

# Phiếu giao nhiệm vụ

## 1. Thông tin sinh viên

- Họ và tên: Lương Đào Quang Anh
- Số điện thoại: 0358289688
- Email: anh.ldq176687@sis.hust.edu.vn
- Lớp: Application Specialist-K62
- Hệ đào tạo: Kỹ sư

## 2. Mục đích và nội dung đồ án tốt nghiệp

Nghiên cứu và xây dựng ứng dụng hỗ trợ điều khiển camera bằng cử chỉ tay với các mục đích sau:

- Kết nối và giao tiếp được với máy chủ.
- Ứng dụng có thể xem được dòng hình ảnh đã qua xử lý.
- Sử dụng cử chỉ tay điều khiển camera trên ứng dụng máy tính và thiết bị biên Jetson Nano.

## 3. Các nhiệm vụ cụ thể của đồ án tốt nghiệp

- Xây dựng ứng dụng xem được dòng hình ảnh đã qua xử lý.
- Xây dựng mô-đun phát hiện cử chỉ tay.
- Điều khiển camera trên thiết bị biên Jetson Nano sử dụng cử chỉ tay.
- Xây dựng bộ dữ liệu để huấn luyện và đánh giá mô hình cử chỉ tay.

## 4. Lời cam đoan

Tôi – *Lương Đào Quang Anh* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS.Đặng Tuấn Linh. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

Hà Nội, tháng 08 năm 2022

Tác giả ĐATN

Lương Đào Quang Anh

**5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và  
cho phép bảo vệ**

Hà Nội, tháng 08 năm 2022

Giáo viên hướng dẫn

Đặng Tuấn Linh

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**Ứng dụng hỗ trợ điều khiển camera bằng cử chỉ tay**

**Lương Đào Quang Anh**  
anh.ldq176687@sis.hust.edu.vn

**Ngành Công nghệ thông tin**

**Giảng viên hướng dẫn:** TS. Đặng Tuấn Linh

Chữ ký GVHD

**Khoa:** Kỹ thuật máy tính

**Trường:** Công nghệ thông tin và Truyền thông

**HÀ NỘI, 08/2022**

# LỜI CẢM ƠN

Lời đầu tiên, tôi xin được cảm ơn sâu sắc nhất đến thầy Đặng Tuấn Linh đã luôn quan tâm, tận tình hướng dẫn và hỗ trợ tôi hoàn thành đồ án tốt nghiệp này. Nhờ sự hướng dẫn của thầy, tôi đã có thể giải quyết những khó khăn xảy ra về cả ý tưởng lẫn công nghệ. Do đó, tôi đã có thể hoàn thành đồ án tốt nghiệp đúng với mong đợi của bản thân.

Tiếp theo, tôi xin gửi lời cảm ơn đến trường Đại học Bách Khoa nói chung và trường Công nghệ thông tin và Truyền thông nói riêng đã luôn tạo điều kiện học tập và truyền đạt cho tôi những kiến thức, kinh nghiệm bổ ích trong học tập và trong cuộc sống sau này.

Cuối cùng, tôi muốn bày tỏ lòng biết ơn đến gia đình, bạn bè đã luôn ủng hộ và giúp đỡ tôi trong quá trình thực hiện đồ án. Đây là nguồn động lực to lớn giúp tôi vững bước trong cả hiện tại và tương lai.

Tôi xin chân thành cảm ơn!

# TÓM TẮT NỘI DUNG ĐỒ ÁN

Cùng với sự phát triển của khoa học và kỹ thuật, việc ứng dụng thị giác máy tính vào camera để hỗ trợ con người ngày càng phổ biến. Những chiếc camera có chức năng thông minh có thể giúp con người rất nhiều việc như tự động quản lý an ninh, hỗ trợ giám sát giao thông công cộng. Để giúp quản lý camera, một hệ thống giúp con người điều khiển chức năng của các camera này là cần thiết.

Trên thế giới cũng đã có những dịch vụ hỗ trợ người dùng có thể thao tác với máy tính và điện thoại để điều khiển các chức năng của camera. Tuy nhiên trong thực tế, không phải mọi lúc có thể sở hữu và sử dụng máy tính hoặc điện thoại. Hơn nữa, đôi khi camera được đặt ở những nơi quá cao hoặc quá hẹp, sẽ khó thao tác bằng bàn phím hay chuột khi muốn điều khiển camera. Đối với người ít tiếp xúc với công nghệ, việc sử dụng máy tính hoặc điện thoại cũng gặp những khó khăn nhất định.

Để giải quyết vấn đề trên, tôi đề xuất giải pháp sử dụng mô hình thị giác máy tính giúp camera có nhận diện được chỉ tay của con người, từ đó sử dụng cử chỉ tay để điều khiển các chức năng camera. Kết thúc đồ án, tôi đã xây dựng được một ứng dụng có thể điều khiển các chức năng của camera bằng chỉ tay. Không những có thể điều khiển trên máy tính thông thường, người dùng còn có thể điều khiển bằng cử chỉ tay thông qua một thiết bị tiêu tốn ít tài nguyên Jetson Nano.

## **ABSTRACT**

Along with the development of science and technology, the use of computer vision in cameras to assist people is increasingly popular. Cameras with intelligent functions can help people with many things such as automatically managing security, supporting public traffic monitoring. To help manage the cameras, a system that helps humans control the functions of these cameras is needed.

In the world, there are also services that support users who can manipulate computers, phones to control camera functions. In reality, however, it is not always possible to own and use a computer or a phone. Moreover, sometimes the camera is placed in places that are too high or too narrow, it will be difficult to manipulate with the keyboard or mouse when you want to control the camera. For people with little exposure to technology, using a computer or phone also encounters certain difficulties.

To solve the above problem, I propose a solution to use a computer vision model to help the camera recognize human hand gestures, thereby using hand gestures to control camera functions. By the end of the project, I have built an application that can control the camera's functions by hand gestures. Not only can it be controlled on computer, users can also control it with hand gestures through a device that consumes less resources Jetson Nano.

## MỤC LỤC

<b>CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI.....</b>	<b>1</b>
1.1 Đặt vấn đề.....	1
1.2 Mục tiêu và phạm vi đề tài.....	2
1.3 Định hướng giải pháp.....	3
1.4 Bố cục đồ án .....	4
<b>CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU.....</b>	<b>5</b>
2.1 Khảo sát hiện trạng .....	5
2.2 Tổng quan chức năng .....	6
2.2.1 Biểu đồ use case tổng quát .....	6
2.2.2 Biểu đồ use case phân rã xem dòng hình ảnh đã qua xử lý.....	7
2.2.3 Biểu đồ use case phân rã xử lý hình ảnh .....	8
2.2.4 Biểu đồ use case phân rã quản lý thông báo và dòng hình ảnh đã lưu .....	9
2.2.5 Biểu đồ use case phân rã quản lý tài khoản .....	10
2.3 Đặc tả chức năng .....	11
2.3.1 Đặc tả use case đăng nhập.....	11
2.3.2 Đặc tả use case xem dòng hình ảnh đã qua xử lý .....	12
2.3.3 Đặc tả use case xử lý hình ảnh.....	13
2.3.4 Đặc tả use case xem thông báo đã lưu .....	14
2.4 Thiết kế ứng dụng .....	14
2.4.1 Thiết kế luồng hoạt động .....	16
2.4.2 Thiết kế giao diện .....	24
2.5 Yêu cầu phi chức năng .....	25
2.5.1 Yêu cầu về kỹ thuật .....	25

2.5.2 Yêu cầu về giao diện người dùng .....	25
<b>CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG.....</b>	<b>26</b>
3.1 Công nghệ sử dụng cho khối ứng dụng máy tính.....	26
3.1.1 Electron .....	26
3.1.2 NodeJS .....	27
3.1.3 Python-Shell.....	27
3.1.4 Firebase .....	28
3.1.5 OpenCV.....	28
3.1.6 TensorFlow.....	29
3.1.7 MediaPipe .....	29
3.1.8 gTTS .....	30
3.1.9 EfficientNet.....	30
3.1.10 Nơi sử dụng trong ứng dụng .....	30
3.1.11 Lý do sử dụng .....	31
3.2 Công nghệ sử dụng cho khối điều khiển camera trên thiết bị Jetson Nano ..	31
3.2.1 FastApi .....	31
3.2.2 MobileNetV2.....	31
3.3 Phương pháp xây dựng .....	32
3.3.1 Mô-đun xây dựng ứng dụng máy tính.....	32
3.3.2 Mô-đun nhận diện cử chỉ tay .....	33
3.3.3 Mô-đun điều khiển camera trên thiết bị Jetson Nano .....	34
<b>CHƯƠNG 4. THỰC NGHIỆM VÀ ĐÁNH GIÁ .....</b>	<b>35</b>
4.1 Xây dựng ứng dụng.....	35
4.1.1 Thư viện và công cụ sử dụng.....	35
4.1.2 Triển khai.....	36
4.1.3 Kết quả đạt được .....	37

4.1.4 Minh họa các chức năng chính .....	37
4.2 Kiểm thử và đánh giá .....	45
4.2.1 Kểm thử mô hình cử chỉ tay .....	45
4.2.2 Kiểm thử ứng dụng máy tính .....	45
4.2.3 Đánh giá .....	49
<b>CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT.....</b>	<b>50</b>
5.1 Mô hình nhận diện cử chỉ tay dựa trên các điểm chính trên bàn tay kết hợp mô hình học sâu.....	50
5.1.1 Đặt vấn đề .....	50
5.1.2 Giải pháp .....	50
5.1.3 Kết quả đạt được .....	51
5.2 Xây dựng bộ dữ liệu để huấn luyện mô hình nhận diện cử chỉ tay .....	52
5.2.1 Đặt vấn đề .....	52
5.2.2 Giải pháp .....	52
5.2.3 Kết quả đạt được .....	53
5.3 Xây dựng ứng dụng điều khiển các chức năng camera .....	53
5.3.1 Đặt vấn đề .....	53
5.3.2 Giải pháp .....	54
5.3.3 Kết quả đạt được .....	54
5.4 Điều khiển camera sử dụng cử chỉ tay thông qua Jetson Nano .....	55
5.4.1 Đặt vấn đề .....	55
5.4.2 Giải pháp .....	56
5.4.3 Kết quả đạt được .....	56
<b>CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....</b>	<b>57</b>
6.1 Kết luận .....	57
6.2 Hướng phát triển.....	57

**TÀI LIỆU THAM KHẢO..... 60**

## DANH MỤC HÌNH VẼ

Hình 2.1	Biểu đồ use case tổng quan . . . . .	6
Hình 2.2	Biểu đồ use case phân rã xem dòng hình ảnh đã qua xử lý . . . . .	7
Hình 2.3	Biểu đồ use case phân rã xử lý hình ảnh . . . . .	8
Hình 2.4	Biểu đồ use case phân rã quản lý thông báo và dòng hình ảnh đã lưu . . . . .	9
Hình 2.5	Biểu đồ use case phân rã quản lý tài khoản . . . . .	10
Hình 2.6	Tổng quan luồng hoạt động của ứng dụng . . . . .	15
Hình 2.7	Luồng đăng nhập . . . . .	16
Hình 2.8	Luồng xem dòng hình ảnh đã xử lý . . . . .	17
Hình 2.9	Luồng xử lý ảnh . . . . .	18
Hình 2.10	Luồng điều khiển bằng cử chỉ tay . . . . .	19
Hình 2.11	Luồng xem thông tin tài khoản . . . . .	20
Hình 2.12	Luồng đổi mật khẩu . . . . .	21
Hình 2.13	Luồng xem thông báo . . . . .	22
Hình 2.14	Luồng xem dòng hình ảnh đã lưu . . . . .	23
Hình 2.15	Bố cục giao diện chung của hệ thống . . . . .	24
Hình 3.1	Tổng quan mô hình camera thông minh . . . . .	26
Hình 3.2	Phương pháp xây dựng ứng dụng . . . . .	33
Hình 4.1	Giao diện chức năng đăng nhập . . . . .	38
Hình 4.2	Kiểm tra an toàn cho trẻ em . . . . .	38
Hình 4.3	Quản lý cửa hàng . . . . .	39
Hình 4.4	Nhận diện đối tượng . . . . .	39
Hình 4.5	Giao diện xử lý ảnh . . . . .	40
Hình 4.6	Giao diện quản lý tài khoản . . . . .	41
Hình 4.7	Điều khiển camera bằng cử chỉ tay . . . . .	42
Hình 4.8	Thiết bị biên Jetson Nano . . . . .	43
Hình 4.9	Thiết bị biên Jetson Nano . . . . .	43
Hình 4.10	Thiết bị biên Jetson Nano . . . . .	43
Hình 5.1	Kết quả mô hình nhận diện cử chỉ tay . . . . .	52
Hình 5.2	Bộ dữ liệu cử chỉ tay . . . . .	53
Hình 5.3	Ứng dụng điều khiển các chức năng camera . . . . .	55

## **DANH MỤC BẢNG BIỂU**

Bảng 2.1	Đặc tả use case đăng nhập . . . . .	11
Bảng 2.2	Đặc tả use case xem dòng hình ảnh đã qua xử lý . . . . .	12
Bảng 2.3	Đặc tả use case xử lý hình ảnh . . . . .	13
Bảng 2.4	Đặc tả use case xem thông báo đã lưu . . . . .	14
Bảng 4.1	Danh sách thư viện và công cụ sử dụng . . . . .	35
Bảng 4.2	Thống kê thông tin sản phẩm . . . . .	37
Bảng 4.3	Kết quả thực nghiệm các mô hình cử chỉ tay . . . . .	42
Bảng 4.4	Kết quả mô hình MobileNetv2 chạy trên thiết bị biên Jetson Nano . . . . .	44
Bảng 4.5	Kiểm thử chức năng đăng nhập . . . . .	46
Bảng 4.6	Kiểm thử chức năng xem dòng đã xử lý . . . . .	47
Bảng 4.7	Kiểm thử chức năng xử lý ảnh . . . . .	48

## **DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT**

<b>Thuật ngữ</b>	<b>Ý nghĩa</b>
API	Giao diện lập trình ứng dụng (Application Programming Interface)
ĐATN	Đồ án tốt nghiệp
Camera	Thiết bị ghi hình ảnh
CPU	Bộ xử lý trung tâm (Central Processing Unit)
CSS	Các tập tin định thiểu theo tầng (Cascading Style Sheets)
FPS	Số khung hình trên giây (Frame per second)
GPU	Bộ xử lý đồ họa (Graphics Processing Unit)
HTML	Ngôn ngữ đánh dấu siêu văn bản (HyperText Markup Language)
Web	Tập hợp các trang chứa thông tin (World Wide Web)

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

## 1.1 Đặt vấn đề

Hiện nay, camera đang được sử dụng nhiều ở cả Việt Nam cũng như trên thế giới và có xu hướng ngày càng tăng [1]. Camera là công cụ hỗ trợ các doanh nghiệp giám sát, quản lý hàng hóa rất tốt, nó cũng giúp đảm bảo an ninh cho các hộ gia đình cùng nhiều ứng dụng khác. Tuy nhiên, đối với các camera thông thường chỉ có chức năng thu hình ảnh, việc giám sát, quản lý sẽ được thực hiện hoàn toàn bởi con người. Để cải thiện vấn đề cần người vận hành, ngày càng nhiều các camera có các chức năng thông minh có thể thay con người giám sát, tự động thông báo khi gặp những dấu hiệu bất thường tùy vào mục đích sử dụng. Đặc biệt vào giai đoạn cách mạng công nghiệp 4.0 hiện nay, với sự bùng nổ của khoa học kỹ thuật, việc sử dụng các camera thông minh đã trở thành lựa chọn được rất nhiều doanh nghiệp và hộ gia đình tin tưởng. Tuy nhiên, đi kèm với đó là một lượng lớn những camera thông thường có nguy cơ bị thay thế gây lãng phí. Để giải quyết vấn đề này, một số nền tảng ra đời cung cấp dịch vụ kết nối những chiếc camera thông thường tới máy chủ trung tâm, sau đó những dòng hình ảnh có thể được xử lý bởi các chức năng thông minh khác nhau rồi trả kết quả lại tới người dùng.

Trên thế giới hiện nay cũng có một số những trang web cung cấp dịch vụ tương tự cho phép lấy dòng video từ camera và xử lý tùy theo mục đích sử dụng của người dùng [2]–[4]. Những trang web kể trên đã phần nào giải quyết vấn đề đặt ra, tuy nhiên việc thao tác điều khiển các chức năng camera cần phải có những thiết bị như máy tính hoặc điện thoại. Với những hệ thống cần máy tính để thao tác, việc kết nối và điều khiển có thể gây tổn tài nguyên do cần khởi chạy máy tính. Một số camera có thể được lắp đặt ở những nơi có địa hình không thuận lợp như quá hẹp hoặc quá cao, khó để sử dụng bàn phím và chuột để điều khiển. Hơn nữa, không phải lúc nào người dùng cũng tiện sử dụng máy tính hoặc điện thoại để điều khiển. Những người già hay những người ít tiếp xúc với công nghệ, việc thao tác với máy tính hoặc định thoại cũng có thể gặp khó khăn.

Nhận thấy sự thiếu sót của những ứng dụng hiện có, tôi đã xây dựng một ứng dụng giúp điều khiển các chức năng của camera. Ngoài sử dụng bàn phím và chuột như thông thường, ứng dụng còn có thể sử dụng cử chỉ tay giúp hỗ trợ trong việc điều khiển. Với mục tiêu điều khiển camera bằng cử chỉ tay trên thiết bị biến, tôi đã sử dụng những mô hình nhẹ khi xây dựng chức năng nhận diện cử chỉ tay, từ đó áp dụng triển khai điều khiển chức năng camera bằng cử chỉ tay trên thiết bị biến Jetson Nano.

Chính vì những lý do trên, tôi đã lựa chọn đề tài ‘Ứng dụng hỗ trợ điều khiển camera sử dụng cử chỉ tay’ để tạo ra một ứng dụng có thể điều khiển camera giúp người dùng dễ dàng thao tác. Để kiểm tra và đánh giá, ứng dụng sẽ được thử nghiệm trên một hệ thống có sẵn gồm máy chủ cung cấp những chức năng thông minh trên dòng camera có sẵn.

## 1.2 Mục tiêu và phạm vi đề tài

Như đã trình bày ở mục 1.1, nhu cầu sử dụng camera tích hợp các chức năng thông minh đang ngày càng tăng cao dẫn tới những hệ thống camera thông minh đã ra đời. Trên thế giới hiện nay, một số trang web cung cấp dịch vụ này tiêu biểu có thể kể tới AnyConnect [2], Guardian [3], Plainsight [4]. Những dịch vụ này giúp kết nối camera với máy chủ, từ đó có thể sử dụng các chức năng thông minh do nhà phát hành cung cấp. Tại Việt nam, BKAV cũng đã cung cấp cho người dùng ứng dụng xử lý dữ liệu camera bằng công nghệ trí tuệ nhân tạo để tự động đưa ra những cảnh báo, phân tích, thống kê đến người dùng. Để quản lý và sử dụng camera được tốt hơn, cần có **một ứng dụng xem được dòng hình ảnh sau khi qua xử lý, ứng dụng có thể điều khiển những chức năng camera, hơn nữa ứng dụng còn cần nhận được thông báo của máy chủ và giúp người dùng giao tiếp với máy chủ thông qua giao diện**.

Đối với các dịch vụ và hệ thống kể trên, để có thể thay đổi chức năng camera người dùng cần sử dụng máy tính hay điện thoại. Truy nhiên, do những khó khăn có thể gặp phải như đã đề cập ở mục 1.1, ứng dụng được thiết kế cần phải đảm bảo việc thay đổi chức năng phải có thể thực hiện được bằng cả cách sử dụng máy tính hoặc điều khiển bằng cử chỉ tay. Để làm được điều đó, **một mô hình nhận diện cử chỉ tay sử dụng các mạng học sâu nhẹ có độ chính xác tối thiểu đạt 90% là cần thiết**.

Do sử dụng máy tính để kết nối tới máy chủ trong một thời gian dài mà chỉ để điều khiển camera và không có nhu cầu sử dụng các chức năng khác sẽ gây tốn tài nguyên lớn, cũng như việc khởi động máy tính chỉ để sử dụng điều khiển camera có thể gây tốn thời gian. Để giải quyết vấn đề này, việc điều khiển các chức năng của camera không chỉ được thực hiện trên máy tính mà còn cần **có thể sử dụng mô hình dự đoán cử chỉ tay trên thiết bị biên Jetson Nano, thiết bị này ngoài việc điều khiển camera còn có nhận được tín hiệu từ máy chủ để thông báo cho người dùng**.

Cuối cùng, để có thể xây dựng được mô hình học sâu nhận diện cử chỉ tay, một bộ dữ liệu gồm hình ảnh các cử chỉ tay là cần thiết. Các bộ dữ liệu hiện có về cử chỉ tay chưa thực sự đa dạng về số lượng và điều kiện thu hình ảnh. Nhiều những

bộ dữ liệu về cử chỉ tay có số lượng hình ảnh ít, một số khác có số lượng nhiều nhưng chất lượng kém. Vì vậy, cần có **một bộ dữ liệu gồm các cử chỉ tay có chất lượng hình ảnh rõ nét không bị biến dạng, được chụp trên nhiều người và có số lượng hình ảnh đủ lớn, tối thiểu 1000 ảnh để phục vụ cho huấn luyện và kiểm thử mô hình.**

Phạm vi của đề tài giới hạn trong việc sử dụng những mô hình ít tham số có thể chạy được trên thiết bị biên Jetson Nano để dự đoán cử chỉ tay. Ứng dụng chạy trong điều kiện ánh sáng bình thường, với khoảng cách tối đa 4m kể từ camera. Đồ án tập trung xây dựng ứng dụng mẫu chưa phải sản phẩm thực tế.

### 1.3 Định hướng giải pháp

Để giải quyết bốn mục tiêu đã đề cập ở mục 1.2, ứng dụng được tạo ra với những giải pháp chính sau đây:

Đầu tiên, với mô-đun xây dựng ứng dụng có thể xem được dòng hình ảnh đã qua xử lý, ứng dụng sử dụng Electron [5] để thiết kế giao diện, kết hợp giữa ngôn ngữ JavaScript và Python để gửi yêu cầu đến cho máy chủ sau đó hiển thị dòng hình ảnh đã qua xử lý lên giao diện cho người dùng. Ứng dụng có những nút bấm, chọn giúp người dùng có thể thay đổi chức năng dễ dàng. Người dùng cũng có thể lựa chọn cách điều khiển chức năng bằng cử chỉ tay sau khi bấm vào nút tương ứng. Đối với dòng hình ảnh, ứng dụng sử dụng thư viện OpenCV [6] giúp người dùng có thể lưu lại dòng vào bộ nhớ máy tính. Những thông báo từ phía máy chủ được gửi đến cho ứng dụng nhờ dịch vụ Firebase [7] có khả năng gửi nhận thông báo nhanh được phát triển bởi Google. Ứng dụng cũng cung cấp cho người dùng giao diện để người dùng có thể thay đổi mật khẩu tài khoản và thay đổi quyền sử dụng đối với các chức năng hiện có của máy chủ. Ngoài ra, người dùng có thể xem hoặc xóa các thông báo, dòng hình ảnh đã lưu trong máy, đồng thời thay đổi đường dẫn lưu dòng hình ảnh.

Thứ hai, đối với mô-đun nhận diện cử chỉ tay. Ứng dụng có đầu vào là hình ảnh của người đang giơ tay, và đầu ra là kết quả dự đoán cử chỉ tay đó. Bước đầu, tôi xây dựng bộ dữ liệu gồm những cử chỉ tay khác nhau từ những bộ dữ liệu sẵn có. Bước hai, tôi đã sử dụng một mô hình để trích xuất những điểm trên bàn tay, từ đó loại bỏ được những phần không cần thiết để thu về bộ dữ liệu chỉ gồm hình ảnh của bàn tay. Sau đó đưa hình ảnh bàn tay vừa lấy được vào mô hình học sâu nhẹ, ít tham số để thu về kết quả mong muốn.

Thứ ba, mô-đun sử dụng cử chỉ tay điều khiển thiết bị biên Jetson Nano để điều khiển các chức năng camera. Do thiết bị biên Jetson Nano có bộ nhớ và tốc độ xử lý chậm, nên việc sử dụng những mô hình mới có độ chính xác cao nhưng tiêu tốn

tài nguyên lớn là không phù hợp, vì vậy, đồ án này đã sử dụng mô hình nhẹ để dự đoán cử chỉ tay trên thiết bị Jetson Nano. Sau khi nhận được hình ảnh cử chỉ tay từ camera được gắn với thiết bị, thiết bị sẽ tiến hành dự đoán, từ cử chỉ tay đó sẽ gửi những yêu cầu tương ứng đến máy chủ để thay đổi chức năng camera. Thiết bị có sử dụng FastAPI [8] giúp xây dựng một máy chủ ảo để nhận thông báo từ máy chủ.

Cuối cùng, để xây dựng một bộ dữ liệu như yêu cầu đã đạt ra, giải pháp là sẽ tạo từ những bộ dữ liệu sẵn có và dữ liệu tự thu thập. Từ đó tiền xử lý để tạo ra một bộ dữ liệu chỉ gồm hình ảnh cử chỉ tay.

Đồ án gồm bốn đóng góp chính sau đây: (i) Đề xuất mô hình nhận diện cử chỉ tay dựa vào các điểm trên khung xương của bàn tay kết hợp mô hình học sâu (ii) Bộ dữ liệu gồm các cử chỉ tay khác nhau (iii) Ứng dụng để sử dụng các chức năng của camera (iv) Điều khiển camera sử dụng cử chỉ tay thông qua Jetson Nano.

#### **1.4 Bố cục đồ án**

Phần còn lại của báo cáo đồ án tốt nghiệp này được tổ chức như sau.

Chương 2 trình bày về thực trạng hiện nay, từ đó đi sâu vào phân tích các yêu cầu cần có và thiết kế ứng dụng.

Chương 3 trình bày về các công nghệ liên quan đến thiết kế giao diện và mạng học sâu có sử dụng trong đồ án này.

Chương 4 trình bày về kết quả đạt được, cách triển khai và quá trình đánh giá ứng dụng.

Chương 5 trình bày về những đóng góp nổi bật của đồ án.

Chương 6 trình bày những kết quả đã đạt được, hướng phát triển trong tương lai.

## CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

### 2.1 Khảo sát hiện trạng

Hiện nay, nhu cầu sử dụng camera đang ngày càng tăng cao tại Việt Nam, chỉ đến năm 2019, tổng số camera tại Việt Nam đã đạt con số 2,6 triệu chiếc [9], con số ngày càng tăng cao cho đến hiện tại. Bên cạnh những chiếc camera chỉ có chức năng xem và lưu trữ hình ảnh, những chiếc camera có tích hợp trí tuệ nhân tạo đang ngày được ứng dụng vào đời sống. Những ứng dụng của camera thông minh có thể kể đến hiện nay như là tự động phát hiện người vượt đèn đỏ, vi phạm giao thông khi lưu thông trên đường, phát hiện người không đeo khẩu trang tại những nơi quy định, nhận diện khuôn mặt để tự động chấm công tại các công ty.

Năm bắt xu hướng sử dụng những camera có tích hợp trí tuệ nhân tạo, nhiều doanh nghiệp đã cho ra đời những camera tích hợp trí tuệ nhân tạo tiêu biểu là thương hiệu AI View [10] được sản xuất bởi BKAV hay nền tảng camera thông minh của Viettel [11]. Nhiều nền tảng trên thế giới được ra đời AnyConnect [2], Guardian [3], Plainsight [4] có thể giúp biến những chiếc camera thông thường sử dụng được những chức năng thông minh nhờ việc kết nối đến một máy chủ chuyên xử lý dòng hình ảnh, sau đó người dùng sẽ thao tác chọn chức năng muốn dùng bằng máy tính hoặc điện thoại. Việc này phần nào hạn chế khả năng linh hoạt của hệ thống khi không thể sử dụng nếu thiếu những thiết bị như máy tính, điện thoại. Trong khi đó, hiện nay những mô hình về thị giác máy tính như cử chỉ tay cũng rất phát triển, nhiều mô hình dự đoán cử chỉ tay với độ chính xác cao được ra đời. Theo như nghiên cứu và tìm hiểu của tôi, tôi chưa thấy tổ chức lớn nào áp dụng những mô hình nhận diện cử chỉ tay để hỗ trợ người dùng trong việc điều khiển các chức năng của camera một hệ thống trí tuệ nhân tạo.

Dựa vào những hiện trạng đã trình bày ở trên, tôi nhận thấy cần có một ứng dụng có thể quản lý camera và sử dụng cử chỉ tay để hỗ trợ điều khiển các chức năng camera. Mô hình nhận dạng cử chỉ tay được sử dụng cần phải nhẹ và có thể chạy được không chỉ trên máy tính thông thường, mà còn phải chạy được trên những thiết bị biên nhỏ gọn như Jetson Nano, giúp người điều khiển camera nhanh gọn hơn.

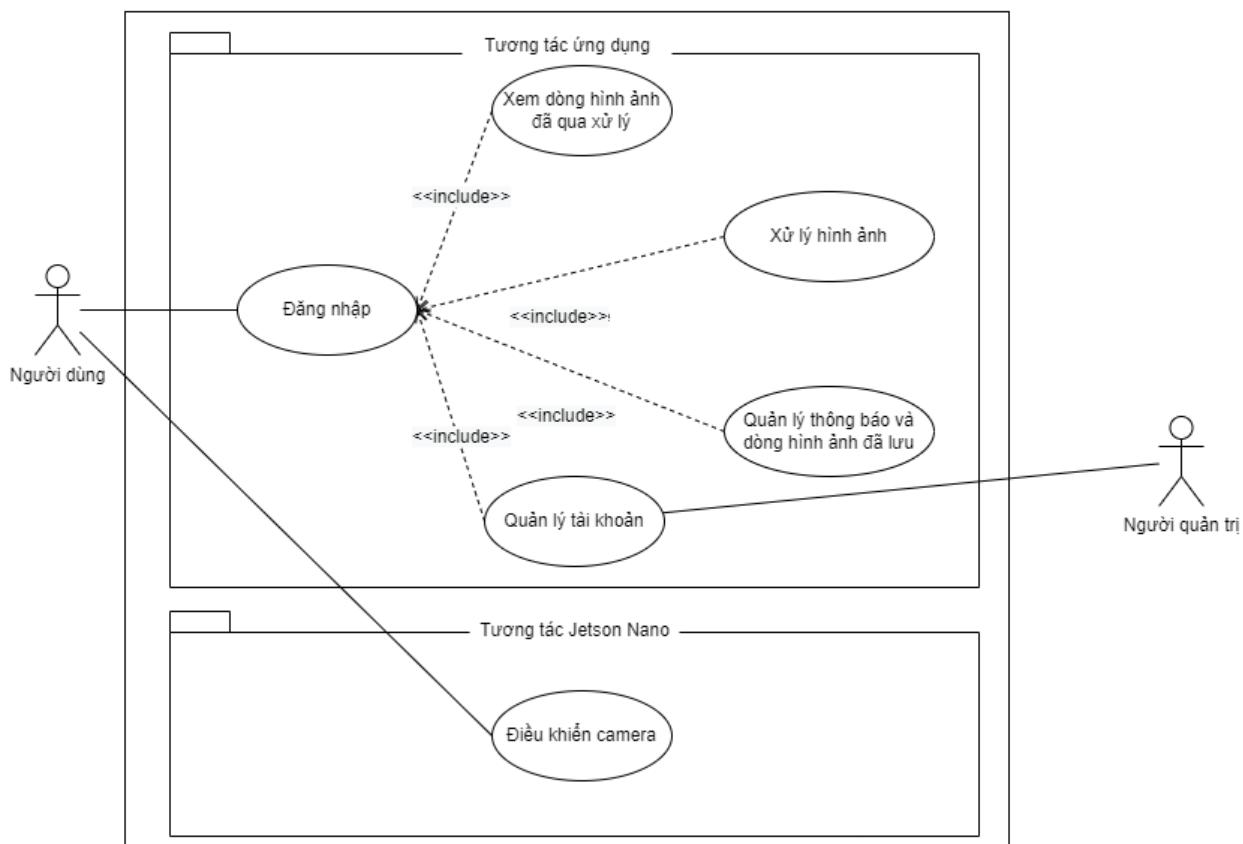
Đối với thiết bị biên Jetson Nano, cần có thể sử dụng cử chỉ tay để điều khiển các chức năng hiện có của camera. Trong quá trình sử dụng, nếu có thông báo từ máy chủ, cần thông báo lại tới người dùng qua loa được gắn kèm. Đối với ứng dụng máy tính, những bài toán chính cần giải quyết là: (i) Đăng nhập (ii) xem dòng hình ảnh đã qua xử lý (iii) xử lý ảnh (iv) quản lý thông báo và dòng hình ảnh đã lưu (v)

quản lý tài khoản. Sau đây tôi xin phân tích tổng quan các chức năng này bằng biểu đồ use case.

## 2.2 Tổng quan chức năng

### 2.2.1 Biểu đồ use case tổng quát

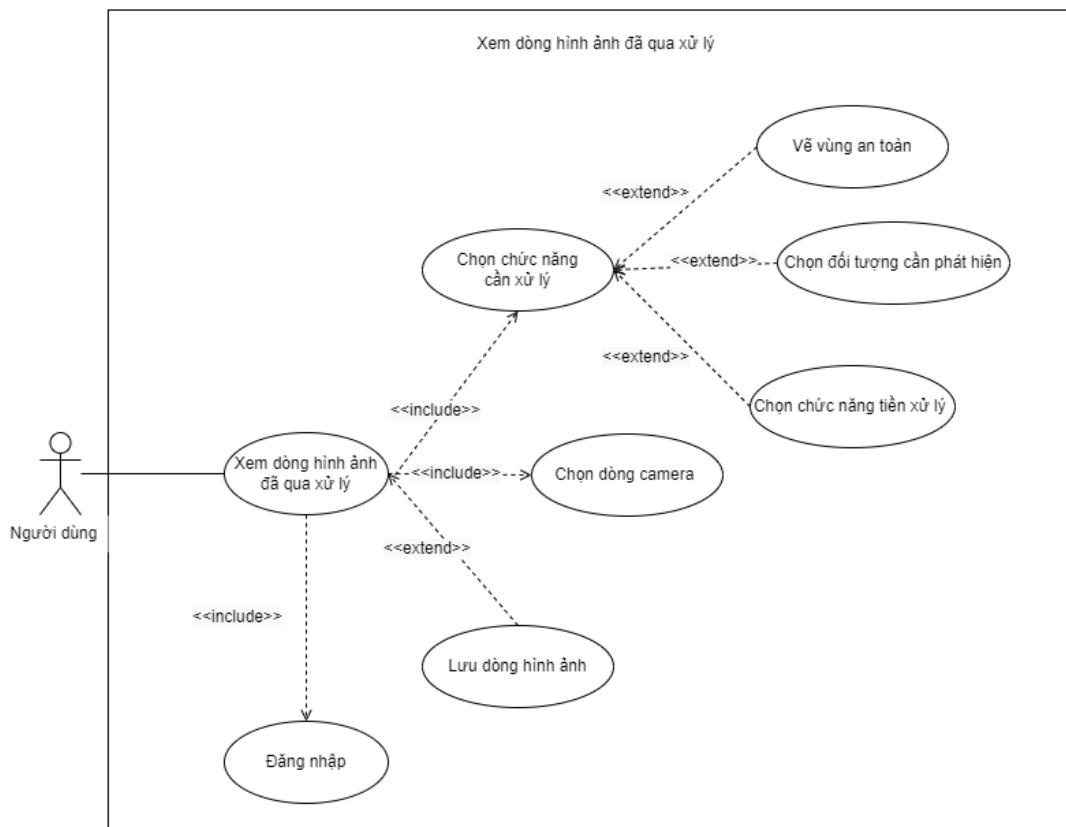
Ứng dụng có một tác nhân duy nhất là người dùng. Người dùng có thể giao tiếp với máy chủ thông qua giao diện của ứng dụng hoặc thông qua Jetson Nano. Đối với ứng dụng, sau khi người dùng đăng nhập, người dùng có thể thực hiện những chức năng của ứng dụng như xem dòng hình ảnh đã qua xử lý, xử lý hình ảnh, quản lý thông báo và dòng hình ảnh đã lưu, quản lý tài khoản.Thêm vào đó, người dùng cũng có thể sử dụng cử chỉ tay để điều khiển camera thông qua thiết bị Jetson Nano. Mô tả chi tiết thể hiện ở hình 2.1



**Hình 2.1:** Biểu đồ use case tổng quan

### 2.2.2 Biểu đồ use case phân rã xem dòng hình ảnh đã qua xử lý

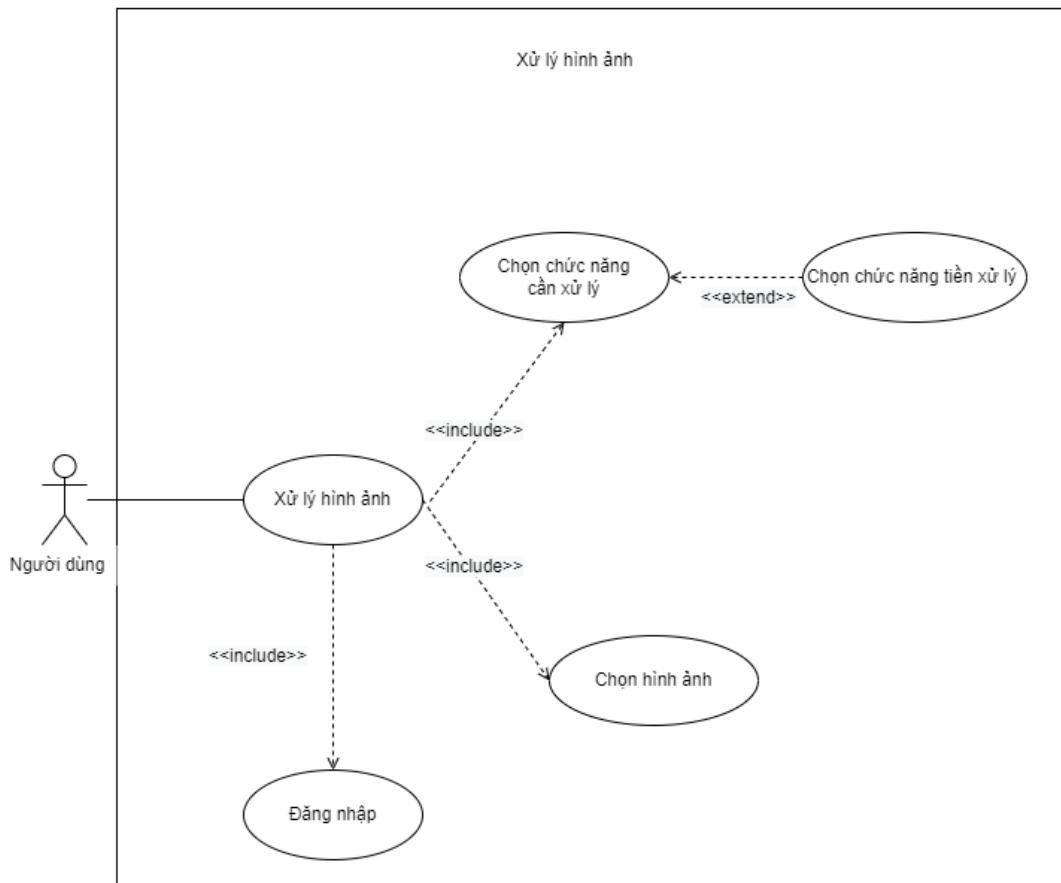
Sau khi đăng nhập người dùng có thể xem dòng hình ảnh từ camera đã qua xử lý bằng cách chọn chức năng và dòng camera muốn xem. Với một vài chức năng, người dùng có thể vẽ vùng an toàn, chọn đối tượng cần phát hiện trong vùng đó, hoặc tiền xử lý dòng hình ảnh trước khi máy chủ thực hiện những chức năng chính. Người dùng cũng có thể lưu lại dòng hình ảnh đang phát. Chi tiết xem tại hình 2.2



**Hình 2.2:** Biểu đồ use case phân rã xem dòng hình ảnh đã qua xử lý

### 2.2.3 Biểu đồ use case phân rã xử lý hình ảnh

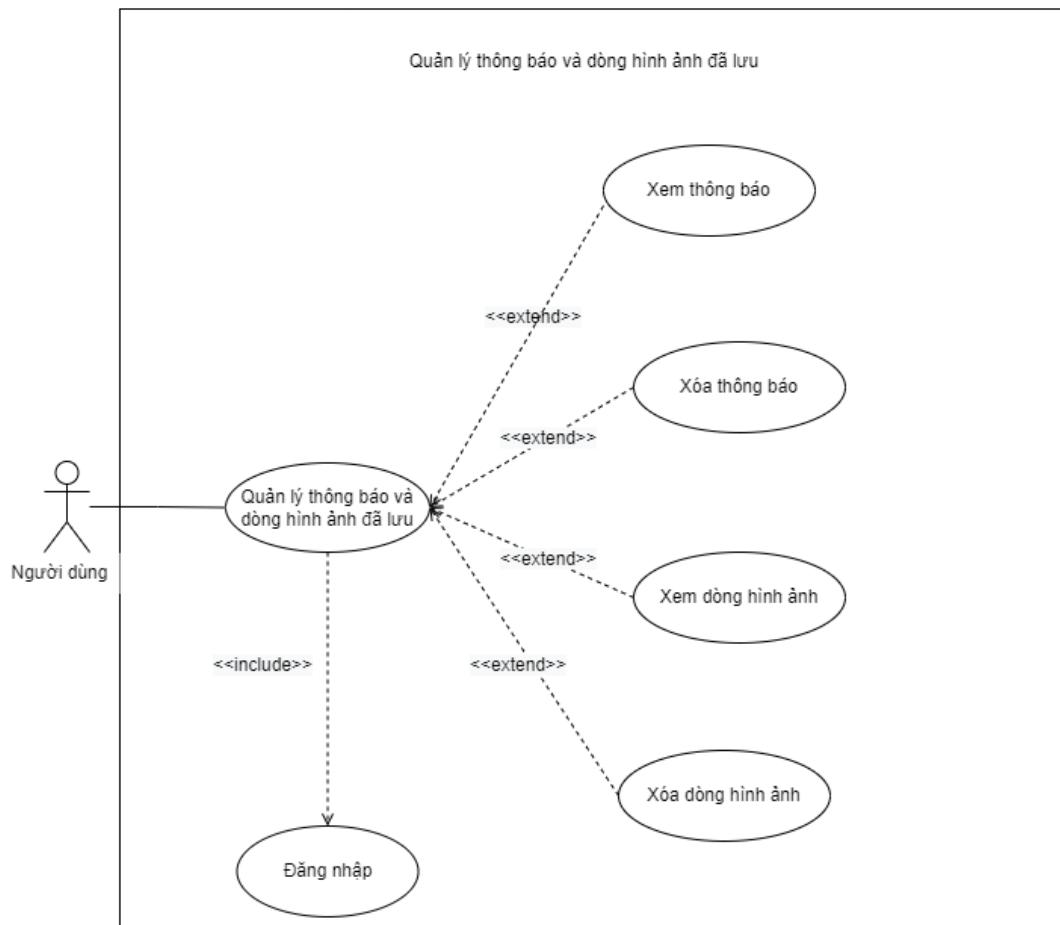
Đối với chức năng xử lý hình ảnh, người dùng có thể sử dụng sau khi đăng nhập. Sau đó người dùng sẽ chọn hình ảnh cần xử lý và chọn chức năng cần xử lý mong muốn. Người dùng có thể kết hợp nhiều thuật toán tiền xử lý để xử lý ảnh. Chi tiết xem tại hình 2.3



**Hình 2.3:** Biểu đồ use case phân rã xử lý hình ảnh

### 2.2.4 Biểu đồ use case phân rã quản lý thông báo và dòng hình ảnh đã lưu

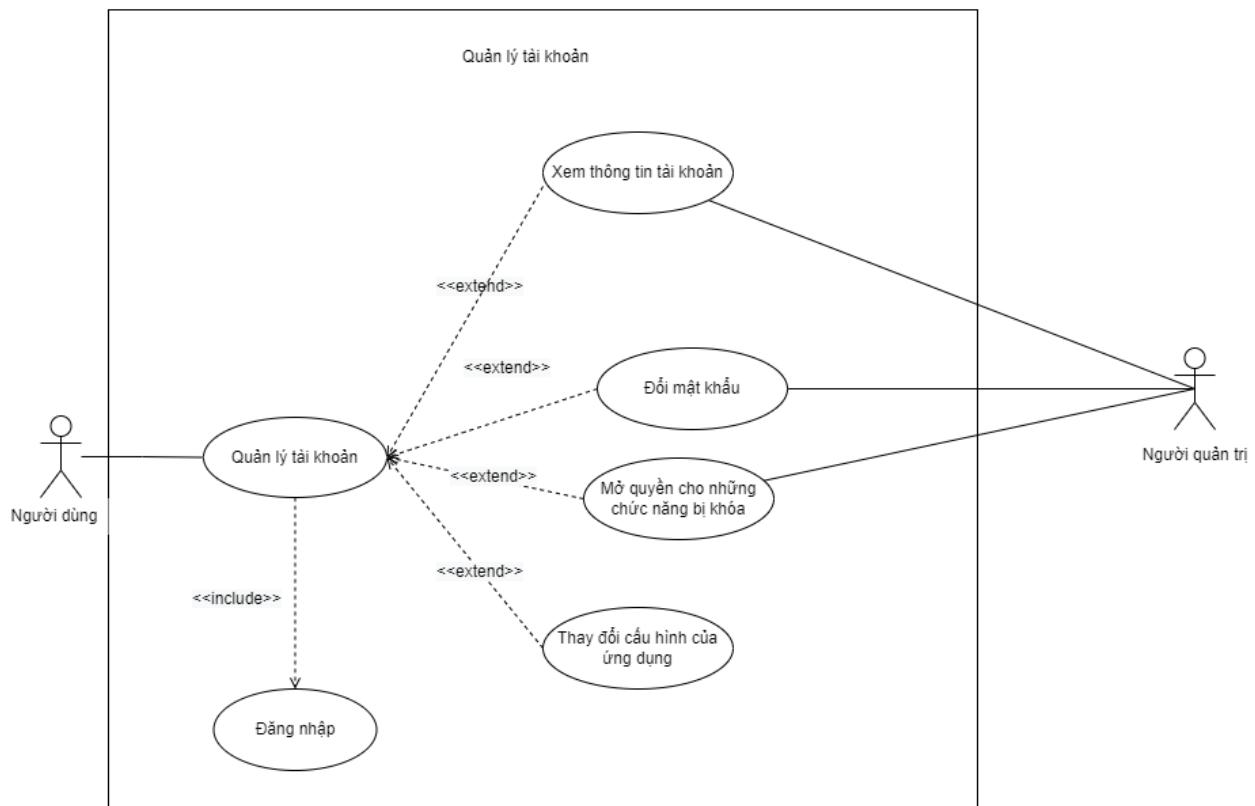
Sau khi đăng nhập, người dùng có thể quản lý thông báo và dòng hình ảnh. Cụ thể, người dùng có thể xem, xóa thông báo hoặc dòng hình ảnh đã lưu. Chi tiết xem tại hình 2.4



**Hình 2.4:** Biểu đồ use case phân rã quản lý thông báo và dòng hình ảnh đã lưu

### 2.2.5 Biểu đồ use case phân rã quản lý tài khoản

Sau khi đăng nhập, người dùng có thể quản lý tài khoản của mình. Thông qua ứng dụng, người dùng có thể xem thông tin tài khoản, thay đổi mật khẩu, mở quyền cho những chức năng bị khóa, thay đổi cấu hình của ứng dụng. Chi tiết xem tại hình 2.5.



**Hình 2.5:** Biểu đồ use case phân rã quản lý tài khoản

### 2.3 ĐẶC TẢ CHỨC NĂNG

Ở phần trước tôi đã trình bày tổng quan các hành động mà người dùng có thể thao tác. Để làm rõ ràng hơn, phần này tôi sẽ đi sâu vào mô tả chi tiết thông qua đặc tả những use case chính trong đồ án này.

#### 2.3.1 Đặc tả use case đăng nhập

Đặc tả use case đăng nhập được mô tả như bảng 2.1.

**Bảng 2.1:** Đặc tả use case đăng nhập

Mã usecase	UC0001	Tên Use case	Đăng nhập
Tác nhân hệ thống		Người dùng	
Tiền điều kiện		Đã có tài khoản trong ứng dụng	
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Chọn chức năng đăng nhập
	2	Hệ thống	Hiển thị giao diện nhập thông tin đăng nhập
	3	Người dùng	Nhập tài khoản, mật khẩu
	4	Người dùng	Ấn nút đăng nhập
Luồng sự kiện thay thế	STT	Thực hiện	Hành động
	5.a	Hệ thống	Thông báo thông tin đăng nhập bị sai
Hậu điều kiện	Không		

### 2.3.2 Đặc tả use case xem dòng hình ảnh đã qua xử lý

Đặc tả use case xem dòng hình ảnh đã qua xử lý được mô tả như bảng 2.2.

Yêu cầu: Người dùng đăng nhập.

**Bảng 2.2:** Đặc tả use case xem dòng hình ảnh đã qua xử lý

Mã usecase	UC0002	Tên Use case	Xem dòng hình ảnh đã qua xử lý
Tác nhân hệ thống	Người dùng		
Tiền điều kiện	Đã đăng nhập		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Chọn chức năng xem dòng hình ảnh
	2	Hệ thống	Hiển thị giao diện để người lựa chọn
	3	Người dùng	Chọn dòng camera
	4	Người dùng	Chọn chức năng
	5	Người dùng	Ấn nút để gửi lên máy chủ
	6	Hệ thống	Hiển thị dòng hình ảnh đã qua xử lý
Luồng sự kiện thay thế	STT	Thực hiện	Hành động
	5.a	Người dùng	Vẽ vùng an toàn
	5.b	Người dùng	Chọn đối tượng cần phát hiện
	5.c	Người dùng	Tiền xử lý dòng hình ảnh
	6.a	Hệ thống	Thông báo lỗi
Hậu điều kiện	Không		

### 2.3.3 Đặc tả use case xử lý hình ảnh

Đặc tả use case xử lý hình ảnh được mô tả như bảng 2.3.

Yêu cầu: Người dùng đăng nhập.

**Bảng 2.3:** Đặc tả use case xử lý hình ảnh

Mã usecase	UC0002	Tên Use case	Xử lý hình ảnh
Tác nhân hệ thống	Người dùng		
Tiền điều kiện	Đã đăng nhập		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Chọn chức năng xử lý hình ảnh
	2	Hệ thống	Hiển thị giao diện để người lựa chọn
	3	Người dùng	Chọn chức năng cần xử lý
	4	Người dùng	Chọn hình ảnh cần xử lý
	5	Người dùng	Chọn kết hợp các chức năng tiền xử lý
	6	Người dùng	Ấn nút để gửi lên máy chủ
Luồng sự kiện thay thế	7	Hệ thống	Hiển thị hình ảnh và kết quả sau khi xử lý
	STT	Thực hiện	Hành động
	5.a	Người dùng	Không chọn
Hậu điều kiện	7.a	Hệ thống	Thông báo kết nối lỗi hoặc thao tác sai
	Không		

### 2.3.4 Đặc tả use case xem thông báo đã lưu

Đặc tả use case xem thông báo đã lưu được mô tả như bảng 2.4.

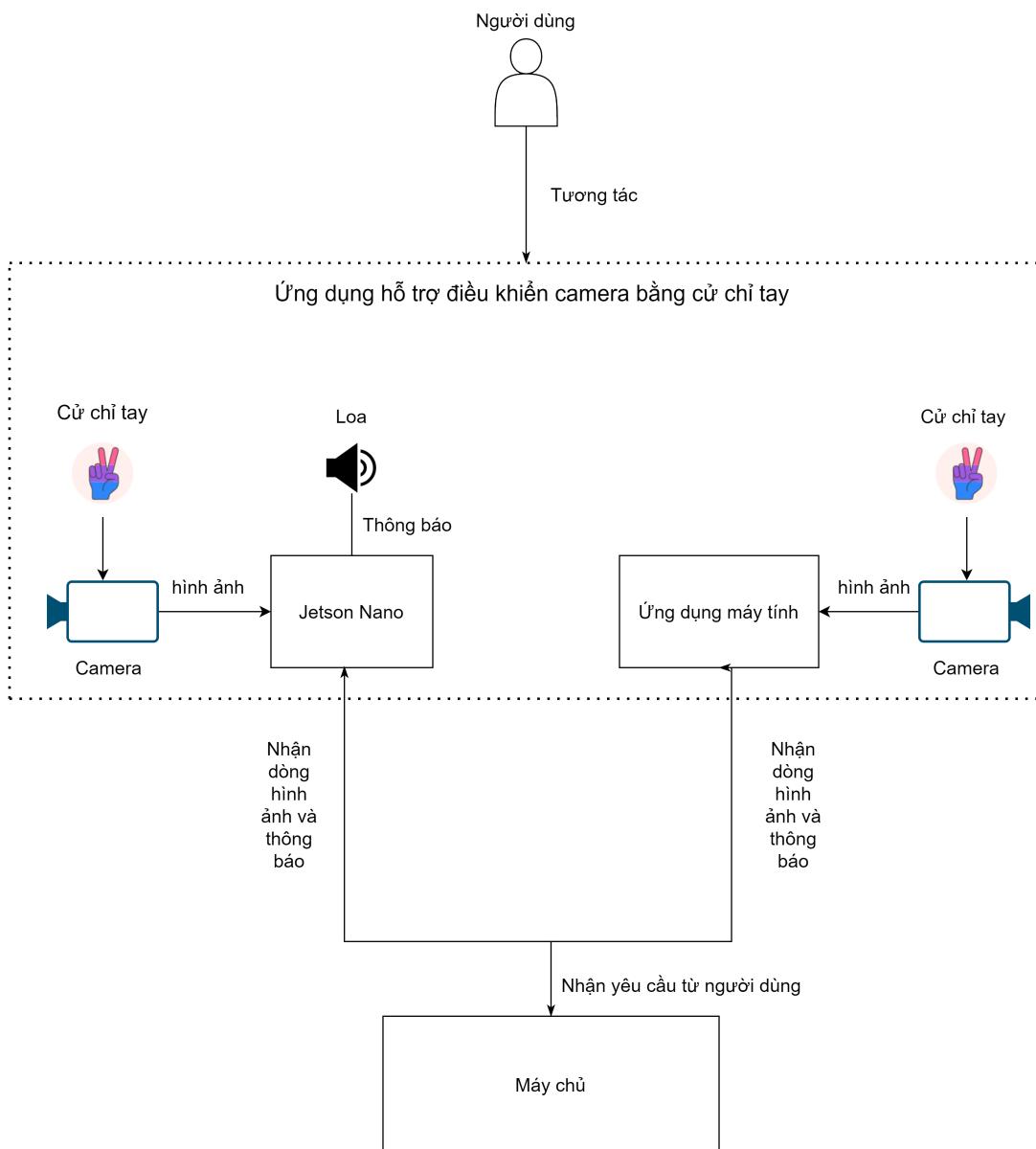
Yêu cầu: Người dùng đăng nhập.

**Bảng 2.4:** Đặc tả use case xem thông báo đã lưu

Mã usecase	UC0002	Tên Use case	Xem thông báo đã lưu
Tác nhân hệ thống	Người dùng		
Tiền điều kiện	Đã đăng nhập		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Chọn chức năng quản lý thông báo
	2	Hệ thống	Hiển thị danh sách thông báo
Luồng sự kiện thay thế	Không		
Hậu điều kiện	Không		

## 2.4 Thiết kế ứng dụng

Hình 2.6 mô tả tổng quan luồng hoạt động của ứng dụng. Để đáp ứng những chức năng đã nêu trên, đồ án này sẽ thực hiện hai phần chính được thiết kế trên hai thiết bị là máy tính và Jetson Nano. Cả hai thiết bị đều nối với camera để thu hình ảnh cử chỉ tay của người dùng. Người dùng có thể sử dụng cử chỉ tay này để điều khiển dòng hình ảnh từ máy chủ. Khi có thông báo nhận về từ máy chủ sẽ được thông báo qua loa được kết nối với thiết bị Jetson Nano. Ứng dụng sử dụng ngôn ngữ Python để xử lý nhận diện cử chỉ tay, sau khi nhận diện sẽ gửi yêu cầu tương ứng với từng cử chỉ tới máy chủ bằng cách gọi đến các API thông qua thư viện FastAPI [8]. Đối với ứng dụng máy tính, được phát triển bằng Electron [5] giúp xây dựng giao diện bằng các công nghệ web. Trong đó, việc giao tiếp giữa hai ngôn ngữ khác nhau JavaScript và Python được thực hiện bằng thư viện Python-Shell [12].



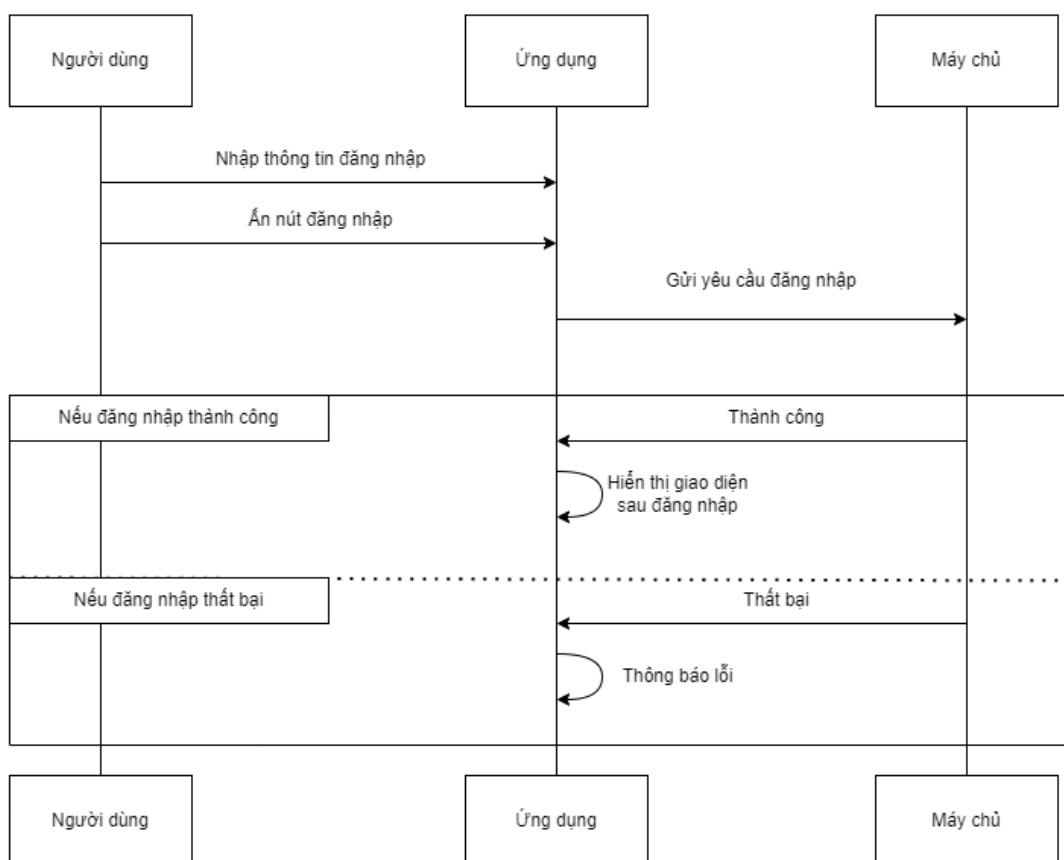
**Hình 2.6:** Tổng quan luồng hoạt động của ứng dụng

### 2.4.1 Thiết kế luồng hoạt động

Để làm rõ hơn kiến trúc của hệ thống, dưới đây là luồng hoạt động đại diện cho những chức năng chính của ứng dụng.

#### a, Luồng đăng nhập

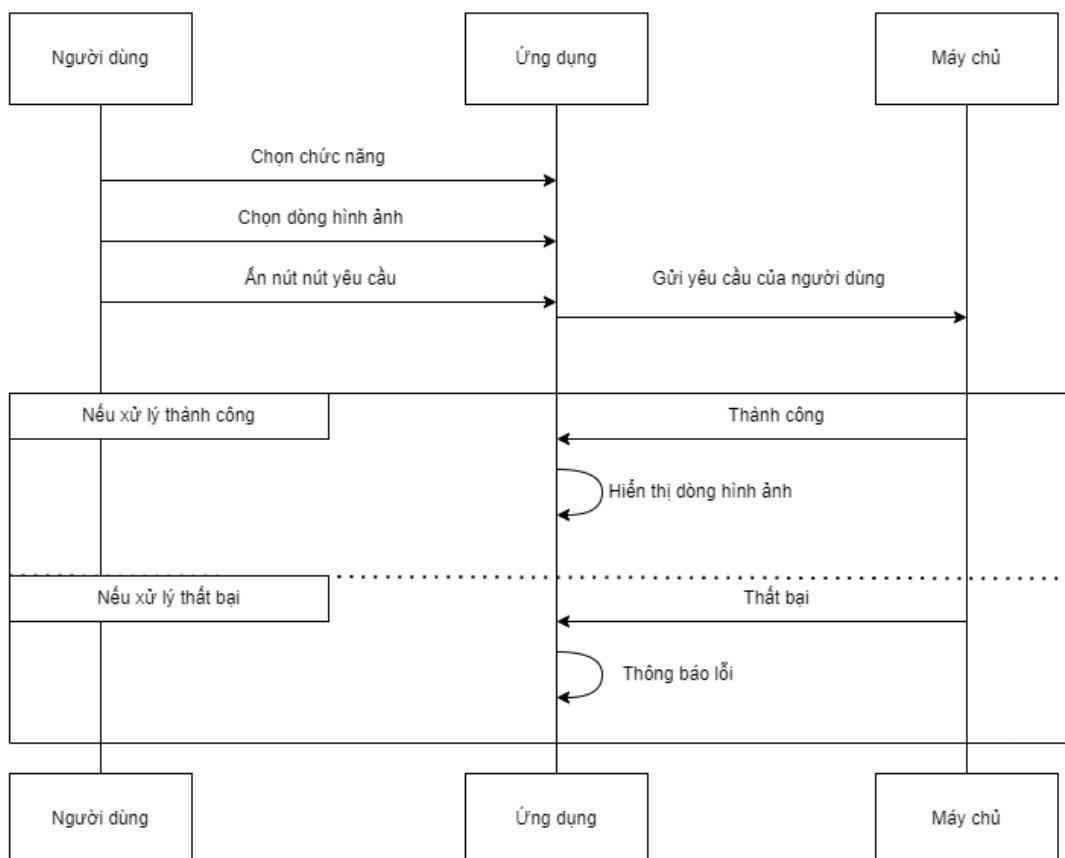
Người dùng để đăng nhập sẽ nhập tài khoản và mật khẩu vào giao diện ứng dụng. Sau đó ấn nút đăng nhập, ứng dụng sẽ xử lý yêu cầu đến máy chủ, nếu thành công người dùng sẽ được chuyển đến giao diện sau khi đăng nhập, nếu thất bại sẽ trả về thông báo lỗi. Chi tiết luồng hoạt động xem tại hình 2.7.



**Hình 2.7:** Luồng đăng nhập

### b, Luồng xem dòng hình ảnh đã xử lý

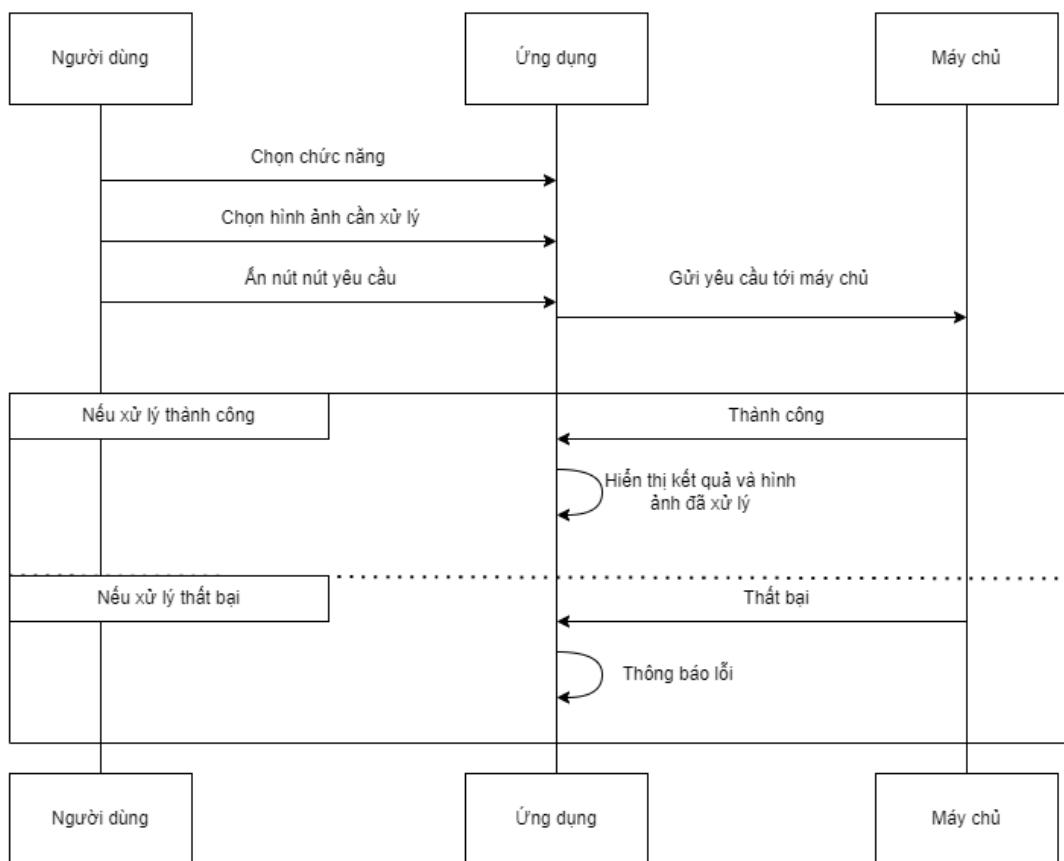
Để xem hình ảnh đã qua xử lý, người dùng truy cập vào chức năng tương ứng. Sau đó chọn chức năng muốn xử lý, chọn dòng hình ảnh muốn xử lý và ấn nút yêu cầu. Tiếp theo ứng dụng sẽ gửi yêu cầu đến máy chủ. Nếu thành công, giao diện ứng dụng sẽ hiển thị dòng, nếu thất bại sẽ hiển thị thông báo lỗi. Chi tiết luồng hoạt động xem tại hình 2.8.



**Hình 2.8:** Luồng xem dòng hình ảnh đã xử lý

### c, Luồng xử lý ảnh

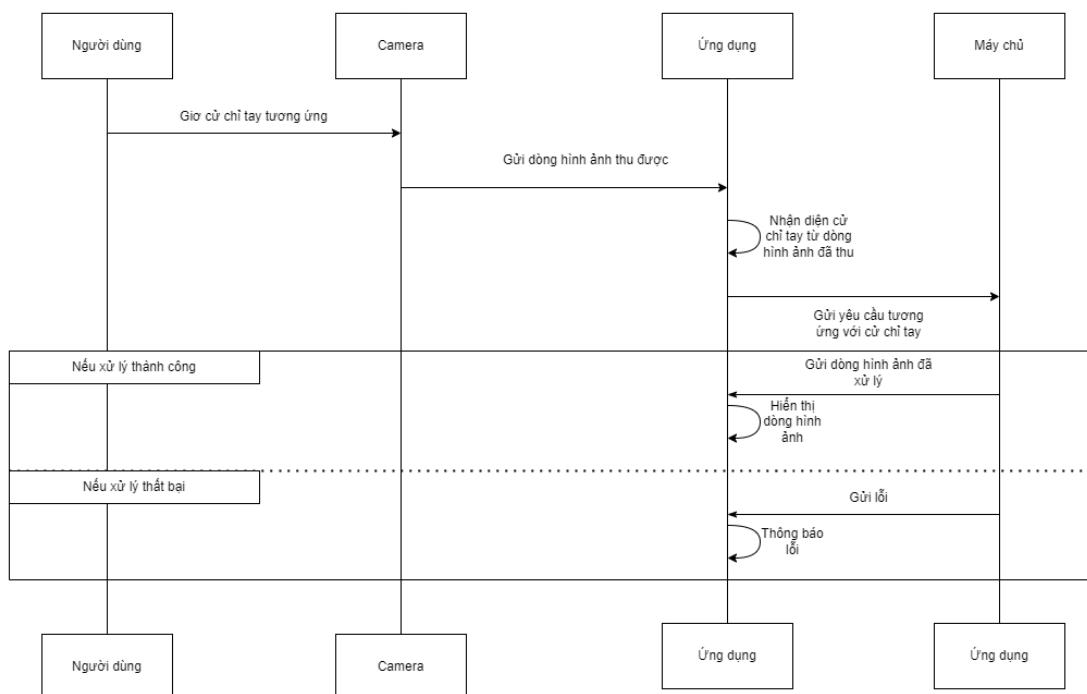
Để xem hình ảnh đã qua xử lý, người dùng truy cập vào chức năng tương ứng. Sau đó chọn chức năng muốn xử lý, chọn hình ảnh muốn xử lý và ấn nút yêu cầu. Tiếp theo ứng dụng sẽ gửi yêu cầu đến máy chủ. Nếu thành công, giao diện ứng dụng sẽ hiển thị kết quả và ảnh đã xử lý, nếu thất bại sẽ hiển thị thông báo lỗi. Chi tiết luồng hoạt động xem tại hình 2.9.



**Hình 2.9:** Luồng xử lý ảnh

#### d, Luồng điều khiển bằng cử chỉ tay

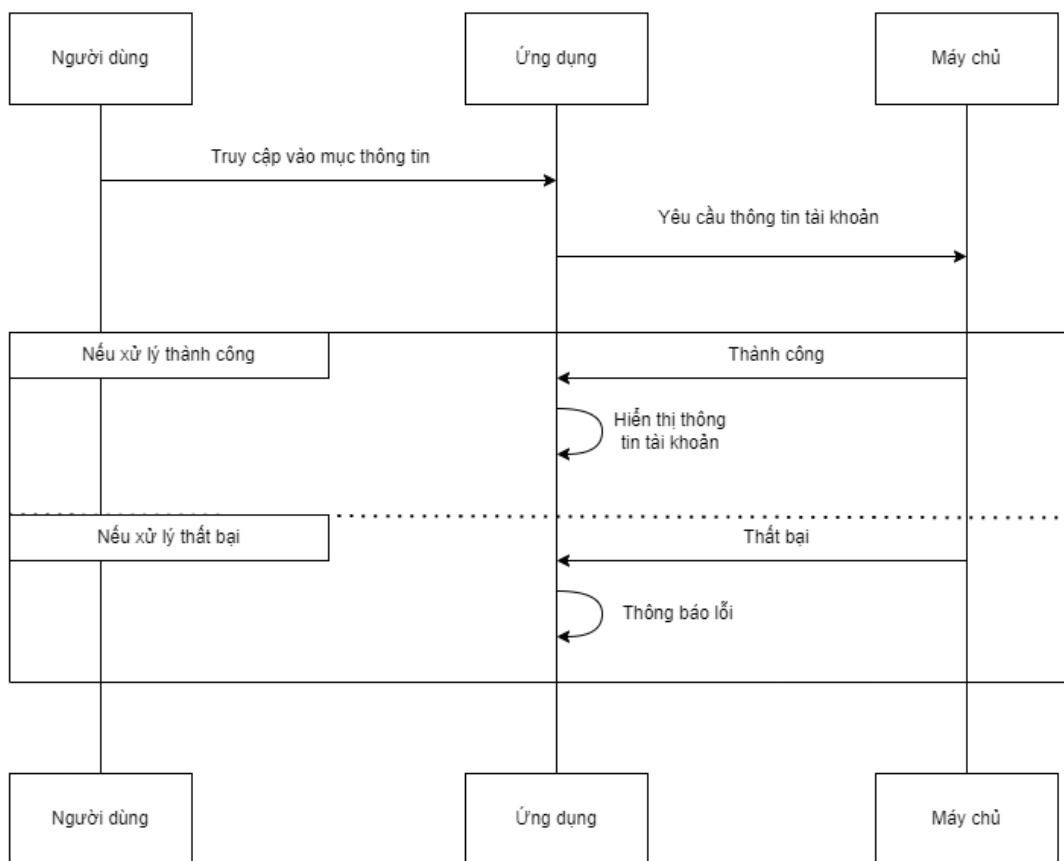
Để điều khiển các chức năng xử lý đối với dòng hình ảnh bằng cử chỉ tay, người dùng sẽ giơ cử chỉ tay tương ứng trước camera. Camera sau khi thu hình ảnh sẽ gửi hình ảnh đó về ứng dụng. Sau đó ứng dụng thực hiện dự đoán cử chỉ tay. Ứng với mỗi cử chỉ tay, ứng dụng sẽ gửi yêu cầu với nội dung tương ứng về máy chủ. Cuối cùng máy chủ sẽ xử lý dòng hình ảnh tương ứng với yêu cầu và trả dòng hình ảnh đó về cho người dùng hiển thị tại ứng dụng. Chi tiết luồng hoạt động xem tại hình 2.10.



**Hình 2.10:** Luồng điều khiển bằng cử chỉ tay

### e, Luồng xem thông tin tài khoản

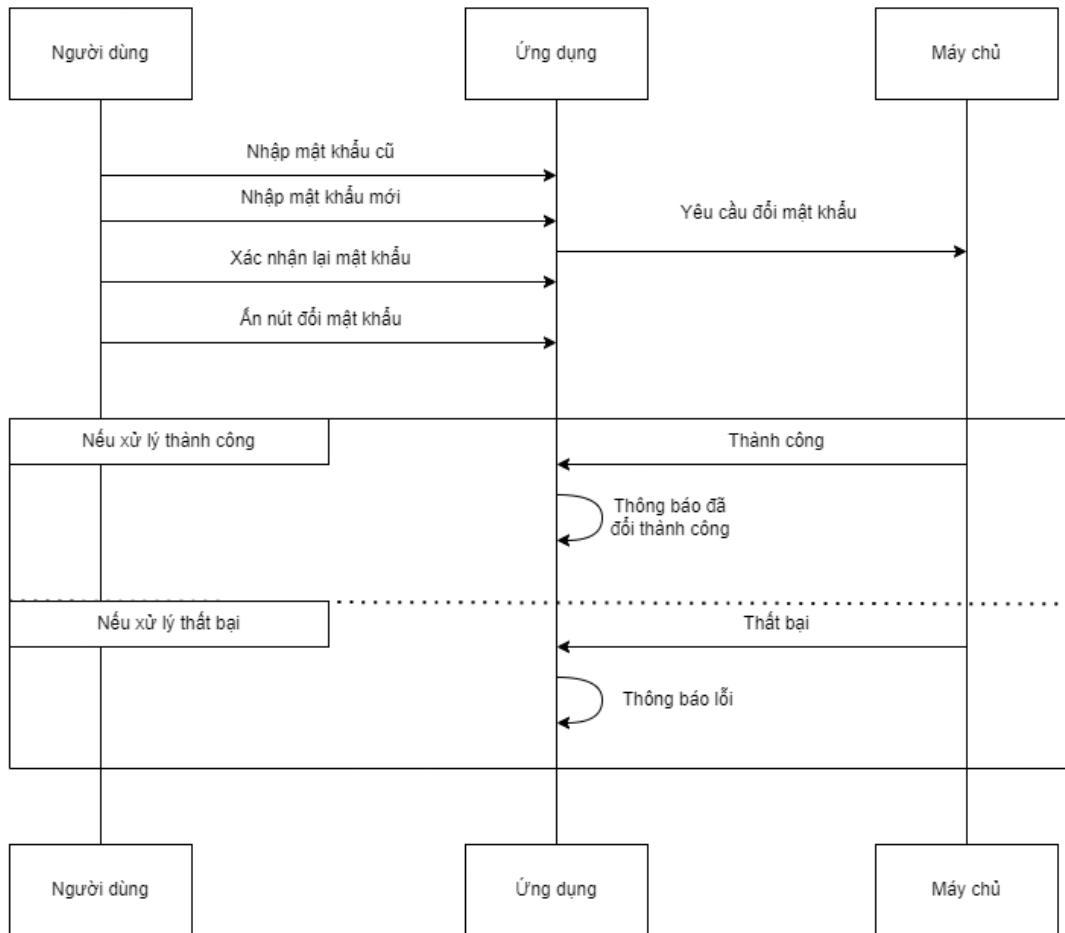
Khi muốn xem thông tin tài khoản, người dùng sẽ truy cập vào chức năng thông tin trong ứng dụng. Sau đó ứng dụng sẽ gửi yêu cầu gửi thông tin tài khoản đến máy chủ. Máy chủ sau khi truy xuất dữ liệu sẽ gửi dữ liệu về ứng dụng. Nếu xử lý thành công, ứng dụng sẽ hiển thị thông tin, nếu thất bại ứng dụng sẽ gửi thông báo lỗi. Chi tiết luồng hoạt động xem tại hình 2.11.



**Hình 2.11:** Luồng xem thông tin tài khoản

### f, Luồng đổi mật khẩu

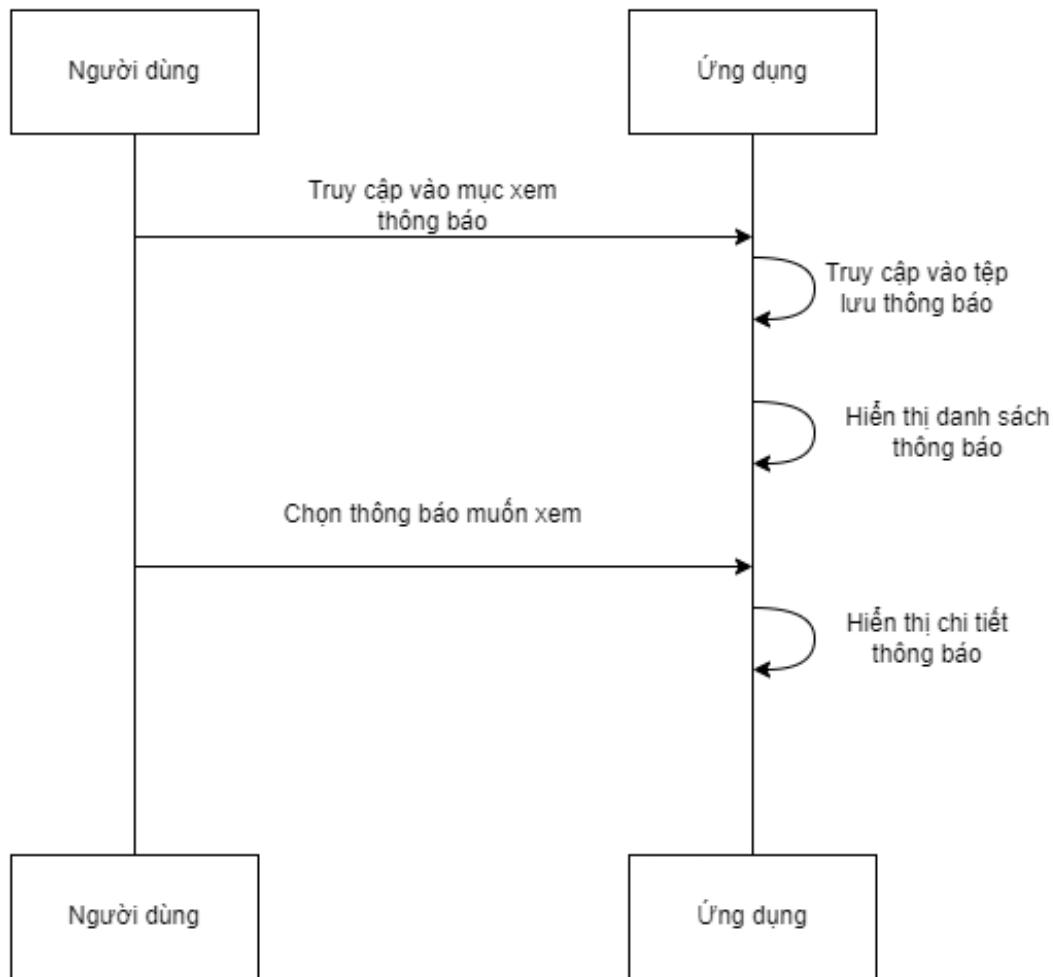
Để đổi mật khẩu, người dùng sẽ phải nhập lần lượt mật khẩu cũ, mật khẩu mới và xác nhận mật khẩu mới, sau đó ấn nút đổi mật khẩu. Ứng dụng sẽ lấy thông tin vừa nhận được để gửi đến cho máy chủ. Tiếp theo, máy chủ sẽ xác nhận và gửi lại kết quả đến ứng dụng. Nếu kết quả là thành công, ứng dụng sẽ gửi thông báo thành công, ngược lại ứng dụng sẽ hiển thị thông báo lỗi. Chi tiết luồng hoạt động xem tại hình 2.12.



**Hình 2.12:** Luồng đổi mật khẩu

### g, Luồng xem thông báo

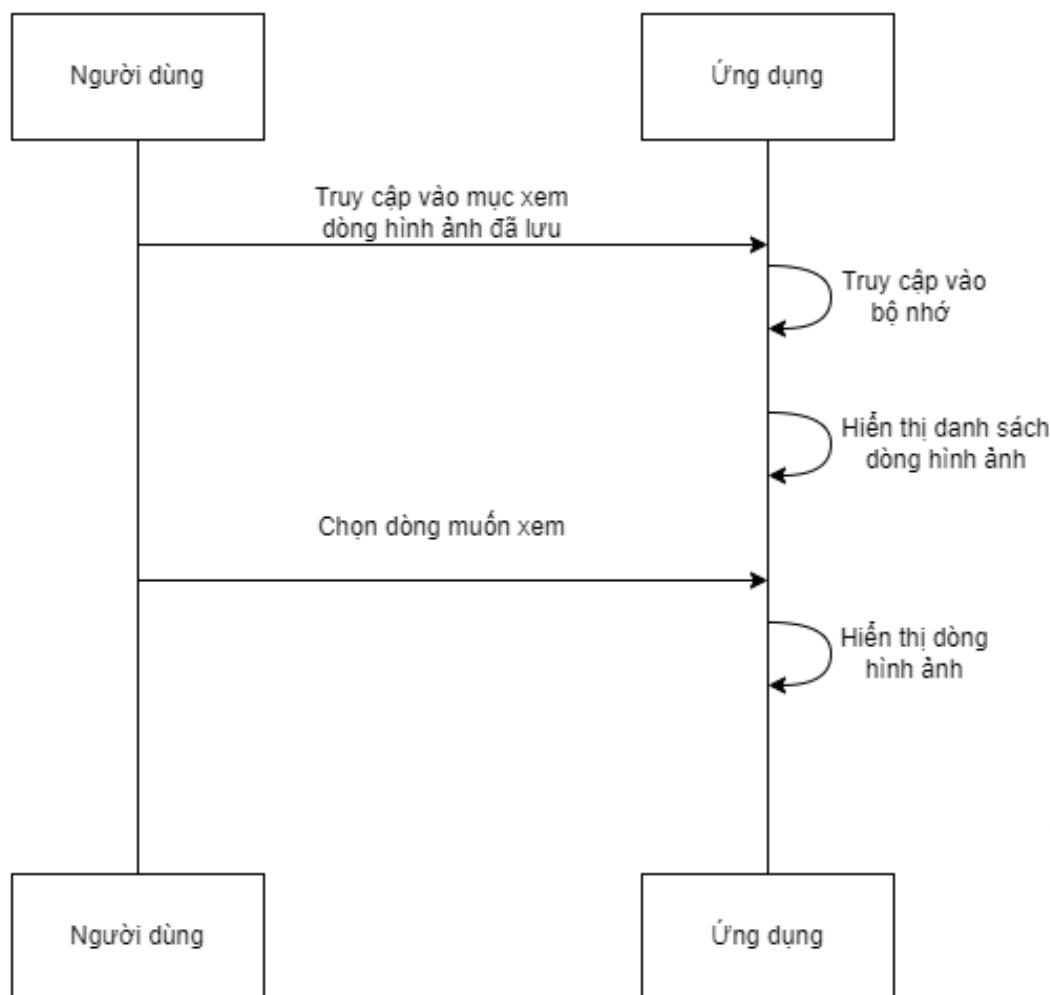
Khi người dùng muốn xem lại thông báo đã nhận được, người dùng sẽ truy cập vào mục xem thông báo. Sau đó ứng dụng sẽ truy cập vào bộ nhớ và hiển thị danh sách thông báo đã lưu. Để xem chi tiết từng thông báo, người dùng sẽ chọn thông báo muốn xem, ứng dụng sẽ hiển thị thông báo chi tiết đó cho người dùng. Chi tiết luồng hoạt động xem tại hình 2.13.



**Hình 2.13:** Luồng xem thông báo

### **h, Luồng xem dòng hình ảnh đã lưu**

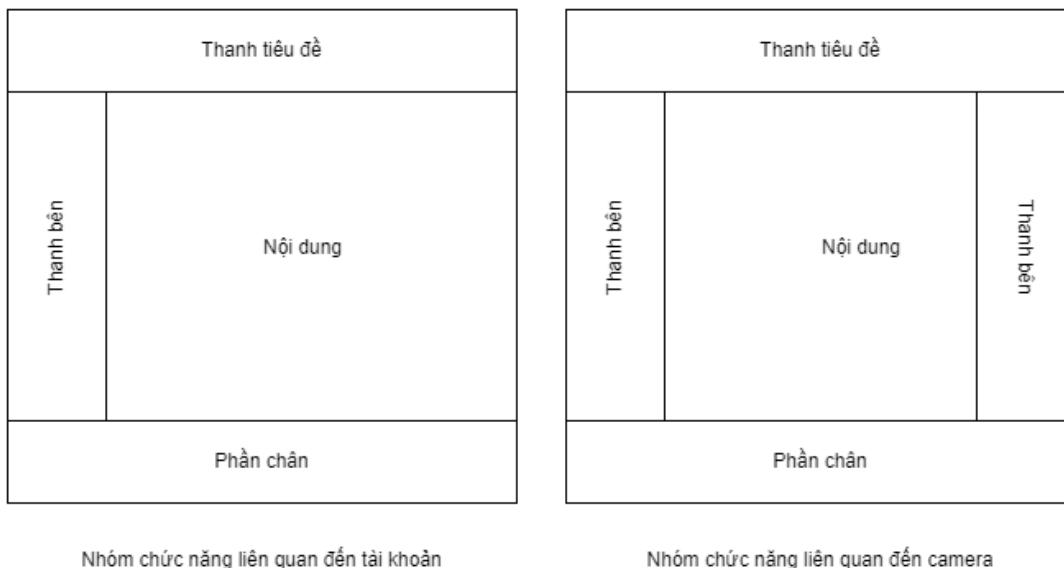
Khi người dùng muốn xem lại dòng hình ảnh đã lưu, người dùng sẽ truy cập vào mục xem dòng đã lưu. Sau đó ứng dụng sẽ truy cập vào bộ nhớ và hiển thị danh sách dòng đã lưu. Để xem chi tiết từng dòng, người dùng sẽ chọn dòng muốn xem, ứng dụng sẽ hiển thị dòng hình ảnh đó. Chi tiết luồng hoạt động xem tại hình 2.14.



**Hình 2.14:** Luồng xem dòng hình ảnh đã lưu

### 2.4.2 Thiết kế giao diện

Ứng dụng được thiết kế với kích thước giao diện là  $800 \times 900$  phù hợp với màn hình laptop và máy tính để bàn. Giao diện có tính phản hồi với người dùng khi người dùng thực hiện một chức năng bất kỳ. Giao diện cũng có sử dụng những biểu tượng và văn bản mô tả nút để người dùng dễ dàng sử dụng, tránh nhầm lẫn khi sử dụng. Trong khi thiết kế giao diện, ứng dụng có sử dụng những màu sắc để làm nổi bật những nút bấm tùy vào chức năng sử dụng. Đối với thiết kế giao diện tổng quan, cấu trúc của các chức năng có 4 thành phần chính là: (i) thanh tiêu đề của giao diện, (ii) phần thanh bên gồm các chức năng có thể sử dụng, (iii) phần nội dung chính, (iv) phần chân của giao diện. Tuy nhiên, đối với nhóm chức năng liên quan đến tài khoản và nhóm chức năng liên quan đến camera, sự sắp xếp bố cục của 4 thành phần có sự khác nhau. Chi tiết xem tại hình 2.15.



**Hình 2.15:** Bố cục giao diện chung của hệ thống

## 2.5 Yêu cầu phi chức năng

### 2.5.1 Yêu cầu về kỹ thuật

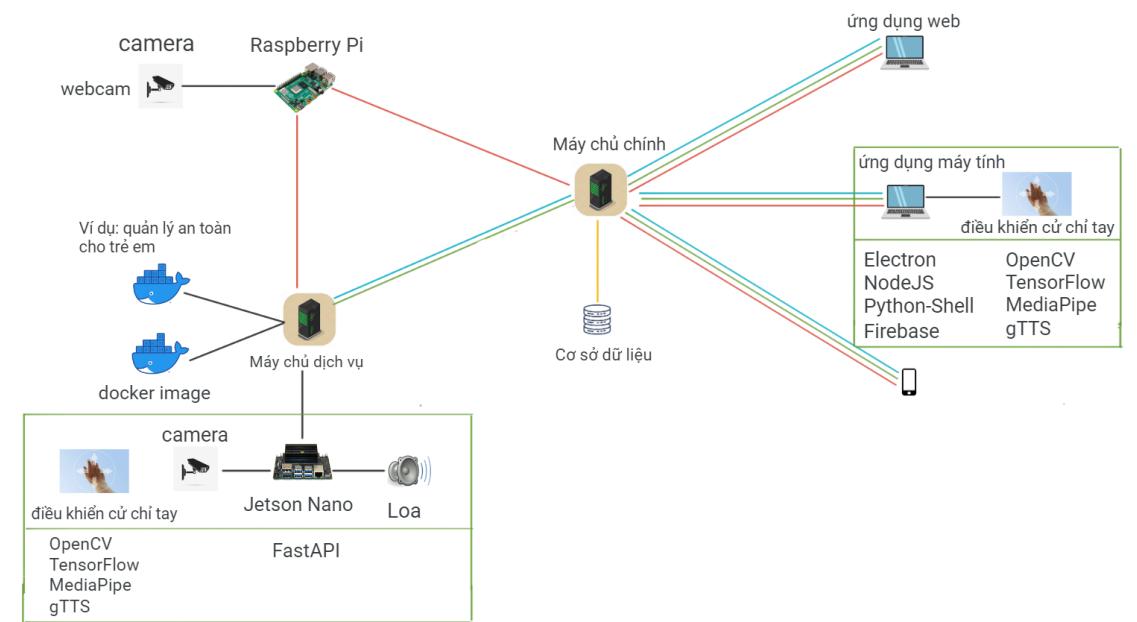
- Người dùng cần đăng nhập để sử dụng các chức năng của ứng dụng.
- Thời gian phản hồi của ứng dụng từ khi thực hiện các chức năng xử lý ảnh và xem dòng hình ảnh đã qua xử lý không quá một giây.
- Thời gian từ khi nhận diện được cử chỉ tay đến khi ứng dụng phản hồi không quá một giây.
- Mô-đun nhận diện cử chỉ tay phải đảm bảo khi chạy đạt số khung hình trên giây trung bình là 8.

### 2.5.2 Yêu cầu về giao diện người dùng

- Giao diện đơn giản, dễ sử dụng.
- Khi thao tác không gây khó hiểu cho người dùng.

## CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

Chương 3 này tôi xin trình bày về những công nghệ đã sử dụng và phương pháp xây dựng ứng dụng. Dưới đây là hình ảnh tổng quan về mô hình được tôi và nhóm đã nghiên cứu và xây dựng để giải quyết những mục tiêu đã đặt ra, trong đó đồ án này tập trung vào hai khối hình chữ nhật trong hình.



**Hình 3.1:** Tổng quan mô hình camera thông minh

Khối thứ nhất là ứng dụng máy tính có thể giao tiếp với máy chủ thực hiện những năng của ứng dụng. Ứng dụng có thể sử dụng bàn phím và chuột hoặc sử dụng cử chỉ tay để điều khiển.

Khối thứ 2 là thiết bị biên Jetson Nano. Người dùng có thể điều khiển camera thông qua cử chỉ tay, sau đó thiết bị biên Jetson Nano sẽ dự đoán và gửi yêu cầu đến máy chủ, nếu có thông báo, thiết bị sẽ thông báo qua Loa được kết nối với Jetson Nano.

Sau đây là những công nghệ tôi sử dụng cho từng khối đã nêu.

### 3.1 Công nghệ sử dụng cho khối ứng dụng máy tính

#### 3.1.1 Electron

##### a, Giới thiệu

Electron [5] là một khung mã nguồn mở cho phép ứng dụng trên máy tính có thể được thiết kế bởi những công nghệ web. Các ứng dụng được thiết kế có thể chạy được trên nhiều nền tảng như Mac, Window và Linux.

**b, Nơi sử dụng trong ứng dụng**

Tôi sử dụng Electron để thiết kế giao diện cho ứng dụng và một số những tác vụ xử lý đơn giản có thể được thực hiện bởi JavaScript bao gồm cả việc gửi nhận yêu cầu đến máy chủ.

**c, Lý do lựa chọn**

Ngoài Electron, còn có rất nhiều những thư viện khác hỗ trợ cho việc lập trình ứng dụng máy tính. Chỉ riêng với ngôn ngữ Python, ta có thể kể đến như PyQt, Tkinter. Những lý do sau đây là nguyên nhân tôi sử dụng Electron (i) Có thể sử dụng công nghệ web để viết ứng dụng (ii) so với những thư viện đã đề cập, Electron giúp thiết kế giao diện một cách nhanh chóng và đẹp hơn (iii) cài đặt đơn giản, dễ dàng triển khai.

**3.1.2 NodeJS****a, Giới thiệu**

Node.js được xây dựng và phát triển từ năm 2009, là nền tảng phát triển độc lập được xây dựng bằng C++ và JavaScript giúp những trang web được xây dựng một cách nhanh chóng và dễ dàng sử dụng. Node.js hướng sự vào ra dữ liệu bằng cách sử dụng các tác vụ thời gian thực một cách nhanh chóng. Do đó, Node.js có khả năng xử lý một số lượng lớn các kết nối. Kết hợp với việc sử dụng luồng đơn để thực thi các yêu cầu, Node.js có thể hỗ trợ cùng lúc chục ngàn yêu cầu.

**b, Nơi sử dụng trong ứng dụng**

Node.js kết hợp với HTML, CSS và Python được dùng để xây dựng và phát triển ứng dụng.

**c, Lý do lựa chọn**

Hiện nay, công nghệ lập trình web còn có nhiều ngôn ngữ có thể thay thế như ruby on rails, C và PHP. Tuy nhiên lý do tôi lựa chọn JavaScript và Node.js là (i) có thư viện npm hỗ trợ nhiều chức năng (ii) cộng đồng sử dụng lớn (iii) hiệu suất cao (iv) dễ dàng triển khai và sử dụng.

**3.1.3 Python-Shell****a, Giới thiệu**

Python-Shell [12] là thư viện của Node.js, giúp giao tiếp giữa ngôn ngữ Python và JavaScript.

**b, Nơi sử dụng trong ứng dụng**

Tôi sử dụng Python-Shell để giao tiếp giữa Python và JavaScript giúp truyền những thông tin nhận từ máy chủ và những dữ liệu sau khi xử lý đến ứng dụng.

### c, Lý do lựa chọn

Để phục vụ việc truyền tin, tôi có thể sử dụng FastAPI hoặc sử dụng SocketIO. Tuy nhiên tôi lựa chọn Python-Shell vì lý do sau đây (i) giao tiếp dễ dàng giữa Python và JavaScript (ii) dễ dàng sử dụng (iii) có nhiều hàm phục vụ việc truyền tin (iv) có thể sử dụng những công nghệ của Python để hỗ trợ nhiều công việc mà JavaScript khó để thực hiện.

#### 3.1.4 Firebase

##### a, Giới thiệu

Firebase [7] là một nền tảng do Google phát triển để tạo các ứng dụng web và di động. Nền tảng này cung cấp những dịch vụ hữu ích hỗ trợ lập trình viên trong quá trình phát triển ứng dụng.

##### b, Nơi sử dụng trong ứng dụng

Đồ án này sử dụng Firebase để nhận thông báo từ phía máy chủ.

##### c, Lý do lựa chọn

Lý do tôi sử dụng Firebase là (i) tốc độ truyền tin nhanh (ii) dễ dàng cài đặt và triển khai (iii) tương tác tốt với ngôn ngữ JavaScript.

#### 3.1.5 OpenCV

Giới thiệu OpenCV là viết tắt của Open Source Computer Vision, đây là một thư viện mã nguồn mở hàng đầu cho những bài toán về học máy, thị giác máy tính. Đây là một thư viện có tính linh hoạt cao, có thể sử dụng tại rất nhiều ngôn ngữ như Python, Java, C và C++. Do tính linh hoạt này, OpenCV có thể sử dụng cho lập trình trên nhiều nền tảng lớn như Windows, Linux, Mac OS, iOS và Android. OpenCV có thể hỗ trợ hiệu quả về tính toán và là thư viện chuyên dùng cho các ứng dụng thời gian thực. Chi tiết xem tại [6]

##### a, Nơi sử dụng trong ứng dụng

OpenCV được sử dụng cho cả lập trình ứng dụng và lập trình trên Jetson Nano với mục đích đọc dòng hình ảnh từ camera, hỗ trợ tiền xử lý hình ảnh để phục vụ cho mô-đun nhận diện cử chỉ tay. Ngoài ra, trên ứng dụng, OpenCV cũng được sử dụng để ghi dòng hình ảnh phục vụ cho mục đích xem lại.

##### b, Lý do lựa chọn

Ngoài OpenCV và TensorFlow cũng có thể được sử dụng cho mục đích tiền xử lý ảnh. Tuy nhiên, tôi lựa chọn thư viện OpenCV cho những mục đích đã đề cập bên trên vì: (i) so với TensorFlow, việc xử lý ảnh bằng OpenCV nhanh gọn hơn, OpenCV cũng cung cấp nhiều hàm phục vụ cho chức năng tiền xử lý (ii) việc đọc

ghi dòng hình ảnh dễ dàng, xử lý tối tốc độ cao (iii) dễ dàng sử dụng do có độ tương thích cao với Python (iv) có cộng đồng sử dụng lớn để hỗ trợ khi sử dụng.

### 3.1.6 TensorFlow

#### a, Giới thiệu

TensorFlow [13] là thư viện mã nguồn mở dành cho học máy lớn bậc nhất thế giới. Thư viện hỗ trợ mạnh mẽ cho việc tính toán trong học máy và học sâu, giúp các bài toán được tiếp cận một cách đơn giản, nhanh chóng hơn. Hơn nữa, TensorFlow cũng cho phép tính toán song song trên nhiều máy tính khác nhau, thậm chí là trên nhiều CPU.

#### b, Nơi sử dụng trong ứng dụng

Tôi sử dụng TensorFlow trong việc huấn luyện mô hình học sâu để dự đoán kết quả cử chỉ tay, tiếp theo sử dụng mô hình đó trong mô-đun nhận diện cử chỉ tay.

#### c, Lý do lựa chọn

Bên cạnh TensorFlow và Pytorch cũng là thư viện được nhiều người sử dụng để huấn luyện mô hình. Nhưng tôi đã lựa chọn TensorFlow mà không phải Pytorch vì những lý do sau đây: (i) so với Pytorch và TensorFlow cung cấp nhiều hàm cho việc hiển thị kết quả, giúp cho người dùng dễ dàng sửa lỗi và theo dõi thông số trong quá trình huấn luyện (ii) TensorFlow tích hợp những mô hình học sâu được sử dụng trong ứng dụng. Vì vậy, việc huấn luyện trở nên nhanh gọn và dễ dàng hơn (iii) cách thức sử dụng đơn giản, dễ dàng tiếp cận kể cả đối với những người mới sử dụng.

### 3.1.7 MediaPipe

#### a, Giới thiệu

MediaPipe là thư viện mã nguồn mở được cung cấp bởi Google, cung cấp những giải pháp trí tuệ nhân tạo đa nền tảng. Rất nhiều những bài toán về thị giác máy tính đều được cung cấp trong thư viện, có thể kể đến như nhận diện khuôn mặt, nhận diện đối tượng, nhận diện tư thế. Những giải pháp được cung cấp không chỉ cho độ chính xác cao mà còn có tốc độ xử lý nhanh. Chi tiết xem tại [14]

#### b, Nơi sử dụng trong ứng dụng

Ứng dụng của tôi sử dụng MediaPipe để trích xuất ra những điểm trên khung xương của bàn tay bao gồm 21 điểm. Từ những điểm này ứng dụng sẽ lấy được ảnh chỉ gồm bàn tay, là đầu vào cho mô hình học sâu để dự đoán cử chỉ tay.

### c, Lý do lựa chọn

Ngoài MediaPipe, còn có những mạng học sâu khác có thể trích xuất được khung xương trên bàn tay như HRNet. Lý do tôi quyết định sử dụng MediaPipe là (i) so với các mạng học sâu khác, MediaPipe không những có tốc độ xử lý nhanh, độ chính xác cao mà phần cứng cũng không yêu cầu cao (ii) đầu ra của MediaPipe gồm những điểm trên bàn tay, phù hợp với yêu cầu của ứng dụng (iii) dễ dàng cài đặt và triển khai trên cả máy tính và Jetson Nano.

#### 3.1.8 gTTS

##### a, Giới thiệu

gTTS [15] là dịch vụ của google giúp chuyển văn bản thành giọng nói bằng nhiều ngôn ngữ khác nhau. gTTS có tốc độ xử lý nhanh, có hỗ trợ lưu tệp âm thanh.

##### b, Nơi sử dụng trong ứng dụng

Tôi sử dụng dịch vụ này để thông báo khi người dùng điều khiển bằng cử chỉ tay, nhằm giúp người dùng xác nhận cử chỉ.Thêm vào đó, khi có thông báo, gTTS cũng hỗ trợ trong việc đọc thông báo đó.

### c, Lý do lựa chọn

Hiện nay, có nhiều thư viện hỗ trợ chuyển giọng nói từ văn bản, có thể kể đến như Wideo Text to Speech, ttsreader, ispeech. Lý do tôi sử dụng gTTS là (i) quá trình cài đặt và triển khai đơn giản (ii) có hỗ trợ nhiều giọng nói (iii) tương tác tốt với ngôn ngữ Python.

#### 3.1.9 EfficientNet

##### a, Giới thiệu

Đối với các mô hình mạng nơ-ron tích chập truyền thống, cách cải thiện hiệu suất hầu hết chỉ là tăng độ sâu. Tuy nhiên, việc tăng chiều sâu của mô hình cũng đòi hỏi tài nguyên và sức mạnh tính toán lớn hơn. Hơn nữa, sau một độ sâu nhất định, sẽ có thể đạt tới trạng thái bão hòa khiến mô hình không còn cải thiện nữa. Để khắc phục nhược điểm này và cải thiện hiệu suất, một giải pháp mới đã ra đời, đó là không chỉ thu phóng chiều sâu, mà còn thu phóng cả theo hướng chiều rộng và độ phân giải. Bằng cách sử dụng phương pháp này với một tỉ lệ thu phóng thích hợp, kết hợp phương pháp khối thặng dư đảo ngược đã tạo ra mô hình EfficientNet với hiệu suất cao. Chi tiết về EfficientNet có thể xem tại [16].

#### 3.1.10 Nơi sử dụng trong ứng dụng

Đồ án đã ứng dụng mô hình EfficientNet cho ứng dụng máy tính.

### 3.1.11 Lý do sử dụng

Ngoài những mô hình đã trình bày, còn có rất nhiều những mô hình có thể sử dụng cho mục đích phân loại có thể kể đến như DenseNet, GoogleNet, ResNet. Tuy nhiên lý do tôi sử dụng mạng trên là (i) mô hình có độ chính xác phù hợp với mục tiêu bài toán (ii) tốc độ xử lý trên CPU đạt ở mức phù hợp với mục tiêu bài toán (iii) dễ dàng sử dụng.

## 3.2 Công nghệ sử dụng cho khôi điều khiển camera trên thiết bị Jetson Nano

Khôi điều khiển trên thiết bị Jetson Nano có sử dụng các công nghệ như đã đề cập trong phần trước để xây dựng chức năng điều khiển cử chỉ tay. Ngoài ra, có sử dụng những công nghệ khác được trình bày dưới đây.

### 3.2.1 FastApi

#### a, Giới thiệu

FastAPI [8] là một khung được sử dụng để xây dựng các API. FastAPI có hiệu năng cao, cú pháp đơn giản, dễ dàng cài đặt và triển khai, có thể lập trình đồng bộ hoặc bất đồng bộ.

#### b, Nơi sử dụng trong ứng dụng

Tôi sử dụng FastAPI làm máy chủ cho thiết bị Jetson Nano, giúp máy chủ có thể gửi tín hiệu đến Jetson Nano.

#### c, Lý do lựa chọn

Hiện nay, ngoài việc sử dụng FastAPI, còn có nhiều công nghệ có thể hỗ trợ như SocketIO và Flask. Tuy nhiên tôi vẫn sử dụng FastAPI vì những lý do sau (i) dễ dàng tiếp cận đối với người mới sử dụng (ii) tốc độ truyền tin cao (iii) hiệu suất cao (iv) có thể xử lý bất đồng bộ.

### 3.2.2 MobileNetV2

#### a, Giới thiệu

Mô hình MobileNetV2 sử dụng mạng tích hợp tách biệt chiều sâu để tính toán. Đối với các mô hình, nguyên nhân chính dẫn tới việc gia tăng tham số làm tăng số lượng tính toán chính là độ sâu. Với phương pháp tính chập tách biệt chiều sâu, sẽ có thể loại bỏ sự phụ thuộc vào chiều sâu mà không làm ảnh hưởng đến kết quả đầu ra, vẫn thu được kết quả có kích thước tương đương thông thường. Ngoài sử dụng tích hợp tách biệt chiều sâu, bản cải thiện của MobileNet là MobileNetV2 còn sử dụng thêm những khôi dư đảo ngược gồm các nối tắt. Theo đó, các khôi ở lớp trước được cộng với các lớp ở liền sau. Các nối tắt được điều chỉnh sao cho số kênh ở đầu vào và đầu ra của mỗi khôi cổ chai được thắt hẹp lại. Điều này sẽ giúp MobileNetV2 xử lý nhanh hơn so với bản ban đầu. Nếu muốn tìm hiểu chi tiết, có

thể xem tại [17].

### **b, Nơi sử dụng trong ứng dụng**

Đồ án đã ứng dụng mô hình MobileNet cho thiết bị Jetson Nano.

### **c, Lý do sử dụng**

Ngoài những mô hình đã trình bày, còn có rất nhiều những mô hình có thể sử dụng cho mục đích phân loại có thể kể đến như DenseNet, GoogleNet và ResNet. Tuy nhiên lý do tôi sử dụng mạng trên là (i) mô hình nhẹ có ít tham số (ii) chạy trên thiết bị Jetson Nano (iii) tài nguyên sử dụng không vượt quá tài nguyên sẵn có của Jetson Nano.

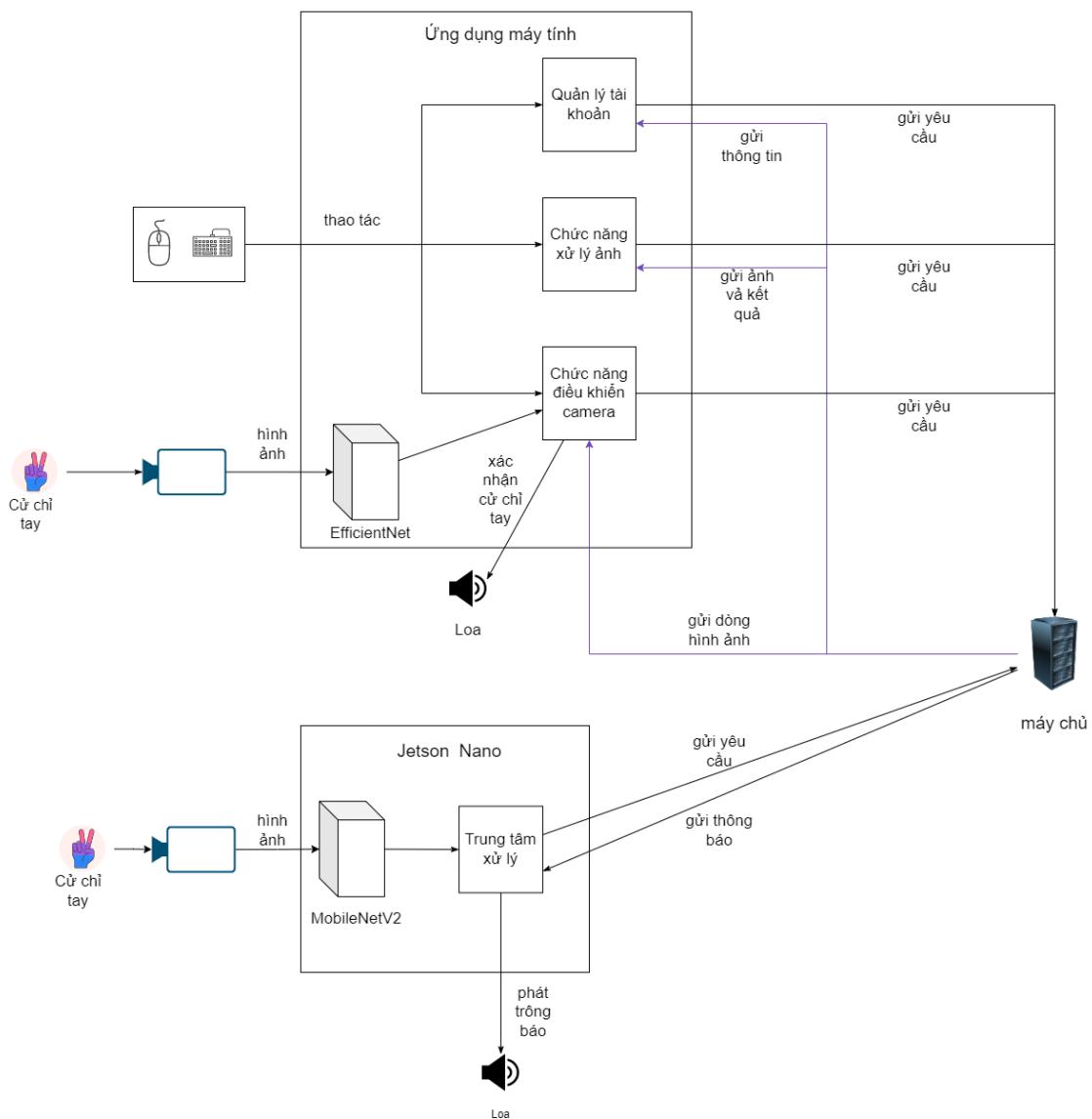
## **3.3 Phương pháp xây dựng**

Từ những công nghệ đã được đề cập bên trên, tôi sử dụng những công nghệ này để xây dựng ứng dụng hỗ trợ điều khiển camera bằng cử chỉ tay. Dưới đây là hình ảnh tổng quát về phương pháp tôi đề xuất.

Hình 3.2 mô tả hoạt động của ứng dụng từ khi nhận dữ liệu đến khi trả kết quả cho người dùng. Từ đó, tôi trình bày phương pháp xây dựng ứng dụng đối với từng mô-đun để đạt được mục tiêu như hình.

### **3.3.1 Mô-đun xây dựng ứng dụng máy tính**

Mục tiêu cuối cùng của mô-đun này là có thể xây dựng được một ứng dụng có thể điều khiển các chức năng camera, xử lý hình ảnh và quản lý tài khoản. Sau khi ứng dụng nhận yêu cầu của người dùng thông qua bàn phím, chuột hoặc thông qua cử chỉ tay đối với chức năng điều khiển camera, ứng dụng sẽ gửi yêu cầu đó cho máy chủ, và hiển thị kết quả trả về từ máy chủ cho người dùng. Để làm được những chức năng này, đầu tiên, ứng dụng sử dụng Electron để xây dựng giao diện ứng dụng cho những chức năng cần thiết là đăng nhập, điều khiển camera, xử lý ảnh và quản lý tài khoản. Về những việc xử lý sau khi người dùng thao tác, JavaScript là ngôn ngữ chính để thiết kế. Sau khi thiết kế thành công giao diện có thể phản hồi lại người dùng, ứng dụng kết hợp với ngôn ngữ Python thông qua thư viện Python-Shell. Bằng việc sử dụng Python, ứng dụng có thể kết hợp nhận yêu cầu từ người dùng bằng cách sử dụng một camera thu hình ảnh cử chỉ tay và đưa vào mô hình EfficientNet phân loại, sau khi nhận diện thành công, ứng dụng sẽ thông báo cho người dùng cử chỉ tay vừa chọn qua loa để người dùng xác nhận. Khi phát hiện hành vi bất thường cần cảnh báo cho người dùng, bằng dịch vụ Firebase, máy chủ có thể gửi ngay lập tức thông báo cho ứng dụng, ứng dụng sau đó sẽ hiển thị cho người dùng thông báo trên màn hình. Để phục vụ việc quản lý được tốt hơn, ứng dụng cũng sẽ tạo ra những chức năng lưu lại thông báo và dòng hình ảnh.

**Hình 3.2:** Phương pháp xây dựng ứng dụng

### 3.3.2 Mô-đun nhận diện cử chỉ tay

Mô-đun này yêu cầu đầu vào là dòng hình ảnh từ camera, đầu ra xác định xem người dùng sử dụng cử chỉ tay nào để điều khiển camera. Do việc điều khiển diễn ra nhanh và thực hiện được trên thiết bị Jetson Nano, nên mô hình để phát hiện cử chỉ tay cũng cần nhẹ để có thể chạy với tốc độ cao cùng với đó là tiêu tốn ít tài nguyên. Với những yêu cầu này, đồ án đã tập trung nghiên cứu những mô hình nhẹ và có khả năng cho ra kết quả tốt. Đầu vào của những mô hình này sẽ là hình ảnh bàn tay. Để có thể tiền xử lý hình ảnh gốc ban đầu sao cho lấy được hình ảnh bàn tay một cách sát nhất, giảm bớt tối đa những ngoại cảnh, tôi đã sử dụng thư viện MediaPipe để trích xuất những điểm trên bàn tay từ đó lấy được hình ảnh tay. Sau quá trình tiền xử lý ảnh gốc ban đầu, tôi tiến hành thử nghiệm huấn luyện trên những mô hình nhẹ. Trong quá trình huấn luyện, tôi cũng kết hợp với đó là thay đổi

tham số như tốc độ học, hàm mất mát, thêm một vài những lớp ở cuối mô hình như lớp gộp, lớp chuẩn hóa và tỉ lệ bỏ qua nút.

Kết quả tôi thu được một mô hình nhận diện cử chỉ tay sử dụng mạng học sâu nhẹ, có tốc độ xử lý nhanh, phù hợp sử dụng trên các thiết bị yêu cầu.

### 3.3.3 Mô-đun điều khiển camera trên thiết bị Jetson Nano

Tại mô-đun này, sau khi nhận yêu cầu từ người dùng sử dụng mô hình cử chỉ tay, tôi sử dụng Python cùng thư viện hỗ trợ để gửi yêu cầu đến máy chủ. Đồng thời, tôi cũng chạy một máy chủ ảo sử dụng thư viện FastAPI. Máy chủ này chịu trách nhiệm lắng nghe máy chủ chính và phát thông báo cho người dùng qua loa khi nhận được thông báo từ máy chủ. Bằng cách sử dụng những công nghệ này, tôi đã thành công giúp thiết bị Jetson Nano có thể gửi và nhận dữ liệu đến máy chủ thông qua cử chỉ tay, từ đó giúp người dùng có thêm những lựa chọn khi muốn điều khiển camera.

## CHƯƠNG 4. THỰC NGHIỆM VÀ ĐÁNH GIÁ

Tại chương 4, tôi xin đi sâu vào trình bày những kết quả đã đạt được của đồ án và kiểm thử sản phẩm để đánh giá độ hoàn thiện.

### 4.1 Xây dựng ứng dụng

#### 4.1.1 Thư viện và công cụ sử dụng

Các thư viện và công cụ sử dụng được trình bày trong bảng 4.1

**Bảng 4.1:** Danh sách thư viện và công cụ sử dụng

Công cụ và thư viện	Mục đích	Địa chỉ URL
Pycharm	IDE lập trình	<a href="https://www.jetbrains.com/pycharm/">https://www.jetbrains.com/pycharm/</a>
Python	Lập trình Jetson Nano, hỗ trợ giao tiếp với máy chủ và xử lý nhận diện cử chỉ tay	<a href="https://www.python.org/">https://www.python.org/</a>
FastApi	Giao tiếp với máy chủ	<a href="https://fastapi.tiangolo.com/">https://fastapi.tiangolo.com/</a>
Electron	Xây dựng giao diện	<a href="https://www.electronjs.org/">https://www.electronjs.org/</a>
Bootstrap	Xây dựng giao diện	<a href="https://getbootstrap.com/">https://getbootstrap.com/</a>
Python-Shell	Hỗ trợ giao tiếp giữa Python và JavaScript	<a href="https://www.npmjs.com/package/python-shell">https://www.npmjs.com/package/python-shell</a>
TensorFlow	Xây dựng mô hình nhận diện cử chỉ tay	<a href="https://www.tensorflow.org/">https://www.tensorflow.org/</a>
MediaPipe	Xây dựng mô hình nhận diện cử chỉ tay	<a href="https://mediapipe.dev/">https://mediapipe.dev/</a>
OpenCV	Xây dựng mô hình nhận diện cử chỉ tay, tiền xử lý hình ảnh và lưu dòng hình ảnh	<a href="https://opencv.org/">https://opencv.org/</a>

### 4.1.2 Triển khai

#### a, Phần cứng

Máy tính cá nhân HP Envy 14, CPU core i7 7700U, 8GB RAM

Camera gắn ngoài có độ phân giải  $1080 \times 1920$

Thiết bị Jetson Nano, 4GB RAM, GPU 128-core Maxwell, CPU Quad-core ARM A57 @ 1.43 GHz

#### b, Cấu hình

Sau quá trình thử nghiệm, tôi nhận thấy những tham số sau là tốt nhất đối với thư viện MediaPipe và mô hình cử chỉ tay.

##### Thư viện MediaPipe:

- STATIC-IMAGE-MODE: False
- MAX-NUM-HANDS: 2
- MODEL-COMPLEXITY: 1
- MIN-DETECTION-CONFIDENCE: 0,8
- MIN-TRACKING-CONFIDENCE: 0,5

Tham số STATIC-IMAGE-MODE được sử dụng trong mô hình phát hiện khung xương bàn tay. Nếu giá trị này là True, đầu vào của mô hình sẽ là ảnh và ngược lại.

Tham số MAX-NUM-HANDS là số bàn tay tối đa có thể phát hiện được.

Tham số MODEL-COMPLEXITY là độ phức tạp của mô hình, có giá trị từ 0 đến 1. Giá trị tham số càng gần 1 độ phức tạp của mô hình càng lớn và độ chính xác, độ trễ cũng tăng theo.

Tham số MIN-DETECTION-CONFIDENCE là giá trị tin cậy tối thiểu để xác nhận đó là một bàn tay.

Tham số MIN-TRACKING-CONFIDENCE là giá trị tối thiểu để các mốc bàn tay được coi là theo dõi thành công, giá trị càng lớn độ tin cậy càng cao nhưng độ trễ cũng tăng lên.

##### Mô hình cử chỉ tay:

- Mô hình EfficientNet: (i) learning rate: 0,0001 (ii) active function: categorical crossentropy (iii) dropout: 0,2 (iv) thêm 3 lớp sau mô hình ban đầu: Global-AveragePooling2D, BatchNormalization và Dense.
- Mô hình MobileNetV2: Cùng với bộ tham số như EfficientNet nhưng thay đổi dropout là 0,18 và cài đặt tham số của ở các lớp *trainable = false* trong 20

vòng đầu, thay đổi lại *trainable = true* trong những vòng còn lại.

#### 4.1.3 Kết quả đạt được

Sau quá trình làm đồ án, kết quả tôi đã xây dựng được một sản phẩm để điều khiển các chức năng thông minh của camera sau khi được kết nối với máy chủ. Sản phẩm gồm 2 thành phần chính là (i) ứng dụng máy tính (ii) sử dụng thiết bị Jetson Nano điều khiển camera. Bằng việc sử dụng sản phẩm của tôi, người dùng có thể giao tiếp với máy chủ, nhận thông báo từ máy chủ khi có sự kiện bất thường xảy ra, sử dụng camera để quản lý tài sản, con người tốt hơn. Thống kê về thông tin sản phẩm được trình bày trong bảng 4.2.

**Bảng 4.2:** Thống kê thông tin sản phẩm

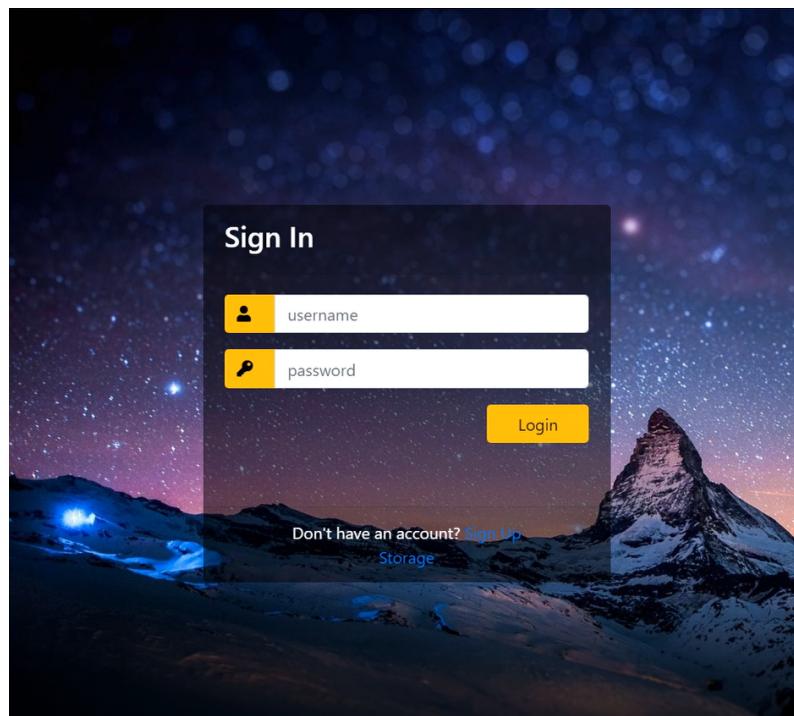
Thông tin	Thống kê
Số dòng code mã nguồn ứng dụng máy tính	5500 dòng
Số dòng code mã nguồn thiết bị Jetson Nano	100 dòng
Dung lượng mã nguồn ứng dụng máy tính	2.5GB
Dung lượng mã nguồn thiết bị Jetson Nano	15MB

#### 4.1.4 Minh họa các chức năng chính

##### a, Các chức năng của ứng dụng máy tính

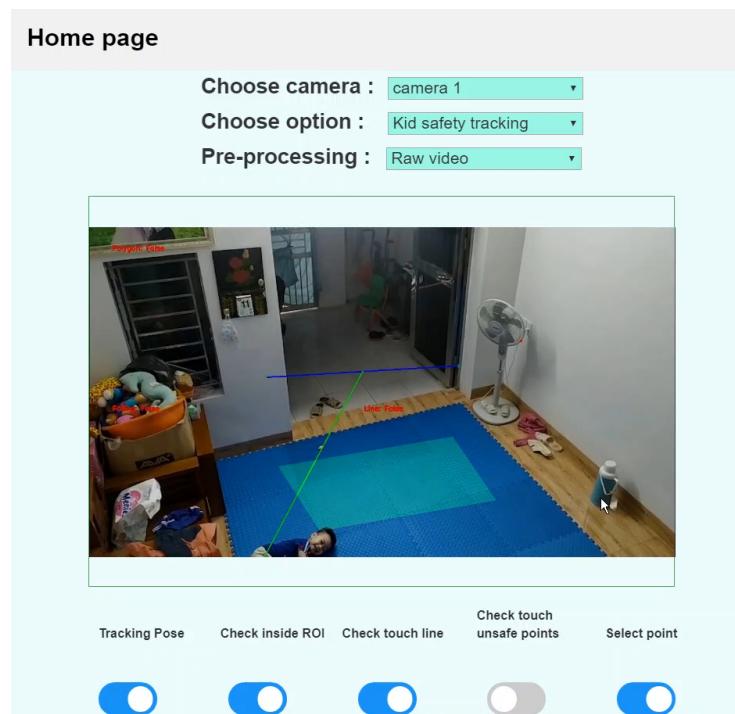
Kịch bản thử nghiệm các chức năng của ứng dụng máy tính được thực hiện theo luồng như sau:

- Đăng nhập và tiến hành sử dụng ứng dụng.
- Chọn chức năng xem dòng video đã qua xử lý, sau đó chọn camera, chức năng cần xử lý và gửi yêu cầu đến máy chủ
- Chọn chức năng xử lý ảnh, chọn ảnh và chức năng cần xử lý và gửi yêu cầu đến máy chủ
- Chọn chức năng quản lý tài khoản và thực hiện các chức năng nhỏ trong đó.

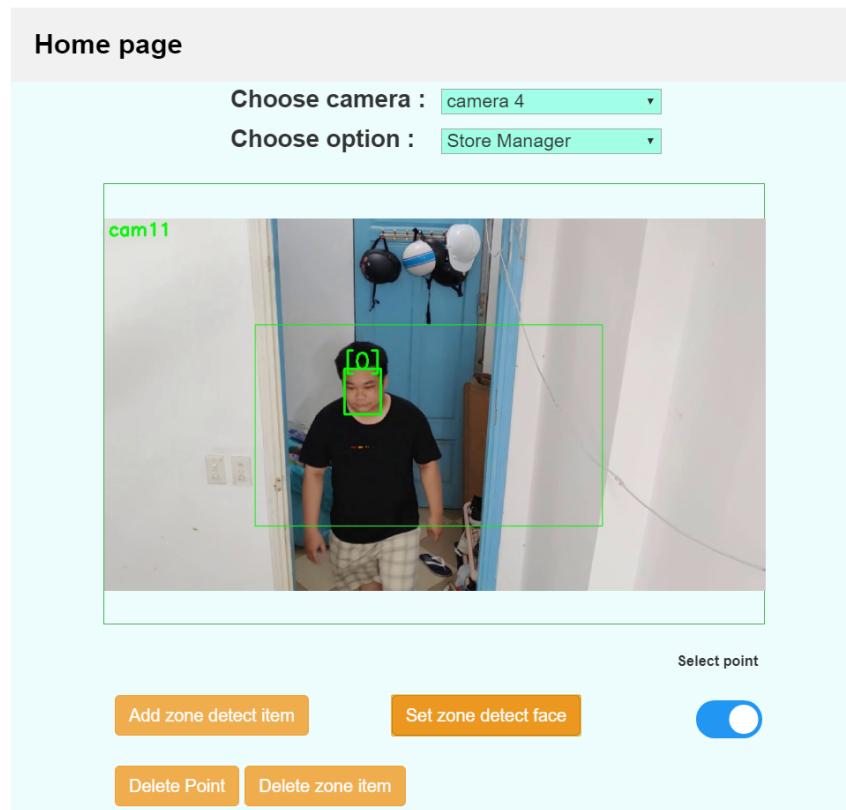


**Hình 4.1:** Giao diện chức năng đăng nhập

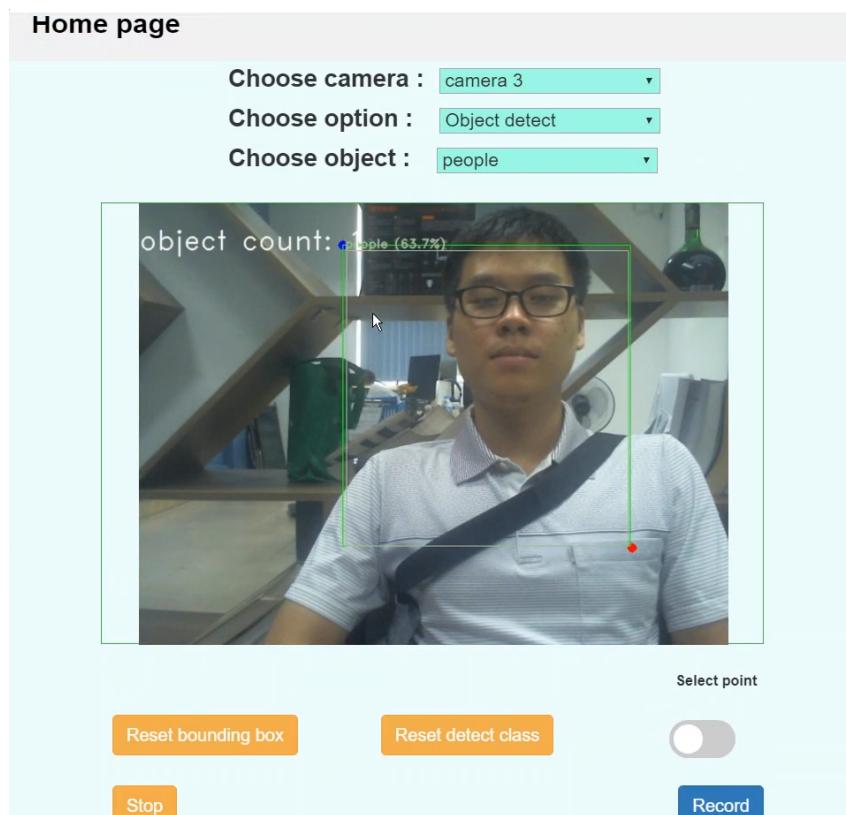
Hình 4.1 là giao diện đăng nhập. Trong giao diện này, người dùng sẽ nhập tài khoản và mật khẩu để đăng nhập. Nếu chưa có tài khoản, có thể ấn vào nút đăng ký để tạo tài khoản mới. Ngoài ra, nếu người dùng đã từng sử dụng ứng dụng và có lưu thông báo hoặc dòng video, người dùng có thể xem lại bằng cách ấn nút lưu trữ để xem lại mà không cần đăng nhập.



**Hình 4.2:** Kiểm tra an toàn cho trẻ em

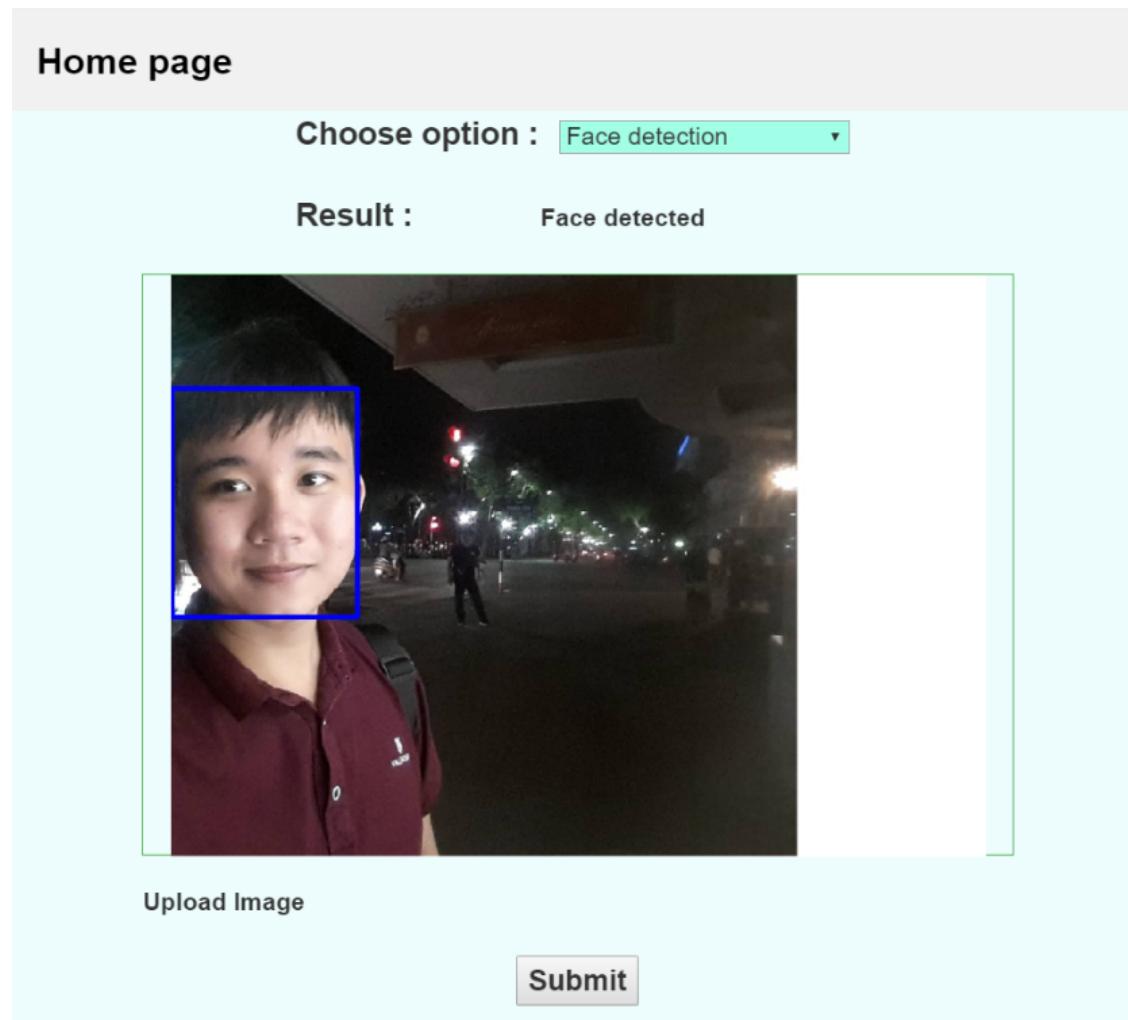


**Hình 4.3:** Quản lý cửa hàng



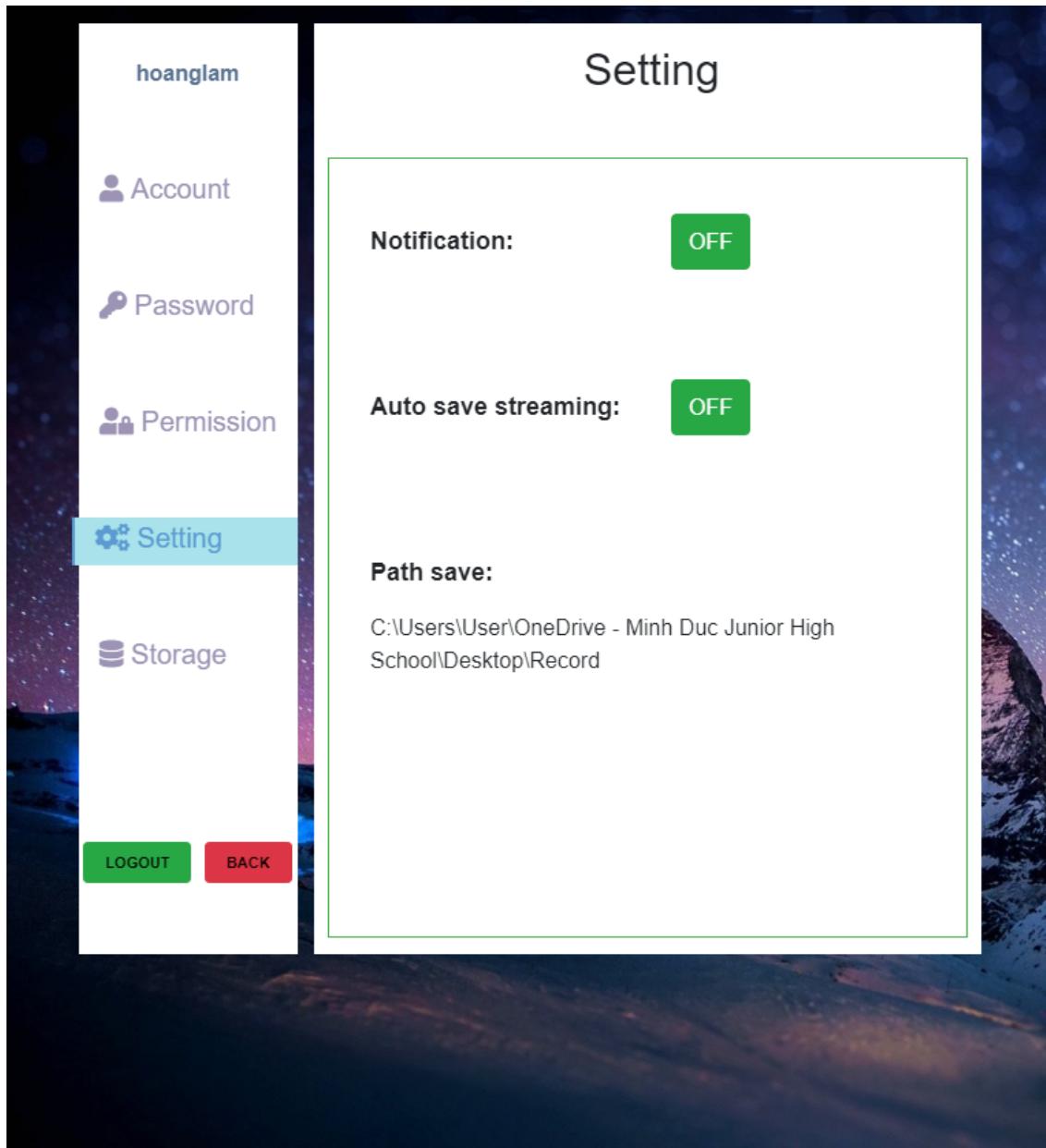
**Hình 4.4:** Nhận diện đối tượng

Ba hình 4.2, 4.3, 4.4 là giao diện cho chức năng xem dòng hình ảnh. Để xem dòng người dùng sẽ chọn camera và chọn chức năng cần sử dụng. Ứng với mỗi chức năng sẽ có những cài đặt chi tiết khác nhau. Người dùng cũng có thể lưu lại dòng hình ảnh bằng cách ấn nút lưu ở giao diện.



**Hình 4.5:** Giao diện xử lý ảnh

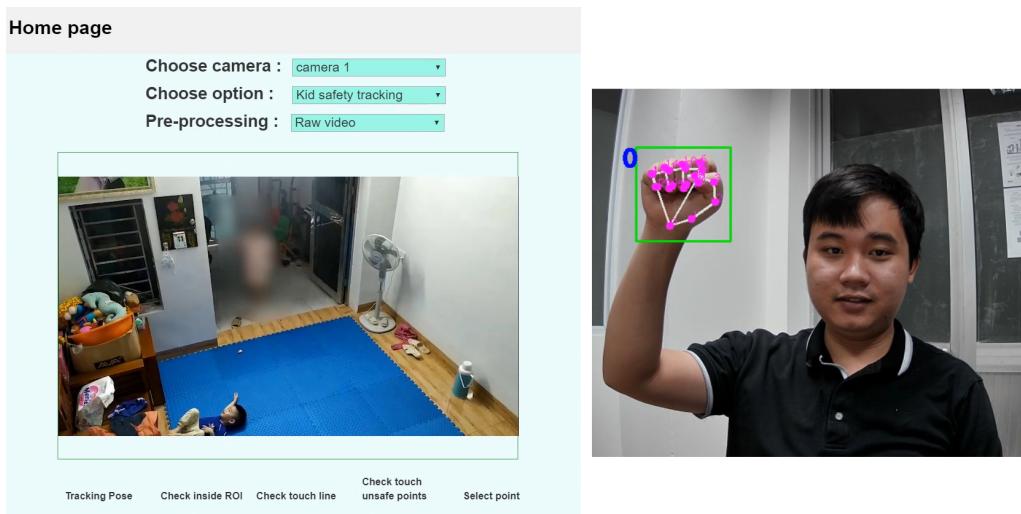
Hình 4.5 là giao diện của chức năng xử lý ảnh. Người dùng sau khi chọn chức năng và hình ảnh cần xử lý sẽ ấn nút gửi đến máy chủ. Sau khi xử lý thành công, máy chủ sẽ trả về kết quả hiển thị cho người dùng.



**Hình 4.6:** Giao diện quản lý tài khoản

Hình 4.6 là giao diện của chức năng quản lý tài khoản, người dùng có thể xem thông tin tài khoản, thay đổi mật khẩu và quyền sử dụng các camera. Người dùng cũng có thể cài đặt cấu hình cho ứng dụng như có nhận thông báo hay có tự động lưu dòng hình ảnh hay không.

## b, Chức năng điều khiển cử chỉ tay trên ứng dụng máy tính và thiết bị biên Jetson Nano



**Hình 4.7:** Điều khiển camera bằng cử chỉ tay

Hình 4.7 là giao diện của chức năng cử chỉ tay. Ứng dụng sử dụng mô hình nhẹ ít tham số để nhận dạng cử chỉ tay. Sau quá trình thực nghiệm, tôi thu được kết trên các mô hình khác nhau như bảng 4.3.

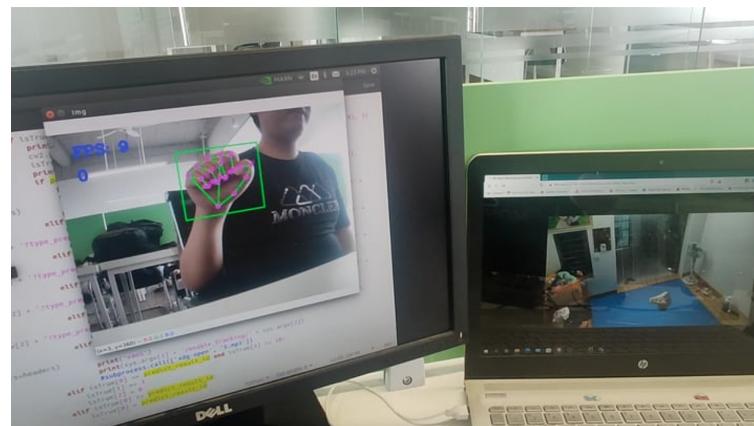
**Bảng 4.3:** Kết quả thực nghiệm các mô hình cử chỉ tay

Mô hình	Số lượng tham số	Độ chính xác
MobileNetV0	3.239.110	94%
MobileNetV2	2.270.790	93%
MobileNetV3Large	3.005.958	94%
MobileNetV3Small	944.886	88%
EfficientNet	4.062.377	96%
NASNetMobile	4.280.282	93%
DenseNet121	7.047.750	98%

Sau khi thực nghiệm, tôi nhận thấy mô hình DenseNet121 cho độ chính xác cao nhất nhưng số lượng tham số lớn hơn so với những mô hình còn lại. Trong khi đó mô hình EfficientNet có số lượng tham số nhỏ hơn 43% nhưng vẫn cho độ chính xác đạt 96%. Tốc độ xử lý của mô hình EfficientNet khi chạy trên máy tính cá nhân cũng đạt số khung hình trên giây trung bình là 9,3. Các chỉ số này phù hợp với mục tiêu đề ra do đó tôi đã lựa chọn mô hình EfficientNet làm mô hình nhận diện cử chỉ tay cho ứng dụng máy tính.



**Hình 4.8:** Thiết bị biên Jetson Nano



**Hình 4.9:** Thiết bị biên Jetson Nano



**Hình 4.10:** Thiết bị biên Jetson Nano

Hình 4.8, 4.9, 4.10 là hình ảnh của thiết bị biên Jetson Nano được nối với camera. Sau khi nhận được cử chỉ tay từ camera, Jetson Nano sẽ dự đoán cử chỉ tay và gửi yêu cầu tương ứng đến máy chủ để điều khiển cử chỉ tay. Thông báo nhận được từ máy chủ sẽ được phát qua loa.

Trong quá trình thực nghiệm, tôi nhận thấy với những mô hình có số lượng tham số lớn hơn 4.000.000 Jetson Nano sẽ không thể khởi chạy được, trong khi đó với những mô hình có số lượng tham số khoảng 3.000.000 tuy có thể khởi chạy nhưng cho tốc độ xử lý chậm biếu thị bởi số khung hình trên giây trung bình chỉ rơi vào khoảng 4 hoặc 5. Nhìn vào bảng 4.3 có thể thấy được mô hình MobileNetV2 là phù hợp do có số lượng tham số nhỏ và có độ chính xác đạt 94%. Trong quá trình chạy thiết bị Jetson Nano sử dụng mô hình MobileNetV2 tôi thu được kết quả như bảng 4.4.

**Bảng 4.4:** Kết quả mô hình MobileNetv2 chạy trên thiết bị biên Jetson Nano

Số khung hình trên giây	8-14
Ram sử dụng	3,5/4 Gb
Bộ nhớ trao đổi sử dụng	1,1/4 G
GPU sử dụng	30% - 95%

Nhìn vào bảng 4.4 ta thấy mô hình MobileNetV2 khi chạy trên thiết bị biên Jetson Nano cho tốc độ khung hình trên giây trung bình lớn hơn 8, những chỉ số sử dụng tài nguyên cũng chưa đạt tối đa. Như vậy, mô hình đã đạt được những yêu cầu đã đề ra.

## 4.2 Kiểm thử và đánh giá

### 4.2.1 Kiểm thử mô hình cử chỉ tay

Tôi đã tiến hành kiểm thử mô hình cử chỉ tay trên thiết bị Jetson Nano và ứng dụng máy tính.

#### a, Chuẩn bị phần cứng

(i) Thiết bị biên Jetson Nano (ii) máy tính cá nhân CPU intel core i7 7700U 8GB RAM (iii) camera có độ phân giải  $1080 \times 1920$ .

#### b, Môi trường kiểm thử

Tôi lần lượt kiểm thử mô hình trong điều kiện ánh sáng bình thường, ánh sáng yếu và ánh sáng mạnh.

#### c, Kết quả thu được

Trong điều kiện ánh sáng bình thường, ứng dụng máy tính nhận diện và thiết bị Jetson Nano và phân biệt được các cử chỉ tay, khoảng cách nhận diện tối đa là  $4m$  đối với máy tính và  $3m$  đối với thiết bị Jetson Nano.

Trong điều kiện ánh sáng yếu và ánh sáng mạnh, khoảng cách tối đa có thể nhận diện được cử chỉ tay là  $2,5m$  đối với ứng dụng máy tính và  $2m$  đối với thiết bị biên Jetson Nano.

### 4.2.2 Kiểm thử ứng dụng máy tính

Để đánh giá độ hoàn thiện của ứng dụng, tôi đã sử dụng phương pháp kiểm thử hộp đen để tiến hành kiểm thử áp dụng với 3 chức năng chính của ứng dụng (i) đăng nhập (ii) xem dòng hình ảnh đã qua xử lý (iii) xử lý ảnh. Sau đó tiến hành kiểm thử đối với các yêu cầu phi chức năng. Dưới đây là kết quả kiểm thử.

#### a, Kiểm thử chức năng đăng nhập

Mô tả yêu cầu: Đối với chức năng đăng nhập, ứng dụng cần xử lý dữ liệu đầu vào được nhập. Nếu dữ liệu bỏ trống, thông tin nhập sai phải thông báo lỗi đến người dùng. Khi nhập đúng thông tin tài khoản, phải chuyển người dùng đến giao diện sau khi đăng nhập.

**Bảng 4.5:** Kiểm thử chức năng đăng nhập

STT	Mục đích	Quy trình		Kết quả thực tế	Đánh giá
		Đầu vào	Đầu ra		
1	Kiểm tra bắt buộc nhập dữ liệu	Bỏ trống tài khoản hoặc mật khẩu	Thông báo yêu cầu nhập tất cả các trường	Thông báo yêu cầu nhập tất cả các trường	Đạt
2	Kiểm tra dữ liệu chứa ký tự đặc biệt	Nhập tài khoản hoặc mật khẩu chứa ký tự đặc biệt	Thông báo lỗi chứa ký tự đặc biệt	Thông báo lỗi chứa ký tự đặc biệt	Đạt
3	Kiểm tra sai dữ liệu	Nhập tài khoản hoặc mật khẩu không chính xác	Thông báo lỗi sai thông tin đăng nhập	Thông báo lỗi sai thông tin đăng nhập	Đạt
4	Kiểm tra dữ liệu chính xác	Nhập tài khoản hoặc mật khẩu chính xác	Chuyển giao diện về màn sau khi đăng nhập	Chuyển giao diện về màn sau khi đăng nhập	Đạt

### b, Kiểm thử chức năng xem dòng đã xử lý

Đây là chức năng giúp cho người dùng có thể điều khiển các chức năng liên quan đến camera. Giúp người dùng xem dòng hình ảnh tương ứng với camera và chức năng đã chọn. Sau đây là nội dung kiểm thử.

Mô tả yêu cầu: Đối với chức năng xem dòng đã qua xử lý, người dùng cần chọn camera và chức năng trước khi yêu cầu máy chủ trả về dòng hình ảnh. Nếu chưa hoàn tất các bước này, ứng dụng sẽ thông báo lỗi yêu cầu thao tác lại. Sau khi lựa chọn thành công, ứng dụng sẽ hiển thị dòng hình ảnh đã qua xử lý cho người dùng. Chi tiết xem tại bảng 4.6 .

**Bảng 4.6:** Kiểm thử chức năng xem dòng đã xử lý

STT	Mục đích	Quy trình		Kết quả thực tế	Đánh giá
		Đầu vào	Đầu ra		
1	Kiểm tra đã hoàn tất các lựa chọn	Không lựa chọn camera hoặc chức năng xử lý	Thông báo yêu cầu hoàn tất mọi thao tác	Thông báo yêu cầu hoàn tất mọi thao tác	Đạt
2	Kiểm tra chức năng có phù hợp với camera	Chọn chức năng mà camera đã chọn không hỗ trợ	Thông báo camera không hỗ trợ chức năng vừa chọn	Thông báo camera không hỗ trợ chức năng vừa chọn	Đạt
3	Kiểm tra hoạt động của ứng dụng	Lựa chọn đúng camera và chức năng	Hiển thị dòng hình ảnh đã qua xử lý tương ứng với lựa chọn	Hiển thị dòng hình ảnh đã qua xử lý tương ứng với lựa chọn	Đạt

### c, Kiểm thử chức năng xử lý ảnh

Đây là chức năng cho phép người dùng xử lý một bức ảnh bất kỳ bằng các chức năng tiền xử, nhận diện do máy chủ cung cấp. Dưới đây là nội dung kiểm thử.

Mô tả yêu cầu: Đối với chức năng xử lý ảnh, ứng dụng cần đảm bảo người dùng đã chọn chức năng và ảnh cần xử lý, nếu người dùng chưa chọn cần đưa ra thông báo cho người dùng. Trong quá trình xử lý, nếu máy chủ có lỗi cần thông báo cho người dùng. Nếu thao tác của người dùng và máy chủ đều chính xác thì ứng dụng sẽ hiển thị kết quả và ảnh đã xử lý cho người dùng. Chi tiết xem tại bảng 4.7 .

**Bảng 4.7:** Kiểm thử chức năng xử lý ảnh

STT	Mục đích	Quy trình		Kết quả thực tế	Đánh giá
		Đầu vào	Đầu ra		
1	Kiểm tra đã hoàn tất các lựa chọn	Không lựa chọn chức năng hoặc ảnh	Thông báo yêu cầu hoàn tất mọi thao tác	Thông báo yêu cầu hoàn tất mọi thao tác	Đạt
2	Kiểm tra ứng dụng khi máy chủ gặp sự cố	Gửi yêu cầu đến máy chủ khi máy chủ gặp sự cố	Thông báo máy chủ đang bị lỗi	Thông báo máy chủ đang bị lỗi	Đạt
3	Kiểm tra hoạt động của ứng dụng	Máy chủ đang hoạt động và người dùng lựa chọn đúng chức năng và ảnh	Hiển thị hình ảnh đã qua xử lý tương ứng với lựa chọn	Hiển thị hình ảnh đã qua xử ly tương ứng với lựa chọn	Đạt

#### d, Các yêu cầu phi chức năng

Tôi đã kiểm thử đối với yêu cầu phi chức năng về kỹ thuật được trình bày ở mục 2.5. Dưới đây là kết quả kiểm thử:

- Ứng dụng không thể sử dụng các chức năng nếu không đăng nhập.
- Thời gian phản hồi của ứng dụng tối đa khi sử dụng các chức năng xử lý ảnh và xem dòng hình ảnh là 0,08 giây.
- Thời gian sau khi nhận cử chỉ tay và phản hồi lại tối đa là 0,039 giây.
- Mô hình cử chỉ tay có số khung hình trên giây trung bình là 9,3 được thực hiện đo trong thời gian một phút.

Đối với yêu cầu phi chức năng về giao diện người dùng, tôi đã thực hiện khảo sát giao diện đối với 20 người. Kết quả khảo sát, giao diện của ứng dụng được đánh giá đơn giản dễ sử dụng, những nút bấm tên dễ hiểu, không bị nhầm khi thao tác. Tuy nhiên giao diện chưa được đẹp về mặt hình thức cần cải thiện thêm.

#### 4.2.3 Đánh giá

Từ những thử nghiệm và kiểm thử trên, tôi đánh giá sản phẩm này đã đạt được những yêu cầu đề ra. Do sử dụng những mô hình nhẹ nên khi triển khai cả ứng dụng và thiết bị đều có tốc độ xử lý đảm bảo yêu cầu. Ứng dụng có thể điều khiển camera bằng cả hai cách thông thường và cử chỉ tay. Trong những điều kiện như thiếu sáng và ánh sáng mạnh, tuy việc nhận diện bị giảm đi độ hiệu quả nhưng ở khoảng cách gần hơn, mô hình vẫn có thể phân biệt được các cử chỉ tay được huấn luyện. Tuy nhiên, ứng dụng vẫn còn gặp những khuyết điểm như giao diện chưa bắt mắt và chưa thực sự chạy trong thực tế, cần có thêm thời gian để cải thiện giúp ứng dụng có thể trở thành một sản phẩm hoàn chỉnh sử dụng được trong thực tế.

## **CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT**

Chương 5 trình bày về một số công hiến nổi bật trong đồ án này.

### **5.1 Mô hình nhận diện cử chỉ tay dựa trên các điểm chính trên bàn tay kết hợp mô hình học sâu**

#### **5.1.1 Đặt vấn đề**

Trong quá trình xây dựng ứng dụng, để điều khiển các chức năng camera, tôi phải sử dụng bàn phím và chuột để thay đổi chức năng. Vì vậy, có đôi lúc tôi muốn điều khiển các chức năng này ở một khoảng cách nhất định nhưng không thể thực hiện được.Thêm vào đó, đối với những người ít có cơ hội tiếp cận với công nghệ, việc sử dụng máy tính, điện thoại có thể gặp những khó khăn. Mong muốn giúp những người này có thể thao tác dễ dàng khi sử dụng ứng dụng chỉ với những thao tác đơn giản, tôi đã nảy ra ý tưởng sử dụng cử chỉ tay đơn giản để điều khiển camera.

Việc nhận diện những cử chỉ tay đối với con người là không khó, nhưng đối với máy tính thì cần phải từ chuỗi hình ảnh đầu vào tính toán hợp lý để dự đoán. Hơn nữa, đối với các góc quay, độ sáng khác nhau có thể ảnh hưởng đến kết quả nhận diện của mô hình.

#### **5.1.2 Giải pháp**

Để giải quyết vấn đề trên, ban đầu tôi đã nghĩ đến việc xây dựng một mô hình học sâu để dự đoán khung xương của tay, từ đó kết hợp những thuật toán để dự đoán cử chỉ tay. Mô hình giai đoạn ban đầu tôi chọn là HRNet, có thể cho kết quả là các điểm trên khung xương của bàn tay dựa trên ảnh đầu vào. Tuy nhiên, để ra được kết quả mà không sử dụng GPU, mô hình HRNet[18] phải tốn thời gian khá lâu khiến cho việc sử dụng mô hình này sẽ khó để thiết kế ứng dụng thời gian thực. Qua quá trình nghiên cứu và thử nghiệm thêm những mô hình khác, tôi đã phát hiện ra thư viện MediaPipe, là thư viện được Goole cung cấp chuyên xử lý các chức năng thông minh liên quan đến thị giác máy tính. Mô hình của thư viện MediaPipe có thể cho kết quả dự đoán các điểm trên khung xương bàn tay rất tốt cùng với tốc độ xử lý cao, có thể lên tới 30 FPS khi chạy với thiết bị biên Jetson Nano. Vì những ưu điểm này, tôi đã sử dụng thư viện MediaPipe[14] với mục đích dự đoán các điểm trên bàn tay.

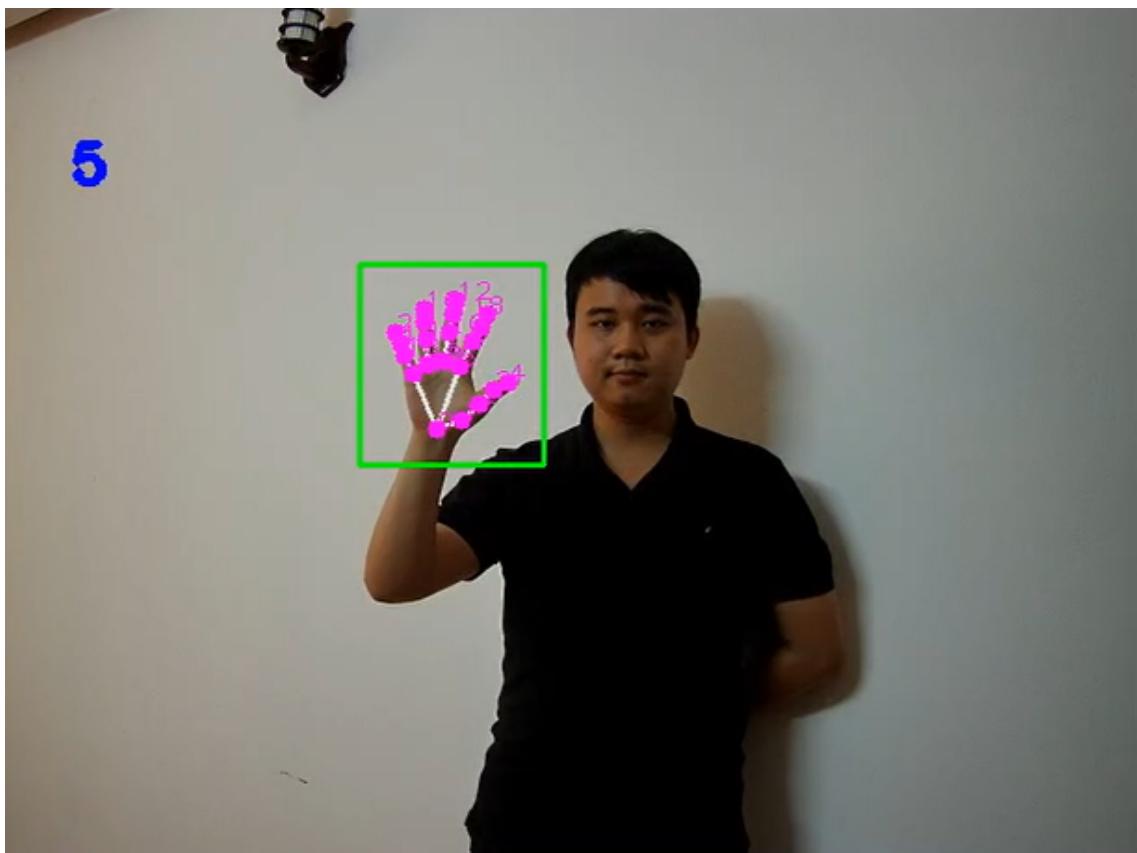
Sau khi có các điểm trên khung xương bàn tay, tôi tìm kiếm những thuật toán để phân loại các điểm này. Mặc dù các thuật toán này có thể phân biệt được những cử chỉ cơ bản, nhưng với những cử chỉ dễ nhầm lẫn như cử chỉ giơ 2 ngón tay chỏ,

út và cử chỉ giơ 2 ngón tay cái, giữa, những thuật toán này tỏ ra không hiệu quả, dẽ nhầm lẫn. Vì vậy, tôi đã nghĩ đến việc sử dụng mạng tích chập gồm những lớp đơn giản để dự đoán cử chỉ với đầu vào là những điểm trên khung bàn tay đã được chuẩn hóa. Kết quả thu được, tôi đã xây dựng được mô hình với độ chính xác xấp xỉ 80%. Tôi đã cố gắng tăng thêm độ phức tạp và tiền xử lý dữ liệu bằng những phương pháp khác nhau, nhưng độ chính xác của mô hình vẫn không tăng quá nhiều.

Để cải thiện độ chính xác cho đầu ra của mô hình, tôi quyết định sử dụng những mạng học sâu nhẹ, vừa đủ để nhận diện cử chỉ tay thông qua hình ảnh. Với các điểm từ khung xương thu được ở bước trước, tôi có thể lấy được hình ảnh bàn tay mà không có quá nhiều những ngoại cảnh. Với ý tưởng sử dụng những mạng học sâu nhẹ, tôi đã thử nghiệm với mạng MobileNetV0, kết quả thu được với đầu vào là tập dữ liệu chỉ gồm bàn tay là 98%. Đây là một độ chính xác khá cao, nhưng khi thử nghiệm trong thực tế, có nhiều khung ảnh có kết quả đầu ra không phù hợp với cử chỉ tay. Nhận thấy mô hình có dấu hiệu tốt với tập huấn luyện nhưng không tốt với thực nghiệm, tôi đã sử dụng thêm những mô hình khác để so sánh kết quả. Những mô hình sau đó cũng cho kết quả tương tự, mô hình chạy trên thực nghiệm kém. Sau quá trình nghiên cứu và thử nghiệm thêm, tôi đã nhận thấy, nếu cứ mỗi 10 đơn vị trong mạng ta bỏ qua 2 đơn vị, sẽ khiến cho tình trạng quá khớp kể trên được cải thiện. Tuy độ chính xác mô hình khó đạt được đến 98% như ban đầu, nhưng kết quả thực nghiệm lại tốt hơn hẳn. Áp dụng phương pháp này với các mạng nhẹ khác nhau, tôi đã tìm ra được mô hình sử dụng mạng EfficientNet có kết quả tốt nhất với 96%. Quá trình thực nghiệm cũng chứng minh độ hiệu quả của mô hình này.

### 5.1.3 Kết quả đạt được

Sau khi thực nghiệm và lựa chọn mô hình EfficientNet[16], tôi đã thử nghiệm trên thiết bị máy tính và camera với thông số đã đề cập ở chương 4. Dưới đây là hình ảnh khi chạy thực nghiệm của mô hình.



**Hình 5.1:** Kết quả mô hình nhận diện cử chỉ tay

Mô hình đã dự đoán kết quả là 5, chính xác với chỉ số tôi đã đặt cho cử chỉ xòe bàn tay.

## 5.2 Xây dựng bộ dữ liệu để huấn luyện mô hình nhận diện cử chỉ tay

### 5.2.1 Đặt vấn đề

Để có thể huấn luyện mô hình nhận diện cử chỉ tay sao cho có độ chính xác cao cả trong quá trình thử nghiệm và thực nghiệm, thì một bộ dữ liệu tốt để huấn luyện và đánh giá là điều không thể thiếu. Bộ dữ liệu yêu cầu phải có số lượng ảnh đủ lớn, các cử chỉ tay khác nhau, số người tham gia chụp ảnh đa dạng, cùng với đó là ánh sáng, độ phân giải của hình ảnh phải đa dạng để mô hình được huấn luyện trong nhiều môi trường. Hiện nay, trong quá trình tôi tìm kiếm, tôi nhận thấy chưa có một bộ dữ liệu hoàn chỉnh nào đáp ứng những tiêu chí vừa nêu.

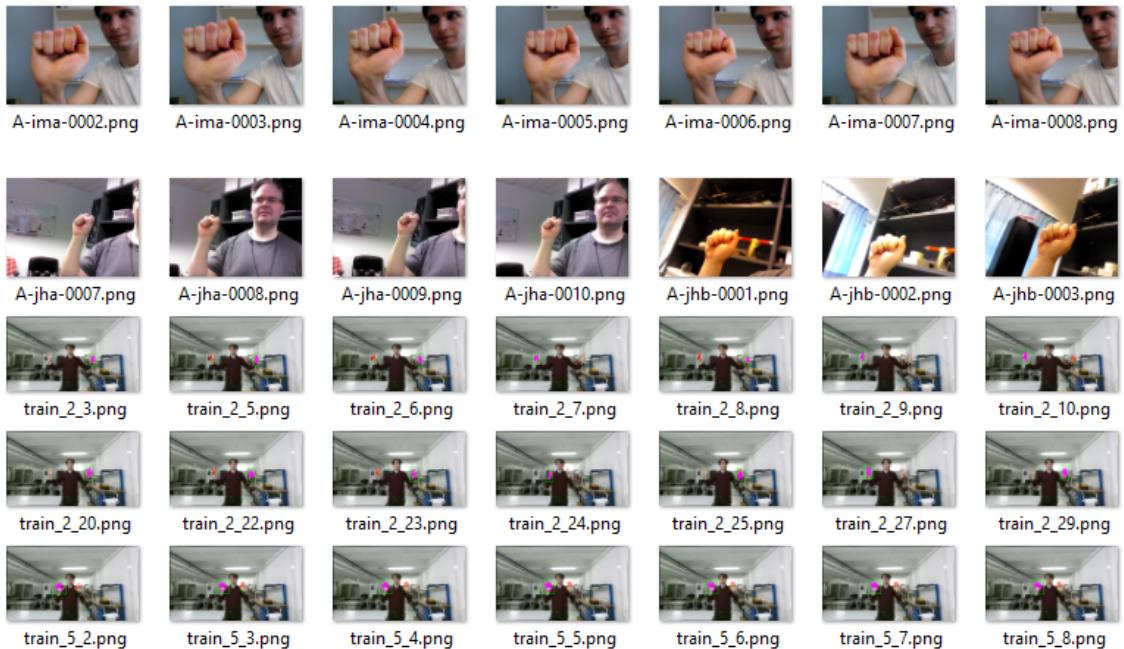
### 5.2.2 Giải pháp

Qua quá trình tìm kiếm để giải quyết vấn đề được nêu ra ở mục 5.2.1, tôi có tìm thấy nhiều những bộ dữ liệu về cử chỉ tay. Tuy nhiên những bộ dữ liệu này lại có những khuyết điểm và không phù hợp với tất cả tiêu chí. Có những bộ dữ liệu có chất lượng ảnh tốt, đa dạng về người chụp và cử chỉ tay nhưng số lượng ảnh lại rất ít, chỉ khoảng 10 ảnh. Một số bộ dữ liệu có số lượng hình ảnh lớn, nhưng chất

lượng hình ảnh kém, khi đưa vào mô hình khiến kết quả đầu ra kém. Để trước vấn đề này, tôi đã nghĩ ra giải pháp kết hợp những bộ dữ liệu sẵn có thành một bộ dữ liệu, gồm 6 cử chỉ tay và được chụp bởi những người từ các nước khác nhau. Bộ dữ liệu thu được nhờ sự kết hợp của ba bộ dữ liệu Hands, Ouhands và Microsoft Kinect and Leap Motion đã đáp ứng đủ về số lượng và chất lượng hình ảnh để huấn luyện và đánh giá mô hình. Sau đó, tự chụp những cử chỉ tay tương tự để tạo nên một bộ dữ liệu hoàn chỉnh.

### 5.2.3 Kết quả đạt được

Sau khi kết hợp các bộ dữ liệu có sẵn, tôi đã thu được kết quả là bộ dữ liệu gồm 6 cử chỉ tay, có tổng số lượng hình ảnh là 6600, tương ứng với mỗi cử tay có khoảng 1100 hình ảnh [19]. Bộ dữ liệu này sẽ được đóng góp cho cộng đồng phục vụ mục đích nghiên cứu và học tập những bài toán liên quan.



**Hình 5.2:** Bộ dữ liệu cử chỉ tay

## 5.3 Xây dựng ứng dụng điều khiển các chức năng camera

### 5.3.1 Đặt vấn đề

Như đã nêu nguyên nhân và hướng giải pháp ở chương 1, để biến những camera thông thường trở nên thông minh, những dòng hình ảnh từ camera đó phải đi qua một máy chủ, nơi chịu trách nhiệm xử lý dòng hình ảnh này bằng những chức năng thông minh và gửi lại dòng hình ảnh cùng thông báo nếu có cho người dùng. Chính vì vậy, để người dùng có thể giao tiếp được với máy chủ và có thể xem dòng hình ảnh đã xử lý, thì một ứng dụng có thể hỗ trợ người dùng thực hiện những chức năng

này là cần thiết. Ứng dụng này sẽ có thể giúp người dùng xác thực danh tính bằng tài khoản, điều khiển những chức năng mà máy chủ cung cấp trên những camera hiện có, nhận thông báo từ máy chủ và quản lý tài khoản của người dùng.

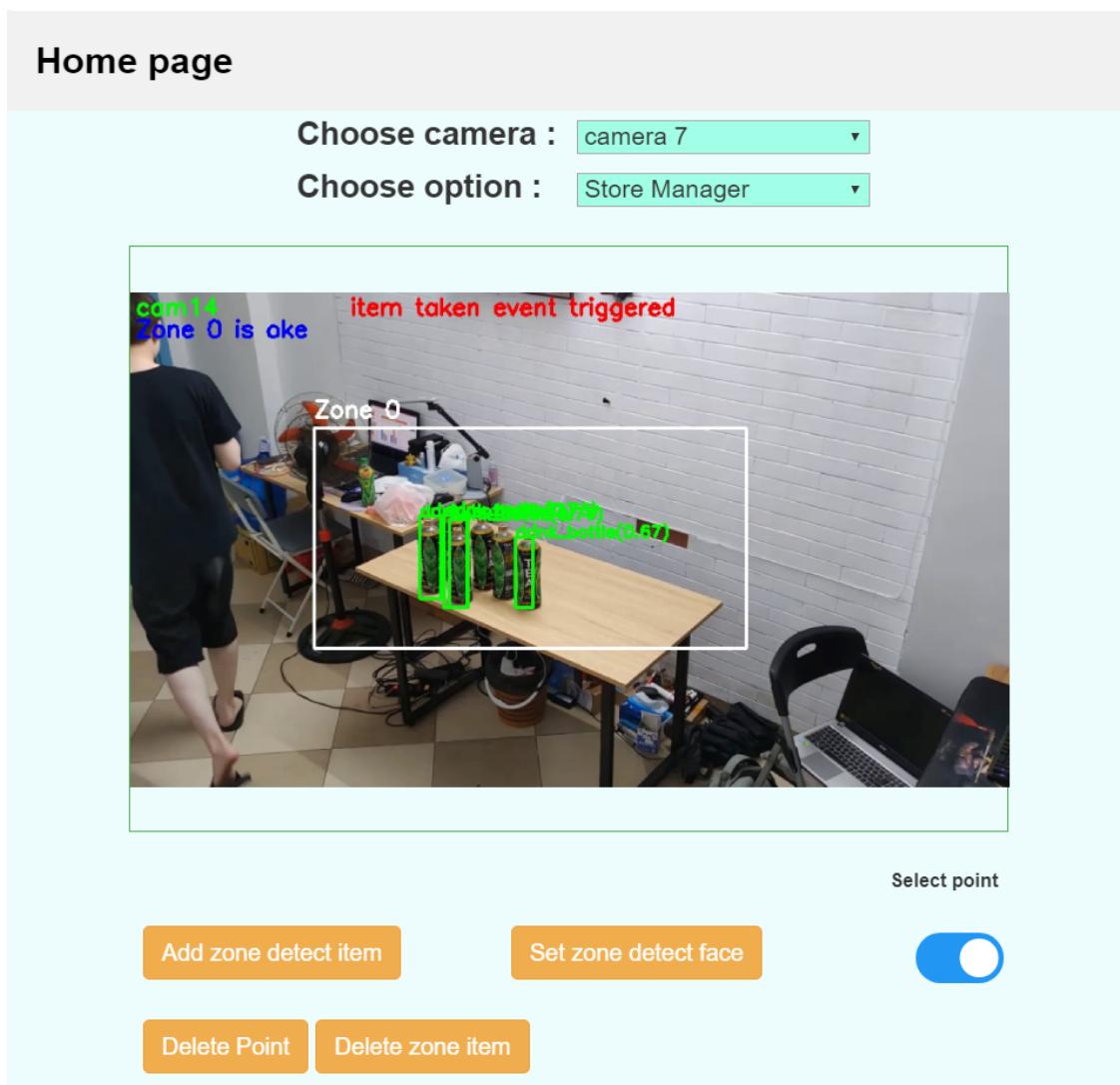
### 5.3.2 Giải pháp

Ban đầu, tôi dự định sẽ sử dụng ngôn ngữ Python cùng thư viện PyQt để lập trình ứng dụng. Tuy nhiên, sau khi triển khai một thời gian, nhận thấy Python không hỗ trợ những dịch vụ như Firebase, giao diện được thiết kế đơn giản, không bắt mắt cùng một số những bất tiện khi lập trình, tôi đã quyết định sử dụng ngôn ngữ khác để xây dựng ứng dụng. Qua quá trình tìm hiểu, tôi thấy Electron hỗ trợ người dùng sử dụng những công nghệ web để xây dựng ứng dụng. Việc sử dụng những công nghệ web sẽ giúp giải quyết những khó khăn về sử dụng các dịch vụ sẵn có, khó khăn về việc kết nối và nhận dữ liệu từ máy chủ. Do đó, tôi đã sử dụng Electron cùng JavaScript là công nghệ chính để xây dựng ứng dụng.

Sau khi đã xây dựng được ứng dụng với những chức năng cơ bản, tôi đã gặp khó khăn trong việc sử dụng JavaScript để gửi nhận tệp hình ảnh từ máy chủ và tiền xử lý hình ảnh trước khi hiển thị cho người dùng. Tuy nhiên, những khó khăn này nếu sử dụng Python đều có thể giải quyết một cách dễ dàng. Vì vậy, tôi đã tìm cách để có thể chạy Python và JavaScript trong cùng một ứng dụng. Tôi đã thử sử dụng một máy chủ ảo bằng Python để giao tiếp với JavaScript bằng công nghệ SocketIO, tuy nhiên cách làm này có nhiều những khuyết điểm. Máy chủ ảo phải luôn hoạt động dẫn đến hiệu suất sử dụng của ứng dụng giảm, cùng với đó việc giao tiếp phức tạp khiến cho việc mở rộng ứng dụng trở nên khó khăn. Để giải quyết vấn đề này, tôi đã sử dụng thư viện Python-Shell, là thư viện hỗ trợ giao tiếp giữa Python và JavaScript với tốc độ cao. Bằng cách sử dụng thư viện này, tôi đã có thể sử dụng Python để thực hiện những chức năng mà bằng JavaScript khó để lập trình như tiền xử lý hình ảnh, sử dụng mô hình nhận diện cử chỉ tay.

### 5.3.3 Kết quả đạt được

Sau quá trình xây dựng, tôi đã xây dựng được một ứng dụng để người dùng có thể thông qua việc giao tiếp với máy chủ thực hiện các chức năng thông minh đối với dòng camera, xử lý hình ảnh và quản lý tài khoản.



**Hình 5.3:** Ứng dụng điều khiển các chức năng camera

## 5.4 Điều khiển camera sử dụng cử chỉ tay thông qua Jetson Nano

### 5.4.1 Đặt vấn đề

Sau khi tôi xây dựng được ứng dụng đã mô tả ở mục 5.3, tôi nhận thấy để có thể sử dụng cử chỉ tay điều khiển camera và nhận thông báo, người dùng cần phải khởi động máy tính. Nếu khi người dùng không muốn sử dụng những chức năng khác xem dòng hình ảnh, việc khởi động máy tính có thể gây tốn thời gian. Do đó, nếu có thể điều khiển các chức năng của camera thông qua cử chỉ tay và nhận thông báo từ máy chủ mà không cần những thao tác khởi động sẽ có thể giúp người dùng dễ dàng sử dụng hơn. Để làm được điều này, tôi đã sử dụng một máy tính nhỏ là thiết bị Jetson Nano có gắn camera, người dùng có thể điều khiển các chức năng thông qua việc giơ cử chỉ tay trước camera và đồng thời máy chủ có thể gửi những thông báo về cho thiết bị này.

### 5.4.2 Giải pháp

Để lập trình trên thiết bị Jetson Nano, tôi cần kết nối thiết bị với máy tính hoặc màn hình máy tính để thao tác. Sau đó sử dụng những mô hình cùng giải pháp đã được áp dụng cho ứng dụng máy tính để cài đặt trên thiết bị Jetson Nano.

Tuy nhiên, trong quá trình cài đặt, tôi nhận thấy vì đây là một thiết bị biên, nên tốc độ xử lý cùng bộ nhớ không cao, hơn nữa những phiên bản cho thư viện TensorFlow trên thiết bị này là những phiên bản cũ. Nếu sử dụng những mô hình và cách cài đặt giống với ứng dụng máy tính, thiết bị sẽ không thể chạy được. Do vậy, trước tiên, tôi tìm hiểu về mô hình nhẹ ít tham số có thể chạy được trên thiết bị biên này. Sau quá trình tìm hiểu và thử nghiệm, tôi nhận thấy mô hình MobileNetV2 với tổng tham số khoảng 2.200.000 cho độ chính xác khoảng 93%. Tuy độ chính xác không cao bằng mô hình EfficientNet sử dụng cho ứng dụng máy tính, nhưng tốc độ xử lý và kết quả khi chạy thực tế không thua quá nhiều so với ứng dụng máy tính.

Sau khi thiết bị có thể nhận diện cử chỉ tay và thông báo cho người dùng bằng loa để người dùng xác nhận, thiết bị sẽ tiến hành gửi yêu cầu đến cho máy chủ để máy chủ xử lý. Máy chủ sau khi phát hiện những hành động bất thường sẽ gửi lại thông báo đến cho thiết bị biên này. Để có thể nhận được thông báo, ban đầu tôi đã gặp khó khăn do Python không hỗ trợ nhận thông báo thông qua dịch vụ Firebase mà tôi sử dụng khi xây dựng ứng dụng. Để giải quyết vấn đề này, tôi đã xây dựng một máy chủ đơn giản chạy bằng FastAPI, với những tốc độ nhận yêu cầu cao và tốn ít tài nguyên hệ thống, máy chủ ảo này có thể nhận thông báo từ máy chủ chính bất cứ lúc nào. Sau khi nhận được thông báo sẽ phát thông báo đó qua loa đã được kết nối với thiết bị Jetson Nano.

### 5.4.3 Kết quả đạt được

Thiết bị Jetson Nano tôi sử dụng đã có thể điều khiển các chức năng của camera thông qua cử chỉ tay và nhận thông báo từ máy chủ.

## CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

### 6.1 Kết luận

Sau thời gian làm đồ án dưới sự hướng dẫn của TS.Đặng Tuấn Linh, tôi đã thành công xây dựng ứng dụng sử dụng cử chỉ tay hỗ trợ điều khiển các chức năng camera. Ứng dụng đã đạt được như mục tiêu đề ra: (i) xây dựng một ứng dụng máy tính có thể điều khiển các chức năng của camera (ii) xây dựng mô hình phát hiện cử chỉ tay (iii) điều khiển camera bằng cử chỉ tay trên thiết bị biên Jetson Nano (iv) xây dựng bộ dữ liệu để huấn luyện và đánh giá mô hình cử chỉ tay.

Để sử dụng các chức năng của camera, người dùng tiến hành đăng nhập và chọn chức năng xem dòng hình ảnh hoặc xử lý ảnh. Đối với chức năng xem dòng hình ảnh, người dùng sẽ chọn camera và chức năng muốn xem, trong đó một số chức năng người dùng có thể chọn vùng để xử lý và chọn tiền xử lý cho dòng hình ảnh, sau đó gửi yêu cầu đến cho máy chủ và máy chủ sẽ gửi dòng hình ảnh lại cho ứng dụng để hiển thị. Đối với chức năng xử lý ảnh, người dùng chọn chức năng và ảnh muốn xử lý và gửi yêu cầu cho máy chủ. Sau khi nhận kết quả từ máy chủ, ứng dụng sẽ hiển thị cho người dùng. Người dùng cũng có thể sử dụng ứng dụng để quản lý thông báo và dòng hình ảnh đã lưu.

Để điều khiển camera bằng cử chỉ tay, người dùng có thể ấn nút khởi động chức năng và tiến hành điều khiển. Đối với mỗi cử chỉ được nhận diện, ứng dụng sẽ thông báo tới người dùng bằng loa để xác nhận lại xem chức năng vừa chọn có chính xác không. Sau đó sẽ tiến hành gửi yêu cầu đến máy chủ dựa trên cử chỉ tương ứng. Nếu có thông báo từ phía máy chủ, thông báo có thể được hiển thị trên màn hình máy tính hoặc kêu qua loa được gắn với thiết bị Jetson Nano.

Trong quá trình thực hiện đồ án, tôi đã học được nhiều công nghệ và kiến thức mới. Tôi cũng hiểu thêm về quy trình thiết kế, phát triển ứng dụng, cách xử lý khi gặp khó khăn trong phát triển.

### 6.2 Hướng phát triển

Vì thời gian phát triển hạn chế, nên đồ án này vẫn chưa được hoàn thiện ở một số tính năng. Từ đánh giá của bản thân và góp ý của những người có chuyên môn xung quanh, tôi thấy ứng dụng cần có những điểm cải thiện như sau:

- Cải thiện tốc độ xử lý và thêm các cử chỉ mới vào mô hình nhận diện cử chỉ tay giúp tăng trải nghiệm của người dùng.
- Cải thiện giao diện ứng dụng thêm bắt mắt hơn.

- Ngoài hỗ trợ cử chỉ tay, ứng dụng cũng nên hỗ trợ điều khiển bằng giọng nói.
- Cải thiện khả năng phản hồi của ứng dụng khi người dùng tương tác.

Trên đây là một số những cải thiện quan trọng mà tôi nghĩ có thể tiếp tục phát triển trong tương lai trước khi có thể đưa ra sử dụng trong thực tế. Tuy hiện nay ứng dụng còn gặp hạn chế về kỹ thuật và thiết kế, nhưng tôi mong rằng trong tương lai không xa ứng dụng có thể được ứng dụng trong thực tế, giúp nhiều người có thể sử dụng camera thông minh để hỗ trợ quản lý những công việc trong cuộc sống.

## TÀI LIỆU THAM KHẢO

- [1] PAUL BISCHOFF, *Surveillance camera statistics*. [Online]. Available: <https://www.comparitech.com/vpn-privacy/the-worlds-most-surveilled-cities/> (visited on 08/03/2022).
- [2] *Smarter ai™ camera platform architecture*. [Online]. Available: <https://anyconnect.com/platform-architecture> (visited on 08/03/2022).
- [3] *Ai smart camera systems for homes businesses*. [Online]. Available: <https://guardianva.com/> (visited on 08/03/2022).
- [4] *Vision ai, simply done*. [Online]. Available: <https://plainsight.ai/> (visited on 08/03/2022).
- [5] *Build cross-platform desktop apps with javascript, html, and css*. [Online]. Available: <https://www.electronjs.org/> (visited on 08/03/2022).
- [6] *Thư viện opencv*. [Online]. Available: <https://opencv.org/> (visited on 08/03/2022).
- [7] Google, *Firebase - make your app the best it can be*. [Online]. Available: <https://firebase.google.com/> (visited on 08/03/2022).
- [8] *Thư viện fastapi*. [Online]. Available: <https://fastapi.tiangolo.com/> (visited on 08/03/2022).
- [9] ——, *Thị trường camera tại việt nam*. [Online]. Available: <https://bom.so/6u6psH> (visited on 08/03/2022).
- [10] BKAV, *Aiview-camera tích hợp ai*. [Online]. Available: <https://www.aiview.ai/> (visited on 08/03/2022).
- [11] Viettel, *Ai camera platform*. [Online]. Available: <https://viettelhightech.vn/detail-solution/ai-camera-platform> (visited on 08/03/2022).
- [12] *Thư viện python shell*. [Online]. Available: <https://www.npmjs.com/package/python-shell> (visited on 08/03/2022).
- [13] *Thư viện tensorflow*. [Online]. Available: <https://www.tensorflow.org/> (visited on 08/03/2022).
- [14] Google, *Mediapipe - live machine learning anywhere*. [Online]. Available: <https://mediapipe.dev/> (visited on 08/03/2022).
- [15] *Dịch vụ google text to speech*. [Online]. Available: <https://pypi.org/project/gTTS/> (visited on 08/03/2022).
- [16] Q. V. L. Mingxing Tan, “Efficientnet: Rethinking model scaling for convolutional neural networks,” *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, pp. 6105–6114, 2019.

- [17] H. A. Z. M. Z. A. C. L. C. Sandler M., “Mobilenetv2: Inverted residuals and linear bottlenecks,” *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.
- [18] X. B. G. C. Y. L. Z. L. S. N. W. J. Yu C., “Lite-hrnet: A lightweight high-resolution network,” *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 10 440–10 450, 2021.
- [19] *Bộ dữ liệu cùi chỉ tay*. [Online]. Available: <https://bom.so/YGqtf1>.

# **PHỤ LỤC**