**TRƯỜNG ĐẠI HỌC KIÊN GIANG**

**KHOA THÔNG TIN – TRUYỀN THÔNG**



**ĐẶNG NGUYỄN TIỀN HẬU**

**HỆ THỐNG PHÂN LOẠI TRÁI CÂY TỰ ĐỘNG ỨNG DỤNG HỌC MÁY VÀ THỊ GIÁC MÁY TÍNH**

**KHOÁ LUẬN TỐT NGHIỆP**

**Ngành: Công nghệ thông tin**

**Mã số ngành: 7480201**

**Kiên Giang – năm 2025**

**TRƯỜNG ĐẠI HỌC KIÊN GIANG**

**KHOA THÔNG TIN – TRUYỀN THÔNG**



**ĐẶNG NGUYỄN TIỀN HẬU**

**MSSV: 21072006018**

**HỆ THỐNG PHÂN LOẠI TRÁI CÂY TỰ ĐỘNG ỨNG DỤNG HỌC MÁY VÀ THỊ GIÁC MÁY TÍNH**

**KHOÁ LUẬN TỐT NGHIỆP**

**Ngành: Công nghệ thông tin**

**Mã số ngành: 7480201**

**GIẢNG VIÊN HƯỚNG DẪN**

**ThS. VÕ HOÀNG NHÂN**

**Kiên Giang – Năm 2025**

# LỜI CẢM ƠN

*Ngày … tháng … năm*

**Sinh viên thực hiện**

*(Ký và ghi rõ họ tên)*

# LỜI CAM ĐOAN

Tôi cam đoan rằng đề tải này do chính tôi thực hiện, các số liệu thu thập và kết quả phân tích trong đề tài là trung thực, đề tài không trùng với bất kỳ đề tài nghiên cứu khoa học nào.

*Ngày … tháng … năm*

**Sinh viên thực hiện**

*(Ký và ghi rõ họ tên)*

# NHẬN XÉT CỦA CƠ QUAN THỰC TẬP

*Ngày … tháng … năm*

**Thủ trưởng đơn vị**

*(Ký và ghi rõ họ tên)*

# NHẬN XÉT CỦA NGƯỜI HƯỚNG DẪN

*Ngày … tháng … năm*

**Giảng viên hướng dẫn**

*(Ký và ghi rõ họ tên)*

BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC KIÊN GIANG**

**CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc lập – Tự do – Hạnh phúc**

# PHIẾU ĐÁNH GIÁ ĐỀ TÀI TỐT NGHIỆP

Họ và tên người đánh giá:

Trách nhiệm trong hội đồng:

Họ và tên sinh viên:……………………………….. MSSV:……………………...

Tên đề tài:

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Tiêu chí chấm điểm** | **Điểm tối đa** | **Kết quả** |
| 1 | **Nội dung luận văn**  - Tính khoa học  - Nội dung của đề tài phù hợp với mục tiêu, phạm vi và tên của đề tài.  - Bố cục luận văn hợp lý, đầy đủ nội dung yêu cầu của một luận văn đại học. | **6,0 đ**  1,0 đ  3,0 đ  2,0 đ |  |
| 2 | **Hình thức luận văn**  - Hình thức trình bày (in ấn, format, hình ảnh,…)  - Cách hành văn.  - Đúng chính tả. | **2,0 đ**  0,5 đ  1,0 đ  0,5 đ |  |
| 3 | **Báo cáo trước Hội đồng**  - Bài báo cáo (hình thức, cấu trúc,…)  - Tác phong trình bày (tự tin, rõ ràng, mạch lạc…)  - Trả lời câu hỏi của Hội đồng. | **2,0 đ**  0,5 đ  0,5 đ  1,0 đ |  |
| 4 | **Điểm thành tích trong nghiên cứu khoa học**  - Có ít nhất 01 bài báo được đăng trên các tạp chí trong danh mục tính điểm Hội đồng Giáo sư Nhà nước hoặc bài tham luận/báo cáo được đăng trong các kỷ yếu của Hội nghị/Hội thảo quốc gia hoặc quốc tế mà sinh viên là tác giả chính.  - Có ít nhất 01 bài báo được đăng trên các tạp chí trong danh mục tính điểm Hội đồng Giáo sư Nhà nước hoặc bài tham luận/báo cáo được đăng trong các kỷ yếu của Hội nghị/Hội thảo quốc gia hoặc quốc tế mà sinh viên là thành viên. | **1,0 đ**  1,0 đ  0,5 đ |  |
| **Tổng các cột 1 + 2 + 3 + 4 (không quá 10 điểm)** | | **10,0 đ** |  |

**Ý KIẾN NHẬN XÉT, ĐỀ NGHỊ**

*… ngày … tháng … năm 20…*

**NGƯỜI ĐÁNH GIÁ**

*(Ký và ghi rõ họ tên)*

# MỤC LỤC

*Trang*

# DANH MỤC BIỂU BẢNG

*Trang*

# DANH MỤC HÌNH

*Trang*

# DANH SÁCH TỪ VIẾT TẮT

|  |  |
| --- | --- |
| **Từ/Cụm từ viết tắt** | **Nội dung đầy đủ** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# DANH SÁCH CÁC KÝ HIỆU (nếu có)

|  |  |
| --- | --- |
| **Ký hiệu** | **Tên ký hiệu** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# CHƯƠNG 1 GIỚI THIỆU ĐỀ TÀI

## 1.1 ĐẶT VẤN ĐỀ

Từ sau đại dịch bệnh của toàn thế giới bùng nổ - Covid 19 thì những lĩnh vực thuộc công nghệ thông tin đang dần được thúc đẩy, phát triển mạnh hơn. Đặc biệt là vào những năm gần đây từ 2021 – 2025 thì Trí tuệ nhân tạo (AI) bỗng dưng trở thành điểm nóng của công nghệ và được các ông lớn (OpenAI, xAI – Elon Musk, Google,…) chống lưng.

Sự ra đời của các AI này làm cho con người phải suy nghĩ lại về cách làm việc trong thời đại công nghệ đang phát triển mạnh. Các nhà máy, xí nghiệp vốn cần rất nhiều công nhân, nhân viên làm việc để đảm bảo mọi quy trình đều chính xác và trôi chảy thì nay với sự hỗ trợ của AI và những thiết bị (robot, vi mạch,…) quá trình làm việc đã được tự động hoá. Điều này tạo nên những lợi ích lớn như: giảm lượng nhân công, tiền lương, chi phí bảo hộ lao động và nâng cao năng suất làm việc,… Sự ra đời của AI và ứng dụng của nó trong thực tiễn đang là yếu tố tạo nên thời kỳ phát triển mới của con người, tương tư như thành tựu của những cuộc cách mạng công nghiệp trong quá khứ.

Nhìn thấy được tiềm năng của AI ở thời đại mới, em mong muốn sử dụng kiến thức, khả năng của bản thân để tạo nên một sản phẩm thực tế, có tiềm năng phát triển trong tương lai để đưa vào cuộc sống giúp đỡ cho con người và quyết định thực hiện đề tài " Hệ thống phân loại trái cây tự động ứng dụng học máy và thị giác máy tính".

## 1.2 MỤC TIÊU NGHIÊN CỨU

### 1.2.1 Mục tiêu chung

* Sản phẩm hoàn thiện của đề tài là một hệ thống phân loại trái cây với quy mô nhỏ, phân loại được các loại trái mà phạm vi đề tài có đề cập. Sau khi đã ổn định thì sẽ định hướng phát triển để mở rộng hệ thống, nhận diện được nhiều loại trái hơn, nâng cấp phần cứng để tự động hoá toàn bộ quy trình.
* Đề tài được thiết kế với quy mô nhỏ do giới hạn về nhân lực, kiến thức, chi phí và thời gian nên chỉ ứng dụng được trong môi trường mô phỏng như phòng thí nghiệm, dự án nghiên cứu sinh viên. Khi được phối hợp với bên ngoài (nhà đầu tư, lập trình viên chuyên sâu thuộc từng mảng riêng: AI, IoT,…) thì đề tài sẽ phù hợp để triển khai trong các cơ sở nhà xưởng, xí nghiệp thu mua, phân phối trái cây ra thị trường.

### 1.2.2 Mục tiêu cụ thể

1.2.2.1 Trái cây và tình trạng

* Đề tài tập trung phân loại 3 loại quả sau:
* Táo Red Delicious Mỹ.
* Nho Ninh Thuận (Việt Nam).
* Lê vàng Việt Nam.

với các đặc trưng gồm: màu sắc và hình dáng. Chi tiết sẽ được nói rõ thêm trong ***"Phạm vi và đối tượng nghiên cứu".***

1.2.2.2 Kiến thức về học máy

* Nắm được các kiến thức cơ bản như: Loss, Early Stopping, các metrics đánh giá mô hình,…
* Tìm hiểu 2 mô hình học máy được sử dụng trong đề tài:
* Convolutional Neural Network (CNN)
* Support Vector Machine (SVM)
* Việc lựa chọn nghiên cứu 2 mô hình học máy nhằm mục đích tạo cơ hội cho bản thân được tìm hiểu sâu về các kiến thức học máy, triển khai và so sánh độ hiệu quả của 2 mô hình, từ đó chọn ra mô hình có hiệu năng tốt nhất.

1.2.2.3 Phần cứng (IoT)

* Chuẩn bị được các thiết bị cần thiết để lắp đặt hệ thống phân loại trái cây.
* Thiết kế sơ đồ nguyên lý, sơ đồ lắp đặt giữa các thành phần.
* Lắp đặt, vận hành và cải tiến để nâng cao độ chính xác và ổn định.

1.2.2.4 Phần mềm (Website)

* Website phải trực quan, dễ sử dụng trong thao tác quản lý các thông tin về công xưởng và các hệ thống phân loại như:
* Số lượng thiết bị.
* Số lượng nhân viên đang làm việc.
* Số lượng tải khoản được sử dụng hệ thống.
* Số lượng kết quả đã phân loại.
* Số lượng khu phân loại.
* Số lượng trái cây và trạng thái được sử dụng.
* Website có biểu đồ thống kê kết quả theo từng khung thời gian (tuần, tháng, năm), cập nhật theo thời gian thực; thống kê tỉ lệ các loại trái cây.
* Website cập nhật trực tiếp cấu hình học máy cho Raspberry Pi tại từng khu phân loại.

1.2.2.5 Phối hợp giữa các thành phần

* Raspberry:
* Sử dụng mô hình học máy đã được huấn luyện và đưa vào dưới dạng tối ưu cho thiết bị IoT (tflite).
* Sau khi nhận diện sẽ thông qua giao thức MQTT và HTTP để tương ứng gửi kết quả đến ESP32 và NestJS Server để ghi nhận kết quả vào cơ sở dữ liệu, hiển thị trên Website và điều khiển quy trình phân loại.
* Raspberry sẵn sàng nhận cấu hình mới (nhãn, mô hình mới) khi đang trong quá trình hoạt động.
* Website (NextJS & NestJS): Server NestJS nhận dữ liệu từ Raspberry và ghi vào cơ sở dữ liệu, đồng thời gửi qua SocketIO tới NextJS để hiển thị trên giao diện quản trị theo thời gian thực.
* ESP32: Nhận kết quả từ Raspberry thông qua MQTT và điều khiển cánh tay robot để gắp trái cây bỏ vào khu vực tương ứng.

## 1.3 PHẠM VI VÀ ĐỐI TƯỢNG NGHIÊN CỨU

### 1.3.1 Phạm vi nghiên cứu

1.3.1.1 Lĩnh vực nghiên cứu

* Trọng tâm của đề tài là các lĩnh vực: học máy (machine learning), thị giác máy tính (compotuer vision), internet vạn vật (IoT), phát triển web nhằm xây dựng hệ thống phân loại trái cây tự động.
* Ứng dụng chính là tự động hoá quy trình phân loại trong ngành nông sản, trọng yếu là công xưởng xử lý, phân phối trái cây.

1.3.1.2 Đối tượng áp dụng

* Hệ thống mà đề tài xây dựng dùng cho 3 loại quả, 2 tình trạng mỗi quả. Như vậy ta có 6 classes:
* Táo Red Delicious Mỹ: loại táo có màu đỏ thẫm, đẹp, được bán khá phổ biến trong các siêu thị, chợ tại Việt Nam.
* Nho Ninh Thuận: loại nho xanh, nhỏ, thành chùm ở Việt Nam.
* Lê vàng Việt Nam: loại lê có màu vàng, vỏ mỏng, láng mịn.

|  |  |  |
| --- | --- | --- |
|  | Ripe  (Trái đẹp, tươi, tốt) | Rot  (Trái xấu, hỏng) |
| Apple | Vỏ có màu đỏ tươi, bóng, độ tròn đồng đều. | Vỏ bị dập, có vết thâm, có vết bị cắn hoặc trầy xước nặng. |
| Grapes | Các trái có màu xanh tươi, bóng, không có quá nhiều trái bị hỏng. | Có nhiều trái/cả chùm bị hỏng, màu xanh ngả vàng thâm, bị dập, trầy xước, nhăn nheo, héo trên vỏ. |
| Pear | Vỏ có màu vàng tươi, láng mịn, tròn đồng đều, chấp nhận một số vết thâm nhỏ xây xát do vận chuyển. | Có vết thâm lớn làm mất cân đối màu sắc quả, quả bị dập, lõm, héo úa. |

1.3.1.3 Công nghệ sử dụng

* Học máy:
* Convolutional Neural Network (CNN)
* Kiến trúc MobileNetV2.
* Support Vector Machine (SVM)
* Website:
* Frontend: NextJS
* Backend: NestJS
* Database: MySQL, TypeORM.
* Khác: RESTful API, SocketIO.
* Phần cứng (IoT):
* Thiết bị: Raspberry, ESP32, Webcam, VL53L0X.
* Phương thức giao tiếp: MQTT, SocketIO.

### 1.3.2 Đối tượng nghiên cứu

1.3.2.1 Đối tượng chính

#### a. Táo Red Delicious Mỹ

Vốn có nguồn gốc từ Mỹ và được tìm ra từ năm 1880. Nhưng đến mãi sau năm 1982, Táo Red Delicious mới được truyền bá và thương mại hoá. Nguồn gốc của Red xuất phát từ câu chuyện về một cuộc thi táo Ben Davis được tổ chức ở vườm ươm Stark Nuseries. Giống Táo Red này thường được gieo trồng, canh tác ở những nơi có khí hậu khô nóng với 300 ngày nắng mỗi năm, đất đai màu mỡ như Washington.



#### b. Nho xanh Ninh Thuận

Nho xanh Ninh Thuận là một loại trái cây đặc sản của vùng đất nắng gió Ninh Thuận. Đây là loại nho có vỏ màu xanh, quả dài, nhỏ hình bầu dục, thịt dày chắc ngọt, nhiều nước, chùm quả to và dài. Nho xanh Ninh Thuận có vị ngọt thanh, chua nhẹ, rất thơm ngon và được nhiều người yêu thích.

Nho xanh Ninh Thuận được trồng chủ yếu ở các huyện Ninh Phước, Ninh Hải, Thuận Bắc của tỉnh Ninh Thuận. Khí hậu nắng nóng quanh năm của vùng đất này là điều kiện lý tưởng để nho xanh phát triển. Nho xanh Ninh Thuận được trồng trên các vùng đất cát pha, thoát nước tốt, được chăm sóc kỹ lưỡng theo quy trình kỹ thuật tiên tiến.

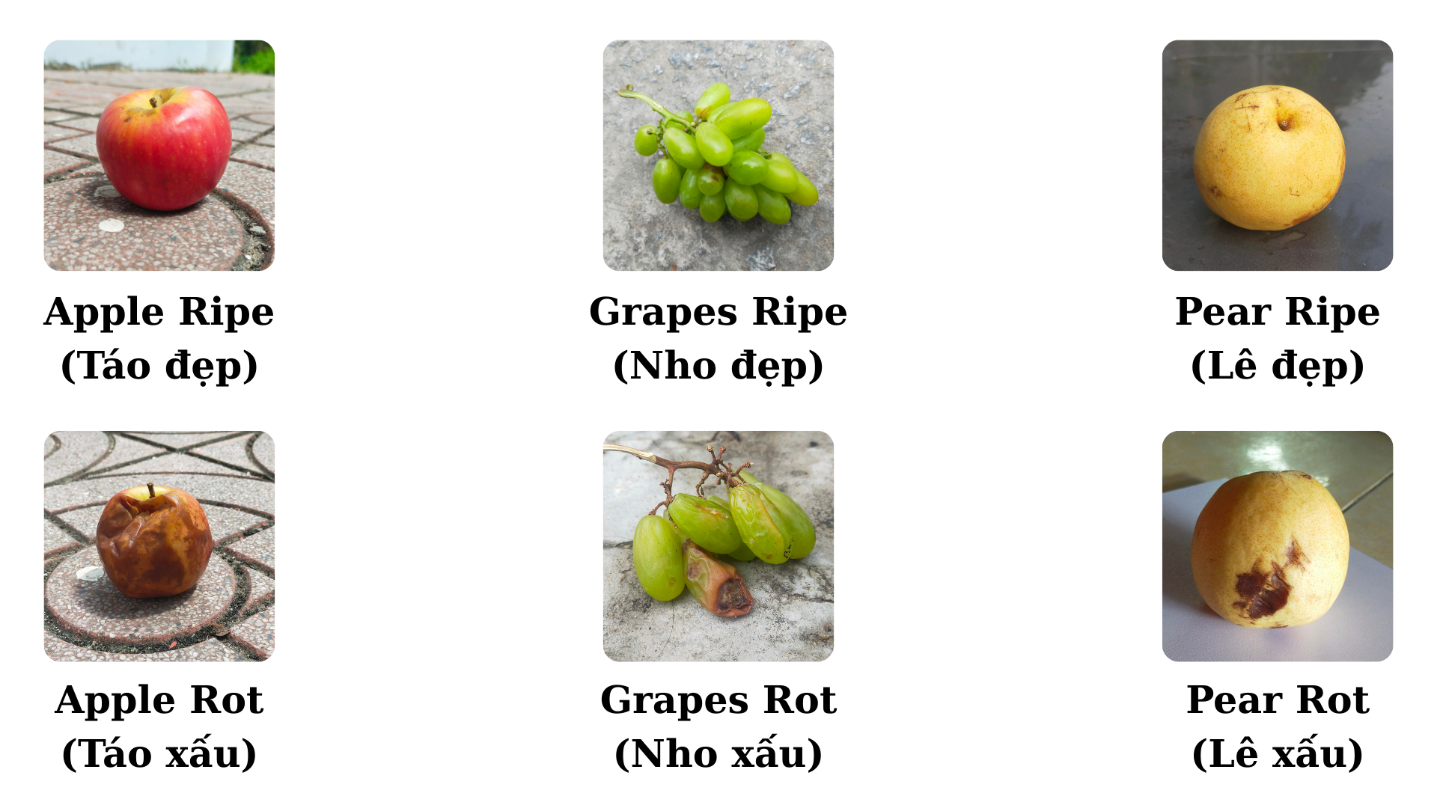


#### c. Lê vàng Việt Nam

Lê vàng Việt Nam, thường được biết đến với tên gọi lê Khang Tân, là một giống lê nhập ngoại từ các vùng ôn đới, hiện được trồng phổ biến tại các khu vực có khí hậu mát mẻ của Việt Nam như Bắc Hà, Sa Pa, Lai Châu, Điện Biên, Lạng Sơn, với độ cao từ 500m trở lên so với mực nước biển. Loại lê này đã thích nghi tốt với điều kiện địa phương và mang lại giá trị kinh tế đáng kể.

### 1.3.3 Tiêu chí phân loại

* Màu sắc: Những quả có màu sắc tươi sáng, độ phân bổ đồng đều sẽ được xếp vào nhóm "Tốt, chín – Ripe". Ngược lại, vỏ có những đốm đen, độ phân bổ màu sắc không đồng đều sẽ được xếp vào nhóm "Xấu, hỏng – Rot".
* Hình dáng: Những quả có độ tròn đồng đều sẽ xếp vào nhóm "Tốt, chín – Ripe" , những quả bị dập, méo, hình dạng không đồng đều sẽ được xếp vào nhóm "Xấu, hỏng – Rot".
* Do giới hạn về chi phí và thời gian nên đề tài chỉ đề cập đến 2 đặc trưng trên, ngoài ra những yếu tố khác như: khối lượng, mùi vị, độ chín,… sẽ không được sử dụng trong đề tài này.



# CHƯƠNG 2 CƠ SỞ LÝ THUYẾT

## 2.1 CÔNG NGHỆ SỬ DỤNG

### 2.1.1 Học máy (Machine Learning)

2.1.1.1 Khái niệm

Học máy (Machine Learning) là một lĩnh vực thuộc trí tuệ nhân tạo, tập trung vào việc nghiên cứu và phát triển các kỹ thuật cho phép hệ thống “học” tự động từ dữ liệu để giải quyết các vấn đề cụ thể.

Hiểu đơn giản, thuật ngữ này nói tới việc con người dạy máy tính nâng cao khả năng thực hiện các tác vụ cụ thể. Cụ thể là cung cấp các dữ liệu và thuật toán có sẵn để máy tính đưa ra dự đoán hoặc tự ra các quyết định. Thông thường, con người chỉ cần lập trình phần mềm với các dòng lệnh cụ thể để máy tính hiểu và thực hiện. Với Machine Learning, máy tính sẽ tự “học” cách giải quyết công việc thông qua những dữ liệu đã được thu thập và cung cấp.

2.1.1.2 Vai trò của học máy

* Tăng khả năng nhận dạng.
* Tăng khả năng thích nghi.
* Triển khai công nghệ AI một cách thông minh.

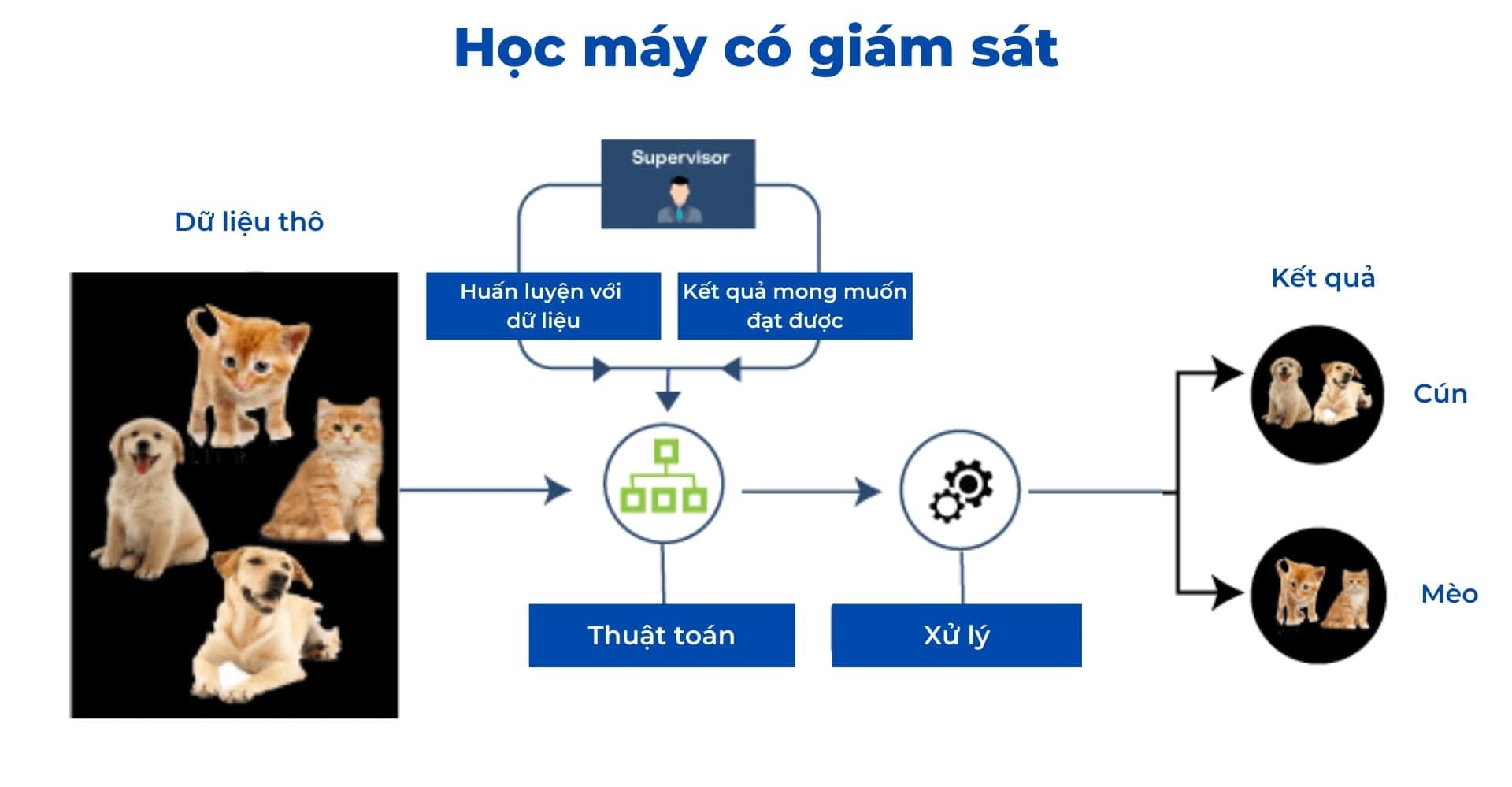
2.1.1.3 Cách hoạt động của học máy

* Quy trình triển khai thuật toán học máy gồm 6 bước:
* Bước 1: Thu thập dữ liệu (Gathering data/Data collection)
* Bước 2: Tiền xử lý dữ liệu (Data preprocessing)
* Trích xuất dữ liệu (Data extraction)
* Làm sạch dữ liệu (Data cleaning)
* Chuyển đổi dữ liệu (Data transformation)
* Chuyển hoá dữ liệu (Feature normalization).
* Trích xuất đặc trưng (Feature extraction).
* Bước 3: Phân tích dữ liệu (Data analysis)
* Bước 4: Xây dựng mô hình máy học (Model building)
* Bước 5: Huấn luyện mô hình (Model training)
* Bước 6: Đánh giá mô hình (Model evaluation)

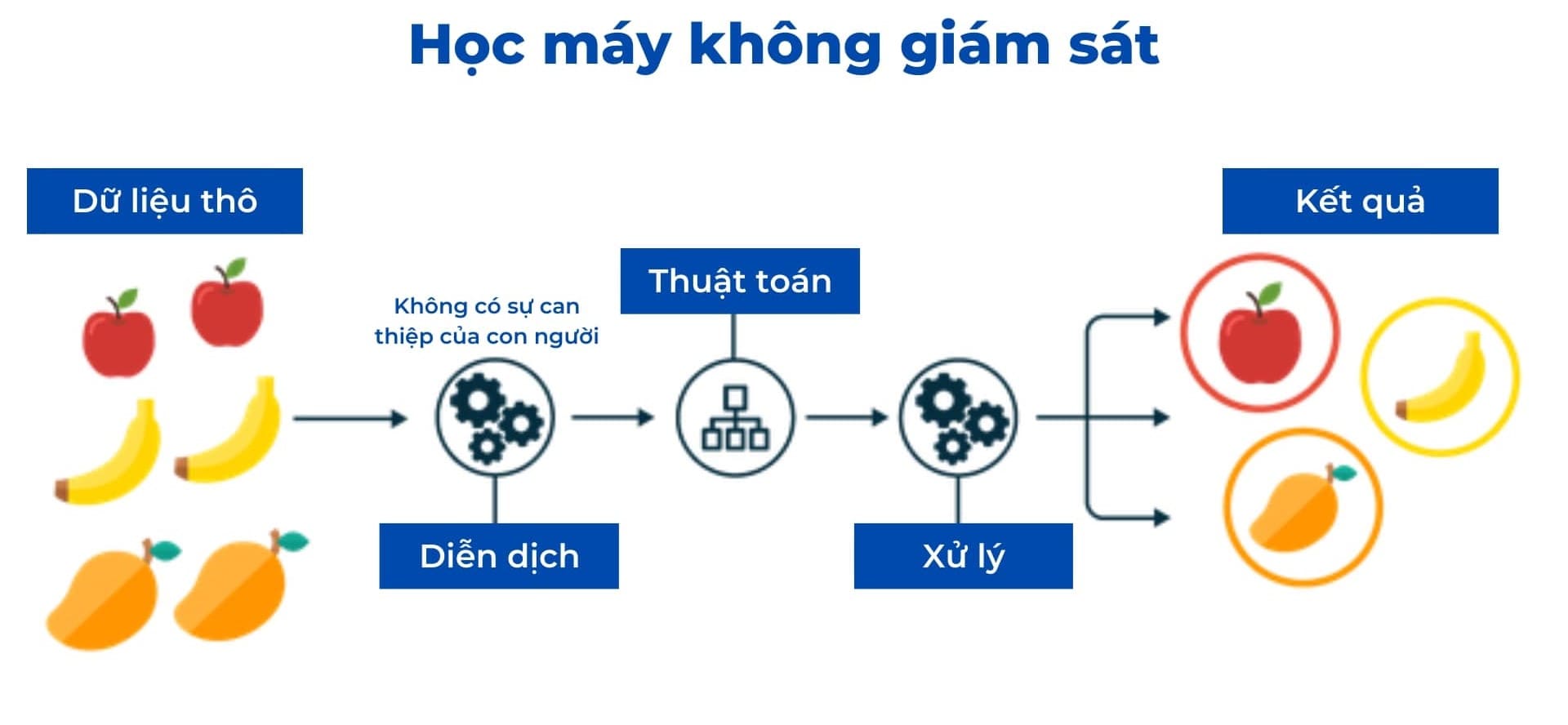


2.1.1.4 Các loại học máy

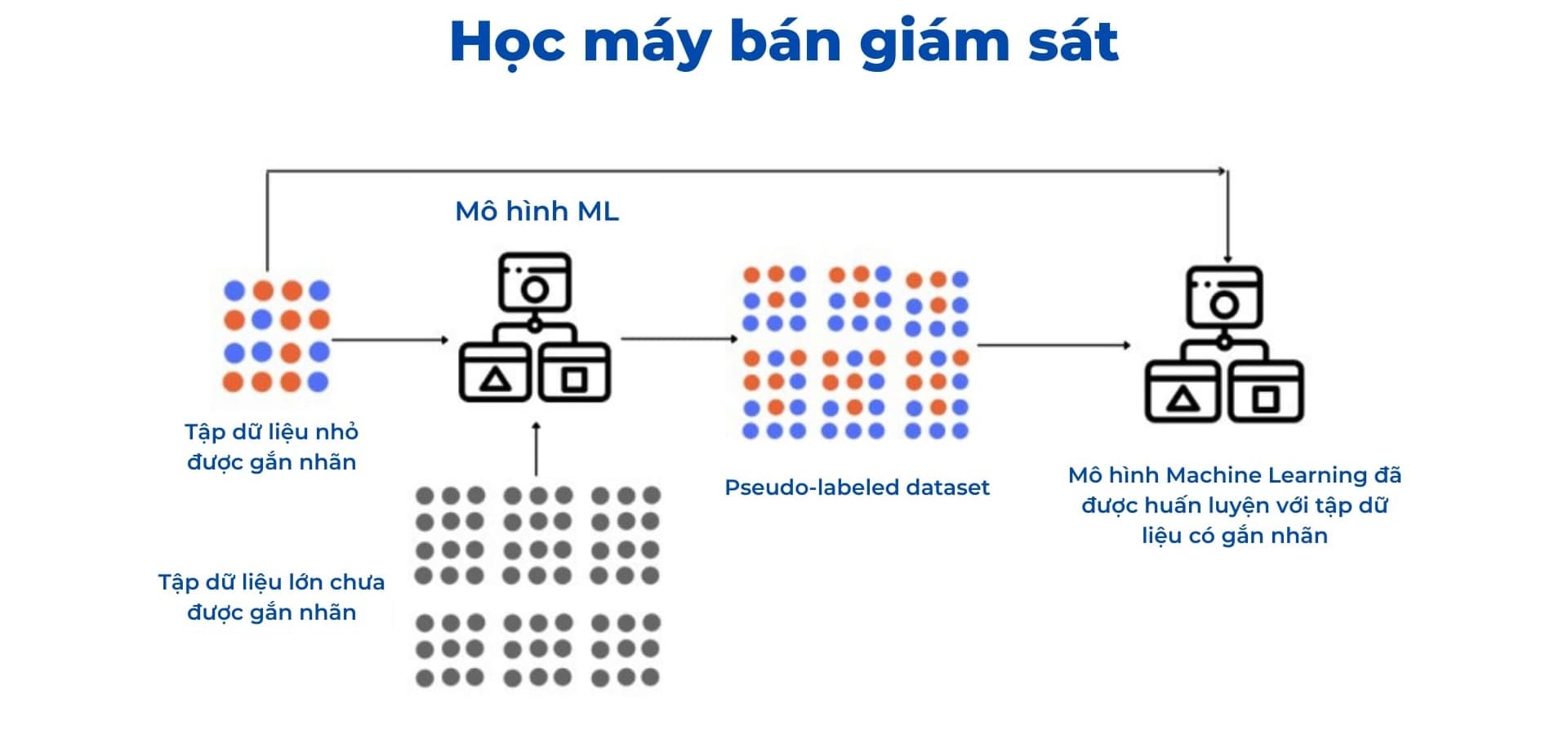
* Học máy có giám sát: Học máy có giám sát (Supervised Machine Learning) là phương pháp mà trong đó máy tính được học từ dữ liệu đã được đánh dấu trước, để phát triển các thuật toán có khả năng phân loại hoặc dự đoán kết quả một cách chính xác. Kỹ thuật này thường được áp dụng cho các bài toán phân lớp (Classification).



* Học máy không giám sát: Trái ngược với học máy có giám sát, máy học không giám sát (Unsupervised Learning) sử dụng thuật toán để phân tích và phân cụm các dữ liệu không có nhãn. Phương pháp này tự động tìm kiếm các mô hình và cấu trúc ẩn trong dữ liệu mà không yêu cầu sự hỗ trợ từ con người.



* Học máy bán giám sát: Trong bối cảnh dữ liệu phát triển nhanh chóng và không có cách nào để chúng được gắn nhãn kịp thời, đó là lý do học máy bán giám sát (Semi-supervised Learning) trở nên rất quan trọng. Phương pháp này kết hợp việc sử dụng dữ liệu đã được gắn nhãn và chưa gắn nhãn để huấn luyện máy tính.



2.1.1.5 Mạng nơ-ron tích chập (Convolutional Neural Network – CNN)

#### a. Khái niệm

Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay.

#### b. Cấu trúc mạng CNN

**Convolutional layer**

Đây chính là lớp đóng vai trò mấu chốt của CNN, khi layer này đảm nhiệm việc thực hiện mọi tính toán. Stride, padding, filter map, feature map là những yếu tố quan trọng nhất của convolutional layer.

* Cơ chế của CNN là tạo ra các filter áp dụng vào từng vùng hình ảnh. Các filter map này được gọi là ma trận 3 chiều, bên trong chứa các parameter dưới dạng những con số.
* Stride là sự dịch chuyển filter map theo pixel dựa trên giá trị từ trái sang phải.
* Padding: Là các giá trị 0 được thêm cùng lớp input.
* Feature map: Sau mỗi lần quét, một quá trình tính toán sẽ được thực hiện. Feature map sẽ thể hiện kết quả sau mỗi lần filter map quét qua input.

**ReLU layer**

Còn có tên gọi khác là activation function, đây là một hàm được kích hoạt trong neural network. Có tác dụng mô phỏng các neuron có tỷ lệ truyền xung qua axon. Trong activation function chúng còn có hàm nghĩa là: Relu, Tanh, Sigmoid, Maxout, Leaky,… Relu layer được ứng dụng phổ biến trong việc huấn luyện nơ-ron do sở hữu nhiều ưu điểm tiên tiến.

**Pooling layer**

Khi nhận phải đầu vào quá lớn, các lớp pooling layer sẽ được xếp giữa những lớp Convolutional layer nhằm mục đích giảm parameter. Pooling layer được chia thành 2 loại phổ biến là max pooling và average.

**Fully connected layer**

Khi 2 lớp convolutional layer và pooling layer nhận được ảnh truyền, lớp này sẽ có nhiệm vụ xuất kết quả. Khi ta nhận được kết quả là model đọc được thông tin ảnh, ta cần phải tạo sự liên kết để cho ra nhiều output hơn. Đây chính là lúc các lập trình viên sử dụng fully connected layer. Hơn nữa, nếu fully connected layer có dữ liệu về hình ảnh thì chúng sẽ chuyển thành mục chưa được phân chia chất lượng.

#### c. MobileNetV2

MobileNetV2 là một kiến trúc dành cho mô hình học sâu (Deep learning) được phát triển bởi nhóm nghiên cứu của Google, công bố vào năm 2018.

Đây là phiên bản cải tiến của MobileNetV1, được thiết kế để tối giản, tương thích với các thiết bị có tài nguyên hạn chế như thiết bị di động, nhúng như Raspberry Pi, ESP32,…

Hiện tại có rất nhiều kiến trúc để triển khai mô hình học máy trên Raspberry, nhưng trong đề tài này em sử dụng kiến trúc MobileNetV2 vì kiến trúc MobileNetV2 đạt độ chính xác tốt và tốc độ nhận diện nhanh. Tiêu chí này là quan trọng vì băng chuyền di chuyển liên tục và tài nguyên của Raspberry không nhiều nên không cần một kiến trúc quá phức tạp, tính toán mất thời gian nhiều.

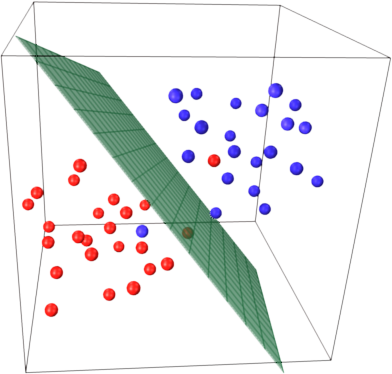
Dưới đây là bảng so sánh một vài đặc điểm MobileNetV2 với các kiến trúc khác:

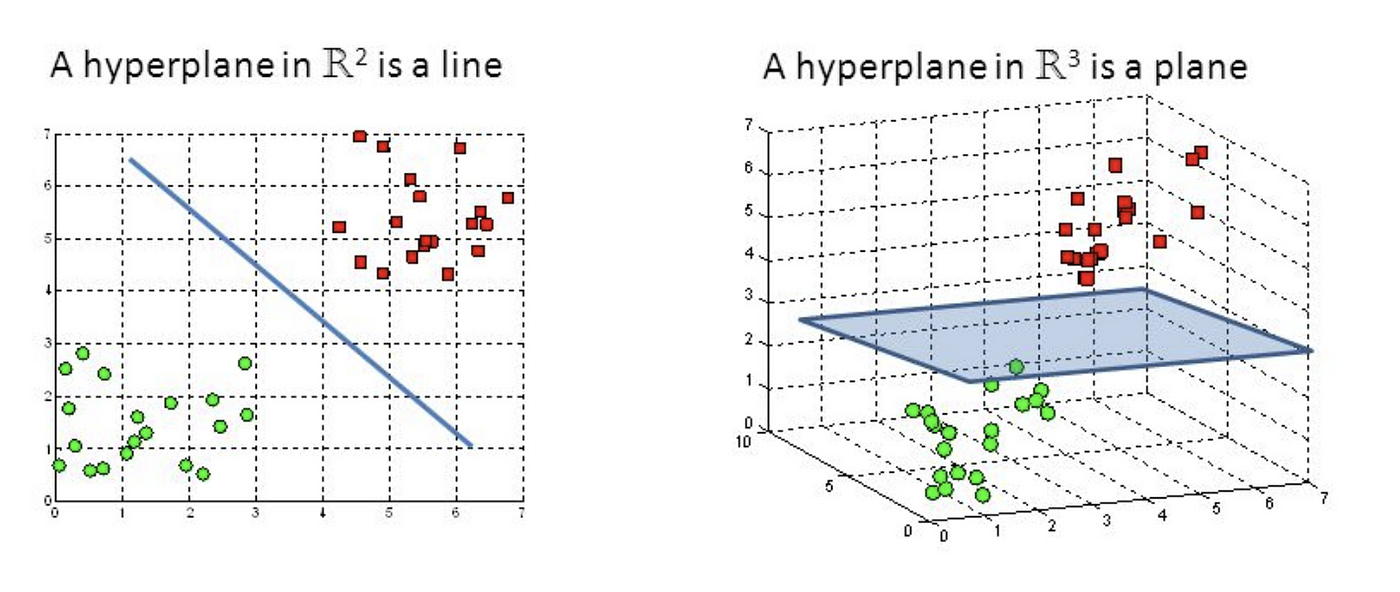
|  |  |  |
| --- | --- | --- |
| Kiến trúc | Ưu điểm | Nhược điểm |
| MobileNetV2 | Nhẹ, nhanh, phù hợp cho thiết bị IoT. | Độ chính xác có thể thấp hơn ResNet, EfficientNet. |
| MobileNetV3 | Tối ưu hơn V2, có phiên bản “Small” cho Edge. | Phiên bản mới nên ít tài liệu, ít hỗ trợ hơn V2 và phức tạp để fine-tune. |
| ResNet50 | Độ chính xác cao, mô hình mạnh. | Nặng, không phù hợp cho thiết bị giới hạn. |
| EfficientNet | Cân bằng tốt giữa độ chính xác và tốc độ. | Triển khai phức tạp hơn, tốn tài nguyên hơn MobileNetV2. |

#### d. Support Vector Machine (SVM)

Support Vector Machine (SVM) là một thuật toán học có giám sát (Supervised Learning Algorithm) được sử dụng phổ biến cho các bài toán phân loại (classification) và hồi quy (regression).

SVM hoạt động dựa trên nguyên lý tìm ra một ***siêu phẳng (hyperplane)*** tối ưu để phân tách các điểm dữ liệu thuộc các lớp khác nhau trong không gian đặc trưng.





Điểm mạnh của SVM là khả năng xử lý các bài toán có số chiều lớn và phân lớp phức tạp, đặc biệt khi dữ liệu không thể phân tách tuyến tính. Trong những trường hợp này, SVM sử dụng hàm nhân (kernel trick) để ánh xạ dữ liệu từ không gian đầu vào sang một không gian chiều cao hơn, nơi có thể tìm được siêu phẳng phân tách hiệu quả. Các loại hàm nhân thường dùng gồm:

* Hàm nhân tuyến tính (Linear Kernel)
* Hàm nhân Gaussian hoặc RBF (Radial Basis Function)
* Hàm nhân bậc hai (Polynomial Kernel)

#### e. Lý do lựa chọn hai mô hình CNN và SVM cho đề tài

Trong đề tài của mình em quyết định chọn CNN và SVM là hai mô hình học máy để triển khai cho đề tài của mình và so sánh, đánh giá chúng vì:

* CNN là mô hình học máy sâu, SVM là mô hình học máy truyền thống (trong đề tài có lợi dụng kiến trúc MobileNetV2 để trích xuất phần đặc trưng ảnh cấp cao). Chọn hai mô hình này sẽ cho em hai hướng tiếp cận khác nhau, từ đó đánh giá được mô hình học sâu hay truyền thống sẽ cho ra kết quả tốt hơn.
* CNN được thiết kế chuyên biệt cho xử lý dữ liệu hình ảnh, nhờ vào các tầng tích chập (convolutional layers) và tầng gộp (pooling layers) giúp tự động trích xuất đặc trưng từ hình ảnh thô. Với dự án này, CNN (cụ thể là MobileNetV2) được chọn vì khả năng nhận diện các đặc trưng phức tạp như hình dạng, màu sắc, và kết cấu của trái cây, đồng thời phù hợp để triển khai trên phần cứng hạn chế như Raspberry Pi nhờ vào kích thước mô hình tối ưu.
* SVM hiệu quả trong việc phân loại nhị phân hoặc đa nhãn (multiclass) khi sử dụng kernel như RBF (Radial Basis Function). SVM được chọn để so sánh với CNN vì tính đơn giản, khả năng hoạt động tốt với dữ liệu có số chiều thấp (sau khi trích xuất đặc trưng thủ công), và ít yêu cầu tài nguyên tính toán hơn so với mô hình học sâu.

Trong đề tài thì mô hình CNN – MobileNetV2 được chọn làm mô hình chính để sử dụng, trong khi SVM chỉ dừng lại với mục đích so sánh.

### 2.1.2 Thị giác máy tính (Computer Vision)

Thị giác máy tính (Computer Vision) là một lĩnh vực con của trí tuệ nhân tạo (AI), nghiên cứu cách để máy tính có thể "nhìn" và "hiểu" thế giới xung quanh thông qua hình ảnh và video. Mục tiêu của thị giác máy tính là phát triển các hệ thống có thể nhận diện, phân tích và xử lý thông tin từ hình ảnh hoặc video, tương tự như cách con người xử lý thị giác của mình.

Thị giác máy tính đóng vai trò quan trọng trong nhiều ứng dụng hiện đại từ nhận diện khuôn mặt, phân loại hình ảnh đến các hệ thống lái xe tự động. Việc ứng dụng thị giác máy tính vào thực tiễn ngày càng trở nên phổ biến nhờ vào sự phát triển của các thuật toán học máy và các mạng nơ-ron nhân tạo.



Thị giác máy tính được phát triển và hoạt động dựa trên các kỹ thuật và thuật toán phức tạp nhằm mô phỏng quá trình nhận diện và xử lý hình ảnh của con người như sau:

* **Thu thập hình ảnh (Image Acquisition):** thu thập dữ liệu hình ảnh hoặc video từ nhiều nguồn khác nhau.
* **Tiền xử lý (Preprocessing):** trước khi bắt đầu bất kỳ bước phân tích nào thì phải tiền xử lý hình ảnh để làm sạch dữ liệu, loại bỏ nhiễu, sửa các biến dạng, điều chỉnh độ sáng hoặc độ tương phản để cải thiện chất lượng hình ảnh. Mục tiêu của tiền xử lý là đảm bảo các thuật toán nhận được đầu vào hình ảnh chính xác và đáng tin cậy.
* **Trích xuất đặc trưng (Feature Extraction):** hệ thống sẽ nhận diện và trích xuất các đặc điểm đặc trưng của hình ảnh như các cạnh, góc, kết cấu, hình dạng hoặc phân bố màu sắc. Việc này giúp đơn giản hóa dữ liệu, giữ lại các thông tin quan trọng và loại bỏ những chi tiết không cần thiết, từ đó tạo ra đầu vào hiệu quả cho các bước phân tích và phân loại sau.
* **Phân loại đối tượng (Object Classification):** Sau khi trích xuất các đặc trưng và hoàn thiện huấn luyện mô hình học máy (nếu có), hệ thống có thể phân loại và nhận diện các đối tượng trong hình ảnh bằng cách so sánh các đặc trưng đã trích xuất với các mẫu mà nó đã học được trong quá trình huấn luyện.
* **Nhận diện đối tượng (Object Identification):** Nhận diện đối tượng nhằm xác định chính xác các đối tượng cụ thể trong một bức tranh.
* **Theo dõi đối tượng (Object Tracking):** Quá trình theo dõi đối tượng trong thị giác máy tính liên quan đến việc theo dõi sự chuyển động của một đối tượng qua các khung hình liên tiếp trong video. Hệ thống sẽ xác định vị trí của đối tượng trong mỗi khung hình và duy trì sự liên tục trong suốt quá trình chuyển động.

### 2.1.3 Internet vạn vật (IoT)

2.1.3.1 Khái niệm

Internet vạn vật là một mô hình công nghệ trong đó các thiết bị vật lý được kết nối với nhau thông qua Internet để thu thập, chia sẻ và xử lý dữ liệu. Những thiết bị này có thể là cảm biến, thiết bị gia dụng, máy móc công nghiệp, phương tiện giao thông, hoặc bất kỳ đối tượng vật lý nào được trang bị bộ vi xử lý, cảm biến và kết nối mạng [13].

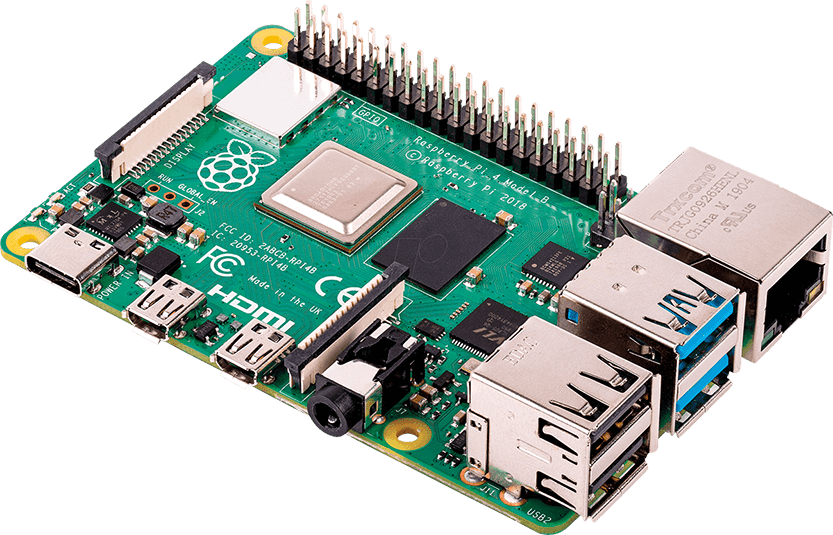
Khác với mô hình máy tính truyền thống, trong IoT, các thiết bị hoạt động tự động và giao tiếp với nhau hoặc với hệ thống trung tâm mà không cần sự can thiệp của con người. Mục tiêu chính là tạo ra một hệ sinh thái thông minh, giúp quản lý, giám sát và tối ưu hoá hoạt động trong thời gian thực.

2.1.3.2 Các thành phần chính trong hệ thống IoT của đề tài

#### a. Raspberry Pi 4 Model B

Raspberry Pi đóng vai trò là trung tâm của toàn bộ dự án. Đảm nhận các nhiệm vụ trong hệ thống phân loại trái cây như:

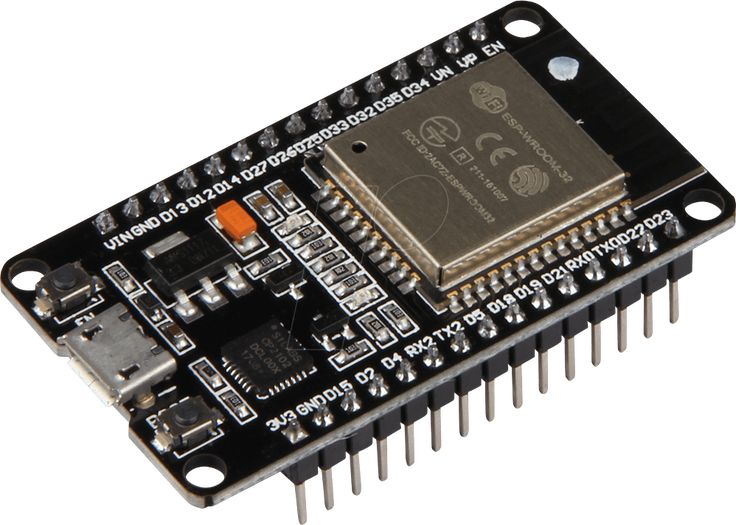
* Kết nối đến 2 webcams thông qua các cổng USB để chụp ảnh lấy dữ liệu nhận diện.
* Triển khai mô hình học máy đã được huấn luyện và đưa ra kết quả.
* Thông qua hai giao thức HTTP và MQTT, Raspberry gửi kết quả đến Server NestJS và ESP32 nhằm lưu lại kết quả phân loại và điều khiển các thiết bị IoT khác phân loại trái cây.



#### b. ESP32

ESP32 đóng vai trò điều khiển các thành phần còn lại trong hệ thống IoT (hiện tại là cánh tay robot, trong tương lai sẽ điều khiển thêm nếu tích hợp thiết bị khác).

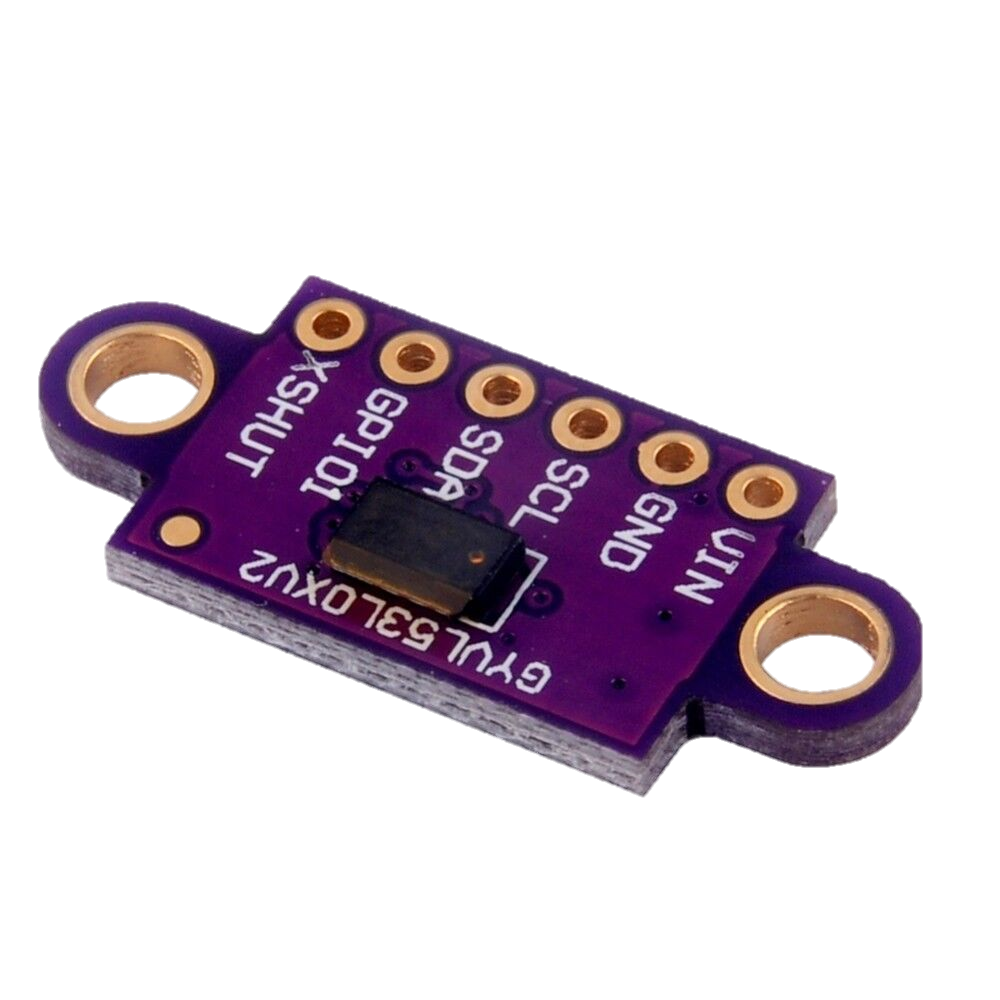
Thông qua kết nối MQTT, ESP32 sẽ nhận kết quả và điều khiển cánh tay phân loại.



#### c. Cảm biến khoảng cách VL53L0X

VL53L0X là một cảm biến đo khoảng cách sử dụng công nghệ Time-of-Flight (ToF) do hãng STMicroelectronics sản xuất. VL53L0X sử dụng tia laser hồng ngoại (class 1 – an toàn cho mắt) để phát xung và đo thời gian phản xạ từ vật thể, từ đó tính ra khoảng cách chính xác.

VL53L0X sử dụng giao tiếp I2C được kết nối với Raspberry Pi 4 và ESP32 (hệ thống sử dụng 2 cảm biến VL53L0X) để biết khi nào sẽ gửi tín hiệu để Raspberry chụp ảnh nhận diện trái cây trên băng chuyền.



#### d. Cánh tay Robot

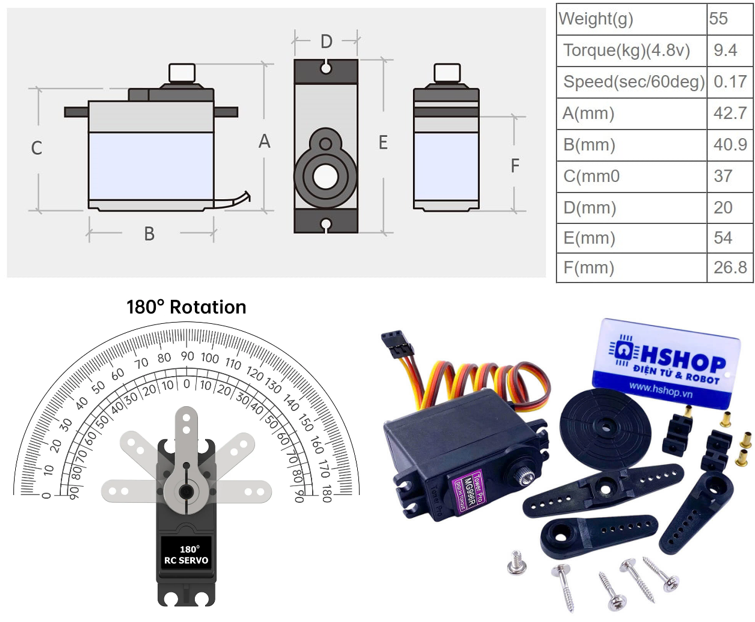
Cánh tay robot được sử dụng trong dự án là bộ khung kim loại 6 bậc tự do (6 DoF ARM Robot Chassis).

Cánh tay robot 6 DoF này được tích hợp cùng ESP32 để gắp và di chuyển trái cây đến với vị trí tương ứng của từng class sau khi nhận kết quả từ Raspberry Pi.

#### e. Động cơ Digital RC Servo MG996R 0-180° TowerPro

Động cơ Digital RC Servo MG996R 0-180° TowerPro với lực kéo, độ bền, độ chính xác và độ ổn định cao, động cơ có cấu tạo hộp số hoàn toàn bằng kim loại với trục chính bằng nhôm hợp kim có độ cứng cao 6061-T6 (màu bạc) giúp giảm trọng lượng và cũng để dễ dàng phân biệt với các loại Fake (có trục bằng đồng hoặc nhựa), ngoài ra trục chính và các trục quay trong hộp số còn được bổ trợ bạc đạn và các vòng đệm kim loại giúp tăng độ bền và độ chính xác của động cơ khi hoạt động.

Động cơ Digital RC Servo MG996R 0-180° được sử dụng trong dự án với vai trò làm các khớp nối, chuyển động cánh tay robot để gắp trái cây vào ô phân loại.



#### f. Băng tải công nghiệp mini

Băng tải mini sử dụng trong đề tài này là loại dùng động cơ DC 12–24V có điều tốc qua Dimmer. Băng tải dùng để truyền trái cây đến vị trí chụp ảnh của Raspberry và vào giá hứng để chờ cánh tay robot gắp đi. Đây là thành phần tối quan trọng sau Raspberry, với việc dẫn truyền tự động trái cây sẽ tăng thêm tính "tự động hoá" trong quy trình phân loại.

Việc hoạt động độc lập không phụ thuộc vào ESP32 giúp băng chuyền sẽ chạy liên tục với tốc độ tuỳ chỉnh, không gặp sự cố nhiễu băng tần PWM trên ESP32 khi phải phối hợp điều khiển cùng lúc băng chuyền và cánh tay Robot.



#### f. Webcam

Để thực hiện tác vụ chụp ảnh trái cây cho Raspberry nhận diện và phân loại, em sử dụng 2 Webcams với độ phân giải 1080p đặt ở 2 góc đầu và cuối băng chuyền, cách băng chuyền khoảng 10cm và chiếu góc xiên về giữa băng chuyền với góc 30 – 45 độ nhằm giữ khoảng cách vật thể vào đúng tiêu cự.

Việc lựa chọn Webcam máy tính thay vì sử dụng Raspberry Camera chuyên dụng có các lợi ích sau:

* Tối ưu về mặt chi phí nhưng đảm bảo chất lượng ảnh tốt hơn nhiều so với Raspberry Camera Module ở mức giá tương tự.
* Sử dụng được 2 camera cùng lúc, thay vì chỉ 1 camera nếu dùng Raspberry Camera Module.

### 2.1.4 Phương thức giao tiếp MQTT & Mosquitto

2.1.4.1 Khái niệm

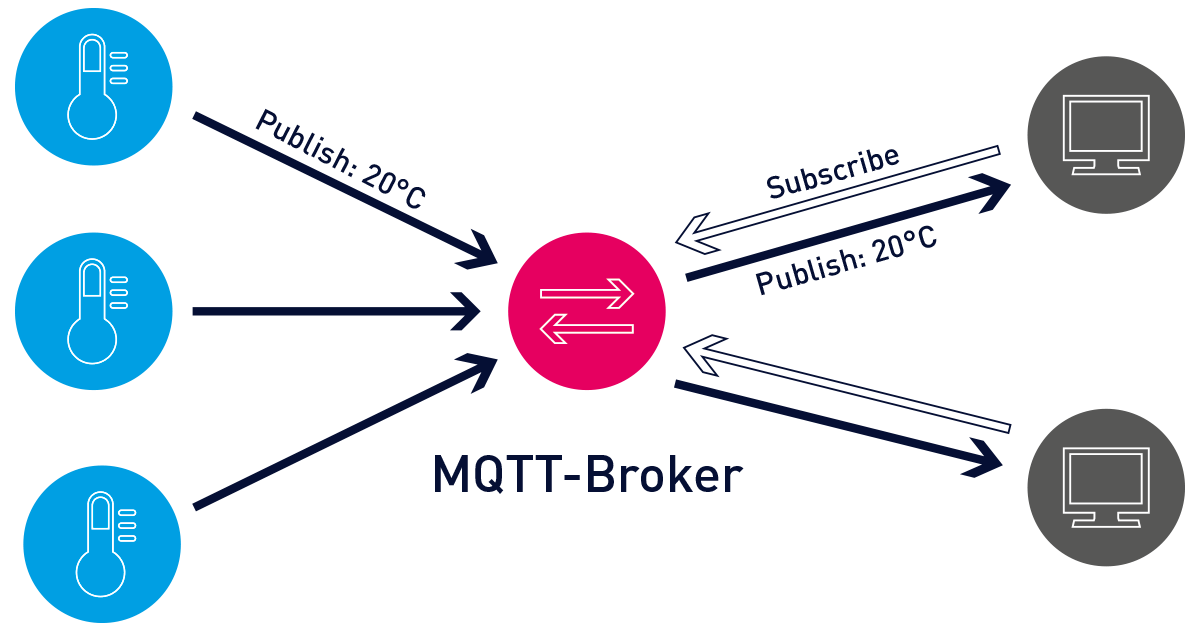
MQTT (Message Queuing Telemetry Transport) là một giao thức truyền thông nhẹ, được thiết kế để trao đổi dữ liệu giữa các thiết bị trong môi trường có băng thông thấp, độ trễ cao, hoặc kết nối không ổn định. Giao thức này được phát triển bởi IBM vào năm 1999 và sau đó trở thành tiêu chuẩn OASIS vào năm 2014. MQTT hoạt động theo mô hình xuất bản/đăng ký (publish/subscribe), với ba thành phần chính:

* Publisher (Nhà xuất bản): Thiết bị hoặc ứng dụng gửi dữ liệu (message) đến một chủ đề (topic).
* Subscriber (Người đăng ký): Thiết bị hoặc ứng dụng nhận dữ liệu từ các chủ đề mà nó đã đăng ký.
* Broker (Môi giới): Máy chủ trung gian quản lý việc truyền thông giữa Publisher và Subscriber, đảm bảo dữ liệu được chuyển đến đúng đích.

MQTT mang những đặc điểm nổi bật như sau:

* Nhẹ và hiệu quả: MQTT sử dụng giao thức TCP/IP với kích thước gói tin nhỏ (header chỉ 2 byte), giúp giảm tải băng thông và tiêu thụ tài nguyên.
* Chất lượng dịch vụ (QoS): MQTT hỗ trợ 3 mức QoS:
* QoS 0: Gửi một lần, không đảm bảo nhận (fire-and-forget).
* QoS 1: Gửi ít nhất một lần, có xác nhận (at least once).
* QoS 2: Gửi chính xác một lần, đảm bảo không trùng lặp (exactly once).
* Khả năng mở rộng: Giao thức phù hợp cho các hệ thống IoT với hàng nghìn thiết bị kết nối đồng thời.
* Bảo mật: MQTT hỗ trợ mã hóa qua SSL/TLS và xác thực người dùng bằng username/password.

Trong đề tài này, em sử dụng MQTT để giao tiếp giữa Raspberry và ESP32. Khi Raspberry hoàn tất quy trình nhận diện – gửi kết quả đến server thì sẽ gửi kết quả đến ESP32 để nó điều khiển cánh tay phân loại.

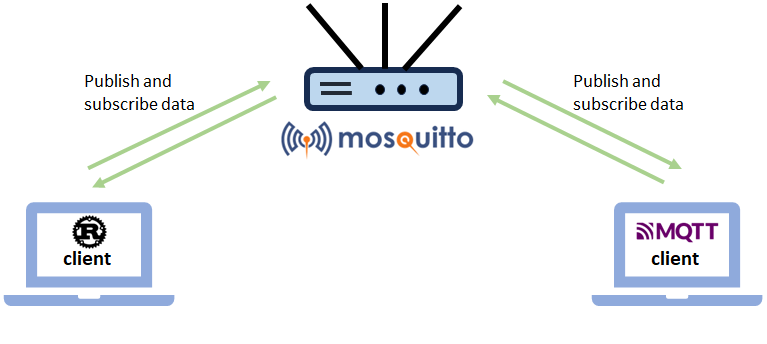


2.1.4.2 Mosquitto

Mosquitto là một phần mềm mã nguồn mở (open-source) hoạt động như một MQTT Broker, được phát triển bởi Eclipse Foundation. Được viết bằng ngôn ngữ C, Mosquitto nổi bật với hiệu suất cao, tính ổn định, và khả năng triển khai trên nhiều nền tảng, từ máy tính cá nhân đến các thiết bị nhúng như Raspberry Pi.

Một số ưu điểm của Mosquitto như:

* Nhẹ và tiết kiệm tài nguyên: Mosquitto tiêu thụ rất ít bộ nhớ và CPU, phù hợp cho các hệ thống có tài nguyên hạn chế.
* Hỗ trợ đầy đủ MQTT: Mosquitto hỗ trợ các phiên bản MQTT 3.1, 3.1.1, và 5.0, bao gồm tất cả các mức QoS và tính năng bảo mật như SSL/TLS, xác thực người dùng.
* Tích hợp dễ dàng: Phần mềm có thể được cài đặt và cấu hình nhanh chóng trên các hệ điều hành như Linux, Windows, và macOS.
* Cộng đồng hỗ trợ mạnh mẽ: Là một dự án mã nguồn mở, Mosquitto được hỗ trợ bởi cộng đồng lớn, với tài liệu phong phú và các bản cập nhật thường xuyên.

Trong dự án này, Mosquitto được cài đặt và hoạt động trên Raspberry Pi 4 như một dịch vụ tự động, quản lý việc truyền thông kết quả phân loại đến ESP32. Raspberry tạo topic và gửi kết quả, ESP32 sẽ đăng ký vào topic đó và nhận kết quả ngay.

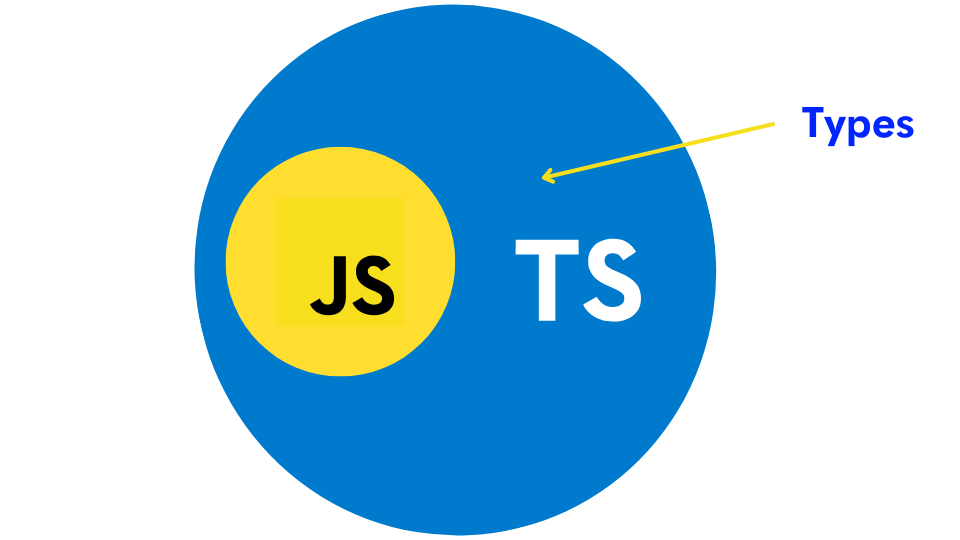
### 2.1.5 Website

2.1.5.1 TypeScript

TypeScript là một siêu tập hợp (superset) của JavaScript, được phát triển bởi Microsoft và ra mắt lần đầu vào năm 2012. TypeScript mở rộng JavaScript bằng cách bổ sung hệ thống kiểu tĩnh (static typing) và các tính năng lập trình hướng đối tượng nâng cao, giúp cải thiện tính bảo trì, khả năng mở rộng và độ an toàn của mã nguồn. TypeScript được biên dịch (transpile) thành JavaScript thuần để chạy trên bất kỳ môi trường nào hỗ trợ JavaScript, như trình duyệt web hoặc Node.js.

TypeScript được thiết kế để hỗ trợ phát triển các ứng dụng quy mô lớn, đặc biệt là trong các dự án phức tạp, nơi việc quản lý mã nguồn và phát hiện lỗi sớm là rất quan trọng. Nó cung cấp các công cụ như kiểm tra kiểu (type checking) tại thời điểm biên dịch, hỗ trợ IntelliSense trong các IDE, và khả năng tích hợp với các framework hiện đại như NestJS, React, hoặc Angular.

Các đặc điểm nổi bật của TypeScript:

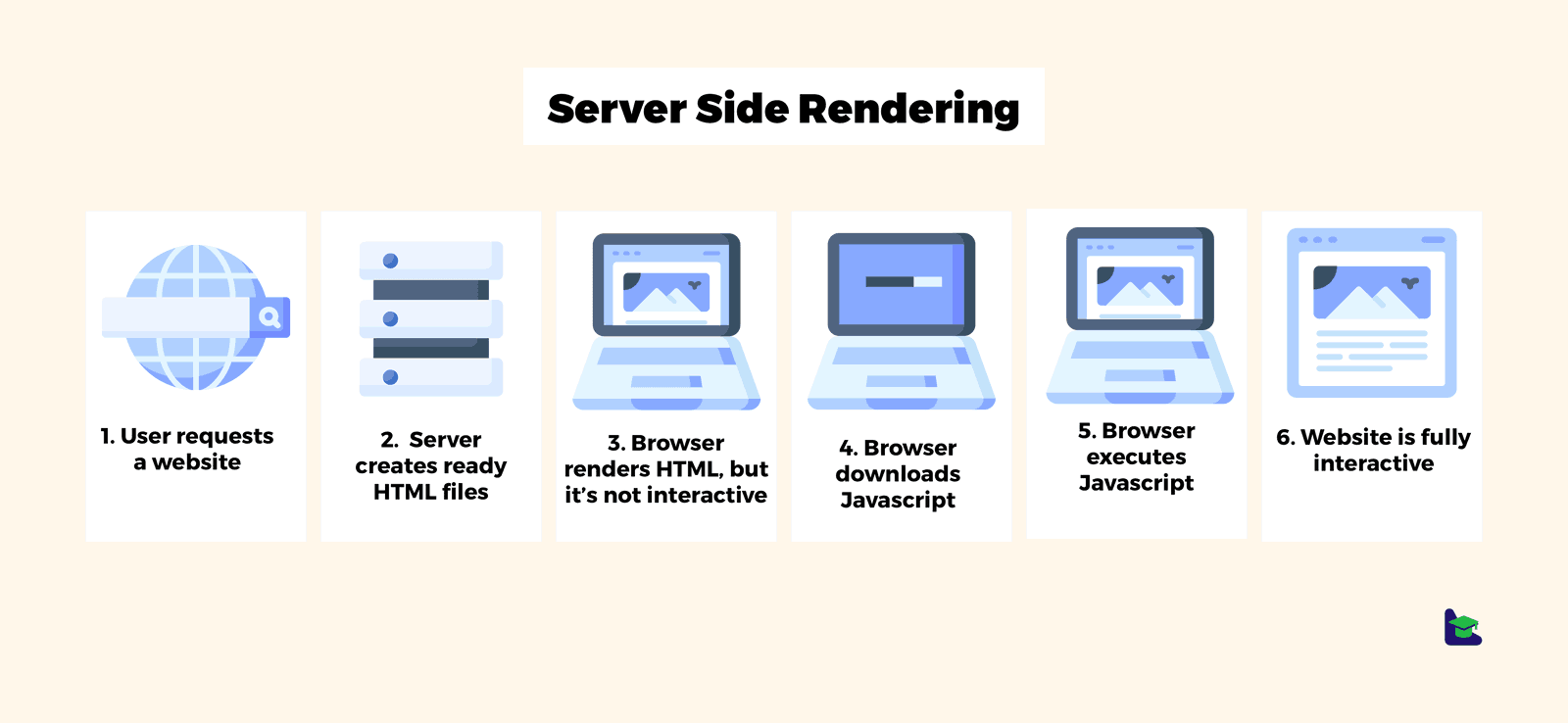
* Hệ thống kiểu tĩnh (Static Typing): TypeScript cho phép định nghĩa kiểu dữ liệu (types) cho biến, tham số, và giá trị trả về, giúp phát hiện lỗi sớm trong quá trình phát triển.
* Hỗ trợ lập trình hướng đối tượng (OOP): TypeScript cung cấp các tính năng như lớp (classes), giao diện (interfaces), kế thừa (inheritance), và sửa đổi truy cập (access modifiers) như public, private, protected.
* Giao diện (Interfaces) và Kiểu (Types): TypeScript sử dụng interfaces và types để định nghĩa cấu trúc dữ liệu, đảm bảo tính nhất quán và dễ tái sử dụng.
* Hỗ trợ IntelliSense: TypeScript cung cấp gợi ý mã (code completion) và kiểm tra lỗi trong các IDE như Visual Studio Code, giúp tăng năng suất lập trình.
* Tương thích ngược với JavaScript: Mọi mã JavaScript hợp lệ đều là mã TypeScript hợp lệ. Điều này cho phép tích hợp dễ dàng với các thư viện JavaScript hiện có.
* Biên dịch thành JavaScript: TypeScript sử dụng trình biên dịch tsc để chuyển đổi mã TypeScript thành JavaScript, cho phép chạy trên bất kỳ môi trường hỗ trợ JavaScript.
* Hỗ trợ các tính năng hiện đại của JavaScript: TypeScript hỗ trợ đầy đủ các tính năng của ECMAScript mới nhất (như async/await, arrow functions).

2.1.5.2 NextJS

NextJS là một framework có mã nguồn mở được xây dựng trên nền tảng của React, cho phép chúng ta xây dựng các trang web tĩnh có tốc độ siêu nhanh và thân thiện với người dùng, cũng như xây dựng các ứng dụng web React.

NextJS được ra đời vào năm 2016, thuộc sở hữu của Vercel. NextJS bắt đầu trở nên phổ biến vào năm 2018 và tiếp tục tăng trưởng mạnh mẽ trong cộng đồng phát triển web. Sự kết hợp của các tính năng như Server-side Rendering (SSR) với Static Site Generation (SSG) đã giúp NextJS trở thành sự lựa chọn hấp dẫn cho nhiều dự án. Một số tính năng chính của NextJS như:

* Routing:
* Automatic Routing: NextJS sẽ tự động tạo các router dựa trên cấu trúc thư mục dự án.
* Nested Routing: NextJS tạo các router lồng nhau nếu ta tạo các thư mục con lồng nhau trong cấu trúc thư mục dự án.
* Dynamic Routes: Tạo route động bằng cách sử dụng dấu **[]** trong tên file.
* Link Component: Tạo liên kết giữa các trang theo hướng tối ưu hiệu suất.
* Query Parameters: Truyền dữ liệu giữa các trang thông qua query parameters (?).
* Rendering:
* Server-side Rendering (SSR)
* Client-side Rendering (CSR)

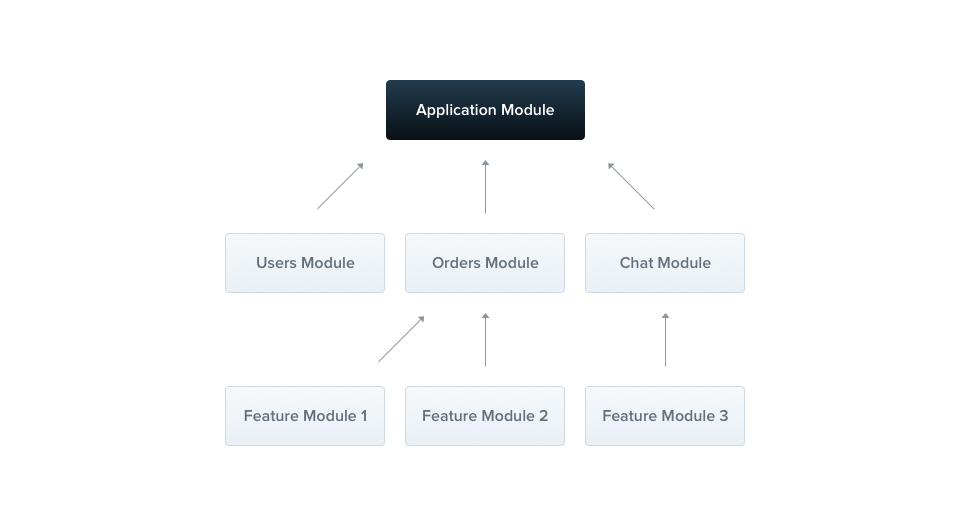




Trong dự án này, NextJS được em sử dụng để xây dựng phần Frontend (giao diện) cho website quản trị.

2.1.5.3 NestJS

NestJS là một framework mã nguồn mở để phát triển ứng dụng server-side (backend applications) bằng ngôn ngữ TypeScript hoặc JavaScript. Được xây dựng trên cơ sở của Node.js để tạo ra một môi trường phát triển hiện đại và mạnh mẽ cho việc xây dựng các ứng dụng web và API.



NestJS triển khai mô hình kiến trúc lõi (core architecture) dựa trên các nguyên tắc của Angular, đặc biệt là sử dụng Dependency Injection (DI) và Modules (Các module).

Cấu trúc của NestJS được xây dựng dựa trên mô hình kiến trúc lõi (core architecture) giúp tạo ra một ứng dụng server-side (backend application) rõ ràng, dễ quản lý và dễ mở rộng. Cấu trúc NestJS thường được tổ chức thành các phần chính sau:

* Module (Các module): Module là một phần cơ bản trong cấu trúc NestJS. Mỗi ứng dụng NestJS bao gồm ít nhất một module gốc (root module) và có thể có nhiều module con. Module là nơi tổ chức các thành phần của ứng dụng như Controllers, Providers và các thành phần khác. Mỗi module đại diện cho một phần chức năng cụ thể của ứng dụng.
* Controller (Bộ điều khiển): Controllers là thành phần chịu trách nhiệm xử lý các yêu cầu HTTP từ phía client và trả về kết quả tương ứng. Controllers là nơi xử lý các request và trả về các response. Các phương thức của controller được chú thích (decorated) bằng các decorator như @Get(), @Post(), @Put(), v.v., để chỉ định các route và phương thức HTTP tương ứng.
* Provider (Các nhà cung cấp): Providers là thành phần chịu trách nhiệm cung cấp các dịch vụ cho ứng dụng. Đây có thể là các service, repository, logger, v.v. Providers sử dụng dependency injection để chèn vào các thành phần khác và có thể được sử dụng bởi các controllers hoặc các providers khác.
* Middleware (Trung gian): Middleware là các hàm xử lý mà NestJS sử dụng để xử lý các yêu cầu HTTP trước khi chúng đến các route xử lý chính. Middleware có thể được sử dụng để thực hiện các thao tác chung như xác thực, ghi log, xử lý lỗi, v.v.
* Filter (Bộ lọc): Filters được sử dụng để xử lý các exception (ngoại lệ) xảy ra trong ứng dụng. Filters cho phép xử lý và thay đổi response trước khi gửi về client khi có exception xảy ra.
* Guard (Bảo vệ): Guards được sử dụng để kiểm tra xem một yêu cầu có thể được xử lý hoặc không. Guards cho phép thực hiện các kiểm tra xác thực hoặc kiểm tra quyền trước khi xử lý một yêu cầu.
* Interceptor (Bộ chặn): Interceptors là các hàm xử lý mà NestJS sử dụng để chặn và thay đổi response trước khi nó được gửi về client. Interceptors có thể được sử dụng để thực hiện các thao tác chung trên response trước khi nó đi ra ngoài.
* Exception (Ngoại lệ): Exception handling (xử lý ngoại lệ) là một phần quan trọng của cấu trúc NestJS. Exception handling cho phép xử lý các exception xảy ra trong ứng dụng và trả về các thông báo lỗi thích hợp cho client.

Trong dự án này, NestJS được sử dụng để xây dựng phần Backend, vừa hỗ trợ các yêu cầu từ phía Frontend, vừa thao tác với Raspberry thông qua HTTP và SocketIO.

2.1.5.4 Kiến trúc RESTful API

RESTful API (Application Programming Interface tuân theo kiến trúc REST) là một giao diện lập trình ứng dụng được thiết kế dựa trên các nguyên tắc và ràng buộc của kiến trúc REST (Representational State Transfer), do Roy Fielding định nghĩa trong luận văn tiến sĩ năm 2000. REST là một phong cách kiến trúc phần mềm dùng để xây dựng các hệ thống phân tán, chủ yếu hoạt động qua giao thức HTTP, nhằm hỗ trợ giao tiếp giữa client (ứng dụng phía người dùng) và server (hệ thống phía máy chủ).

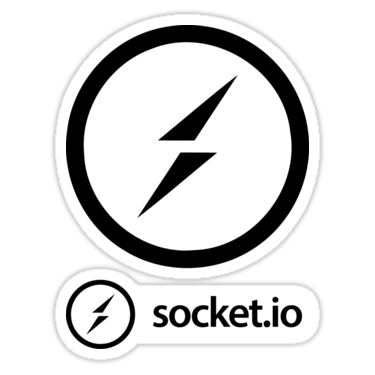


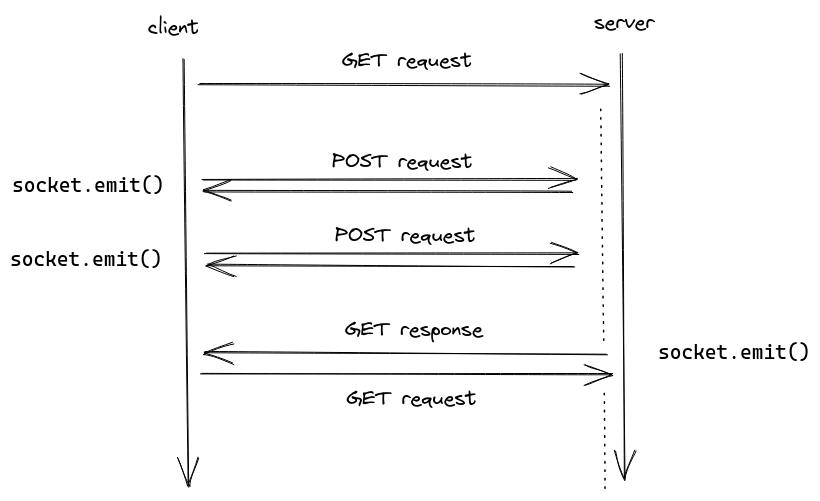
RESTful API cho phép tương tác với các tài nguyên. Các tài nguyên này được biểu diễn dưới dạng dữ liệu (thường là JSON hoặc XML) và được truy cập hoặc thao tác bằng các phương thức HTTP tiêu chuẩn như GET, POST, PUT, DELETE,… tương ứng với các hoạt động CRUD (Create, Read, Update, Delete). Kiến trúc RESTful API đặc biệt tương thích với framework NestJS.

Các nguyên tắc cốt lõi của RESTful API:

* Kiến trúc Client-Server: Tách biệt rõ ràng giữa client và server, cho phép phát triển độc lập. Client chịu trách nhiệm hiển thị giao diện người dùng, trong khi server xử lý logic nghiệp vụ và lưu trữ dữ liệu. Điều này cải thiện tính bảo trì và khả năng mở rộng.
* Không trạng thái (Stateless): Mỗi yêu cầu từ client đến server phải chứa đầy đủ thông tin cần thiết để server xử lý. Server không lưu trữ trạng thái của client giữa các yêu cầu, giúp hệ thống dễ mở rộng và giảm tải.
* Khả năng lưu trữ (Cacheable): Phản hồi từ server có thể được lưu trữ (cache) ở phía client hoặc các tầng trung gian (như proxy) để cải thiện hiệu suất và giảm tải cho server. Các tiêu đề HTTP như Cache-Control được sử dụng để chỉ định khả năng lưu trữ.
* Hệ thống phân tầng (Layered System): Hệ thống có thể được tổ chức thành các tầng (layers), chẳng hạn như tầng bảo mật, cân bằng tải hoặc lưu trữ. Client không cần biết mình đang giao tiếp trực tiếp với server hay qua một tầng trung gian, điều này tăng tính linh hoạt và bảo mật.
* Giao diện thống nhất (Uniform Interface): RESTful API cung cấp một giao diện chuẩn hóa, bao gồm:
* Xác định tài nguyên: Mỗi tài nguyên được đại diện bằng một URI duy nhất (ví dụ: /users, /products/123).
* Thao tác tài nguyên qua biểu diễn: Client tương tác với tài nguyên thông qua các biểu diễn (representations) như JSON hoặc XML.
* Thông điệp tự mô tả: Các yêu cầu và phản hồi chứa đủ thông tin (qua tiêu đề HTTP, mã trạng thái) để mô tả ý nghĩa.
* HATEOAS (Hypermedia as the Engine of Application State): Phản hồi có thể chứa các liên kết (links) để dẫn dắt client thực hiện các hành động tiếp theo, dù nguyên tắc này ít được áp dụng trong thực tế.
* Code theo yêu cầu (Code on Demand) - Tùy chọn: Server có thể gửi mã thực thi (như JavaScript) đến client để mở rộng chức năng, nhưng nguyên tắc này hiếm khi được sử dụng trong RESTful API.

### 2.1.6 SocketIO

Socket.IO là một thư viện JavaScript mạnh mẽ, được sử dụng để xây dựng các ứng dụng giao tiếp thời gian thực (real-time), hai chiều (bidirectional) giữa client và server. Socket.IO dựa trên giao thức WebSocket, nhưng bổ sung các tính năng như khả năng tự động kết nối lại (reconnection), hỗ trợ fallback sang các phương thức khác (như HTTP long-polling) khi WebSocket không khả dụng, và quản lý các sự kiện (event-driven communication). Thư viện này hoạt động trên cả phía client (trình duyệt hoặc ứng dụng) và server (thường được triển khai với Node.js), giúp đơn giản hóa việc phát triển các ứng dụng yêu cầu cập nhật dữ liệu tức thời.



Socket.IO cho phép truyền dữ liệu theo mô hình event-based, trong đó client và server có thể gửi hoặc nhận các sự kiện tùy chỉnh (custom events) thông qua các kênh giao tiếp (channels). Điều này làm cho Socket.IO trở thành lựa chọn lý tưởng cho các ứng dụng như chat thời gian thực, thông báo đẩy, trò chơi trực tuyến, hoặc các hệ thống giám sát dữ liệu.

Các đặc điểm chính của SocketIO:

* Giao tiếp hai chiều (Bidirectional Communication): Socket.IO hỗ trợ giao tiếp đồng thời giữa client và server, cho phép cả hai phía gửi và nhận dữ liệu mà không cần chờ yêu cầu từ phía còn lại.
* Hỗ trợ WebSocket và Fallback: Socket.IO sử dụng WebSocket làm giao thức chính, nhưng tự động chuyển sang các phương thức thay thế như HTTP long-polling nếu WebSocket không được hỗ trợ, đảm bảo khả năng tương thích với nhiều môi trường.
* Quản lý sự kiện (Event-Driven): Socket.IO sử dụng mô hình phát-xử lý sự kiện (emit-handle events). Client và server có thể phát (emit) các sự kiện tùy chỉnh và lắng nghe (listen) các sự kiện từ phía đối diện.
* Khả năng tự động kết nối lại (Automatic Reconnection): Socket.IO tự động xử lý các sự cố ngắt kết nối và cố gắng kết nối lại, đảm bảo trải nghiệm người dùng liền mạch.
* Phòng và không gian tên (Rooms and Namespaces): Socket.IO hỗ trợ tổ chức người dùng vào các "phòng" (rooms) hoặc "không gian tên" (namespaces) để quản lý giao tiếp nhóm hoặc phân tách logic giao tiếp. Ví dụ, một phòng chat riêng có thể được tạo để chỉ gửi tin nhắn đến một nhóm người dùng cụ thể.
* Hỗ trợ đa nền tảng: Socket.IO hoạt động trên nhiều nền tảng, bao gồm trình duyệt web, ứng dụng di động (thông qua các thư viện client tương thích) và server Node.js.
* Khả năng mở rộng: Socket.IO có thể được tích hợp với các công cụ như Redis để hỗ trợ mở rộng quy mô (scalability) trong các hệ thống phân tán, cho phép nhiều instance server xử lý đồng thời các kết nối.

Trong dự án này, SocketIO hỗ trợ việc cập nhật dữ liệu tự động theo thời gian thực từ NestJS đến NextJS tại các biểu đồ thống kê, danh sách kết quả nhận diện khi có kết quả mới. Ngoài ra, SocketIO còn hỗ trợ cập nhật trực tiếp cấu hình cho Raspberry khi nó đang hoạt động từ Website.

### 2.1.7 Cơ sở dữ liệu

2.1.7.1 MySQL – Hệ quản trị cơ sở dữ liệu

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở (open-source RDBMS - Relational Database Management System) được sử dụng rộng rãi trong các ứng dụng web và doanh nghiệp. MySQL lưu trữ dữ liệu dưới dạng bảng, hỗ trợ các truy vấn SQL tiêu chuẩn và cung cấp hiệu suất cao, khả năng mở rộng và độ tin cậy.

Đặc điểm nổi bật của MySQL:

* Hiệu suất cao: MySQL được tối ưu hóa cho các ứng dụng yêu cầu xử lý nhanh các truy vấn đọc và ghi.
* Khả năng mở rộng: Hỗ trợ các cơ chế như replication, clustering để xử lý khối lượng dữ liệu lớn.
* Tính phổ biến: MySQL được sử dụng rộng rãi, có cộng đồng lớn và nhiều công cụ hỗ trợ như phpMyAdmin, MySQL Workbench.
* Tính tương thích: MySQL hoạt động tốt trên nhiều nền tảng (Windows, Linux, macOS) và tích hợp dễ dàng với các framework như NestJS.



2.1.7.2 TypeORM – Thư viện ORM cho MySQL

TypeORM là một thư viện Object-Relational Mapping (ORM) được viết bằng TypeScript, cho phép ánh xạ các đối tượng trong mã nguồn (entities) với các bảng trong cơ sở dữ liệu quan hệ như MySQL. TypeORM giúp đơn giản hóa việc thao tác dữ liệu, từ việc tạo, truy vấn, cập nhật đến xóa, thông qua các API lập trình thay vì viết truy vấn SQL thủ công.

Một số đặc điểm nổi bật của TypeORM:

* Hỗ trợ TypeScript: TypeORM tận dụng hệ thống kiểu tĩnh của TypeScript để định nghĩa các thực thể (entities) với cấu trúc rõ ràng, giảm lỗi khi thao tác dữ liệu.

|  |
| --- |
| import { Entity, Column, PrimaryGeneratedColumn } from 'typeorm';  @Entity()  export class User {  @PrimaryGeneratedColumn()  id: number;  @Column()  name: string;  @Column()  email: string;  } |

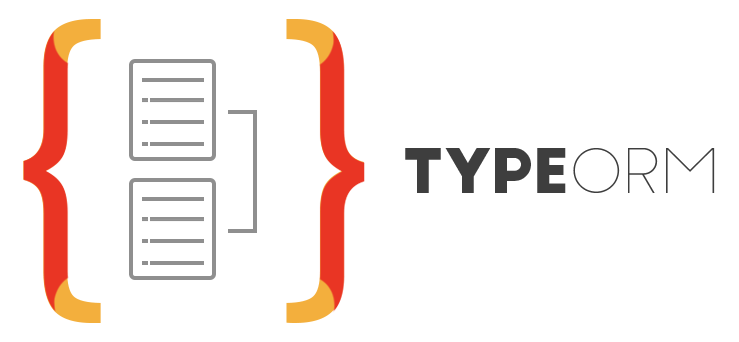
* Hỗ trợ nhiều cơ sở dữ liệu: TypeORM tương thích với MySQL, PostgreSQL, SQLite, và các RDBMS khác, giúp dễ dàng chuyển đổi cơ sở dữ liệu nếu cần.
* Truy vấn đơn giản: TypeORM cung cấp API trực quan để thực hiện các thao tác CRUD. Ví dụ:

|  |
| --- |
| const userRepository = dataSource.getRepository(User);  const user = await userRepository.findOneBy({ id: 1 }); |

Việc lựa chọn sử dụng thư viện TypeORM trong dự án nhờ các tiêu chí sau:

* Tận dụng TypeScript để đảm bảo tính nhất quán và an toàn kiểu dữ liệu khi làm việc với MySQL.
* Đơn giản hóa việc phát triển các API RESTful và ứng dụng thời gian thực trong NestJS.
* Hỗ trợ các tính năng như migration (quản lý schema cơ sở dữ liệu) và truy vấn phức tạp mà không cần viết SQL thủ công.
* Tăng khả năng bảo trì mã nguồn nhờ cấu trúc thực thể rõ ràng và tích hợp tốt với Dependency Injection của NestJS.

Trong dự án, MySQL đóng vai trò là hệ thống lưu trữ dữ liệu, trong khi TypeORM hoạt động như một tầng trung gian, giúp ứng dụng NestJS tương tác với MySQL một cách hiệu quả. TypeORM chuyển đổi các thao tác lập trình trong TypeScript thành các truy vấn SQL gửi đến MySQL, đồng thời ánh xạ các kết quả trả về thành các đối tượng TypeScript.



### 2.1.8 Postman

Postman là công cụ mạnh mẽ và phổ biến dùng để phát triển, kiểm thử và quản lý các API. Postman đặc biệt hữu ích trong việc kiểm thử RESTful API, giúp các nhà phát triển và tester tương tác với các endpoint, xác minh chức năng và đảm bảo chất lượng của API trong các dự án như những dự án sử dụng NestJS.

Postman cho phép tạo, gửi và lưu các yêu cầu HTTP (như GET, POST, PUT, DELETE), quản lý các bộ sưu tập (collections) yêu cầu, và tích hợp với các quy trình phát triển phần mềm như CI/CD. Ngoài ra, Postman hỗ trợ các tính năng như kiểm thử tự động, mock server và tài liệu hóa API, giúp tối ưu hóa quy trình phát triển và kiểm thử.

Một số chức năng được hỗ trợ trong Postman:

* Giao diện người dùng trực quan: Postman cung cấp giao diện đồ họa dễ sử dụng để tạo và gửi các yêu cầu HTTP, với các tab để quản lý tiêu đề (headers), tham số (query params), và thân yêu cầu (body) ở các định dạng như JSON, form-data.
* Hỗ trợ các phương thức HTTP: Postman hỗ trợ đầy đủ các phương thức HTTP tiêu chuẩn như GET, POST, PUT, PATCH, DELETE, phù hợp để kiểm thử RESTful API.
* Quản lý bộ sưu tập (Collections): Người dùng có thể tổ chức các yêu cầu thành các bộ sưu tập, giúp quản lý và chia sẻ dễ dàng trong nhóm phát triển. Các bộ sưu tập có thể được xuất/nhập dưới dạng JSON.
* Kiểm thử tự động: Postman hỗ trợ viết các kịch bản kiểm thử (test scripts) bằng JavaScript để tự động kiểm tra phản hồi của API, như mã trạng thái, nội dung JSON, hoặc thời gian phản hồi.
* Môi trường (Environments): Postman cho phép tạo các môi trường (ví dụ: development, staging, production) để quản lý các biến như URL cơ sở, token xác thực, hoặc tham số động.
* Mock Server: Postman cung cấp tính năng tạo mock server, cho phép mô phỏng phản hồi API trước khi backend được triển khai, hỗ trợ phát triển đồng thời giữa frontend và backend.
* Tài liệu hóa API: Postman tự động tạo tài liệu API từ các bộ sưu tập, bao gồm mô tả endpoint, phương thức, tham số và ví dụ phản hồi, giúp nhóm phát triển dễ dàng chia sẻ thông tin.
* Tích hợp CI/CD: Postman hỗ trợ tích hợp với các công cụ như Jenkins, GitHub Actions thông qua Newman (công cụ dòng lệnh của Postman), cho phép chạy các kịch bản kiểm thử API trong quy trình CI/CD.
* Hỗ trợ xác thực: Postman hỗ trợ nhiều phương thức xác thực như Basic Auth, OAuth 2.0, Bearer Token, giúp kiểm thử các API yêu cầu bảo mật.

Trong dự án này, Postman thường xuyên được sử dụng để kiểm tra các API Endpoint trong NestJS: dữ liệu, logic, lỗi,… đảm bảo các Endpoint đều đúng yêu cầu, ràng buộc.

### 2.1.9 Git

Git là một hệ thống quản lý phiên bản phân tán (Distributed Version Control System – DVCS), nó là một trong những hệ thống quản lý phiên bản phân tán phổ biến nhất hiện nay. Git cung cấp cho mỗi lập trình viên kho lưu trữ (repository) riêng chứa toàn bộ lịch sử thay đổi.



Git được sử dụng rộng rãi trong các dự án phát triển phần mềm, đặc biệt trong các dự án sử dụng các framework như NestJS, để quản lý mã nguồn, hỗ trợ cộng tác nhóm, và tích hợp với các quy trình phát triển hiện đại như CI/CD.

Một số đặc điểm nổi bật của Git:

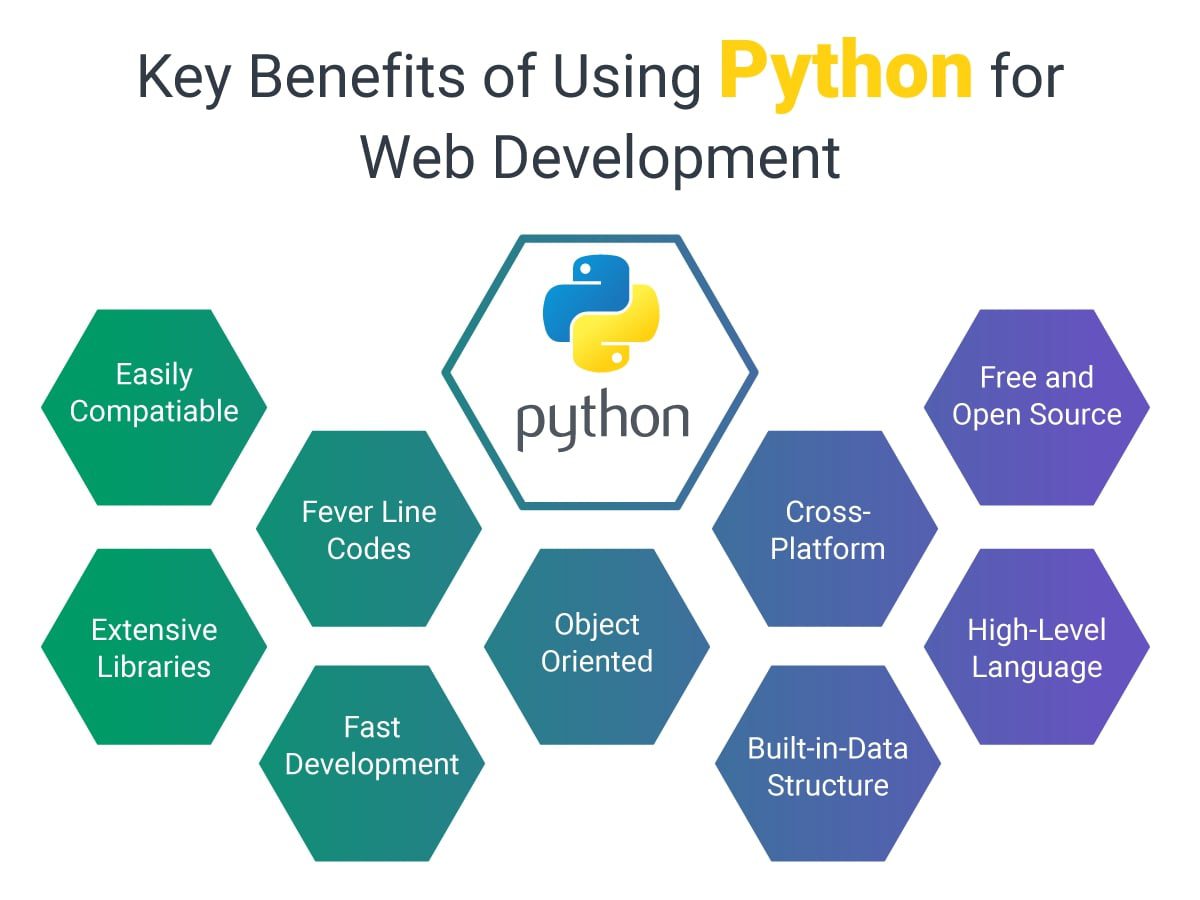
* Quản lý phiên bản phân tán: Mỗi nhà phát triển có một bản sao đầy đủ của kho lưu trữ, cho phép làm việc độc lập mà không cần kết nối liên tục với server trung tâm.
* Hệ thống nhánh (Branching) mạnh mẽ: Git hỗ trợ tạo và quản lý các nhánh dễ dàng, cho phép phát triển song song các tính năng, sửa lỗi, hoặc thử nghiệm mà không ảnh hưởng đến mã nguồn chính.
* Ghi nhận thay đổi (Commits): Git lưu trữ lịch sử thay đổi dưới dạng các commit, mỗi commit đi kèm với thông điệp mô tả và mã định danh duy nhất (hash).
* Khả năng quay lại (Rollback): Git cho phép hoàn tác các thay đổi hoặc quay về trạng thái trước đó thông qua các lệnh như git revert hoặc git reset.
* Tích hợp với nền tảng lưu trữ: Git tích hợp tốt với các dịch vụ như GitHub, GitLab, hoặc Bitbucket, cung cấp giao diện web để quản lý kho lưu trữ, pull requests, và đánh giá mã nguồn (code review).
* Hỗ trợ CI/CD: Git là nền tảng cho các pipeline CI/CD, cho phép tự động hóa kiểm thử và triển khai khi mã nguồn được đẩy lên kho lưu trữ.
* Nhẹ và nhanh: Git được tối ưu hóa để xử lý các dự án lớn với hiệu suất cao, ngay cả khi kho lưu trữ chứa hàng nghìn commit.

Git là công cụ quan trọng trong đề tài này, hỗ trợ việc lưu trữ các phiên bản mã nguồn trên Github để dễ dàng truy vấn, cải tiến hoặc quay về nếu xảy ra lỗi.

### 2.1.10 Python

Python là một ngôn ngữ lập trình bậc cao, mã nguồn mở và đa nền tảng. Python được sử dụng rộng rãi để phát triển các ứng dụng web, phát triển phần mềm, khoa học dữ liệu và máy học (ML).

Một số ưu điểm khiến Python được ưa chuộng:

* Cú pháp đơn giản và dễ đọc: Cú pháp của Python rất giống với ngôn ngữ tiếng Anh tự nhiên, dễ học và dễ đọc, giúp lập trình viên tập trung vào giải quyết vấn đề hơn là việc ghi nhớ cú pháp phức tạp.
* Đa dụng: Python có thể được sử dụng trong nhiều lĩnh vực khác nhau như phát triển web, khoa học dữ liệu, trí tuệ nhân tạo, tự động hóa, phân tích dữ liệu, và nhiều ứng dụng khác.
* Thư viện phong phú: Python có một kho thư viện đồ sộ, hỗ trợ nhiều tác vụ khác nhau. Các thư viện như NumPy, Pandas, TensorFlow, và Django giúp lập trình viên tiết kiệm thời gian và công sức khi phát triển các ứng dụng phức tạp.
* Cộng đồng lớn mạnh: Python có một cộng đồng lập trình viên rộng lớn và năng động, cung cấp nhiều tài liệu, hướng dẫn, và hỗ trợ qua các diễn đàn, nhóm thảo luận và các khóa học trực tuyến.
* Đa nền tảng: Python có thể chạy trên nhiều hệ điều hành khác nhau như Windows, macOS, Linux, Raspberry Pi,… giúp cho việc phát triển và triển khai ứng dụng trở nên dễ dàng.
* Khả năng mở rộng và tích hợp tốt: Python có thể dễ dàng tích hợp với các ngôn ngữ lập trình khác và các công nghệ hiện có, giúp nó trở thành một lựa chọn lý tưởng cho nhiều dự án khác nhau.
* Hỗ trợ từ các tổ chức lớn: Nhiều công ty và tổ chức lớn như Google, Facebook, NASA sử dụng Python và đóng góp vào việc phát triển ngôn ngữ này, làm tăng uy tín và sự tin cậy của nó.

Trong dự án này, Python được sử dụng để lập trình trên Raspberry với các chức năng quan trọng như:

* Triển khai mô hình đã được huấn luyện trên Raspberry.
* Kết nối và sử dụng cảm biến khoảng cách laser VL53L0X.
* Gửi request đến server NestJS.
* Sử dụng và kết nối với SocketIO của NestJS.
* Tải file mô hình mới trực tiếp từ NestJS khi nhận cấu hình mới.
* Sử dụng MQTT để giao tiếp với ESP32.

## 2.2 CÁC NGHIÊN CỨU TRƯỚC ĐÂY

### 2.2.1 Sa et al. (2016) - Hệ thống nhận diện trái cây dựa trên mạng nơ-ron tích chập (CNN)

* Nội dung nghiên cứu: Nhóm tác giả đề xuất hệ thống nhận diện trái cây dựa trên mạng nơ-ron tích chập (CNN), cụ thể là mô hình AlexNet. Dữ liệu huấn luyện là ảnh thực tế của các loại trái cây như táo và cam, được thu thập trong môi trường nông nghiệp và bán lẻ. Mô hình CNN học đặc trưng hình ảnh như màu sắc, hình dạng, và kết cấu để phân loại từng loại quả. Độ chính xác đạt được khoảng 90%, cho thấy tính ứng dụng cao của học sâu trong lĩnh vực nông nghiệp thông minh.
* Ưu điểm:
* Sử dụng tập dữ liệu lớn, hình ảnh thu từ môi trường thực tế, tăng tính khả dụng cho ứng dụng triển khai.
* Mô hình AlexNet có độ sâu hợp lý, phù hợp với phần cứng phổ biến vào thời điểm đó.
* Cho độ chính xác cao (~90%) trong phân loại hai loại quả điển hình.
* Hạn chế:
* Chỉ dừng lại ở mức phân loại hình ảnh tĩnh, không xử lý luồng ảnh thời gian thực.
* Chưa tích hợp với hệ thống IoT hoặc thiết bị điều khiển như robot, băng chuyền, cảm biến.
* Thiếu giao diện quản lý hoặc phần mềm hỗ trợ người dùng cuối (dashboard, giám sát).
* Hệ thống chưa đánh giá hiệu năng trên thiết bị nhúng như Raspberry Pi, giới hạn tính triển khai thực tế.

### 2.2.2 Hossain et al. (2019) - Hệ thống phân loại trái cây sử dụng học sâu với mô hình ResNet

* Nội dung nghiên cứu: Nghiên cứu đề xuất một hệ thống phân loại trái cây tự động sử dụng học sâu, cụ thể là mô hình ResNet. Hệ thống được thiết kế để ứng dụng trong môi trường công nghiệp, với khả năng nhận diện và phân loại các loại trái cây khác nhau thông qua hình ảnh. Mô hình được huấn luyện trên tập dữ liệu lớn, bao gồm hình ảnh của nhiều loại trái cây trong điều kiện ánh sáng và góc chụp đa dạng. Kết quả cho thấy hệ thống đạt độ chính xác khoảng 92% trong việc phân loại trái cây.
* Ưu điểm:
* Tích hợp IoT: Hệ thống được thiết kế để hoạt động trong môi trường công nghiệp, có khả năng kết nối và truyền dữ liệu qua mạng, phù hợp với các ứng dụng IoT.
* Hiệu quả cao: Mô hình ResNet cho phép nhận diện chính xác các loại trái cây khác nhau, ngay cả trong điều kiện ánh sáng và góc chụp thay đổi.
* Khả năng mở rộng: Hệ thống có thể được mở rộng để nhận diện thêm nhiều loại trái cây khác nhau bằng cách huấn luyện lại mô hình với dữ liệu mới.
* Hạn chế:
* Thiếu giao diện quản lý: Nghiên cứu không đề cập đến việc phát triển giao diện người dùng hoặc hệ thống quản lý để giám sát và điều khiển quá trình phân loại.
* Chưa tối ưu cho thiết bị nhúng: Mô hình ResNet có cấu trúc phức tạp, đòi hỏi tài nguyên tính toán lớn, chưa phù hợp để triển khai trên các thiết bị nhúng như Raspberry Pi mà không có các biện pháp tối ưu hóa thêm.

### 2.2.3 Elwakeel et al. (2023) - Thiết kế hệ thống phân loại trái cây tự động với chi phí thấp, sử dụng cảm biến màu RGB

* Nội dung nghiên cứu: Nghiên cứu này trình bày việc thiết kế và tối ưu hóa một hệ thống phân loại trái cây tự động, chi phí thấp, sử dụng cảm biến màu RGB. Hệ thống bao gồm các thành phần phần cứng như khung máy, băng tải, vùng quét, hộp nhựa, các thành phần điện tử (động cơ bước, cảm biến màu RGB, Arduino Mega, driver động cơ) và phần mềm (Arduino IDE và C++). Mục tiêu là tạo ra một hệ thống phân loại trái cây dựa trên màu sắc, có thể hoạt động hiệu quả và chính xác trong môi trường nông nghiệp.
* Ưu điểm:
* Chi phí thấp và dễ triển khai: Sử dụng các thành phần phổ biến và giá rẻ như Arduino và cảm biến màu RGB, phù hợp với các mô hình nông nghiệp quy mô nhỏ.
* Hiệu quả trong phân loại: Hệ thống có khả năng phân loại trái cây dựa trên màu sắc một cách chính xác, giúp nâng cao chất lượng sản phẩm.
* Dễ dàng tích hợp và mở rộng: Thiết kế mô-đun cho phép dễ dàng tích hợp với các hệ thống khác hoặc mở rộng chức năng trong tương lai.
* Hạn chế:
* Phụ thuộc vào màu sắc: Hệ thống chỉ dựa vào màu sắc để phân loại, có thể gặp khó khăn với các loại trái cây có màu tương tự hoặc trong điều kiện ánh sáng không ổn định.
* Chưa tích hợp học máy: Không sử dụng các thuật toán học máy hoặc mạng nơ-ron để cải thiện độ chính xác và khả năng phân loại.
* Thiếu giao diện người dùng: Chưa có giao diện quản lý hoặc hệ thống giám sát từ xa để theo dõi và điều khiển quá trình phân loại.

### 2.2.4 Khoảng trống nghiên cứu và định hướng phát triển đề tài

2.2.4.1 Hạn chế của các nghiên cứu trước đây

1. **Tập trung nhiều vào phần nhận diện hình ảnh, thiếu một hệ thống tự động toàn diện và tổng thể:** Nghiên cứu của Sa et al. (2016) hay của Hossain et al. (2019) chủ yếu sử dụng các mô hình học máy như CNN, ResNet để nhận diện hình ảnh trái cây, đạt độ chính xác tốt trong môi trường thử nghiệm. Tuy nhiên, nghiên cứu của họ chưa có phần tích hợp thêm hệ thống phần cứng hay website quản lý.
2. **Thiếu tích hợp IoT với các mô hình học máy hiện đại:** Các hệ thống nhúng như Arduino hoặc ESP32 có thể xử lý tác vụ đơn giản, nhưng hiếm nghiên cứu nào khai thác sức mạnh của các mô hình học máy nhẹ như MobileNetV2, vốn phù hợp với các thiết bị có tài nguyên hạn chế. Điều này dẫn đến việc các hệ thống vẫn còn thủ công hoặc sử dụng thuật toán đơn giản, thiếu độ chính xác và khả năng mở rộng.
3. **Thiếu giao diện người dùng để phục vụ quản lý:** Một điểm yếu chung trong nhiều nghiên cứu là thiếu giao diện trực quan để giám sát, quản lý hoặc tương tác với hệ thống. Các công nghệ phát triển website như NextJS, NestJS hoặc MySQL rất ít được đề cập, khiến hệ thống thiếu khả năng vận hành thực tế trong môi trường công xưởng – nơi cần kiểm soát trạng thái thiết bị, lưu lịch sử phân loại, hoặc xuất báo cáo.

2.2.4.2 Định hướng phát triển của đề tài

Từ những nghiên cứu trên, rút ra được nội dung nghiên cứu, điểm mạnh và điểm yếu thì em đã định hướng được những yếu tố mà đề tài của mình cần có như sau:

1. Xây dựng hệ thống phân loại trái cây thông minh, tự động và trực quan:

* Thị giác máy tính và máy học: Sử dụng OpenCV để tiền xử lý dữ liệu, kết hợp với các mô hình học máy như CNN MobileNetV2, SVM để nhận diện và phân loại trái cây theo 6 lớp.
* Hệ thống IoT: Raspberry Pi đảm nhiệm việc xử lý ảnh và chạy mô hình học máy, ESP32 điều khiển cánh tay robot và băng chuyền để di chuyển vật phẩm.
* Giao tiếp và điều khiển: Sử dụng các giao thức như MQTT, HTTP/REST, SocketIO để truyền dữ liệu giữa vi điều khiển và máy chủ, đảm bảo tính linh hoạt và mở rộng.
* Giao diện Website quản lý trực tuyến: Sử dụng NextJS cho frontend và NestJS cho backend, tích hợp MySQL để lưu trữ kết quả phân loại, trạng thái thiết bị và thông tin người dùng.

1. Tối ưu cho thiết bị nhúng, đảm bảo độ chính xác cao: Mục tiêu đặt ra là đạt độ chính xác ≥ 90% cho 6 lớp trái cây, đồng thời đảm bảo mô hình hoạt động mượt mà trên thiết bị có tài nguyên hạn chế như Raspberry Pi, từ đó giảm chi phí triển khai.
2. Giải pháp thực tiễn cho công nghiệp hóa nông nghiệp: Hệ thống hướng đến ứng dụng thực tế trong các cơ sở phân loại nông sản (trái cây) quy mô nhỏ và vừa, giúp giảm chi phí nhân công, tăng năng suất và tính đồng nhất trong khâu phân loại. Ngoài ra, khả năng giám sát từ xa qua website giúp nâng cao hiệu quả quản lý và bảo trì hệ thống.

# CHƯƠNG 3 PHƯƠNG PHÁP NGHIÊN CỨU

# NHẬN XÉT CỦA NGƯỜI HƯỚNG DẪN