàng theo dõi và đánh giá mức độ an toàn cũng như mức tiêu thụ năng lượng trong nhà.

2.4 Quản lý dữ liệu và lịch sử

Trong một hệ thống nhà thông minh hiện đại, dữ liệu đóng vai trò cốt lõi trong việc giám sát, đánh giá hiệu suất cũng như đảm bảo khả năng truy xuất thông tin về sau. Việc quản lý dữ liệu và lưu trữ lịch sử một cách có hệ thống không chỉ giúp người dùng nắm được trạng thái thiết bị theo thời gian thực, mà còn hỗ trợ phát hiện các bất thường, tối ưu hóa điều khiển và nâng cao mức độ an toàn.

2.4.1. Thiết kế cơ sở dữ liệu MySQL

Để đảm bảo tính ổn định và dễ triển khai, hệ thống sử dụng hệ quản trị cơ sở dữ liệu quan hệ MySQL. Đây là nền tảng lưu trữ phổ biến, mã nguồn mở, hỗ trợ truy vấn linh hoạt và có thể mở rộng tốt khi triển khai trong thực tế. Cơ sở dữ liệu được thiết kế theo hướng phân lớp chức năng, trong đó mỗi bảng đảm nhiệm một loại thông tin riêng biệt. Một số bảng dữ liệu tiêu biểu bao gồm:

- Bảng users: chứa thông tin người dùng đã đăng ký, bao gồm tên, mã người dùng, ảnh khuôn mặt, embedding khuôn mặt đã mã hóa bằng FaceNet.

- Bảng sensors: lưu dữ liệu cảm biến theo thời gian thực như giá trị đo từ cảm biến khí gas MQ2, khoảng cách đo từ cảm biến siêu âm HC-SR04.
- Bảng events: ghi nhận các sự kiện an ninh như phát hiện người trước cửa, cảnh báo khí gas, cửa mở bởi nhận diện cử chỉ hoặc khuôn mặt, hoặc truy cập trái phép.
- Bảng logs: nhật ký hệ thống theo thời gian, có thể bao gồm ảnh chụp người không xác định, thời gian tương tác với thiết bị, hoặc kết quả xử lý từ hệ thống.

Tất cả các bảng đều có cột timestamp để lưu thời gian chính xác sự kiện xảy ra. Điều này giúp truy vết lịch sử hoạt động của hệ thống một cách chính xác.

2.4.2. Lưu trữ dữ liệu định kỳ từ Home Assistant và các thành phần khác

Dữ liệu được thu thập từ nhiều nguồn khác nhau trong hệ thống. Các cảm biến kết nối với chip vi xử lý ESP32 có nhiệm vụ gửi dữ liệu về backend Server thông qua giao thức HTTP. Sau khi Server nhận được thông tin, dữ liệu sẽ được phân loại và lưu vào bảng tương ứng trong MySQL. Một số quy trình cụ thể:

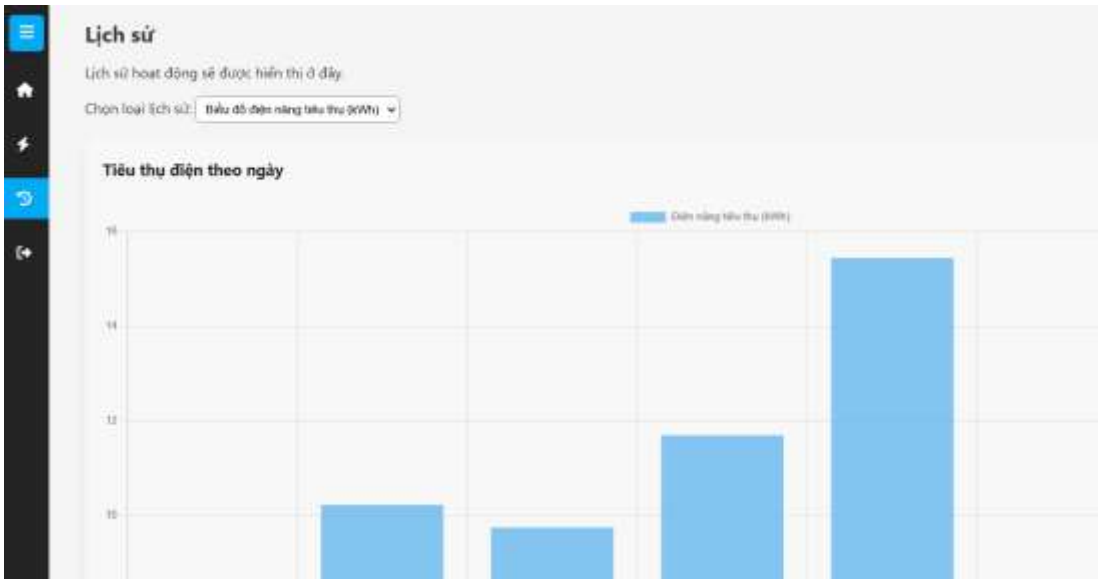
- Cảm biến khí gas MQ2 gửi giá trị đo nồng độ khí. Nếu vượt ngưỡng cho phép, Server ghi một bản ghi mới vào bảng events kèm cảnh báo "Gas Detected".
- Cảm biến siêu âm gửi khoảng cách đo đến Server mỗi 1–2 giây. Nếu phát hiện người đứng gần cửa, hệ thống kích hoạt camera nhận diện khuôn mặt.
- Nếu một người không xác thực được qua FaceNet, hệ thống sẽ lưu ảnh chụp khuôn mặt vào ổ đĩa và ghi đường dẫn ảnh vào bảng logs, kèm timestamp và ghi chú "Stranger".
- Khi người dùng điều khiển thiết bị (đèn, quạt...) qua dashboard, hệ thống ghi lại thời điểm và hành động tương ứng vào events, giúp quản trị viên theo dõi hành vi sử dụng.

2.4.3. Truy xuất và hiển thị lịch sử trên giao diện web

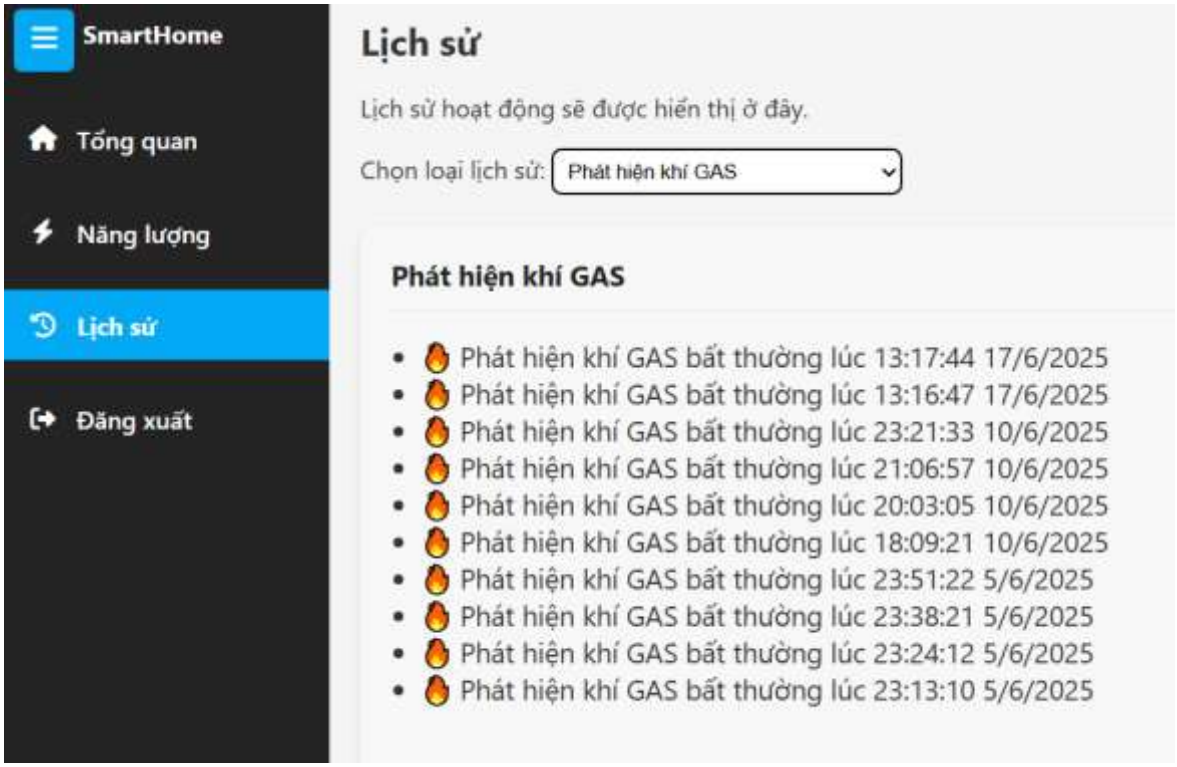
Dữ liệu lịch sử được hiển thị trực quan trên giao diện người dùng bằng bảng thống kê các sự kiện gần đây, biểu đồ kèm thời gian. Dashboard sử dụng Flask kết hợp với giao thức WebSocket để truy vấn dữ liệu từ MySQL và hiển thị realtime các mục sau:

- Lịch sử phát hiện người trước cửa.
- Ảnh người không xác thực được.
- Báo động khí gas, báo động người lạ.
- Biểu đồ sản lượng điện năng tiêu thụ.

Hệ thống hỗ trợ phản hồi thời gian thực thông qua giao thức WebSocket. Nghĩa là khi có sự kiện xảy ra, dữ liệu sẽ được hiển thị ngay lập tức trên dashboard mà không cần tải lại trang. Điều này giúp người dùng phát hiện kịp thời và can thiệp nhanh chóng nếu có vấn đề xảy ra (ví dụ: phát hiện xâm nhập, rò rỉ khí gas).



Hình 2.16 Tab lịch sử tiêu thụ điện



Hình 2.17 Tab lịch sử cảnh báo

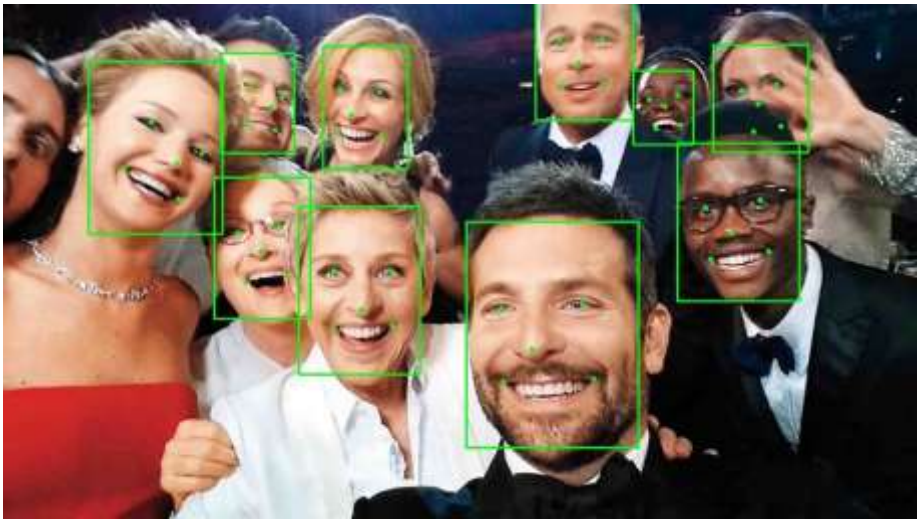
CHƯƠNG III: TÍCH HỢP HỆ THỐNG BẢO MẬT BẰNG NHẬN DIỆN HÌNH ẢNH

3.1 Phát hiện và nhận diện khuôn mặt

Nhận diện khuôn mặt (Face Recognition) là một trong những thách thức lớn mà các nhà nghiên cứu về machine learning – deep learning đã và đang phải đối mặt. Bài toán này có thể được áp dụng ở rất nhiều lĩnh vực khác nhau, đặc biệt trong những lĩnh vực yêu cầu độ chính xác và bảo mật cao như eKYC trong E-Comercial và nhận diện danh tính qua surveillance camera (CCTV). Ta sẽ chia bài toán này ra thành 2 vấn đề chính: Phát hiện khuôn mặt (Face Detection) và Phân biệt khuôn mặt (Face Verification). Để đi sâu hơn về quá trình hiểu và áp dụng hoàn chỉnh một bài Face Recognition, chúng ta sẽ cùng tìm hiểu và suy diễn mô hình (inference) 2 mạng nổi tiếng cho 2 vấn đề trên: MTCNN và FaceNet.

3.1.1 Ứng dụng mô hình MTCCN để phát hiện khuôn mặt

MTCNN là viết tắt của Multi-task Cascaded Convolutional Networks là một mô hình mạng nơ-ron tích chập được thiết kế để phát hiện khuôn mặt và các điểm đặc trưng trên khuôn mặt (facial landmarks) trong hình ảnh.



Hình 3.1 Phát hiện khuôn mặt bằng mô hình MTCNN

Mô hình này bao gồm 3 mạng CNN xếp chồng và đồng thời hoạt động khi phát hiện khuôn mặt. Mỗi mạng có cấu trúc khác nhau và đảm nhiệm vai trò khác nhau. Tượng trưng cho 3 mạng CNN này là P-Net, R-Net và O-Net. Đầu ra của mô hình MTCNN là vị trí khuôn mặt và các điểm trên mặt như: mắt, mũi, miệng…

3.1.1.1 Mạng P-Net

Trước hết, một bức ảnh thường có nhiều hơn một người – một khuôn mặt. Ngoài ra, những khuôn mặt thường có kích thước khác nhau. Ta cần một phương

thức để có thể nhận dạng toàn bộ số khuôn mặt đó, ở các kích thước khác nhau. Mô hình MTCNN đưa ra giải pháp đó là bằng cách thay đổi kích thước ảnh, để tạo một loạt các bản sao từ ảnh gốc với kích cỡ khác nhau, từ to đến nhỏ, để tạo thành một tháp ảnh (Image Pyramid).

Hình 3.2 Image Pyramid

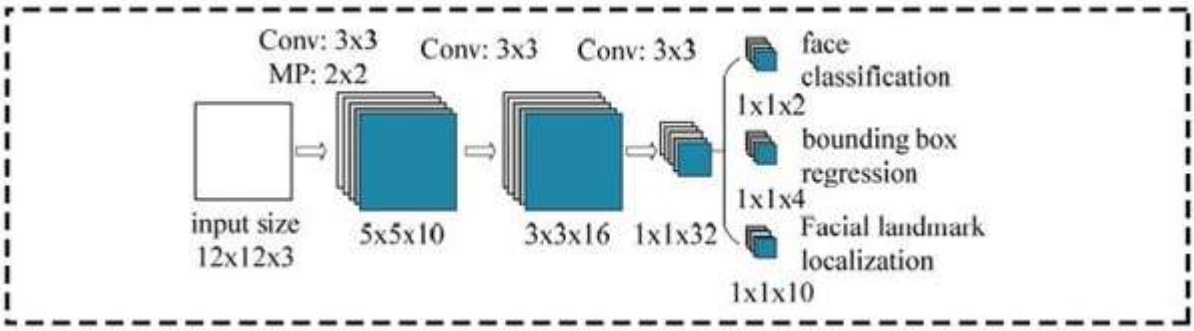
Trong bước đầu tiên của quá trình phát hiện khuôn mặt bằng mô hình MTCNN, ảnh đầu vào sẽ được xử lý theo kỹ thuật Image Pyramid – tức là tạo ra nhiều phiên bản của ảnh gốc với các tỷ lệ kích thước khác nhau (thu nhỏ dần). Mục tiêu của bước này là giúp mạng có thể phát hiện được các khuôn mặt với kích thước khác nhau trong ảnh, dù chỉ sử dụng một cửa sổ trượt (kernel) với kích thước cố định.

Cụ thể, tại mỗi mức của Image Pyramid, ta áp dụng một kernel kích thước 12×12 pixel và thực hiện bước trượt (stride) = 2 để quét qua toàn bộ ảnh. Mỗi vùng 12×12 thu được sẽ được coi là một "cửa sổ quan sát" và được đưa vào mạng P-Net (Proposal Network – Mạng đề xuất) để đánh giá xem vùng đó có khả năng chứa khuôn mặt hay không.

Kết quả đầu ra của P-Net là một tập các bounding boxes (hộp giới hạn) tương ứng với các vùng nghi ngờ có khuôn mặt. Mỗi bounding box bao gồm:

- Tọa độ 4 góc xác định vị trí của hộp trong ảnh (đã được chuẩn hóa về khoảng [0, 1]),
- Và một điểm độ tin cậy (confidence score) biểu thị xác suất có khuôn mặt trong vùng đó.

Nhờ kỹ thuật Image Pyramid kết hợp với P-Net, hệ thống có thể phát hiện khuôn mặt ở nhiều kích thước khác nhau một cách hiệu quả, mà không cần phải huấn luyện nhiều kernel với kích thước khác nhau.

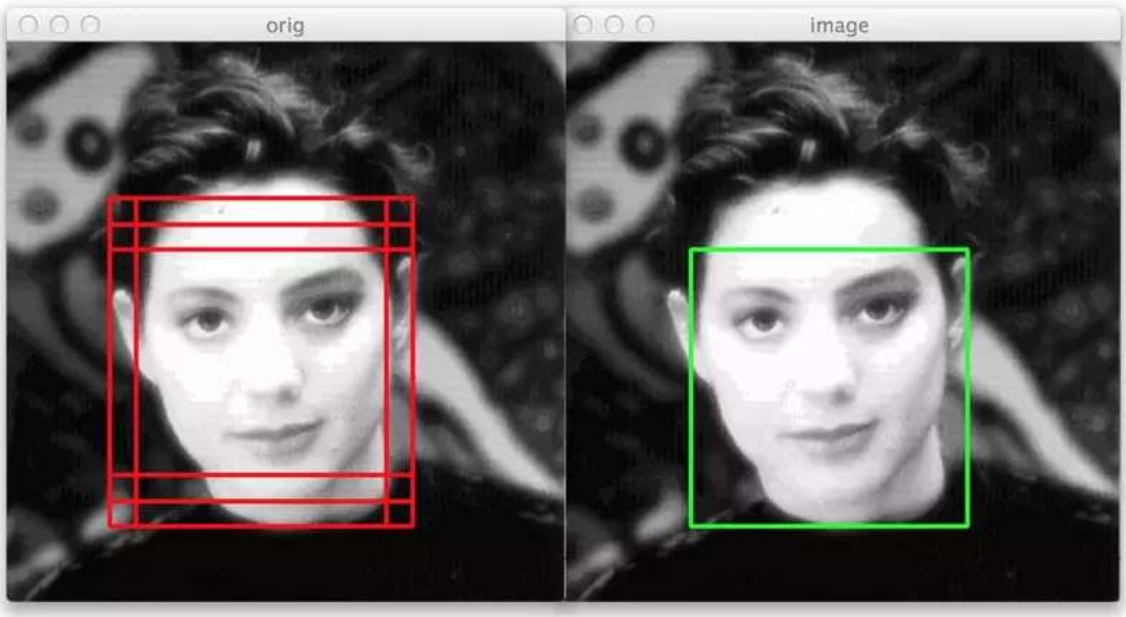


Hình 3.3 Mạng P-Net

Để loại bỏ bớt các hộp giới hạn (bounding boxes) không cần thiết trên các ảnh và các vùng cục bộ (kernels), hệ thống sử dụng hai phương pháp chính:

1. Thiết lập ngưỡng độ tin cậy (Threshold confidence) – nhằm loại bỏ các hộp có chỉ số confident (độ tin cậy) thấp, vốn không đủ chắc chắn để được xem là một khuôn mặt hợp lệ.
2. NMS (Non-Maximum Suppression – loại trừ cực đại không trùng lặp) – kỹ thuật này được sử dụng để loại bỏ các bounding box có mức độ trùng lặp (Intersection over Union – IoU) vượt quá một ngưỡng định sẵn. Khi nhiều hộp phát hiện trùng nhau, thuật toán sẽ giữ lại hộp có độ tin cậy cao nhất, đồng thời loại bỏ các hộp còn lại có mức IoU lớn hơn ngưỡng đã thiết lập.

Hình ảnh minh họa dưới đây thể hiện rõ cách hoạt động của thuật toán NMS: những hộp bị chồng lấn sẽ bị loại bỏ, chỉ giữ lại duy nhất một hộp có độ tin cậy (confidence score) cao nhất.



Hình 3.4 Ảnh trước và sau khi xoá các vùng bị trùng nhau

Sau khi loại bỏ các hộp (bounding boxes) không hợp lệ, bước tiếp theo là chuyển các tọa độ hộp còn lại về hệ tọa độ của ảnh gốc. Do các tọa độ hiện tại đã được chuẩn hóa (normalize) về khoảng [0,1][0,1][0,1] trong không gian của từng *kernel* (vùng cục bộ trên ảnh), nên ta cần thực hiện các phép biến đổi ngược như sau:

Bước 1: Tính toán kích thước thực tế của kernel dựa trên ảnh gốc (chiều rộng và chiều cao).

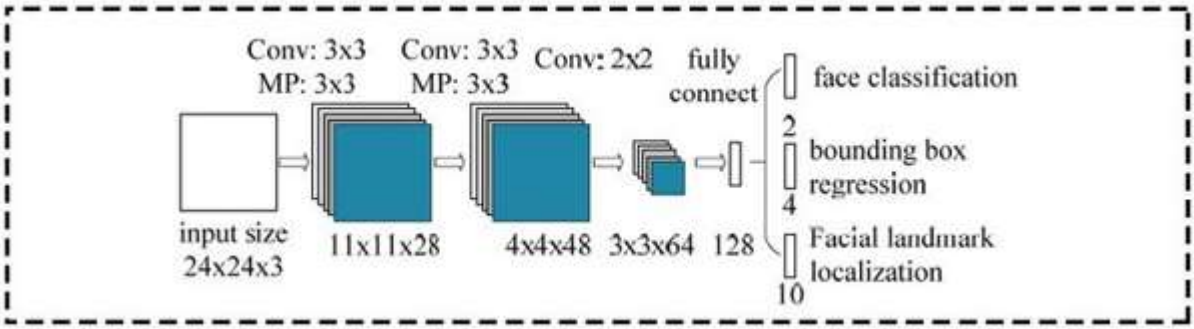
Bước 2: Giải chuẩn hóa (denormalize) các tọa độ của hộp bằng cách nhân với kích thước kernel tương ứng.

Bước 3: Tịnh tiến tọa độ các hộp này bằng cách cộng thêm tọa độ góc trên bên trái của kernel trong ảnh gốc.

Kết quả thu được là các tọa độ của hộp (bounding boxes) tương ứng trên toàn ảnh (full-resolution image). Sau đó, để chuẩn bị cho bước xử lý tiếp theo, ta sẽ chuyển đổi tất cả các hộp về dạng hình vuông bằng cách thay đổi kích thước (resize) tương ứng. Cuối cùng, các hộp này được đưa vào mạng kế tiếp (mạng R) để tiếp tục xử lý.

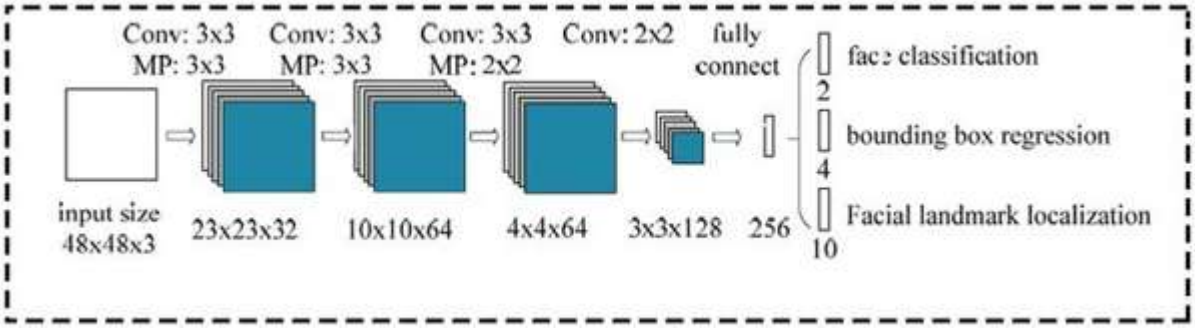
3.1.1.2 Mạng R-Net

Mạng R (Refine Network - Mạng tinh chỉnh) thực hiện các bước như mạng P. Tuy nhiên, mạng còn sử dụng một phương pháp tên là đệm ảnh (padding), nhằm thực hiện việc chèn thêm các điểm ảnh có giá trị 0 (zero-pixels) vào các phần thiếu của bounding box nếu bounding box bị vượt quá biên của ảnh. Tất cả các bounding box lúc này sẽ được thay đổi kích thước (resize) về kích thước 24x24, được coi như 1 kernel và đưa vào mạng R. Kết quả sau cũng là những tọa độ mới của các box còn lại và được đưa vào mạng tiếp theo, mạng O.



Hình 3.5 Mạng R-Net

3.1.1.3 Mạng O-Net



Hình 3.6 Mạng O-Net

Cuối cùng là mạng O (Output Network – Mạng đầu ra), mạng này cũng thực hiện các bước tương tự như mạng R, trong đó các ảnh đầu vào được đổi kích thước (resize) về 48×48 pixels để phù hợp với kiến trúc mạng.

Tuy nhiên, kết quả đầu ra của mạng O không chỉ đơn thuần là các tọa độ của bounding box (hộp giới hạn) như ở các bước trước, mà bao gồm ba thành phần chính:

- out[0]: Bốn tọa độ (coordinates) xác định vị trí của bounding box.
- out[1]: Tọa độ của năm điểm đặc trưng (landmarks) trên khuôn mặt, bao gồm hai mắt, một mũi, và hai điểm ở hai bên miệng.
- out[2]: Chỉ số độ tin cậy (confidence score) tương ứng với mỗi bounding box.

Toàn bộ kết quả này sẽ được lưu trữ dưới dạng một dictionary (từ điển dữ liệu) với ba key (khóa) tương ứng: box, landmark, và confidence.

Hình 3.7 Hình ảnh sau khi đi qua 3 mạng P-Net, R-Net và O-Net

Như vậy phần phát hiện khuôn mặt (Face Detection) với mô hình MTCNN đã được giải quyết, tiếp theo với bài toán xác minh khuôn mặt (Face Verification), ta sẽ sử dụng mạng FaceNet để tiến hành phân biệt các khuôn mặt.

3.1.2 Ứng dụng mô hình deep learning FaceNet để xác minh khuôn mặt

Ở bài toán xác minh khuôn mặt, nhiệm vụ chính là đánh giá xem ảnh khuôn mặt hiện tại có đúng với thông tin, mặt của một người khác đã có trong hệ thống hay không. FaceNet là một mạng nơ-ron sâu (deep neural network) được sử dụng để trích xuất đặc trưng từ hình ảnh khuôn mặt của một người. Mô hình này được giới thiệu vào năm 2015 bởi các nhà nghiên cứu của Google — Schroff và cộng sự. Để tiếp cận bài toán này, trước tiên ta cần chỉ ra một số khái niệm có trong FaceNet như sau:

- **Embedding vector:** là một vector đặc trưng có kích thước cố định, được mô hình học sâu tạo ra trong quá trình huấn luyện. Vector này đại diện cho những đặc điểm quan trọng của đối tượng (ví dụ như khuôn mặt), giúp phân biệt giữa các đối tượng khác nhau. Embedding thường được dùng để so sánh sự giống nhau giữa các đối tượng bằng cách tính khoảng cách giữa các vector trong không gian. Các vector càng gần nhau thì càng giống nhau, từ đó giúp nhận dạng hoặc phân loại chính xác hơn**.**
- **Inception V1:** là một kiến trúc mạng nơ-ron tích chập (CNN) do Google giới thiệu năm 2014. Đặc điểm nổi bật của mạng là sử dụng các khối Inception**,** cho phép xử lý song song nhiều lớp Convolution khác nhau (1x1, 3x3, 5x5) trên cùng một đầu vào. Kết quả từ các lớp này sẽ được kết nối (concatenate) lại để tạo đầu ra chung. Cách thiết kế này giúp mạng học được nhiều đặc trưng hơn so với mạng CNN truyền thống. Đồng thời, việc sử dụng lớp Convolution 1x1 giúp giảm số lượng tham số và tăng tốc độ huấn luyện.

Mô hình FaceNet nhận đầu vào là hình ảnh khuôn mặt của một người và xuất ra một vector gồm 128 số, đại diện cho những đặc trưng quan trọng nhất của khuôn mặt đó. Trong Machine Learning, vector này được gọi là embedding.

Toàn bộ thông tin quan trọng từ hình ảnh đã được "nhúng" (embedded) vào trong vector này. Nói cách khác, FaceNet chuyển khuôn mặt của một người thành một vector 128 chiều, như một dạng nén dữ liệu khuôn mặt. Lý tưởng là, các khuôn mặt giống nhau sẽ có embedding tương tự nhau. Embedding là một vector, và ta có thể hiểu vector như một điểm trong hệ tọa độ Đề-các. Điều này có nghĩa là ta có thể vẽ vị trí của một hình ảnh khuôn mặt trong hệ tọa độ thông qua embedding của nó.

Một cách khả thi để nhận diện một người trong một hình ảnh chưa từng thấy trước đó là tính toán embedding của khuôn mặt đó, tính khoảng cách đến các hình ảnh của những người đã biết, và nếu embedding của khuôn mặt đó đủ gần với embedding của người A, thì ta kết luận rằng hình ảnh này chứa khuôn mặt của người A.

Để huấn luyện FaceNet, chúng ta cần rất nhiều hình ảnh khuôn mặt. Để đơn giản hóa, ta sẽ giả sử chỉ có vài hình ảnh của hai người. Tuy nhiên, cùng một nguyên lý có thể được áp dụng khi bạn có hàng ngàn hình ảnh của nhiều người khác nhau.

Ở giai đoạn đầu của quá trình huấn luyện, mô hình FaceNet tạo ra các vector ngẫu nhiên cho mỗi hình ảnh, điều đó có nghĩa là các hình ảnh bị phân tán ngẫu nhiên khi vẽ lên không gian.

Mô hình FaceNet học theo cách sau:

1. Chọn ngẫu nhiên một ảnh gốc (anchor image).
2. Chọn ngẫu nhiên một ảnh khác của cùng người với ảnh gốc → gọi là ví dụ dương (positive example).
3. Chọn ngẫu nhiên một ảnh của người khác với ảnh gốc → gọi là ví dụ âm (negative example).
4. Điều chỉnh các tham số của mạng FaceNet sao cho:
    - Embedding của ảnh dương gần hơn ảnh gốc,
    - So với embedding của ảnh âm phải xa hơn.

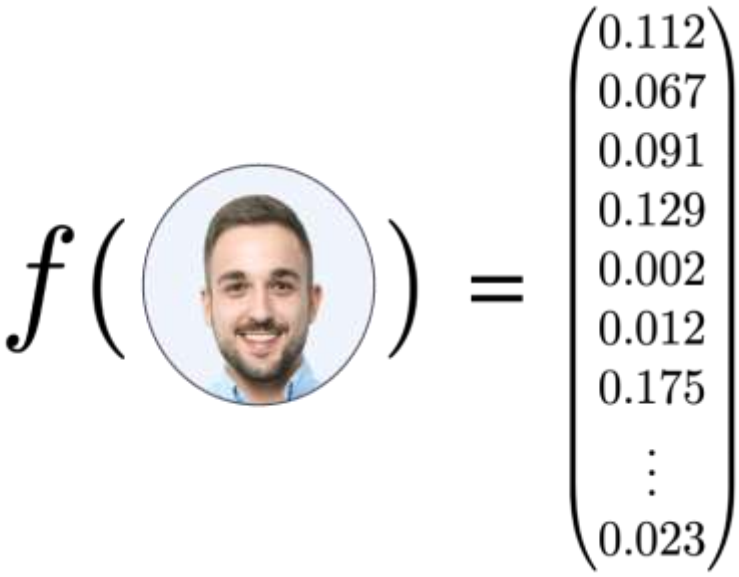
Lặp lại quá trình này liên tục cho đến khi không còn thay đổi đáng kể nào nữa.

Kết quả: Tất cả các khuôn mặt của cùng một người sẽ nằm gần nhau trong không gian embedding, và cách xa khuôn mặt của người khác.

Trong quá trình huấn luyện, chúng ta không trực tiếp nói cho FaceNet biết rằng các con số trong vector nên đại diện cho đặc trưng nào. Điều duy nhất chúng ta yêu cầu là: Các vector embedding của những khuôn mặt giống nhau phải gần nhau, và của những khuôn mặt khác nhau thì phải cách xa nhau.

Việc biểu diễn khuôn mặt như thế nào bằng vector là nhiệm vụ mà mô hình FaceNet tự học trong quá trình huấn luyện. Để làm được điều đó, FaceNet phải tìm ra những đặc điểm then chốt của khuôn mặt giúp phân biệt khuôn mặt này với khuôn mặt khác.

Trong quá trình huấn luyện, FaceNet sẽ thử rất nhiều tổ hợp đặc trưng khác nhau cho đến khi tìm được cách biểu diễn tốt nhất. Tuy nhiên, mạng nơ-ron không biểu diễn đặc trưng giống cách con người nghĩ (như kích thước, khoảng cách,...). Đó là lý do vì sao rất khó để giải thích từng con số trong vector embedding. Nhưng chúng ta có thể chắc chắn rằng những đặc điểm như khoảng cách giữa hai mắt đang được ẩn giấu đâu đó bên trong các con số ấy.



Hình 3.8 Mô tả về cách hoạt động của FaceNet

Đây là một hàm nhận hình ảnh đầu vào và xuất ra embedding khuôn mặt — tức là một bản tóm tắt đặc trưng của khuôn mặt đó.

Điều này có nghĩa là:

- Cùng một người → cùng embedding (gần nhau trong không gian)
- Khác người → khác embedding (cách xa nhau)

Nhờ đó, hệ thống có thể nhanh chóng nhận diện và phân biệt giữa các khuôn mặt.

3.1.3 Ngưỡng xác thực và xử lý ảnh đầu vào

Ngưỡng xác thực là giá trị ngưỡng để xác định xem một khuôn mặt có trùng khớp với khuôn mặt đã huấn luyện hay không.

Sau khi hệ thống tạo vector embedding cho khuôn mặt đầu vào, nó sẽ tính khoảng cách (thường là khoảng cách Euclidean hoặc cosine) giữa embedding này và embedding của người đã lưu.

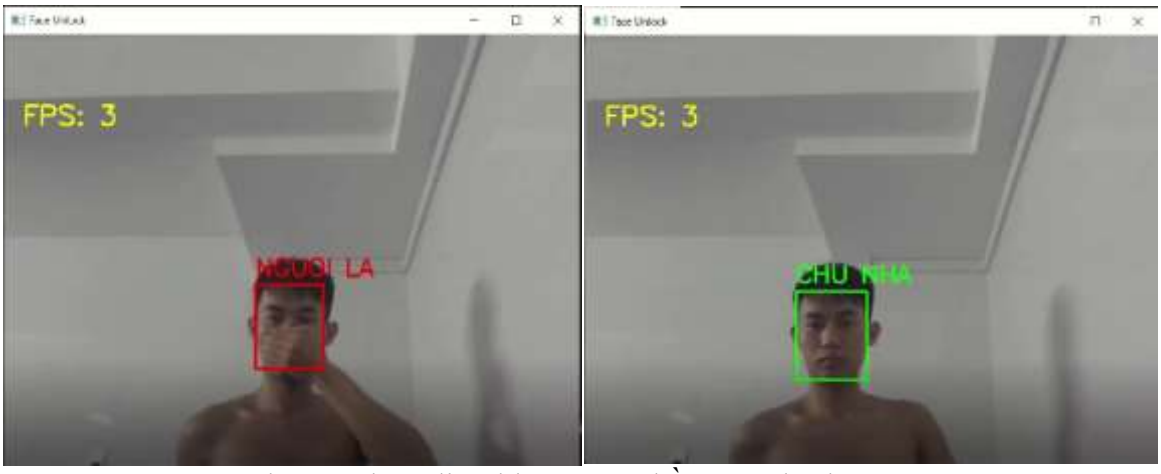
Nếu khoảng cách nhỏ hơn một ngưỡng định trước (threshold), thì hệ thống chấp nhận rằng đó là cùng một người.

Nếu lớn hơn ngưỡng, hệ thống từ chối xác thực.

Việc chọn ngưỡng là cân bằng giữa độ chính xác và bảo mật. Nếu chọn ngưỡng quá thấp dễ bị từ chối sai, nếu ngưỡng quá cao dễ chấp nhận nhầm người.

Trước khi ảnh khuôn mặt được đưa vào mô hình để tạo embedding, cần qua một số bước xử lý như sau:

- Phát hiện khuôn mặt: dùng mô hình MTCNN để xác định vùng có khuôn mặt.
- Căn chỉnh khuôn mặt: đảm bảo các đặc trưng như mắt, mũi, miệng nằm đúng vị trí chuẩn.
- Cắt và thay đổi kích thước ảnh: đưa khuôn mặt về đúng kích thước mà mô hình yêu cầu.
- Chuẩn hóa ảnh: chuẩn hóa giá trị pixel (ví dụ: về khoảng [0,1] hoặc [-1,1]) để giúp mô hình xử lý hiệu quả hơn.



Hình 3.9 Nhận diện khuôn mặt bằng mô hình Facenet

3.2 Tích hợp với hệ thống mở cửa

Hệ thống không chỉ dừng lại ở việc phân biệt người quen hay người lạ, mà còn kết hợp với thiết bị điều khiển cửa (động cơ servo), cảm biến thông qua vi điều khiển ESP32.

Để tiết kiệm tài nguyên xử lý và chỉ kích hoạt camera khi có người thực sự đứng trước cửa, hệ thống sử dụng cảm biến siêu âm HC-SR04 giao tiếp với vi điều khiển ESP32 để phát hiện chuyển động hoặc sự hiện diện ở khoảng cách gần.

- **Nguyên lý hoạt động:** Cảm biến siêu âm HC-SR04 phát ra sóng âm và đo thời gian phản xạ sóng để tính khoảng cách đến vật thể phía trước. Khi phát hiện có người trong vùng xác định (10cm), vi điều khiển sẽ gửi yêu cầu đến Server để kích hoạt camera và tiến hành nhận diện khuôn mặt.
- **Cơ chế thực hiện:**
    - Cảm biến siêu âm liên tục đo khoảng cách.

- Nếu khoảng cách nhỏ hơn ngưỡng, vi điều khiển ESP32 sẽ gửi lệnh POST /trigger_unlock đến Server.
- Server nhận được yêu cầu sẽ kích hoạt camera (cv2.VideoCapture) và thực hiện xác thực khuôn mặt như mô tả ở phần trước.

- **Ưu điểm:**
  - Giảm tiêu thụ tài nguyên hệ thống vì camera chỉ hoạt động khi có người.
  - Tăng cường tính bảo mật, hạn chế xử lý sai trong điều kiện không cần thiết.
  - Tăng tuổi thọ cho camera và tiết kiệm điện năng.

3.2.1 Quy trình mở cửa

Quá trình mở cửa thông minh sau khi xác thực khuôn mặt được thực hiện theo các bước sau:

Bước 1: Khi hệ thống xác thực thành công là người quen sẽ gọi API POST đến vi điều khiển qua địa chỉ đã thiết lập trước.
Bước 2: Vi điều khiển ESP32 nhận lệnh và điều khiển động cơ servo để mở cửa trong một khoảng thời gian nhất định.
Bước 3: Một sự kiện được gửi về giao diện dashboard để cập nhật trạng thái mở cửa.

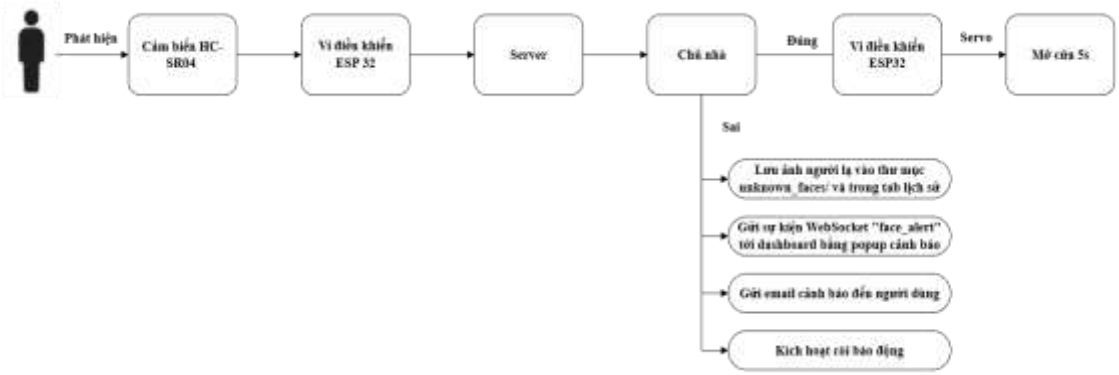
3.2.2 Xử lý khi không nhận dạng được

Nếu hệ thống phát hiện khuôn mặt không trùng hợp sẽ:

- Lưu ảnh người lạ vào thư mục unknown_faces/ và trong tab lịch sử.
- Gửi sự kiện WebSocket "face_alert" tới dashboard bằng một popup cảnh báo.
- Gửi email cảnh báo đến người dùng (qua SMTP Gmail).
- Kích hoạt còi báo động qua vi điều khiển (/alarm).

Tổng hợp luồng hoạt động mở cửa thông minh:

- Người đến trước cửa → Cảm biến HC-SR04 phát hiện → Vi điều khiển gửi tín hiệu tới Server
- Server mở camera, chụp ảnh và xử lý khuôn mặt:
- Nếu đúng chủ nhà: Gửi lệnh mở cửa về vi điều khiển.
- Nếu người lạ: Lưu ảnh, gửi cảnh báo và kích hoạt còi.



Hình 3.10 Sơ đồ luồng hoạt động mở cửa thông minh

3.3 Giao diện phản hồi

Giao diện người dùng (dashboard) đóng vai trò trung tâm trong việc cung cấp thông tin bảo mật theo thời gian thực. Hệ thống được xây dựng với khả năng phản hồi ngay lập tức thông qua kết nối giao thức WebSocket, giúp người dùng giám sát và điều khiển từ xa một cách hiệu quả, đặc biệt trong các tình huống có yếu tố an ninh như phát hiện người lạ hay mở cửa thành công.

3.3.1 Phản hồi thời gian thực qua giao thức Websocket

Hệ thống sử dụng Socket.IO (một thư viện WebSocket cho Python và JavaScript) để truyền dữ liệu từ server đến client trong thời gian thực. Sử dụng giao thức WebSocket giúp giảm độ trễ phản hồi, không cần tải lại trang và tiết kiệm băng thông.

Các sự kiện chính được truyền qua WebSocket gồm:

- face_alert: gửi khi hệ thống phát hiện người lạ đứng trước cửa.
- gas_alert: cảnh báo khi phát hiện khí gas bất thường.

3.3.2 Hiển thị hình ảnh người lạ gần nhất trên dashboard

Khi có người lạ xuất hiện, hệ thống sẽ:

- Lưu lại ảnh khuôn mặt dưới dạng stranger_HHMMSS_DDMMYYYY.jpg vào thư mục unknown_faces.
- Tạo một API endpoint /latest_stranger_image trả về hình ảnh người lạ mới nhất.

Điều này giúp người dùng nhanh chóng biết ai đã đến gần cửa và phản ứng kịp thời, ngay cả khi không có mặt tại nhà.

3.4 Kết luận chương 3

Việc tích hợp công nghệ nhận diện khuôn mặt với cảm biến và thiết bị điều khiển cửa giúp tăng cường khả năng bảo mật, tiện ích, thông minh và khả năng tự động hóa hệ thống ngôi nhà. Đây là một bước quan trọng để đưa hệ thống tiến gần hơn đến mô hình nhà thông minh hiện đại, tối ưu tài nguyên và thân thiện với người dùng.

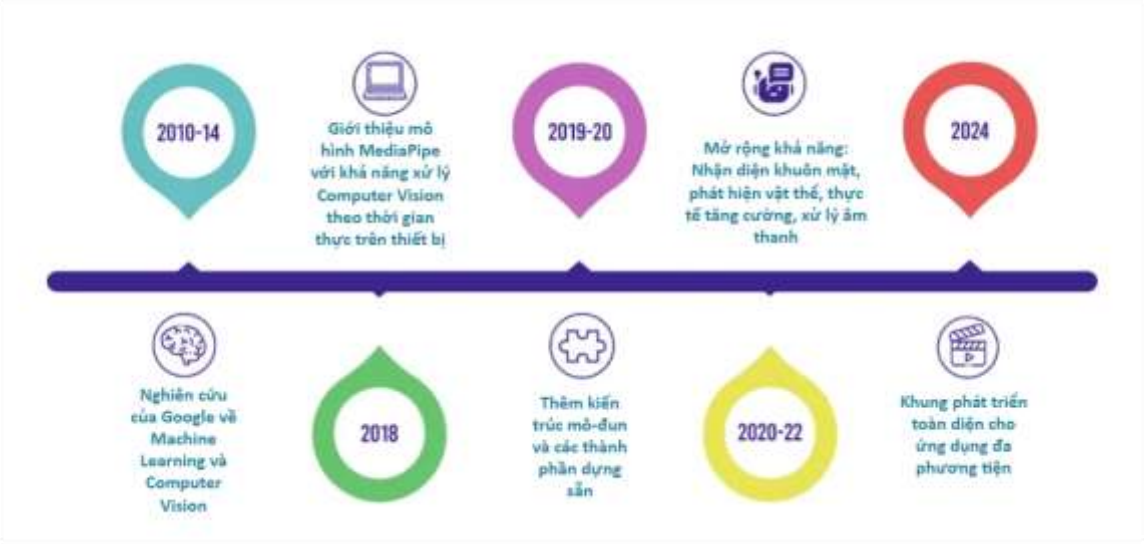
CHƯƠNG IV: THIẾT KẾ HỆ THỐNG ĐIỀU KHIỂN BẰNG CỬ CHỈ

4.1 Nhận diện cử chỉ tay bằng MediaPipe

Để đưa ngôi nhà thông minh nay trở thành thông minh hơn nữa. Thay vì điều khiển các thiết bị bằng công tắc vật lý thông thường, trong chương này ta sẽ ứng dụng một model Machine Learning có tên là mô hình MediaPipe để điều khiển thiết bị thông qua cử chỉ của bàn tay.

Khả năng nhận biết hình dạng và chuyển động của bàn tay có thể là một thành phần quan trọng trong việc nâng cao trải nghiệm người dùng trên nhiều lĩnh vực và nền tảng công nghệ khác nhau. Ví dụ, nó có thể là nền tảng cho việc hiểu ngôn ngữ ký hiệu, điều khiển bằng cử chỉ tay, và còn có thể cho phép phủ nội dung số lên thế giới thực trong các ứng dụng thực tế tăng cường (AR). Mặc dù việc này là tự nhiên đối với con người, nhưng việc nhận diện bàn tay trong thời gian thực một cách ổn định và chính xác lại là một nhiệm vụ khó khăn trong Computer Vision, vì bàn tay thường tự che khuất hoặc che lẫn nhau (ví dụ: ngón tay che lòng bàn tay, hay bắt tay) và thường thiếu các mẫu tương phản cao.

Những tiến bộ trong trí tuệ nhân tạo (AI) trong vài năm qua – chẳng hạn như các phát triển của mô hình MediaPipe từ Google – đã khiến việc tạo ra các trải nghiệm tuyệt vời kết hợp giữa thế giới số và thế giới thực trở nên khả thi. Nguồn gốc của MediaPipe bắt đầu từ đầu những năm 2010, khi Google đang nỗ lực cải tiến các công nghệ Machine Learning và Computer Vision. Mô hình MediaPipe lần đầu tiên được sử dụng vào năm 2012 để phân tích video và âm thanh theo thời gian thực trên YouTube.



Hình 4.1 Dòng thời gian phát triển của mô hình MediaPipe

Vào năm 2018, MediaPipe bắt đầu giải quyết các vấn đề liên quan đến việc triển khai các mô hình Computer Vision phức tạp trên các thiết bị như điện thoại thông minh và máy tính nhúng. Đến năm 2020, nhu cầu xử lý đa phương tiện một cách nhanh chóng và hiệu quả ngày càng tăng, vì vậy mô hình MediaPipe đã được cập nhật để đáp ứng yêu cầu này. Hiện nay, mô hình MediaPipe vẫn là một khung phát triển mạnh mẽ dành cho các lập trình viên muốn xây dựng các ứng dụng đa phương tiện sáng tạo với hiệu năng cao.

Các Tính Năng Cốt Lõi và Công Nghệ của mô hình MediaPipe

MediaPipe tích hợp nhiều tính năng nổi bật. Một trong số đó là khả năng tận dụng sức mạnh xử lý song song của đơn vị xử lý đồ họa (GPU) để tăng tốc độ xử lý. Bằng cách sử dụng GPU cho các tác vụ yêu cầu hiệu năng tính toán cao, mô hình MediaPipe có thể xử lý các tác vụ đa phương tiện phức tạp trong thời gian thực. Nhờ khả năng xử lý song song, MediaPipe có thể thực hiện đồng thời nhiều tác vụ, chẳng hạn như xử lý nhiều luồng video hoặc chạy đồng thời nhiều mô hình thị giác máy tính.

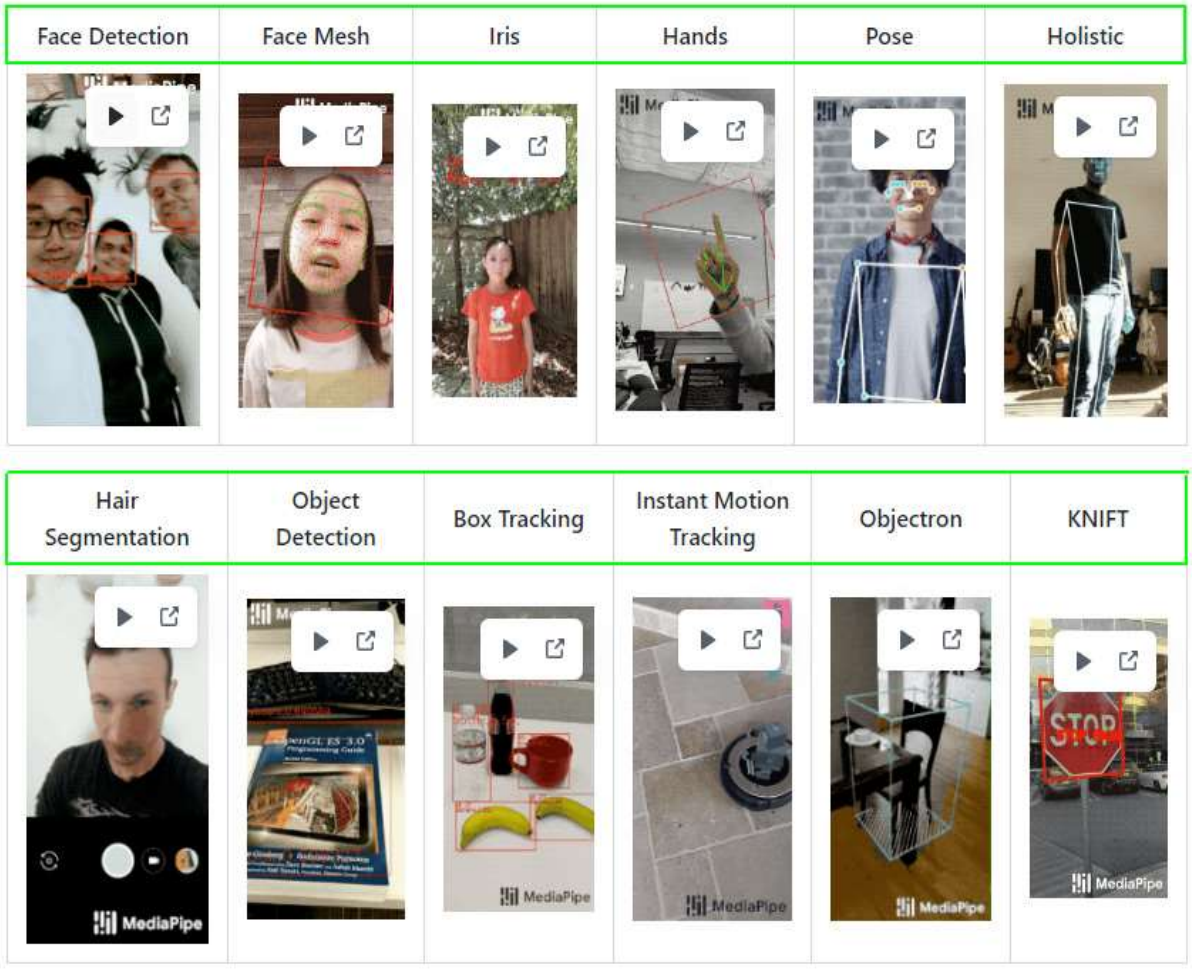
Tuy nhiên, đó chỉ là phần nổi của tảng băng chìm. MediaPipe còn tích hợp OpenCV – thư viện mã nguồn mở mạnh mẽ chuyên về Computer Vision. OpenCV cung cấp nhiều công cụ và thuật toán hỗ trợ xử lý ảnh và video. Nhờ tích hợp thư viện OpenCV, mô hình MediaPipe dễ dàng triển khai các chức năng như thu nhận video, xử lý và hiển thị trong các pipeline của mình. Ngoài ra, MediaPipe còn kết hợp với TensorFlow – nền tảng Machine Learning của Google – giúp tích hợp các mô hình học máy đã huấn luyện sẵn hoặc tùy chỉnh một cách dễ dàng. Điều này giúp đơn giản hóa các tác vụ như nhận diện khuôn mặt hoặc hiểu lời nói. MediaPipe cũng hỗ trợ nhiều ngôn ngữ lập trình phổ biến như C++, Java và Python, giúp dễ dàng tích hợp vào các dự án phần mềm.

Các giải pháp (solutions) trong MediaPipe là những ví dụ mã nguồn mở được xây dựng sẵn, dựa trên các mô hình TensorFlow hoặc TensorFlow Lite đã được huấn luyện trước.

Các giải pháp này được xây dựng dựa trên nền tảng MediaPipe Framework. Hiện tại, mô hình MediaPipe cung cấp 16 giải pháp chính, bao gồm:

1. Face Detection – Phát hiện khuôn mặt
2. Face Mesh – Lưới khuôn mặt 3D
3. Iris – Theo dõi mống mắt
4. Hands – Nhận diện và theo dõi bàn tay
5. Pose – Ước lượng tư thế cơ thể
6. Holistic – Kết hợp toàn diện giữa khuôn mặt, tay và tư thế

7. Selfie Segmentation – Tách nền ảnh selfie
8. Hair Segmentation – Tách vùng tóc
9. Object Detection – Phát hiện đối tượng
10. Box Tracking – Theo dõi hộp đối tượng
11. Instant Motion Tracking (IMT) – Theo dõi chuyển động tức thời
12. Objectron – Nhận diện và theo dõi đối tượng 3D
13. KNIFT – Nhận diện đặc trưng hình ảnh (Keypoint NIFT)
14. AutoFlip – Tự động cắt và điều chỉnh khung hình video
15. MediaSequence – Chuỗi dữ liệu video có cấu trúc
16. YouTube-8M – Tập dữ liệu video quy mô lớn từ YouTube



Hình 4.2 Minh hoạ các giải pháp của mô hình MediaPipe

4.1.1 Phân tích cử chỉ tay

Trong dự án lần này, chúng ta sử dụng một giải pháp trong các giải pháp của MediaPipe đó là nhận diện và theo dõi bàn tay để điều khiển các thiết bị như đèn, quạt, TV, và nhạc thông qua các thao tác giơ ngón tay. Hệ thống được tích hợp với camera thời gian thực và cung cấp giao diện điều khiển thông qua Server.

Hệ thống nhận diện cử chỉ tay bằng MediaPipe hoạt động theo quy trình gồm nhiều bước, từ thu nhận hình ảnh đến phân tích cử chỉ và đưa ra hành động tương ứng. Cụ thể như sau:

Bước 1: Thu nhận hình ảnh đầu vào

- Sử dụng camera (webcam) để ghi lại hình ảnh thời gian thực.
- Hình ảnh được truyền trực tiếp vào pipeline xử lý của mô hình MediaPipe.

Bước 2: Phát hiện bàn tay (Hand Detection)

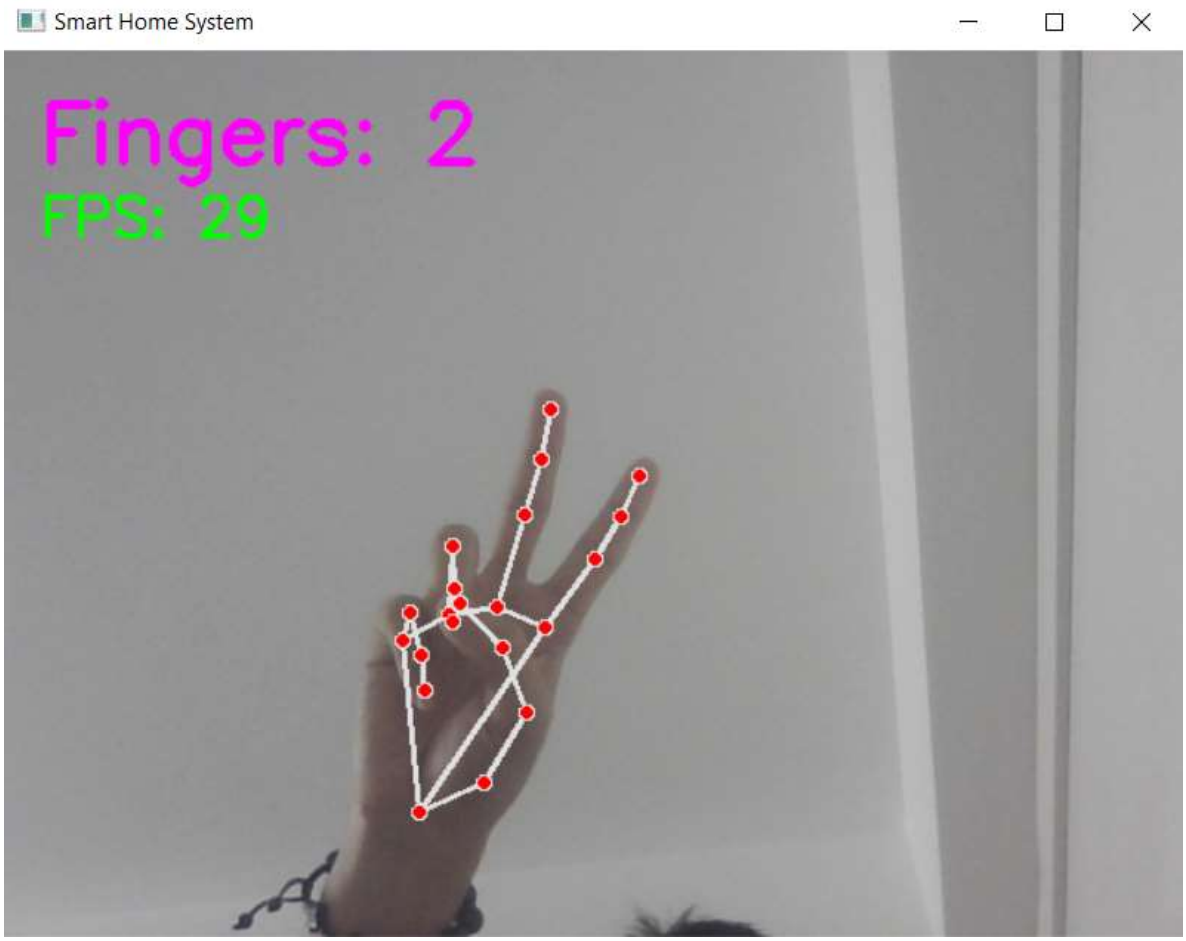
- MediaPipe sử dụng mô hình học sâu để phát hiện bàn tay trong khung hình.
- Nếu phát hiện có bàn tay, hệ thống sẽ cắt vùng chứa bàn tay và chuyển sang bước tiếp theo.

Bước 3: Trích xuất điểm đặc trưng (Hand Landmarks)

- MediaPipe Hands cung cấp 21 điểm landmark trên mỗi bàn tay, bao gồm:
    - 5 đầu ngón tay
    - Các khớp giữa ngón tay
    - Cổ tay (wrist)
- Mỗi điểm landmark có tọa độ (x, y, z) trong không gian ảnh.

Bước 4: Phân tích cử chỉ dựa trên số ngón tay giơ lên

- Hệ thống xác định trạng thái của từng ngón tay (giơ lên hay gập lại) bằng cách so sánh vị trí tương đối giữa các điểm landmark:
    - Ví dụ: nếu đầu ngón tay nằm trên khớp giữa (theo trục y), thì ngón tay được coi là giơ lên.
- Đếm tổng số ngón tay giơ lên để ánh xạ với một hành động cụ thể.



Hình 4.3 Mô tả cách hoạt động phát hiện cử chỉ tay của mô hình MediaPipe

5. Ánh xạ cử chỉ → hành động

Mỗi số lượng ngón tay giơ lên tương ứng với một lệnh điều khiển:

Bảng 4.1 Ánh xạ giữa cử chỉ tay và hành động điều khiển thiết bị

Số ngón tay	Thiết bị	Hành động
1	Đèn phòng khách	Bật đèn
2	Đèn phòng khách	Tắt đèn
3	Đèn phòng ngủ	Bật đèn
4	Đèn phòng ngủ	Tắt đèn

4.1.2 Tích hợp với camera an ninh và Server

- Để đưa hình ảnh làm đầu vào cho MediaPipe xử lý, ta sử dụng luồng hình ảnh được trích xuất từ camera an ninh trong nhà, mô hình sẽ xử lý các hình ảnh này từ đó gửi tín hiệu về Server để điều khiển các thiết bị.
- Luồng xử lý:

1. Camera ghi nhận hình ảnh.
2. MediaPipe xử lý và nhận diện cử chỉ.

3. Server gửi lệnh điều khiển tương ứng.

4.2 Điều khiển thiết bị qua cử chỉ

4.2.1 Gửi lệnh điều khiển đến Home Assistant

Trong hệ thống điều khiển thiết bị thông minh, Server đóng vai trò trung gian giữa hệ thống nhận diện cử chỉ tay và nền tảng Home Assistant. Khi người dùng thực hiện một cử chỉ tay cụ thể (ví dụ: giơ 1 ngón tay để bật đèn), Server sẽ tiếp nhận tín hiệu này và gửi yêu cầu HTTP (REST API) đến Home Assistant để thực hiện hành động tương ứng. Home Assistant cần được cấu hình API access và tạo Long-Lived Access Token để xác thực.

4.2.2 Cập nhật trạng thái thiết bị trên giao diện

Trạng thái thiết bị được xây dựng trên giao diện theo thời gian thực, nhằm đảm bảo sự đồng bộ hai chiều giữa Server và giao diện người dùng. Tương tự như thiết kế hệ thống nhận diện khuôn mặt, các trạng thái của thiết bị cũng được sử dụng giao thức Websocket để truyền thông hai chiều cho phép server và client trao đổi dữ liệu liên tục mà không cần gửi lại yêu cầu HTTP. Khi trạng thái thiết bị thay đổi (do cử chỉ, giọng nói hoặc thao tác từ giao diện), server sẽ gửi thông báo cập nhật đến tất cả client đang kết nối.

- Từ hệ thống → người dùng: Khi thiết bị được điều khiển bởi cử chỉ hoặc giọng nói, Server nhận phản hồi từ Home Assistant và gửi sự kiện thông qua giao thức WebSocket đến giao diện.
- Từ người dùng → hệ thống: Khi người dùng nhấn nút điều khiển trên giao diện, yêu cầu được gửi đến Server, sau đó cập nhật trạng thái thiết bị và phản hồi lại.

4.2.3 Giao diện trực quan và phản hồi thời gian thực

Thành phần giao diện

a) Hình ảnh camera với overlay cử chỉ

- Hiển thị luồng video trực tiếp từ camera.

b) Tên thiết bị và trạng thái hiện tại

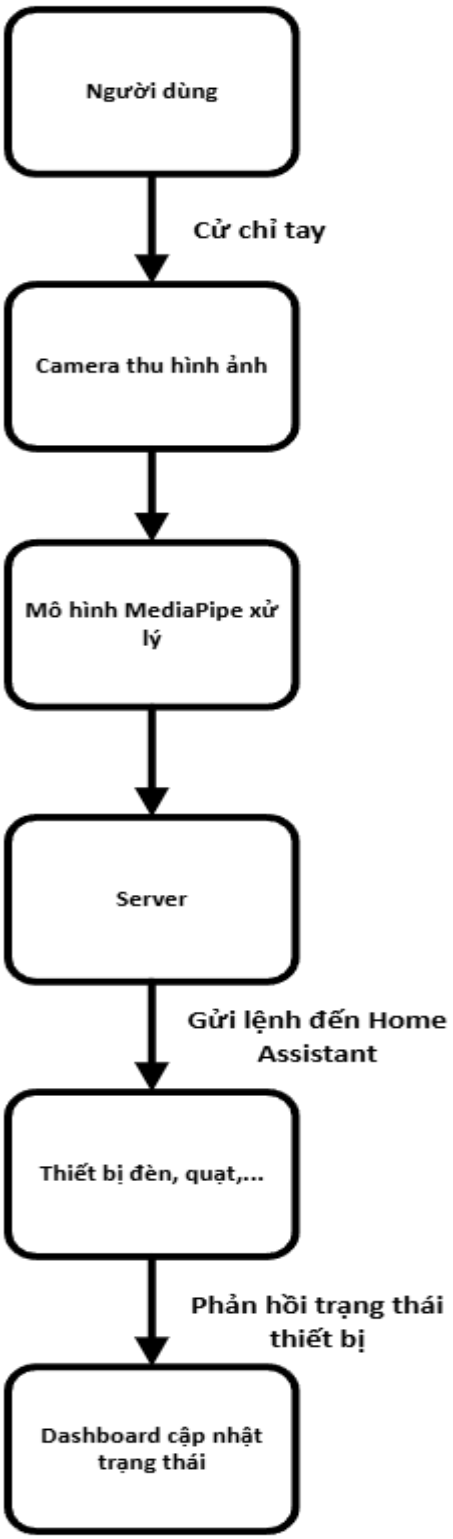
- Mỗi thiết bị có một khối hiển thị riêng gồm:
    - Tên thiết bị (ví dụ: “Đèn phòng khách”)
    - Trạng thái hiện tại (Bật/Tắt)
    - Nút điều khiển thủ công

c) Phản hồi thời gian thực

- Khi có thay đổi trạng thái (từ cử chỉ, giọng nói hoặc thao tác thủ công), giao diện cập nhật theo thời gian thực.

- Trạng thái thiết bị được đồng bộ với Server thông qua giao thức WebSocket.

Dưới đây là sơ đồ luồng xử lý hệ thống điều khiển bằng cử chỉ tay:



Hình 4.4 Sơ đồ luồng xử lý hệ thống điều khiển bằng cử chỉ tay

4.3 Kết luận chương 4

Trong chương này, hệ thống điều khiển thiết bị thông minh bằng cử chỉ tay đã được thiết kế và triển khai thành công, sử dụng mô hình MediaPipe của Google để nhận diện bàn tay và phân tích số lượng ngón tay giơ lên. Giải pháp cho thấy tính hiệu quả cao khi hoạt động thời gian thực, độ chính xác tốt và có khả năng mở rộng linh hoạt.

Việc tích hợp giữa MediaPipe – Server – Home Assistant – giao diện giúp tạo ra một hệ sinh thái đồng bộ, nơi người dùng có thể tương tác với thiết bị thông minh một cách tự nhiên, không cần tiếp xúc vật lý hay can thiệp trực tiếp vào hệ thống.

Giao diện được thiết kế trực quan, cho phép theo dõi trạng thái thiết bị và hình ảnh camera theo thời gian thực thông qua WebSocket, từ đó nâng cao trải nghiệm sử dụng.

Từ kết quả triển khai, có thể khẳng định rằng giải pháp điều khiển bằng cử chỉ tay là một hướng đi hiện đại, hiệu quả và đầy tiềm năng cho các hệ thống nhà thông minh thế hệ mới.

CHƯƠNG V: KẾT LUẬN

5.1. Kết quả đạt được

Sau quá trình nghiên cứu, thiết kế và triển khai hệ thống nhà thông minh tích hợp nhận diện khuôn mặt, điều khiển bằng cử chỉ tay và giao tiếp thời gian thực, nhóm thực hiện đã đạt được những kết quả nổi bật sau:

5.1.1 Hệ thống hoạt động ổn định, nhận diện chính xác

- Các thành phần chính như nhận diện khuôn mặt (FaceNet + MTCNN), nhận diện cử chỉ tay (MediaPipe), và xử lý trung gian (Server) đều hoạt động ổn định và cho kết quả chính xác trong môi trường thử nghiệm thực tế.
- Hệ thống có khả năng nhận diện khuôn mặt và cử chỉ tay trong thời gian thực, với độ trễ thấp, cho phép thực hiện các hành động điều khiển thiết bị gần như ngay lập tức.
- Cảm biến, camera và các thiết bị đầu ra (đèn, quạt, servo...) đều tương tác mượt mà với hệ thống điều khiển trung tâm.

5.1.2 Giao diện thân thiện, cập nhật thời gian thực

Giao diện người dùng được xây dựng trên nền web với hình ảnh trực quan, chia thành các vùng hiển thị rõ ràng. Các trạng thái thiết bị và hình ảnh camera được cập nhật liên tục thông qua giao thức WebSocket, cho phép người dùng theo dõi hệ thống một cách hiệu quả và tương tác thuận tiện. Các thao tác điều khiển như bật/tắt thiết bị có thể thực hiện thông qua dashboard hoặc cử chỉ tay, tất cả đều phản hồi theo thời gian thực.

5.1.3 Tích hợp nhiều công nghệ hiện đại: Machine Learning, IoT, WebSocket

- Ứng dụng thành công các công nghệ Machine Learning như MediaPipe (cử chỉ tay), MTCNN & FaceNet (khuôn mặt).
- Kết hợp linh hoạt với nền tảng Home Assistant – hệ điều hành mã nguồn mở dành cho nhà thông minh – để thực thi các hành động điều khiển thiết bị.
- Giao tiếp thời gian thực giữa frontend (dashboard) và backend (Server) thông qua giao thức WebSocket.
- Hệ thống sử dụng vi điều khiển ESP32 để đọc cảm biến (như siêu âm HC-SR04) và điều khiển servo.
- Hệ thống áp dụng REST API để gửi lệnh điều khiển từ Server đến Home Assistant, đảm bảo khả năng mở rộng và tích hợp với các hệ thống khác.

5.1.4 Khả năng hoạt động đa kênh điều khiển

- Hệ thống có thể nhận lệnh điều khiển thiết bị từ nhiều nguồn: nhận diện khuôn mặt, cử chỉ tay, nút bấm vật lý, và giao diện.
- Dữ liệu từ các kênh điều khiển được thống nhất qua Server, đảm bảo tính đồng bộ và tránh xung đột điều khiển.
- Trạng thái thiết bị luôn được cập nhật đầy đủ và chính xác trên giao diện người dùng.

5.1.5 Tính ứng dụng thực tiễn cao

Đề tài thể hiện rõ tính ứng dụng thực tiễn trong bối cảnh nhà thông minh đang ngày càng phát triển tại Việt Nam. Trong xu thế hiện đại hóa và tự động hóa đời sống, việc tích hợp các công nghệ điều khiển thông minh vào không gian sống không còn là một lựa chọn xa vời, mà đã trở thành nhu cầu tất yếu.

Hệ thống được xây dựng không chỉ khẳng định được năng lực kỹ thuật, mà còn cho thấy tiềm năng ứng dụng rộng rãi. Với chi phí triển khai hợp lý, khả năng mở rộng linh hoạt và độ ổn định cao, hệ thống có thể dễ dàng áp dụng trong nhiều môi trường như gia đình, văn phòng, lớp học hoặc cơ sở y tế. Đặc biệt, tính năng điều khiển không chạm bằng cử chỉ mang lại trải nghiệm tương tác tiện lợi, an toàn, phù hợp với bối cảnh phòng chống dịch bệnh hoặc hỗ trợ các đối tượng đặc biệt như người cao tuổi, trẻ nhỏ – những người gặp khó khăn khi thao tác với thiết bị truyền thống.

5.2. Hạn chế của hệ thống

Mặc dù hệ thống đạt được nhiều kết quả khả quan, vẫn còn tồn tại một số hạn chế:

5.2.1 Chưa hỗ trợ nhiều người dùng và phân quyền truy cập

Hiện tại hệ thống chỉ lưu trữ embedding của một số lượng nhỏ người dùng và chưa có cơ chế phân quyền. Điều này gây hạn chế khi muốn triển khai trong môi trường có nhiều thành viên, ví dụ như gia đình đông người hoặc văn phòng.

Người dùng chưa thể tạo tài khoản cá nhân và hệ thống cũng chưa phân biệt vai trò quản trị viên hay khách. Việc bổ sung phân quyền sẽ tăng tính bảo mật và cá nhân hóa hành vi điều khiển.

5.2.2 Điều khiển bằng giọng nói chưa được triển khai hoàn chỉnh

- Mặc dù kiến trúc hệ thống hỗ trợ mở rộng tính năng điều khiển bằng giọng nói, nhưng chức năng này chưa được tích hợp trong bản thử nghiệm hiện tại.
- Chưa có xử lý ngôn ngữ tự nhiên (NLP) hoặc nhận diện giọng nói tiếng Việt.

5.2.3 Phụ thuộc vào kết nối nội bộ hoặc internet

- Một số chức năng hoặc truyền dữ liệu WebSocket phụ thuộc vào mạng LAN hoặc WiFi ổn định.

- Nếu mất mạng hoặc Server ngừng hoạt động, hệ thống không có chế độ fallback hoặc xử lý ngoại lệ tạm thời.

5.2.4 Chưa triển khai mô hình Machine Learning tùy chỉnh

- Các mô hình sử dụng (MediaPipe, FaceNet) đều là mô hình tiền huấn luyện (pre-trained), chưa tối ưu hóa riêng cho môi trường sử dụng cụ thể.
- Hệ thống chưa huấn luyện lại mô hình (fine-tune) theo tập dữ liệu người dùng thực tế.

5.2.5 Khả năng xử lý nhiều thiết bị cùng lúc còn hạn chế

- Số lượng thiết bị điều khiển chưa vượt quá 5 thiết bị cùng lúc.
- Giao diện người dùng chưa có nhóm thiết bị hoặc phân khu điều khiển theo phòng.

5.3. Hướng phát triển

5.3.1 Tích hợp điều khiển bằng giọng nói

Trong tương lai, việc bổ sung tính năng điều khiển thiết bị thông minh bằng giọng nói sẽ giúp nâng cao trải nghiệm người dùng và tăng tính linh hoạt trong điều khiển. Người dùng có thể sử dụng khẩu lệnh tự nhiên như "Bật đèn phòng khách" hoặc "Tắt quạt phòng ngủ" mà không cần thao tác trực tiếp. Điều này đặc biệt hữu ích trong các tình huống tay bị bận, người khuyết tật hoặc khi người dùng ở xa tầm nhìn camera.

Việc tích hợp có thể thực hiện thông qua Google Assistant, hoặc sử dụng các thư viện mã nguồn mở như SpeechRecognition, Vosk hoặc Whisper. Một thách thức cần giải quyết là xử lý tiếng Việt với nhiều phương ngữ khác nhau. Do đó, hệ thống cần có bước chuyển đổi giọng nói thành văn bản (speech-to-text), sau đó ánh xạ lệnh đã chuẩn hóa sang hành động tương ứng trong Home Assistant.

5.3.2 Phát triển ứng dụng di động

Song song với giao diện người dùng trên trình duyệt, việc phát triển một ứng dụng di động (Android/iOS) sẽ giúp người dùng dễ dàng điều khiển hệ thống từ bất cứ đâu. Ứng dụng nên có các tính năng như:

- Giao diện điều khiển thiết bị theo phòng/khu vực
- Nhận thông báo đẩy (push notification) khi có sự kiện xảy ra: mở cửa, phát hiện người lạ...
- Hỗ trợ điều khiển từ xa thông qua mạng Internet
- Đăng nhập và phân quyền người dùng

Điều này không chỉ mở rộng khả năng sử dụng mà còn giúp hệ thống trở nên chuyên nghiệp và thương mại hóa được.

5.3.3 Phân quyền người dùng và bảo mật

Để hệ thống có thể phục vụ nhiều người cùng lúc và tránh rủi ro truy cập trái phép, cần bổ sung cơ chế phân quyền người dùng. Các vai trò như quản trị viên (admin), thành viên (user), khách (guest) cần được phân biệt rõ ràng về quyền hạn. Ví dụ, admin có thể thay đổi cấu hình thiết bị, trong khi user chỉ được phép bật/tắt thiết bị, còn guest chỉ được xem trạng thái.

Ngoài ra, các biện pháp bảo mật như xác thực bằng token, mã hóa đường truyền (HTTPS), kiểm soát truy cập theo IP cũng cần được triển khai nhằm tăng cường tính an toàn của hệ thống.

5.3.4 Triển khai mô hình Machine Learning riêng

Hiện tại, hệ thống sử dụng các mô hình Machine Learning tiền huấn luyện (pre-trained). Trong tương lai, nhóm có thể xây dựng tập dữ liệu riêng gồm khuôn mặt và cử chỉ tay của người dùng thực tế để huấn luyện lại mô hình (fine-tuning). Điều này sẽ giúp tăng độ chính xác, giảm sai số trong điều kiện ánh sáng yếu hoặc môi trường nhiễu.

Ngoài ra, có thể áp dụng các kỹ thuật tăng cường dữ liệu (data augmentation), học chuyển giao (transfer learning) để cải thiện chất lượng mô hình mà không cần tập dữ liệu quá lớn.

5.3.5 Tối ưu hệ thống cho môi trường lớn hơn

Nếu mở rộng triển khai tại văn phòng, bệnh viện, trường học… hệ thống cần hỗ trợ số lượng lớn thiết bị và người dùng. Giao diện dashboard nên được phân khu rõ ràng theo từng phòng hoặc tầng lầu. Backend cần tối ưu hiệu năng xử lý đồng thời nhiều luồng dữ liệu, đảm bảo tính ổn định khi tải cao.

Các giải pháp mở rộng như phân cụm thiết bị, ưu tiên xử lý tác vụ quan trọng, hoặc ghi log thông minh theo thời gian thực sẽ giúp hệ thống vận hành hiệu quả hơn trong môi trường phức tạp.

TÀI LIỆU THAM KHẢO

[1] F. Schroff, D. Kalenichenko, và J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in *Proc. IEEE CVPR*, Jun. 2015, pp. 815–823.

[2] L. Dulčić, "Face Recognition with FaceNet and MTCNN," Ars Futura Blog, Nov. 15, 2019. [Online]. Available: https://arsfutura.com/blog/face-recognition-with-facenet-and-mtcnn. [Accessed: Jun. 12, 2025].

[3] N. C. Thắng. "[Mì AI] Nhận dạng khuôn mặt ngon hơn với MTCNN và Facenet" *Youtube,* Sep. 20, 2019. [Video file]. Available: https://www.youtube.com/watch?v=snmYELRQvME. [Accessed: May. 28, 2025].

[4] N. C. Thắng, "[Face Recog 2.0] Nhận diện khuôn mặt trong video bằng MTCNN và Facenet," Mì AI Blog, Sep. 11, 2019. [Online]. Available: https://miai.vn/2019/09/11/face-recog-2-0-nhan-dien-khuon-mat-trong-video-bang-mtcnn-va-facenet/ [Accessed: May. 28, 2025].

[5] Quang Tran, "Nhận diện khuôn mặt với mạng MTCNN và FaceNet (Phần 1)," Viblo, July. 15, 2021. [Online]. Available: https://viblo.asia/p/nhan-dien-khuon-mat-voi-mang-mtcnn-va-facenet-phan-1-Qbq5QDN4lD8 [Accessed: Jun. 12, 2025].

[6] Roboflow, "What is MediaPipe?," Roboflow Blog, Apr. 10, 2024. [Online]. Available: https://blog.roboflow.com/what-is-mediapipe/#mediapipe-use-cases [Accessed: Jun. 10, 2025].

[7] Kukil, P.Durai, "Introduction to MediaPipe," LearnOpenCV, March. 1, 2022**.** [Online]. Available: https://learnopencv.com/introduction-to-mediapipe/. [Accessed: Jun. 12, 2025].

[8] Vohungvi, "Phát hiện cử chỉ chỉ bằng MediaPipe," Thị Giác Máy Tính. Aug. 30, 2022. [Online]. Available: https://thigiacmaytinh.com/phat-hien-cu-chi-co-the-bang-mediapipe/ [Accessed: Jun. 12, 2025].

[9] ESPBoards, "NodeMCU-32S," ESPBoards.dev. [Online]. Available: https://www.espboards.dev/esp32/nodemcu-32s/. [Accessed: Jun. 15, 2025].

[10] Lumi, "Home Assistant là gì? 3 ứng dụng hữu ích cho nhà thông minh của bạn. 11, Oct, 2024. [Online]. Available: https://lumi.vn/home-assistant-la-gi.html [Accessed: Jun. 11, 2025].

[11] geeksforgeeks, "What is web socket and how it is different from the HTTP?", 04, Apr, 2025 [Online]. Available: https://www.geeksforgeeks.org/what-is-web-socket-and-how-it-is-different-from-the-http/ [Accessed: Jun. 04, 2025].

[12] Mamtaz Alam, “Gas Leakage Detector with Email Alert Notification using ESP32”, 27, Nov, 2023 [Online]. Available: https://how2electronics.com/gas-leakage-detector-email-alert-notification-esp32/

[13] trvqhuy, “Công cụ theo dõi điện năng tiêu thụ từ EVN Việt Nam dành cho HomeAssistant”, 11, Feb, 2025 [Online]. Available: https://github.com/trvqhuy/nestup_evn [Accessed: Jun. 04, 2025].

Các trang Web tham khảo :

- https://fptshop.com.vn/tin-tuc/danh-gia/python-flask-176037
- https://www.home-assistant.io/docs/glossary/

PHỤ LỤC

1. Đường dẫn truy cập dashboard:

https://smarthome.smarthomediy.click

2. Arduino code

```
#include <WiFi.h>
#include <WebServer.h>
#include <ESP32Servo.h>
#include <HTTPClient.h>

const char* ssid = "Your wifi";
const char* password = "pass";
const char* flask_host = "IpServer";
const int flask_port = 5000;


#define TRIG_PIN 16
#define ECHO_PIN 17
#define SERVO_PIN 32
#define LED 2
#define BUZZER_PIN 23
#define MQ2_PIN 33
#define GAS_THRESHOLD 2500


WebServer server(80);
Servo doorServo;

long distanceThreshold = 10; // cm

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500); Serial.print(".");
  }
  Serial.println("WiFi connected");
  Serial.println(WiFi.localIP());

  // Gửi đăng ký IP đến Flask server
  HTTPClient http;
  http.begin(String("http://") + flask_host + ":" + flask_port + "/register_esp");
  http.addHeader("Content-Type", "application/json");
  int code = http.POST("{}");
```

```
  if (code > 0) {
    Serial.println(" Đã đăng ký IP với Flask Server.");
  } else {
    Serial.printf(" Lỗi khi đăng ký IP: %s\n", http.errorToString(code).c_str());
  }
  http.end();


  pinMode(TRIG_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(LED, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(MQ2_PIN, INPUT);


  doorServo.attach(SERVO_PIN);
  doorServo.write(0); // Đóng cửa

  server.on("/unlock", HTTP_POST, []() {
    openDoor();
    server.send(200, "text/plain", "OK");
  });

  server.on("/alarm", HTTP_POST, []() {
    Serial.println("Nhận lệnh báo động từ Flask!");
    for (int i = 0; i < 5; i++) {
      digitalWrite(BUZZER_PIN, HIGH);
      delay(200);
      digitalWrite(BUZZER_PIN, LOW);
      delay(200);
    }
    server.send(200, "text/plain", "ALARM DONE");
  });


  server.begin();
}
int lastMq2State = -1;  // Khởi tạo với giá trị không hợp lệ

void loop() {
  server.handleClient();

  int mq2State = digitalRead(MQ2_PIN);
  if (mq2State != lastMq2State) {
    Serial.printf(" Giá trị MQ2 thay đổi: %d\n", mq2State);
    lastMq2State = mq2State;
  }

  if (mq2State == LOW) {
```

```
    Serial.println(" CẢNH BÁO: Phát hiện khí gas!");


    // Gửi POST về Flask để cảnh báo
    HTTPClient http;
    http.begin(String("http://") + flask_host + ":" + flask_port + "/gas_alert");
    http.addHeader("Content-Type", "application/json");
    http.POST("{\"gas\": true}");
    http.end();

    // Còi báo động
    digitalWrite(BUZZER_PIN, HIGH);
    delay(3000);
    digitalWrite(BUZZER_PIN, LOW);

  }
  if (detectPerson()) {
    Serial.println(" Có người trước cửa...");

    // Đợi 3 giây trước khi gửi POST đến Flask
    delay(3000);

    // Gửi tín hiệu đến Flask server
    HTTPClient http;
    http.begin(String("http://") + flask_host + ":" + flask_port + "/trigger_unlock");
    http.addHeader("Content-Type", "application/json");
    int code = http.POST("{}");
    if (code > 0) {
      Serial.printf(" Gửi thành công! Mã phản hồi: %d\n", code);
      String response = http.getString();
      Serial.println(" Phản hồi: " + response);
    } else {
      Serial.printf(" Lỗi gửi: %s\n", http.errorToString(code).c_str());
    }
    http.end();

    // Nhấp nháy LED báo hiệu
    digitalWrite(LED, HIGH); delay(100);
    digitalWrite(LED, LOW); delay(100);
  }

  delay(1000);
}



bool detectPerson() {
  digitalWrite(TRIG_PIN, LOW); delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH); delayMicroseconds(10);
```

```
  digitalWrite(TRIG_PIN, LOW);
  long duration = pulseIn(ECHO_PIN, HIGH);
  long distance = duration * 0.034 / 2;
  return (distance > 0 && distance < distanceThreshold);
}

void openDoor() {
  Serial.println("Mở cửa...");
  doorServo.write(90); // quay 90 độ mở cửa
  delay(5000);
  Serial.println("Đóng cửa...");        // giữ cửa mở 5s
  doorServo.write(0);  // đóng lại
}
```

3. Code Flask Server, cử chỉ tay và bảo mật nhận diện khuôn mặt

Truy cập : https://github.com/minh838699/smart-home-project?fbclid=IwY2xjawK8i-lleHRuA2FlbQIxMQABHk36Sslw_RawzeFYAP9Lsl9oFSH4F5EdMShfdg1GZozAujx2Rcl1E5ak4kyF_aem_XbIIRsL1xxJhN0gyOvfRig