

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG  
KHOA CÔNG NGHỆ THÔNG TIN



**ĐỒ ÁN TỐT NGHIỆP**  
**NGÀNH: CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI:**  
**NGHIÊN CỨU VÀ ỨNG DỤNG YOLOv8 TRONG**  
**NHẬN DẠNG VÀ PHÂN LOẠI ĐỘ CHÍN TRÁI**  
**MÍT PHỤC VỤ NÔNG NGHIỆP THÔNG MINH**

Người hướng dẫn : TS. NGUYỄN HÀ HUY CƯỜNG  
Sinh viên thực hiện :  
1. NGUYỄN VĂN DUY MSSV:2151220251  
2. LÊ NAM MSSV:2151220149  
Lớp : 21CT5 – 21CT3

Đà Nẵng, ...../202...

**TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN TỐT NGHIỆP**  
**NGÀNH: CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI:**  
**NGHIÊN CỨU VÀ ỨNG DỤNG YOLOv8 TRONG**  
**NHẬN DẠNG VÀ PHÂN LOẠI ĐỘ CHÍN TRÁI**  
**MÍT PHỤC VỤ NÔNG NGHIỆP THÔNG MINH**

Người hướng dẫn : **TS. NGUYỄN HÀ HUY CƯỜNG**  
Sinh viên thực hiện :  
1. **NGUYỄN VĂN DUY** MSSV:2151220251  
2. **LÊ NAM** MSSV:2151220149  
Lớp : 21CT5 – 21CT3

**Đà Nẵng, ...../202...**

## **NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

## **NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN**

## TÓM TẮT

Tên đề tài: Nghiên cứu và ứng dụng YOLOv8 trong nhận dạng và phân loại độ chín trái mít phục vụ nông nghiệp thông minh.

Sinh viên thực hiện:

1. Nguyễn Văn Duy Mã số SV: 2151220251 Lớp: 21CT5

2. Lê Nam Mã số SV: 2151220149 Lớp: 21CT3

Tóm tắt đề tài: Xây dựng website nền tảng AI giám sát và nhận dạng phân loại độ chín của trái mít, tích hợp model YOLOv8 để nhận dạng vật thể có các chức năng như phân tích ảnh, phân tích video, so sánh mô hình, chatbot hỗ trợ. Ngoài ra chúng em còn xây dựng một Mobile App cũng có những chức năng trọng yếu như trên để có thể giúp người dùng sử dụng một cách trực quan và dễ dàng hơn.

## LỜI NÓI ĐẦU

Trong xu thế phát triển của nông nghiệp thông minh, việc ứng dụng trí tuệ nhân tạo và thị giác máy tính vào giám sát, đánh giá chất lượng nông sản ngày càng trở nên cần thiết. Đối với cây mít – một loại nông sản có giá trị kinh tế cao tại Việt Nam – việc xác định độ chín và phát hiện trái bị sâu bệnh hiện nay vẫn chủ yếu dựa vào kinh nghiệm thủ công, dễ gây sai lệch, làm giảm chất lượng và hiệu quả thu hoạch.

Từ thực tế đó, đề tài “Nghiên cứu và ứng dụng YOLOv8 trong nhận dạng và phân loại độ chín trái mít phục vụ nông nghiệp thông minh” được thực hiện với mục tiêu xây dựng một hệ thống thị giác máy tính có khả năng phát hiện vị trí trái mít và phân loại mức độ chín/tình trạng sâu bệnh từ ảnh trong điều kiện thực tế. Kết quả của đề tài hướng đến việc hỗ trợ người nông dân và các đơn vị sản xuất nông nghiệp trong việc ra quyết định thu hoạch, phân loại và quản lý chất lượng trái mít một cách khách quan, nhanh chóng và hiệu quả hơn.

Trong quá trình thực hiện đồ án, nhóm đã nhận được sự hướng dẫn tận tình của giảng viên hướng dẫn thầy TS. Nguyễn Hà Huy Cường đã hướng dẫn tận tình nhóm chúng em, cùng sự hỗ trợ của các thầy cô trong Khoa Công nghệ Thông tin, Trường Đại Học Kiến trúc Đà Nẵng. Nhóm chúng em xin chân thành cảm ơn thầy và quý thầy cô đã tạo điều kiện thuận lợi, góp ý chuyên môn và định hướng để nhóm có thể hoàn thành đề tài này.

Chúng em xin chân thành cảm ơn !

## **CAM ĐOAN**

Chúng em xin cam đoan rằng đồ án tốt nghiệp với đề tài: “Nghiên cứu và ứng dụng YOLOv8 trong nhận dạng và phân loại độ chín trái mít phục vụ nông nghiệp thông minh”. Là kết quả nghiên cứu, tìm hiểu và thực hiện của chính bản thân chúng em dưới sự hướng dẫn của thầy TS. Nguyễn Hà Huy Cường.

Trong quá trình thực hiện đồ án về các số liệu, hình ảnh, mã nguồn, kết quả thực nghiệm và nội dung trình bày trong báo cáo được thu thập, xử lý và trình bày một cách trung thực, rõ ràng và minh bạch. Những tài liệu, công trình nghiên cứu của các tác giả khác nếu có sử dụng trong báo cáo đều được trích dẫn rõ ràng nguồn gốc theo đúng quy định.

Đề tài này chưa từng được dùng để bảo vệ hoặc xét tốt nghiệp cho bất kỳ chương trình đào tạo nào trước đây. Chúng em hoàn toàn chịu trách nhiệm trước nhà trường về tính trung thực và nguyên bản của toàn bộ nội dung đồ án.

Sinh viên thực hiện

# MỤC LỤC

|                                                                          |           |
|--------------------------------------------------------------------------|-----------|
| Tóm tắt                                                                  |           |
| Lời nói đầu .....                                                        | i         |
| Cam đoan .....                                                           | ii        |
| Mục lục .....                                                            | iii       |
| Danh mục bảng biểu, hình vẽ.....                                         | vi        |
| Danh sách các ký hiệu, chữ viết tắt .....                                | x         |
| <b>MỞ ĐẦU .....</b>                                                      | <b>1</b>  |
| <b>Chương 1. CƠ SỞ LÝ THUYẾT VÀ CÔNG CỤ XÂY DỰNG.....</b>                | <b>5</b>  |
| <b>1.1. Cơ sở lý thuyết.....</b>                                         | <b>5</b>  |
| 1.1.1. Ngôn ngữ lập trình Python .....                                   | 5         |
| 1.1.2. Framework học sâu và mô hình YOLOv8 .....                         | 6         |
| 1.1.3. Thư viện xử lý dữ liệu và ảnh: NumPy, Pandas, OpenCV .....        | 8         |
| 1.1.4. Nền tảng gắn nhãn và quản lý dữ liệu Roboflow .....               | 9         |
| 1.1.5. Nền Tảng Teachable Machine cho mô hình trên ứng dụng di động..... | 10        |
| <b>1.2. Công cụ xây dựng hệ thống .....</b>                              | <b>11</b> |
| 1.2.1. Môi trường thực thi và phát triển VS Code và môi trường ảo. ....  | 11        |
| 1.2.2. Khởi tạo Frontend và Backend cho dự án .....                      | 13        |
| 1.2.3. Môi trường huấn luyện: Google Colab .....                         | 15        |
| 1.2.4. Cơ sở dữ liệu MongoDB Atlas .....                                 | 20        |
| 1.2.5. Ngôn ngữ Dart và Framework Flutter cho ứng dụng di động.....      | 22        |
| 1.2.6. Nền tảng Supabase backend cho ứng dụng di động .....              | 23        |
| 1.2.7. Công cụ hỗ trợ làm việc nhóm và quản lý mã nguồn .....            | 24        |
| <b>Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG.....</b>                     | <b>27</b> |
| <b>2.1. Khảo sát thực tế và xác định yêu cầu .....</b>                   | <b>27</b> |

|                                                                            |           |
|----------------------------------------------------------------------------|-----------|
| 2.1.1. Khảo sát thực tế.....                                               | 27        |
| 2.1.2. Mục tiêu hệ thống.....                                              | 28        |
| 2.1.3. Xác định yêu cầu.....                                               | 29        |
| <b>2.2. Mô hình Use Case tổng quát và chi tiết.....</b>                    | <b>33</b> |
| 2.2.1. Xác định các actor trong hệ thống .....                             | 33        |
| 2.2.2. Sơ đồ Use Case tổng quát của hệ thống .....                         | 33        |
| 2.2.3. Mô tả các chức năng (use case) của hệ thống.....                    | 35        |
| 2.2.4. Use Case chi tiết.....                                              | 37        |
| 2.2.5. Sơ đồ ngữ cảnh hệ thống (Context Diagram) .....                     | 45        |
| 2.2.6. Sơ đồ thực thể (ERD) .....                                          | 47        |
| 2.2.7. Sơ đồ tuần tự .....                                                 | 49        |
| <b>2.3. Phân tích và thiết kế ứng dụng di động AgriVision Mobile .....</b> | <b>55</b> |
| 2.3.1. Sơ đồ use case tổng quát cho ứng dụng di động.....                  | 55        |
| 2.3.2. Sơ đồ tuần tự chức năng chụp ảnh phân tích.....                     | 57        |
| 2.3.3. Sơ đồ tuần tự chức năng đăng ký tài khoản.....                      | 58        |
| 2.3.4. Sơ đồ tuần tự chức năng đăng nhập tài khoản.....                    | 59        |
| 2.3.5. Sơ đồ hoạt động chức năng nhận dạng qua Camera .....                | 59        |
| <b>Chương 3. TRIỂN KHAI HỆ THỐNG VÀ KẾT QUẢ ĐẠT ĐƯỢC .....</b>             | <b>61</b> |
| <b>3.1. Cấu hình hệ thống .....</b>                                        | <b>61</b> |
| 3.1.1. Môi trường phát triển cục bộ .....                                  | 61        |
| 3.1.2. Môi trường huấn luyện mô hình .....                                 | 61        |
| 3.1.3. Cấu hình cơ sở dữ liệu.....                                         | 62        |
| <b>3.2. Quy trình cài đặt và chạy hệ thống .....</b>                       | <b>63</b> |
| 3.2.1. Cấu trúc thư mục chương trình .....                                 | 63        |
| 3.2.2. Cài đặt môi trường và các thư viện .....                            | 63        |
| 3.2.3. Cấu hình kết nối cơ sở dữ liệu và tham số hệ thống .....            | 64        |
| 3.2.4. Khởi tạo Backend.....                                               | 64        |

|                                                                               |           |
|-------------------------------------------------------------------------------|-----------|
| 3.2.5. Khởi chạy frontend.....                                                | 65        |
| <b>3.3. Kết quả đạt được của website.....</b>                                 | <b>65</b> |
| 3.3.1. Giao diện trang chủ của Website .....                                  | 65        |
| 3.3.2. Giao diện đăng ký và đăng nhập .....                                   | 66        |
| 3.3.3. Giao diện phân tích ảnh và kết quả phân tích.....                      | 67        |
| 3.3.4. Giao diện phân tích video và kết quả phân tích.....                    | 68        |
| 3.3.5. Giao diện webcam và kết quả nhận dạng webcam .....                     | 69        |
| 3.3.6. Giao diện chức năng thống kê và phân tích số liệu .....                | 69        |
| 3.3.7. Giao diện chức năng so sánh mô hình.....                               | 70        |
| 3.3.8. Giao diện AI hỗ trợ Chatbot .....                                      | 71        |
| 3.3.9. Giao diện thông tin người dùng .....                                   | 72        |
| <b>3.4. Quy trình cài đặt và chạy ứng dụng AgriVision Mobile .....</b>        | <b>72</b> |
| 3.4.1. Chuẩn bị môi trường phát triển.....                                    | 72        |
| 3.4.2. Cấu trúc thư mục dự án .....                                           | 73        |
| 3.4.3. Chạy ứng dụng trên thiết bị giả lập (Android Emulator) .....           | 73        |
| <b>3.5. Kết quả đạt được của App Mobile .....</b>                             | <b>75</b> |
| 3.5.1. Giao diện đăng ký, đăng nhập và trang chủ .....                        | 75        |
| 3.5.2. Giao diện chức năng Phân tích ảnh, Camera, Video .....                 | 75        |
| 3.5.3. Giao diện chức năng Webcam, Chatbot AI và so sánh mô hình YOLOv8 ..... | 76        |
| 3.5.4. Giao diện chức năng thống kê dữ liệu và thông tin tài khoản.....       | 76        |
| 3.5.5. Giao diện phần cài đặt của App Mobile.....                             | 77        |
| 3.5.6. Giao diện phần giới thiệu và đổi mật khẩu.....                         | 77        |
| <b>KẾT LUẬN .....</b>                                                         | <b>78</b> |
| <b>TÀI LIỆU THAM KHẢO .....</b>                                               | <b>79</b> |

## DANH MỤC BẢNG BIỂU, HÌNH VẼ

|                                                                                    |    |
|------------------------------------------------------------------------------------|----|
| Bảng 2.1. Mô tả các Actor trong hệ thống.....                                      | 33 |
| Bảng 2.2. Mô tả chức năng của người dùng .....                                     | 35 |
| Bảng 2.3. Mô tả các chức năng của Admin .....                                      | 36 |
| Bảng 2.4. Use Case chi tiết chức năng đăng ký .....                                | 37 |
| Bảng 2.5. Use case chi tiết chức năng đăng nhập .....                              | 39 |
| Bảng 2.6. Use case chi tiết chức năng tải ảnh và nhận dạng .....                   | 41 |
| Bảng 2.7. Use case chi tiết chức năng so sánh 2 mô hình YOLOv8 .....               | 43 |
| Bảng 2.8. Mô tả các chức năng của sơ đồ ngữ cảnh .....                             | 45 |
|                                                                                    |    |
| Hình 1.1. Hình ảnh về ngôn ngữ Python .....                                        | 5  |
| Hình 1.2. Hình ảnh hiệu suất YOLOv8 với các phiên bản trước.....                   | 7  |
| Hình 1.3. Hình ảnh kiến trúc của YOLOv8.....                                       | 7  |
| Hình 1.4. Hình ảnh trái mít được vẽ bounding box nhãn bằng OpenCV.....             | 9  |
| Hình 1.5. Hình ảnh Logo Roboflow .....                                             | 9  |
| Hình 1.6. Hình ảnh giao diện Roboflow đang gắn nhãn bounding box cho trái mít .... | 10 |
| Hình 1.7. Teachable Machine đã train xong dữ liệu .....                            | 11 |
| Hình 1.8. Hình ảnh về Visual Studio Code.....                                      | 12 |
| Hình 1.9. Hình ảnh tạo môi trường ảo cho dự án.....                                | 12 |
| Hình 1.10. Hình ảnh đã kích hoạt môi trường ảo.....                                | 13 |
| Hình 1.11. Hình ảnh cài đặt các thư viện sử dụng trong đề tài.....                 | 13 |
| Hình 1.12. Hình ảnh lưu các thư viện vào file requirement.txt .....                | 13 |
| Hình 1.13. Hình ảnh khởi động backend bằng lệnh.....                               | 14 |
| Hình 1.14. Hình ảnh khởi động Frontend bằng lệnh.....                              | 14 |
| Hình 1.15. Hình ảnh về Google Colab .....                                          | 15 |
| Hình 1.16. Cài đặt thư viện với các ô lệnh trong Google Colab.....                 | 16 |

|                                                                                   |    |
|-----------------------------------------------------------------------------------|----|
| Hình 1.17. Hình ảnh nhập dữ liệu ở Google Colab.....                              | 17 |
| Hình 1.18. Hình ảnh thiết lập mô hình YOLOv8 trên Google Colab .....              | 17 |
| Hình 1.19. Hình ảnh đào tạo mô hình YOLOv8 trong Google Colab .....               | 18 |
| Hình 1.20. Hình ảnh biểu đồ mô hình YOLOv8 sau huấn luyện.....                    | 19 |
| Hình 1.21. Hình ảnh ma trận Confusion Matrix của mô hình YOLOv8.....              | 19 |
| Hình 1.22. Hình ảnh kết quả đầu ra mô hình YOLOv8 với ảnh nhận dạng trái mít..... | 20 |
| Hình 1.23. Hình ảnh về MongoDB Atlas .....                                        | 20 |
| Hình 1.24. Ngôn ngữ lập trình Dart.....                                           | 23 |
| Hình 1.25. Nền tảng Framework Flutter.....                                        | 23 |
| Hình 1.26. Nền tảng Supabase backend Service.....                                 | 24 |
| Hình 1.27. Hình ảnh làm việc nhóm trên Trello .....                               | 25 |
| Hình 1.28. Repository GitHub để quản lý mã nguồn và phân chia công việc nhóm....  | 26 |
| Hình 1.29. Lịch sử commit trên Github phía bên web .....                          | 26 |
| Hình 1.30. Lịch sử commit trên Github phía bên App Mobile.....                    | 26 |
| Hình 2.1. Hình ảnh người dân đánh giá mít dựa vào kinh nghiệm nuôi trồng .....    | 27 |
| Hình 2.2. Hình ảnh về sơ đồ Use Case tổng quát.....                               | 34 |
| Hình 2.3. Sơ đồ Use Case đăng ký chi tiết .....                                   | 37 |
| Hình 2.4. Sơ đồ use case đăng nhập chi tiết .....                                 | 39 |
| Hình 2.5. Use case tải ảnh và nhận dạng .....                                     | 41 |
| Hình 2.6. Use Case chi tiết so sánh hai mô hình YOLOv8 .....                      | 43 |
| Hình 2.7. Sơ đồ ngữ cảnh hệ thống .....                                           | 45 |
| Hình 2.8. Sơ đồ thực thể ERD.....                                                 | 47 |
| Hình 2.9. Sơ đồ tuần tự chức năng tải ảnh trái mít và nhận dạng.....              | 49 |
| Hình 2.10. Sơ đồ tuần tự chức năng đăng nhập .....                                | 51 |
| Hình 2.11. Sơ đồ tuần tự chức năng so sánh hai mô hình .....                      | 53 |
| Hình 2.12. Sơ đồ Use case tổng quát ứng dụng di động Mobile App.....              | 55 |
| Hình 2.13. Sơ đồ tuần tự chức năng chụp ảnh nhận dạng trái mít.....               | 57 |

|                                                                                  |    |
|----------------------------------------------------------------------------------|----|
| Hình 2.14. Sơ đồ tuần tự chức năng đăng ký của App Mobile.....                   | 58 |
| Hình 2.15. Sơ đồ tuần tự chức năng đăng nhập tài khoản của App Mobile.....       | 59 |
| Hình 2.16. Sơ đồ hoạt động chức năng nhận dạng qua Camera trên App Mobile .....  | 60 |
| Hình 3.1. Hình ảnh Google Colab khi đã train xong mô hình YOLOv8 .....           | 62 |
| Hình 3.2. Chuỗi mã connection string của MongoDB Atlas .....                     | 62 |
| Hình 3.3. Các collection chính của hệ thống .....                                | 63 |
| Hình 3.4. Cấu trúc thư mục chương trình .....                                    | 63 |
| Hình 3.5. Hình ảnh đoạn mã đọc biến môi trường khởi tạo kết nối MongoDB .....    | 64 |
| Hình 3.6. Giao diện trang chủ chính của website .....                            | 65 |
| Hình 3.7. Giao diện đăng ký của website .....                                    | 66 |
| Hình 3.8. Giao diện đăng nhập của website.....                                   | 66 |
| Hình 3.9. Giao diện phân tích ảnh của website.....                               | 67 |
| Hình 3.10. Kết quả nhận diện trái mít và phân tích của AI .....                  | 67 |
| Hình 3.11. Giao diện phân tích video của website.....                            | 68 |
| Hình 3.12. Kết quả phân tích video và phân tích AI của video .....               | 68 |
| Hình 3.13. Giao diện webcam và kết quả nhận diện của webcam.....                 | 69 |
| Hình 3.14. Kết quả thống kê và phân tích số liệu .....                           | 69 |
| Hình 3.15. Giao diện chức năng so sánh hai mô hình.....                          | 70 |
| Hình 3.16. Kết quả so sánh và phân tích dữ liệu của hai mô hình YOLOv8 .....     | 70 |
| Hình 3.17. Xuất File PDF phân tích từ hệ thống .....                             | 71 |
| Hình 3.18. Giao diện Chatbot hỗ trợ AgriVision.....                              | 71 |
| Hình 3.19. Kết quả trả lời của Chatbot.....                                      | 72 |
| Hình 3.20. Giao diện thông tin người dùng .....                                  | 72 |
| Hình 3.21. Cấu trúc thư mục của dự án App Mobile .....                           | 73 |
| Hình 3.22. Khởi động thiết bị giả lập thành công .....                           | 74 |
| Hình 3.23. Giao diện trang đăng ký, đăng nhập và trang chủ của App Mobile .....  | 75 |
| Hình 3.24. Giao diện chức năng phân tích ảnh camera và video của App Mobile..... | 75 |

|                                                                                     |    |
|-------------------------------------------------------------------------------------|----|
| Hình 3.25. Giao diện chức năng webcam, chatbot AI và so sánh mô hình.....           | 76 |
| Hình 3.26. Giao diện chức năng thống kê dữ liệu và thông tin tài khoản người dùng . | 76 |
| Hình 3.27. Giao diện phần cài đặt và mã QR tải ứng dụng.....                        | 77 |
| Hình 3.28. Giao diện giới thiệu và đổi mật khẩu .....                               | 77 |

## DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

| STT | Ký tự viết tắt | Tên đầy đủ                          |
|-----|----------------|-------------------------------------|
| 1   | AI             | Artificial Intelligence             |
| 2   | ML             | Machine Learning                    |
| 3   | CNN            | Convolutional Neural Network        |
| 4   | YOLO           | You Only Look Once                  |
| 5   | mAP            | mean Average Precision              |
| 6   | IoU            | Intersection over Union             |
| 7   | GPU            | Graphics Processing Unit            |
| 8   | API            | Application Programming Interface   |
| 9   | REST API       | Representational State Transfer API |
| 10  | UI             | User Interface                      |
| 11  | IDE            | Integrated Development Environment  |
| 12  | CSDL           | Cơ sở dữ liệu                       |
| 13  | VS Code        | Visual Studio Code                  |
| 14  | JSON           | JavaScript Object Notation          |
| 15  | URL            | Uniform Resource Locator            |
| 16  | SDK            | Software Development Kit            |
| 17  | IoT            | Internet of Things                  |

## MỞ ĐẦU

### 1. Mục đích thực hiện đề tài

Trong những năm gần đây, các công nghệ như trí tuệ nhân tạo (AI), học sâu (Deep Learning) và thị giác máy tính (Computer Vision) phát triển rất nhanh và dần xuất hiện ngày càng nhiều trong đời sống. Đối với sinh viên công nghệ thông tin, việc chỉ dừng lại ở mức tìm hiểu lý thuyết trên lớp là chưa đủ, mà cần có những đề tài cụ thể để vận dụng kiến thức đã học vào một bài toán thực tế, qua đó rèn luyện tư duy, kỹ năng tự nghiên cứu và khả năng làm việc nhóm.

Với bối cảnh đó, nhóm chúng em mong muốn thực hiện một đề tài vừa có yếu tố kỹ thuật, vừa gắn với đời sống sản xuất ở Việt Nam. Cây mít là một loại nông sản quen thuộc, có giá trị kinh tế khá cao và được trồng nhiều ở các địa phương. Tuy nhiên, việc đánh giá độ chín và tình trạng sâu bệnh của trái mít hiện nay phần lớn vẫn dựa vào kinh nghiệm và cảm quan của người trồng, dễ dẫn tới thu hoạch không đúng thời điểm hoặc phân loại chưa chính xác, ảnh hưởng trực tiếp đến chất lượng và giá bán.

Mục đích chính của đề tài là nghiên cứu, xây dựng và thử nghiệm một hệ thống có khả năng hỗ trợ người dùng nhận dạng và phân loại độ chín trái mít thông qua hình ảnh, dựa trên mô hình YOLOv8. Thông qua đó, chúng em mong muốn góp phần chứng minh rằng các mô hình học sâu hiện đại không chỉ mang tính lý thuyết mà hoàn toàn có thể áp dụng cho những bài toán gần gũi trong nông nghiệp, từ đó mở ra thêm những hướng ứng dụng mới trong tương lai.

Ngoài ra, việc thực hiện đề tài còn giúp nhóm củng cố lại kiến thức về học sâu, lập trình Python, xử lý ảnh, đồng thời làm quen với quy trình xây dựng một hệ thống tương đối hoàn chỉnh: từ khâu thu thập dữ liệu, huấn luyện mô hình cho đến xây dựng giao diện ứng dụng và đánh giá kết quả.

### 2. Mục tiêu đề tài

Từ mục đích trên, đề tài được định hướng với mục tiêu tổng quát là xây dựng một hệ thống thị giác máy tính có khả năng phát hiện và phân loại trái mít theo độ chín hoặc tình trạng bề mặt dựa trên mô hình YOLOv8, phục vụ nhu cầu giám sát và hỗ trợ ra quyết định trong bối cảnh nông nghiệp thông minh.

Cụ thể hơn, đề tài đặt ra một số mục tiêu sau:

- Xây dựng và chuẩn hóa một bộ dữ liệu hình ảnh trái mít với số lượng và chất lượng phù hợp cho việc huấn luyện mô hình, đảm bảo có sự đa dạng về bối cảnh, góc chụp, điều kiện ánh sáng và trạng thái trái mít.
- Nghiên cứu và áp dụng mô hình YOLOv8 cho bài toán phát hiện và phân loại trái mít, tiến hành huấn luyện và tinh chỉnh các tham số để đạt được độ chính xác ở mức chấp nhận được trong điều kiện dữ liệu thực nghiệm.
- Xây dựng một ứng dụng demo (trên nền web và/hoặc thiết bị di động) cho phép người dùng tải ảnh hoặc sử dụng webcam để hệ thống tự động nhận dạng vị trí trái mít, phân loại độ chín hoặc tình trạng bề mặt, và hiển thị kết quả một cách trực quan.
- Đánh giá chất lượng mô hình dựa trên các chỉ số phổ biến trong lĩnh vực thị giác máy tính như Precision, Recall, F1-score, mAP,... đồng thời thử nghiệm hệ thống trong nhiều trường hợp khác nhau để nhận xét khả năng ứng dụng trong thực tế.

Thông qua việc hoàn thành các mục tiêu trên, đề tài không chỉ dừng lại ở mức xây dựng một mô hình nhận dạng đơn lẻ mà còn hướng tới một hệ thống tổng thể có thể tiếp tục phát triển và cải tiến trong tương lai.

### **3. Phạm vi và đối tượng nghiên cứu**

Để đảm bảo tính khả thi trong phạm vi một đồ án tốt nghiệp, đề tài được giới hạn rõ về đối tượng và phạm vi nghiên cứu.

Về đối tượng nghiên cứu, đề tài tập trung vào:

- Mô hình học sâu YOLOv8, với vai trò là lõi chính cho bài toán phát hiện và phân loại đối tượng trên ảnh.
- Dữ liệu hình ảnh trái mít dưới dạng ảnh màu (RGB), có chứa một hoặc nhiều trái mít trong khung hình, được chụp trong các bối cảnh khác nhau như vườn cây, chợ, khu sơ chế,... với độ chín và tình trạng bề mặt khác nhau (ví dụ: mít non, mít chín, mít bị sâu bệnh).
- Hệ thống ứng dụng demo cho phép người dùng tương tác với mô hình thông qua giao diện web hoặc ứng dụng di động, từ đó kiểm tra khả năng nhận dạng và phân loại trái mít trong những tình huống gần với thực tế.

Về phạm vi nghiên cứu, đề tài chỉ xem xét bài toán phát hiện và phân loại trái mít dựa trên hình ảnh 2D của trái còn nguyên vẹn. Hệ thống đánh giá dựa trên đặc trưng bề mặt vỏ trái mít, không xét đến các yếu tố bên trong như độ ngọt, cấu trúc múi hay mùi

hương. Các trường hợp ảnh quá mờ, bị che khuất phần lớn hoặc chụp trong điều kiện ánh sáng quá cực đoan không phải là trọng tâm xử lý của đề tài mà chủ yếu được xem như các tình huống khó, dùng để tham khảo trong phần đánh giá.

Bên cạnh đó, việc triển khai hệ thống cũng được giới hạn trong môi trường máy tính cá nhân và thiết bị di động phổ biến. Đề tài chưa đi sâu vào việc tối ưu mô hình cho các nền tảng nhúng, cảm biến IoT hay tích hợp với những hệ thống quản lý nông nghiệp quy mô lớn, mà chủ yếu dừng lại ở mức demo minh họa tính khả thi và tiềm năng ứng dụng.

#### **4. Phương pháp nghiên cứu**

Để thực hiện đề tài một cách có hệ thống, chúng em kết hợp nhiều phương pháp nghiên cứu khác nhau, được triển khai theo từng giai đoạn.

Trước hết, nhóm sử dụng phương pháp nghiên cứu tài liệu. Chúng em tìm hiểu các khái niệm và kiến thức nền tảng liên quan đến trí tuệ nhân tạo, học sâu, mạng nơ-ron tích chập, thị giác máy tính cũng như các mô hình phát hiện đối tượng, trong đó tập trung vào dòng YOLO và phiên bản YOLOv8. Ngoài các tài liệu lý thuyết, chúng em còn tham khảo thêm những bài báo, blog kỹ thuật, tài liệu hướng dẫn sử dụng thư viện Ultralytics YOLOv8, PyTorch, các công cụ gắn nhãn dữ liệu, cũng như một số nghiên cứu ứng dụng thị giác máy tính trong nông nghiệp để có cái nhìn thực tế hơn.

Tiếp theo, chúng em áp dụng phương pháp phân tích và mô hình hóa hệ thống. Từ bài toán thực tế là nhận dạng và phân loại độ chín trái mít, nhóm xác định các yêu cầu chính của hệ thống, bao gồm: nguồn dữ liệu như thế nào, đầu vào – đầu ra của mô hình, các chức năng người dùng cần, luồng xử lý của hệ thống (từ ảnh đầu vào → tiền xử lý → mô hình suy luận → hiển thị kết quả). Việc phân tích này giúp chúng em xây dựng được kiến trúc tổng quan, phân chia rõ vai trò của từng thành phần như mô hình YOLOv8, ứng dụng web/di động, cơ sở dữ liệu,...

Sau đó, đề tài sử dụng phương pháp thiết kế và thực nghiệm. Ở bước này, chúng em tiến hành:

- Thu thập, lựa chọn và gắn nhãn dữ liệu ảnh trái mít theo các lớp đã xác định.
- Thiết lập cấu trúc dữ liệu theo chuẩn YOLOv8, lựa chọn biến thể mô hình phù hợp và điều chỉnh các siêu tham số huấn luyện.
- Huấn luyện và tinh chỉnh mô hình trên tập dữ liệu đã chuẩn bị, đồng thời theo dõi các chỉ số đánh giá để có những điều chỉnh cần thiết.

Song song với việc huấn luyện mô hình, chúng em xây dựng một ứng dụng demo (web và/hoặc di động) để tích hợp mô hình đã huấn luyện, cho phép người dùng tải ảnh hoặc sử dụng webcam, sau đó hệ thống sẽ tự động nhận dạng và hiển thị kết quả nhận diện trái mít trên giao diện.

Cuối cùng, chúng em áp dụng phương pháp kiểm thử và đánh giá thực nghiệm. Về phía mô hình, chúng em sử dụng tập dữ liệu kiểm tra để tính toán các chỉ số như Precision, Recall, F1-score, mAP,... cũng như quan sát các trường hợp dự đoán sai, nhằm lẫn giữa các lớp để rút ra nhận xét. Về phía ứng dụng, nhóm thử nghiệm hệ thống trên nhiều loại ảnh khác nhau về góc chụp, ánh sáng và bối cảnh, ghi nhận những lỗi phát sinh (nếu có) và điều chỉnh dần để hệ thống hoạt động ổn định hơn. Kết quả đánh giá là cơ sở để chúng em kết luận về mức độ đáp ứng của mô hình YOLOv8 đối với bài toán đề ra và đề xuất những hướng phát triển tiếp theo.

## **5. Cấu trúc của đồ án tốt nghiệp**

Đồ án tốt nghiệp được trình bày gồm các phần chính sau:

- **Mở đầu:** Trình bày mục đích thực hiện đề tài, mục tiêu nghiên cứu, đối tượng và phạm vi nghiên cứu, phương pháp nghiên cứu và cấu trúc của đồ án.
- **Chương 1:** Cơ sở lý thuyết và các công nghệ, công cụ sử dụng trong đề tài.
- **Chương 2:** Phân tích và thiết kế hệ thống nhận dạng và phân loại độ chín trái mít.
- **Chương 3:** Triển khai hệ thống, kết quả thực nghiệm và đánh giá.
- **Kết luận:** Tổng kết kết quả đạt được, nêu hạn chế và đề xuất hướng phát triển trong tương lai.

## Chương 1. CƠ SỞ LÝ THUYẾT VÀ CÔNG CỤ XÂY DỰNG

### 1.1. Cơ sở lý thuyết

#### 1.1.1. Ngôn ngữ lập trình Python

##### 1.1.1.1. Khái niệm về 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). Python được Guido van Rossum giới thiệu vào năm 1991 và đã trải qua 3 giai đoạn phát triển khác nhau tương ứng với các version, mới nhất hiện nay là Python version 3x (3.12.3 vào 9 tháng 4 2024). Python có cú pháp rõ ràng và ngắn gọn, giúp cho việc học và sử dụng ngôn ngữ này trở nên dễ dàng.



Hình 1.1. Hình ảnh về ngôn ngữ Python

Có thể thấy Python là ngôn ngữ lập trình được sử dụng nhiều nhất trên thế giới hiện nay, vì Python hỗ trợ rất nhiều phong cách lập trình như (hướng đối tượng, hàm, thủ tục) và có hệ sinh thái thư viện rất phong phú, đặc biệt mạnh trong các lĩnh vực:

- Khoa học dữ liệu, phân tích dữ liệu.
- Trí tuệ nhân tạo (AI), học máy (Machine learning), học sâu (deep learning)
- Phát triển web, tự động hoá, xử lý ảnh...

Ta có thể thấy, Python là ngôn ngữ lập trình được sử dụng nhiều nhất thế giới hiện nay. Và các lập trình viên lại lựa chọn ngôn ngữ này, Trong các dự án AI hiện nay, Python gần như là ngôn ngữ chuẩn vì:

- Cú pháp đơn giản dễ đọc.
- Đa dụng

- Sở hữu nhiều thư viện phong phú
- Cộng đồng lớn mạnh và đa nền tảng

#### *1.1.1.2. Ưu điểm và nhược điểm của Python*

- Ưu điểm: Cú pháp dễ hiểu, giúp rút ngắn thời gian phát triển, thư viện phong phú, phù hợp cho AI/ML, dễ thử nghiệm và debug chia sẻ mã nguồn.
- Nhược điểm: Tốc độ thực thi của Python chậm hơn với ngôn C/C++, Tiêu thụ bộ nhớ lớn, và không phù hợp với các ứng dụng đòi hỏi hiệu suất cao.

### **1.1.2. Framework học sâu và mô hình YOLOv8**

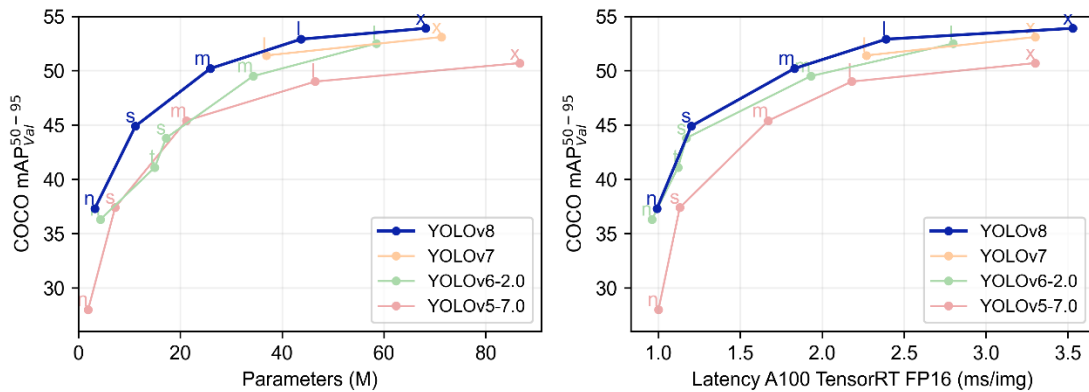
#### *1.1.2.1. PyTorch là gì ?*

PyTorch là framework học sâu mã nguồn mở do Meta (Facebook) phát triển, gồm một số đặc điểm chính như. Cung cấp kiểu dữ liệu Tensor, tương tự mảng nhiều chiều, hỗ trợ tính toán trên CPU và GPU, cơ chế autograd cho phép tự động tính đạo hàm, thuận tiện cho việc tối ưu mô hình, API trực quan, gần với cách viết Python thuần, dễ thử nghiệm với mô hình mới. Trong đề tài này, PyTorch đóng vai trò nền tảng mà trên đó Ultralytics YOLOv8 được cài đặt và vận hành.

#### *1.1.2.2. Giới thiệu về mô hình YOLOv8*

YOLO (You Only Look Once) là một trong những mô hình học sâu nổi bật và phổ biến nhất trong lĩnh vực phát hiện đối tượng (Object Detection). Không giống như các mô hình truyền thống như R-CNN hay Faster R-CNN vốn yêu cầu nhiều bước xử lý trung gian, YOLO được thiết kế để nhận dạng đối tượng trong một lần duy nhất, giúp rút ngắn thời gian xử lý và phù hợp với các ứng dụng yêu cầu tốc độ cao.

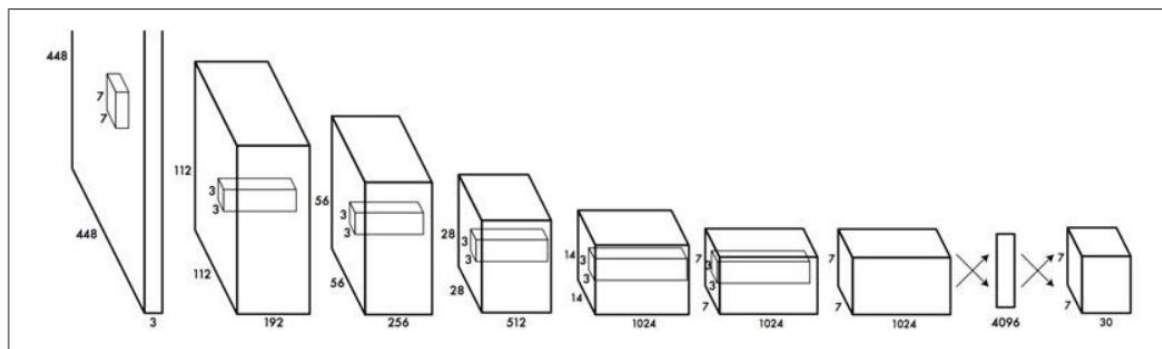
YOLOv8 là phiên bản mới nhất được phát hành bởi Ultralytics vào đầu năm 2023, được cải tiến mạnh mẽ so với các phiên bản tiền nhiệm như YOLOv5, YOLOv6, YOLOv7. Mô hình này kế thừa những ưu điểm vượt trội về tốc độ và độ chính xác, đồng thời cải tiến cấu trúc bên trong để hỗ trợ tốt hơn cho các bài toán phát hiện đối tượng, phân vùng ảnh (segmentation), phân loại ảnh (classification), và thậm chí là định vị tư thế (pose estimation).



Hình 1.2. Hình ảnh hiệu suất YOLOv8 với các phiên bản trước

Khác với các mô hình trước đây như R-CNN hay SSD cần nhiều bước để nhận diện đối tượng, YOLO hoạt động theo nguyên lý “một lần nhìn duy nhất:”, nghĩa là chỉ cần một lần truyền ảnh qua mô hình để dự đoán được các bounding box và lớp tương ứng. Điều này giúp tăng đáng kể tốc độ xử lý, rất phù hợp với các ứng dụng thời gian thực.

#### ❖ Cấu trúc tổng quan của YOLOv8



Hình 1.3. Hình ảnh kiến trúc của YOLOv8

#### ❖ YOLOv8 bao gồm ba thành phần chính trong kiến trúc:

- **Backbone:** Chịu trách nhiệm trích xuất đặc trưng từ ảnh đầu vào. Trong YOLOv8, backbone sử dụng kiến trúc CSPDarknet hoặc một số biến thể nhẹ hơn, giúp tăng khả năng học đặc trưng mà không tăng nhiều tham số.
- **Neck:** Là cầu nối giữa backbone và head, dùng để tổng hợp các đặc trưng ở nhiều cấp độ khác nhau. Neck thường sử dụng FPN (Feature Pyramid Network) hoặc PANet nhằm tăng khả năng nhận diện đối tượng ở nhiều kích thước.
- **Head:** Đầu ra của mô hình, nơi diễn ra quá trình dự đoán các bounding box, class (nhãn đối tượng), confidence score (độ tin cậy) tương ứng.

❖ **Ngoài ra YOLOv8 còn sử dụng các kỹ thuật như:**

- Anchor-free detection: Không cần định nghĩa trước các anchor box, giúp đơn giản hoá quá trình huấn luyện.
- Auto-learning bounding box: Cải thiện độ chính xác trong việc dự đoán vị trí và kích thước vật thể.
- Loss Function cải tiến: Tối ưu hàm mất mát giúp mô hình hội tụ nhanh và chính xác hơn.

❖ **Ưu điểm nổi bật của YOLOv8:**

- Tốc độ xử lý nhanh: YOLOv8 có thể chạy real-time trên cả CPU và GPU, thích hợp cho ứng dụng yêu cầu phản hồi thời gian thực như camera giám sát hay thiết bị nhúng.
- Độ chính xác cao: Được tinh chỉnh và cải tiến thuật toán, YOLOv8 đạt điểm mAP (mean Average Precision) rất cao trên nhiều bộ dữ liệu benchmark.
- Dễ sử dụng và triển khai: Thư viện Ultralytics hỗ trợ cài đặt nhanh qua pip, có sẵn có model pre-trained, tài liệu rõ ràng và cú pháp lập trình ngắn gọn.
- Tuỳ biến tốt: Cho phép dễ dàng fine-tune trên dataset riêng hỗ trợ export mô hình sang nhiều định dạng (ONNX, TorchScript, CoreML...) để triển khai linh hoạt trên nhiều nền tảng.

### 1.1.3. Thư viện xử lý dữ liệu và ảnh: NumPy, Pandas, OpenCV

❖ **NumPy**

NumPy (Numerical Python) cung cấp kiểu dữ liệu ndarray – mảng nhiều chiều hiệu quả, cùng với hàng loạt phép toán vector hoá và ma trận.

- Là nền tảng cho nhiều thư viện khác (Pandas, scikit-learn, PyTorch..).
- Trong đề tài, NumPy dùng để thao tác trên mảng pixel, chuyển đổi dữ liệu giữa các định dạng.

❖ **Pandas**

Pandas là thư viện xử lý dữ liệu dạng bảng (DataFrame), cung cấp:

- Các phép đọc/ghi file CSV, Excel, JSON...
- Các thao tác lọc, nhóm, thống kê, tổng hợp dữ liệu.

Trong đề tài, Pandas được dùng để:

- Lưu log các lần huấn luyện (chỉ số mAP, loss theo epoch).
- Tổng hợp kết quả đánh giá theo lớp (non, chín, sâu bệnh).

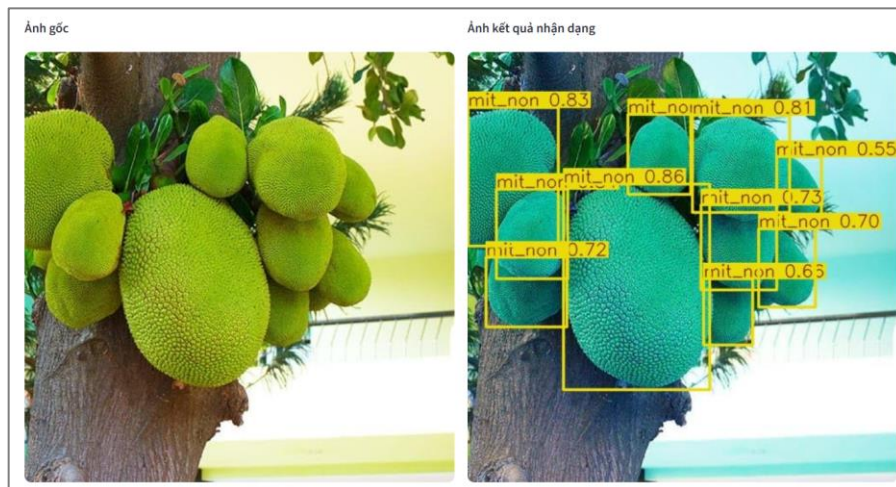
## ❖ OpenCV

OpenCV là thư viện thị giác máy tính mã nguồn mở, hỗ trợ:

- Đọc, ghi, hiển thị ảnh và video.
- Các phép biến đổi ảnh (resize, crop, blur, chuyển đổi màu..)
- Vẽ Bounding box, nhãn lên ảnh.

Trong đề tài, OpenCV dùng để:

- Hiển thị kết quả suy luận (ảnh trái mít với hộp bao + nhãn lớp).
- Thực hiện một số tiền xử lý đơn giản.



Hình 1.4. Hình ảnh trái mít được vẽ bounding box nhãn bằng OpenCV

### 1.1.4. Nền tảng gắn nhãn và quản lý dữ liệu Roboflow

#### ❖ Roboflow là gì?

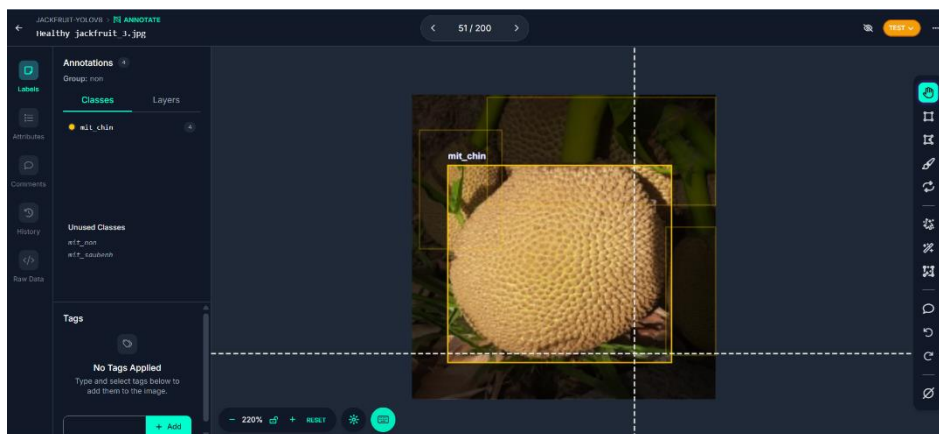
Roboflow là một nền tảng thị giác máy tính được thiết kế cho các nhà phát triển và doanh nghiệp. Nó cung cấp các công cụ để xây dựng và triển khai các mô hình thị giác máy tính, bao gồm tự động chú thích, cơ sở hạ tầng đào tạo mô hình và giải pháp triển khai. Nền tảng hỗ trợ nhiều ngành công nghiệp và trường hợp sử dụng, đồng thời cung cấp các tính năng như quản lý dữ liệu, gắn nhãn hợp tác, đánh giá mô hình và triển khai đồng cạnh. Roboflow nhằm mục đích tăng tốc lộ trình phát triển thị giác máy tính với công cụ hàng đầu và hướng dẫn chuyên môn.



Hình 1.5. Hình ảnh Logo Roboflow

### ❖ Quy trình lý thuyết khi sử dụng Roboflow

- Tạo Project cho bài toán (ví dụ: Object Detection – jackfruit).
- Tải ảnh lên Roboflow
- Gắn nhãn bounding box cho mỗi trái mít, gắn các lớp (mit\_chín, mit\_non, mit\_saubenh).
- Chọn và cấu hình các augmentation nếu cần.
- Chia train/val/test trên Roboflow.
- Export dataset sang định dạng YOLOv8 (thư mục ảnh + nhãn .txt + file data.yaml).



Hình 1.6. Hình ảnh giao diện Roboflow đang gắn nhãn bounding box cho trái mít

### ❖ Ưu điểm, nhược điểm của Roboflow

- Ưu điểm: Giao diện gắn nhãn thân thiện, hỗ trợ các công cụ gắn nhãn, dễ sử dụng, dễ quản lý nhiều phiên bản dataset, tích hợp sẵn export YOLOv8 tiết kiệm thời gian.
- Nhược điểm: bản miễn phí có giới hạn dung lượng/feature, phụ thuộc kết nối internet.

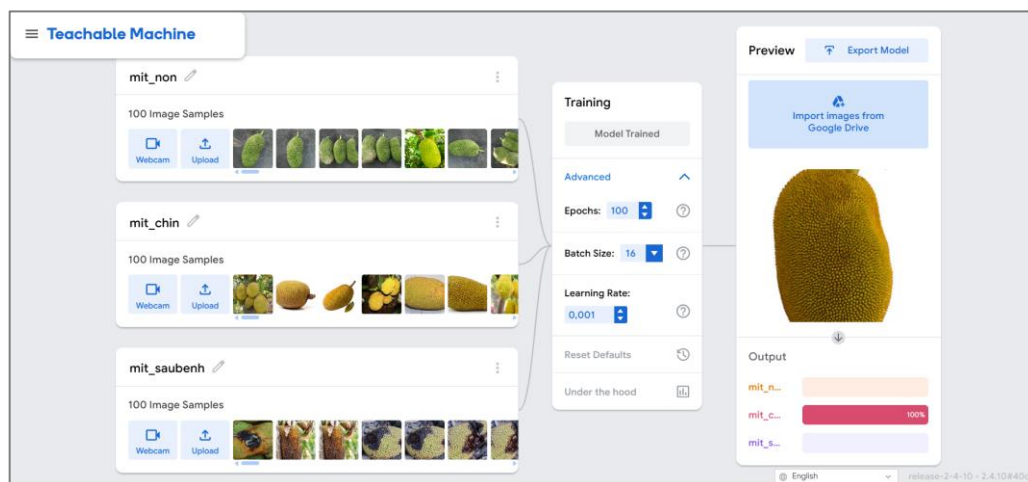
#### 1.1.5. Nền Tảng Teachable Machine cho mô hình trên ứng dụng di động

Teachable Machine là nền tảng học máy trực tuyến do Google phát triển, cho phép người dùng tạo nhanh các mô hình phân loại hình ảnh, âm thanh hoặc tư thế mà không cần nhiều kiến thức chuyên sâu về lập trình hay học sâu. Người dùng chỉ cần định nghĩa các lớp cần phân loại, tải dữ liệu đào tạo tương ứng, sau đó tiến hành huấn luyện trực tiếp trên giao diện web. Sau khi huấn luyện, Teachable Machine cho phép xuất mô hình dưới nhiều định dạng phổ biến như TensorFlow, TensorFlow Lite hoặc TensorFlow.js để tích hợp vào các ứng dụng thực tế.

Trong đề tài của chúng em, Teachable Machine được sử dụng để xây dựng một mô hình phân loại hình ảnh đơn giản phục vụ ứng dụng di động AgriVision Mobile. Các lớp phân loại được định nghĩa tương ứng với một số trạng thái của trái mít, ví dụ: mít non, mít chín, mít sâu bệnh. Dữ liệu ảnh trái mít được gắn nhãn và được chia theo từng lớp và tải lên Teachable Machine để huấn luyện mô hình classification.

❖ **Quy trình sử dụng Teachable Machine trong đề tài gồm các bước chính:**

- Chuẩn bị tập ảnh trái mít đã phân theo từng mức độ và tình trạng.
- Tạo project Image Classification trên Teachable Machine, định nghĩa các lớp tương ứng.
- Tải ảnh vào từng lớp tương ứng, cấu hình epoch, batch size cơ bản rồi tiến hành huấn luyện.
- Theo dõi độ chính xác trên giao diện.
- Xuất mô hình ở dạng phù hợp, chúng em đã chọn TensorFlow Lite để tích hợp vào ứng dụng Flutter.



Hình 1.7. Teachable Machine đã train xong dữ liệu

## 1.2. Công cụ xây dựng hệ thống

### 1.2.1. Môi trường thực thi và phát triển VS Code và môi trường ảo.

#### 1.2.1.1. Visual Studio Code ?

Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn miễn phí, đa nền tảng, được phát triển bởi Microsoft. Nó đã nhanh chóng trở thành một trong những trình soạn thảo phổ biến nhất dành cho các nhà phát triển, nhờ giao diện thân thiện, tính năng mạnh mẽ và khả năng mở rộng cao.



Hình 1.8. Hình ảnh về Visual Studio Code

#### ❖ Ưu điểm của Visual Studio Code

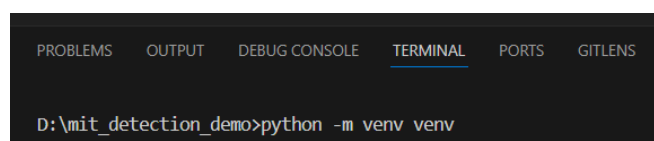
- Miễn phí và mã nguồn mở: VS Code hoàn toàn miễn phí và được cấp phép theo giấy phép MIT. Điều này nghĩa là bạn có thể sử dụng và sửa đổi nó tự do mà không phải trả bất kỳ khoản phí nào.
- Đa nền tảng: VS Code có sẵn trên Windows, macOS và Linux, cho phép bạn sử dụng nó trên bất kỳ hệ điều hành nào. Điều này mang lại sự thuận tiện cho các lập trình viên khi họ có thể làm việc trên nhiều thiết bị khác nhau mà không cần thay đổi môi trường phát triển.
- Giao diện thân thiện: VS Code có giao diện đơn giản, dễ sử dụng và trực quan, phù hợp cho cả người mới bắt đầu và các nhà phát triển có kinh nghiệm.
- Hỗ trợ nhiều ngôn ngữ lập trình: VS Code hỗ trợ hơn 30 ngôn ngữ lập trình phổ biến, bao gồm JavaScript, Python, Java, C++, C#, Golang, PHP và nhiều hơn nữa. Bạn có thể sử dụng VS Code để viết mã cho bất kỳ dự án nào, bất kể ngôn ngữ lập trình nào bạn đang sử dụng.

Bên cạnh đó để hỗ trợ quá trình phát triển và tổ chức soạn thảo code cũng như quản lý mã nguồn, thì đề tài em sử dụng môi trường phát triển là Visual Studio Code (VS Code) và môi trường ảo (virtual environment).

##### 1.2.1.2. Tạo môi trường ảo (Virtual environment)

Sau khi đã cài đặt Python và VS Code về máy, chúng ta sẽ tiến hành cài đặt và tạo môi trường ảo cho dự án. Đầu tiên mở Terminal trong VS Code (hoặc Command Prompt / PowerShell), di chuyển đến thư mục chứa dự án và tạo môi trường ảo.

Bước 1: Tạo môi trường ảo cho dự án.



Hình 1.9. Hình ảnh tạo môi trường ảo cho dự án

Lệnh trên sẽ tạo một thư mục con tên là venv chứa toàn bộ nội dung Python riêng cho dự án.

Bước 2: kích hoạt môi trường ảo (trên Windows):

```
D:\mit_detection_demo>python -m venv venv  
D:\mit_detection_demo>venv\Scripts\activate  
(venv) D:\mit_detection_demo>
```

Hình 1.10. Hình ảnh đã kích hoạt môi trường ảo

Sau khi kích hoạt thành công, tên môi trường là (venv) sẽ xuất hiện ở đầu dòng lệnh, thể hiện rằng mọi thư viện cài đặt tiếp theo sẽ được cài vào môi trường này, không ảnh hưởng đến hệ thống toàn cục.

Bước 3: Cài đặt các thư viện cần thiết cho dự án

Trong môi trường ảo đã kích hoạt, tiến hành cài đặt các thư viện chính phục vụ đề tài:

```
(venv) D:\mit_detection_demo>pip install ultralytics opencv-python numpy pandas streamlit pymongo
```

Hình 1.11. Hình ảnh cài đặt các thư viện sử dụng trong đề tài

- Ultralytics: Thư viện YOLOv8, phục vụ huấn luyện và suy luận mô hình.
- Opencv-python: xử lý ảnh, vẽ bounding box, đọc và ghi ảnh.
- Numpy: hỗ trợ tính toán trên mảng nhiều chiều.
- Pandas: thao tác dữ liệu dạng bảng, log kết quả, thống kê.
- Streamlit: xây dựng ứng dụng web demo.
- Pymongo: kết nối và thao tác với cơ sở dữ liệu MongoDB.

Sau khi cài đặt, có thể sử dụng lệnh:

```
(venv) D:\mit_detection_demo>pip freeze > requirements.txt
```

Hình 1.12. Hình ảnh lưu các thư viện vào file requirements.txt

Cách này để lưu lại danh sách thư viện và phiên bản tương ứng vào file requirements.txt. File này giúp việc cài đặt lại môi trường trên máy khác trở nên dễ dàng (chỉ cần chạy pip install -r requirements.txt trong môi trường máy ảo mới).

### 1.2.2. Khởi tạo Frontend và Backend cho dự án

1. Kích hoạt môi trường ảo mỗi khi bắt đầu làm việc với dự án.

- Mỗi khi bắt đầu làm việc với dự án, người phát triển cần kích hoạt môi trường ảo (venv\Scripts\activate hoặc source venv/bin/activate).

## 2. Backend

- Chạy backend: Backend của hệ thống (API phục vụ suy luận mô hình, truy xuất dữ liệu...) được xây dựng dựa trên FastAPI và chạy bằng uvicorn.
- Giả sử mã nguồn backend nằm trong thư mục backend/ và ứng dụng chính được định nghĩa trong backend/main.py với đối tượng app, ta có thể khởi động backend bằng lệnh:

```
(venv) D:\mit_detection_demo>uvicorn backend.main:app --reload
INFO:     Will watch for changes in these directories: ['D:\mit_detection_demo']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [9136] using StatReload
```

Hình 1.13. Hình ảnh khởi động backend bằng lệnh

- Backend.main.app chỉ ra module backend.main và biến ứng dụng app bên trong.
- Tham số --reload cho phép uvicorn tự động tải lại (hot reload) mỗi khi mã nguồn thay đổi, rất thuận tiện trong giai đoạn phát triển.
- Lệnh này sẽ khởi chạy một server HTTP (mặc định port 8000), cung cấp các endpoint API để frontend (Streamlit) hoặc các hệ thống khác có thể gọi đến.

## 3. Frontend

- Frontend của hệ thống được xây dựng bằng Streamlit, mã nguồn đặt trong thư mục frontend/, ví dụ file chính là frontend/app.py.
- Sau khi backend đã chạy, mở một terminal khác (vẫn trong môi trường ảo) và khởi động giao diện người dùng:

```
(venv) D:\mit_detection_demo>streamlit run frontend/app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.16:8501
```

Hình 1.14. Hình ảnh khởi động Frontend bằng lệnh

- Lệnh này sẽ mở ứng dụng Streamlit (thường tại địa chỉ http://localhost:8501).
- Ứng dụng sẽ cho phép người dùng tải ảnh chụp ảnh.
- Gửi yêu cầu đến backend (FastAPI/uvicorn) để thực hiện suy luận YOLOv8.

- Nhận kết quả (bounding box, nhãn, xác suất) và hiển thị trực quan cho người dùng.

#### 4. Quy trình làm việc điển hình

- Kích hoạt môi trường ảo.
- Mở hai cửa sổ terminal trong VS Code: Terminal 1: chạy backend bằng `uvicorn backend.main:app --reload`, Terminal 2: chạy frontend bằng `streamlit run frontend/app.py`.
- Tiến hành chỉnh sửa mã nguồn; khi backend thay đổi, `--reload` sẽ tự cập nhật; khi frontend thay đổi, Streamlit cũng sẽ tự reload giao diện.

### 1.2.3. Môi trường huấn luyện: Google Colab

#### ❖ Giới thiệu về Google Colab

Google Colab (Google Colaboratory) là một dịch vụ do Google cung cấp, cho phép người dùng chạy mã Python trực tuyến trên nền tảng cloud thông qua giao diện notebook tương tự như Jupyter Notebook. Điểm nổi bật của Google Colab là:



Hình 1.15. Hình ảnh về Google Colab

- Hỗ trợ sẵn nhiều thư viện liên quan đến học máy và học sâu (như NumPy, Pandas, Matplotlib, PyTorch, TensorFlow, OpenCV...).
- Cho phép sử dụng tài nguyên phần cứng tăng tốc như GPU (ví dụ: NVIDIA Tesla T4) hoặc TPU trong một khoảng thời gian nhất định, hoàn toàn miễn phí ở mức cơ bản.
- Tích hợp chặt chẽ với Google Driver, giúp lưu trữ dữ liệu, trọng số mô hình và notebook huấn luyện một cách thuận tiện.

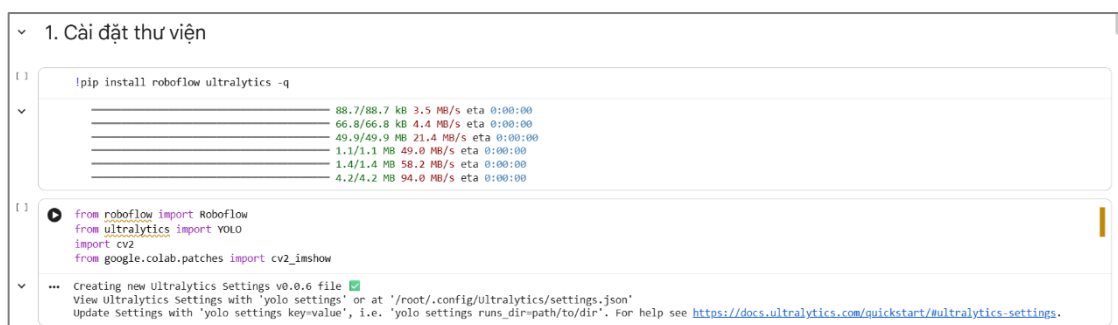
Để huấn luyện mô hình YOLOv8 cho bài toán nhận dạng và phân loại độ chín trái mít, đề tài sử dụng Google Colab làm môi trường huấn luyện chính. Google Colab cho phép chạy mã Python trên hạ tầng cloud, hỗ trợ GPU miễn phí, tích hợp sẵn nhiều thư viện học sâu và kết nối trực tiếp với Google Driver, rất phù hợp cho các bài toán cần tài nguyên tính toán lớn.

#### 1. Cài đặt thư viện

Ở đoạn đầu, chúng ta tiến hành:

- Cài cấu hình runtime sử dụng GPU trong Colab (Runtime → Change runtime type → GPU).
- Cài đặt các thư viện cần thiết bằng pip, bao gồm: ultralytics để sử dụng YOLOv8, opencv-python để xử lý ảnh, numpy, pandas để thao tác dữ liệu, matplotlib, seaborn để trực quan hoá biểu đồ, đánh giá mô hình.
- Import các thư viện này vào trong notebook để sử dụng trong các phần sau.

Đây là bước chuẩn bị môi trường phần mềm và phần cứng (GPU) cho toàn bộ quá trình huấn luyện.



```

1. Cài đặt thư viện

!pip install roboflow ultralytics -q

from roboflow import RoboFlow
from ultralytics import YOLO
import cv2
from google.colab.patches import cv2_imshow
    
```

Hình 1.16. Cài đặt thư viện với các ô lệnh trong Google Colab

## 2. Nhập dữ liệu

Ở bước này, dữ liệu dùng để huấn luyện và đánh giá mô hình được lấy trực tiếp từ Google Driver cá nhân. Đầu tiên notebook sử dụng module drive của thư viện google.colab để gắn (mount) Google Drive vào thư mục làm việc của Colab:

```

from google.colab import drive
drive.mount('/content/drive')
    
```

Sau khi lệnh trên thực thi thành công, toàn bộ nội dung Google Drive sẽ xuất hiện tại đường dẫn /content/drive. Colab hiển thị thông báo Mounted at / content/drive, cho biết Drive đã được gắn kết thành công.

Tiếp theo, chúng ta khai báo đường dẫn đến thư mục chứa bộ tài liệu của đề tài:

```
dataset_path = "/content/drive/MyDrive/Đồ_Án_Tốt_Nghiệp/dataset"
```

Trong đó:

- MyDrive/Đồ\_Án\_Tốt\_Nghiệp/dataset là thư mục trên Google Drive lưu toàn bộ dữ liệu ảnh trái mít và các file nhãn đã gán.

- Biến `dataset_path` được sử dụng ở các bước sau để truy cập đến ảnh và nhãn trong quá trình huấn luyện, đánh giá và kiểm thử mô hình YOLOv8.

```
2. Nhập dữ liệu

[ ] from google.colab import drive
    drive.mount('/content/drive')

... Mounted at /content/drive

[ ] dataset_path = "/content/drive/MyDrive/Đồ_Án_Tốt_Nghiệp/dataset"
```

Hình 1.17. Hình ảnh nhập dữ liệu ở Google Colab

### 3. Thiết lập mô hình YOLOv8

Trong phần này chúng ta tiến hành thiết lập mô hình YOLOv8

- Đầu tiên khởi tạo mô hình bằng thư viện Ultralytics ở bước trước đó:
  - `from ultralytics import YOLO`
  - `model = YOLO("yolov8n.pt")`
  - `model = YOLO("yolov8s.pt")`
- Lựa chọn các biến thể YOLOv8n hoặc YOLOv8s dựa vào cấu hình GPU và yêu cầu cân bằng giữa tốc độ suy luận và độ chính xác.
- Thiết lập các tham số huấn luyện quan trọng.
- EPOCHS: số epoch huấn luyện
- BATCH\_SIZE: kích thước batch
- IMG\_SIZE: kích thước ảnh đầu vào (thường 640x640).

Như vậy, mô hình YOLOv8 đã sẵn sàng để huấn luyện trên tập dữ liệu trái mít của đề tài.

```
3. Thiết lập mô hình YOLOv8

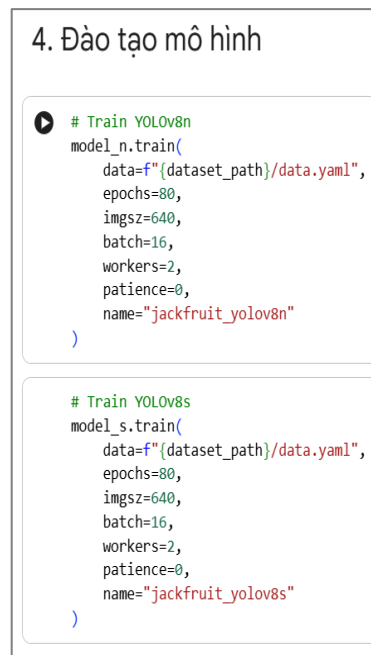
[ ] model_n = YOLO("yolov8n.pt")

[ ] model_s = YOLO("yolov8s.pt")
```

Hình 1.18. Hình ảnh thiết lập mô hình YOLOv8 trên Google Colab

### 4. Đào tạo mô hình

Phần đào tạo mô hình trong notebook thực hiện lệnh huấn luyện YOLOv8 với các siêu tham số đã lựa chọn như:



Hình 1.19. Hình ảnh đào tạo mô hình YOLOv8 trong Google Colab

Trong đó:

- Data: đường dẫn đến file data.yaml mô tả tập dữ liệu.
- Epochs: số vòng lặp huấn luyện.
- Imgsz: kích thước ảnh đầu vào (thường 640 x 640).
- Batch: kích thước batch, phụ thuộc vào dung lượng GPU.
- Name, project: đặt tên thư mục lưu kết quả trong run/detect.

Trong quá trình huấn luyện, YOLOv8 tự động lưu lại:

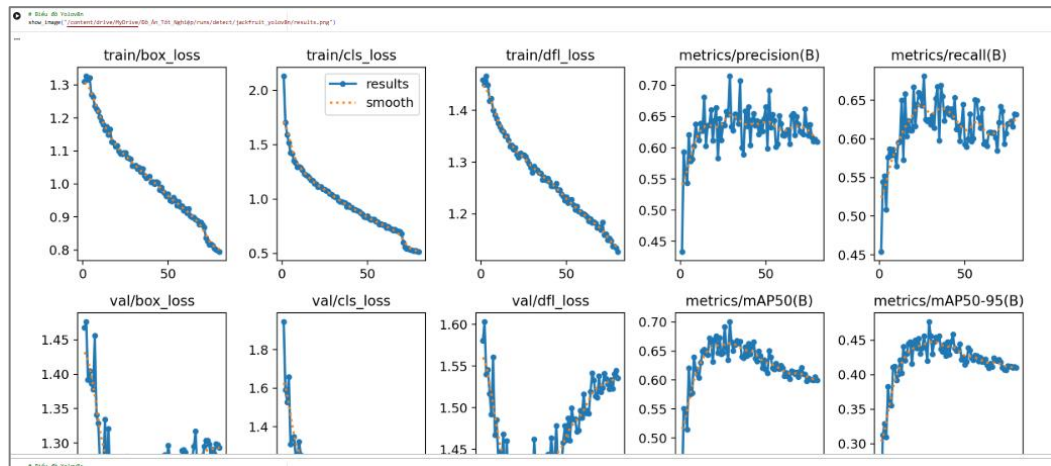
- Giá trị loss (box loss, cls loss, dfl loss) theo từng epoch.
- Các chỉ số đánh giá như mAP@0.5, mAP0.5:0.95, Precision, Recall trên tập validation.

## 5. Đánh giá mô hình sau huấn luyện

Sau khi huấn luyện kết thúc, chúng ta chuyển sang bước đánh giá mô hình trên tập validation/test:

- Sử dụng hàm model.val() hoặc lệnh yolo val để tính toán các chỉ số
- Precision, Recall, F1-score, mAP@0.5 , mAP@0.5:0.95 cho từng lớp và trung bình.

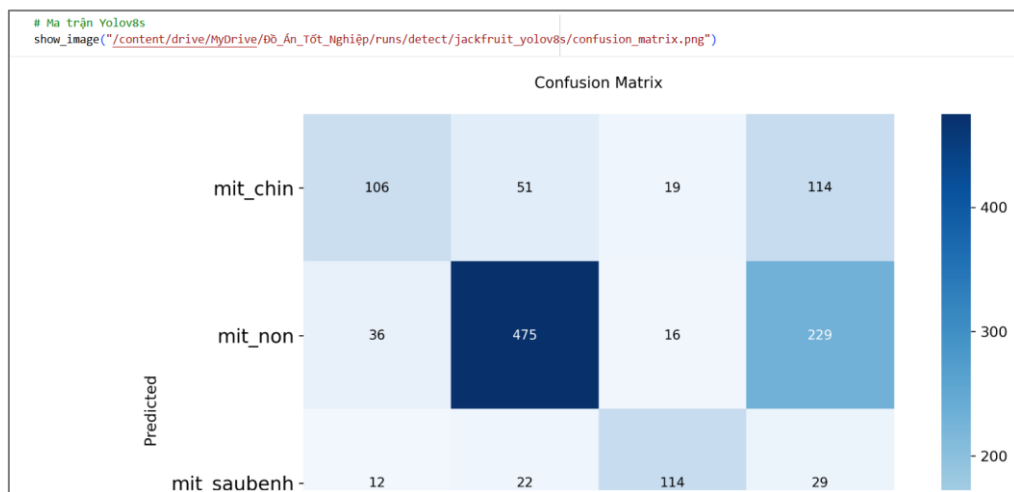
Và Sau khi kết quả đánh giá huấn luyện đã cho ra một biểu đồ so sánh từ YOLOv8



Hình 1.20. Hình ảnh biểu đồ mô hình YOLOv8 sau huấn luyện

Tiếp theo là ma trận nhầm lẫn (confusion matrix) cho các lớp: mít chín, mít non mít sâu bệnh, background. Ma Trận nhầm lẫn cho phép:

- Quan sát xem mô hình thường nhầm lẫn giữ những lớp nào, chẳng hạn:
- Mít non bị nhận nhầm thành mít chín.
- Mít sâu bệnh bị nhận nhầm thành mít chín nếu vùng sâu bệnh nhỏ...



Hình 1.21. Hình ảnh ma trận Confusion Matrix của mô hình YOLOv8

## 6. Đánh giá kết quả và đầu ra của mô hình

Sau khi hoàn thành quá trình huấn luyện mô hình YOLOv8, bước tiếp theo là tiến hành đánh giá hiệu quả mô hình thông qua kiểm thử thực tế trên tập dữ liệu mới. Mục tiêu là kiểm tra khả năng tổng quát của mô hình và xác định mức độ chính xác trong việc nhận dạng trái mít theo đúng 3 lớp: mit\_chin, mit\_non, mit\_saubenh.



Hình 1.22. Hình ảnh kết quả đầu ra mô hình YOLOv8 với ảnh nhận dạng trái mít

Kết quả mô hình đã nhận dạng chính xác nhiều trái mít trong ảnh, với mỗi bbox hiển thị nhãn lớp (mit\_chin, mit\_non, mit\_saubenh) tương ứng hiển thị trên ảnh. Và mô hình cũng phân biệt được mít chín, mít non và sâu bệnh với độ tin cậy cao, đặt biệt khi trái mít rõ nét và không bị che khuất.

#### 1.2.4. Cơ sở dữ liệu MongoDB Atlas

##### ❖ Giới thiệu chung

MongoDB Atlas là dịch vụ MongoDB trên nền tảng cloud (Database as a Service) do chính MongoDB cung cấp. Thay vì phải tự cài đặt và quản trị server MongoDB trên máy cục bộ hoặc VPS, người dùng chỉ cần:

- Tạo một cluster trên Atlas.
- Cấu hình người dùng, mật khẩu, IP được phép truy cập.
- Kết nối tới cluster thông qua một connection string (URL).

MongoDB Atlas vẫn được sử dụng mô hình NoSQL dạng document như MongoDB thông thường: dữ liệu được lưu dưới dạng các document JSON/BSON trong các collection thuộc một database, nhưng toàn bộ phần hạ tầng (cài đặt, backup, scaling) được Atlas quản lý.



Hình 1.23. Hình ảnh về MongoDB Atlas

##### ❖ Vai trò của MongoDB Atlas trong hệ thống

Trong đề tài, MongoDB Atlas được sử dụng làm cơ sở dữ liệu chính để:

- Lưu thông tin người dùng: Tài khoản đăng ký, đăng nhập của người dùng
- Lưu thông tin ảnh/video đầu vào: Tên file, đường dẫn (hoặc URL lưu trữ), nguồn dữ liệu (upload, webcam, video), thời gian xử lý.
- Lưu kết quả nhận dạng từ mô hình YOLOv8: Danh sách các trái mít được phát hiện trên ảnh/frame video, thông tin chi tiết từng đối tượng: lớp (mít non, mít chín, mít sâu bệnh..) xác xuất, tọa độ bounding box, phiên bản mô hình đã sử dụng

Việc dùng Atlas thay vì MongoDB cục bộ giúp:

- Dễ dàng truy cập từ nhiều nơi (backend chạy local, deploy server, máy khác..) chỉ cần đúng connection string.
- Hạn chế phải tự quản lý backup, update, bảo trì server DB.
- Phù hợp với định hướng sau này nếu muốn triển khai hệ thống lên cloud.

#### ❖ **Ưu điểm và hạn chế của MongoDB Atlas**

Ưu điểm:

- Cloud-managed: không cần cài đặt và vận hành máy chủ MongoDB thủ công.
- Hỗ trợ backup, scaling, bảo mật (IP whitelist, user/password, TLS) ngay trong giao diện web.
- Vẫn giữ đầy đủ ưu điểm của MongoDB document: linh hoạt, dễ lưu dữ liệu AI (bbox, nhãn, log mô hình..).
- Có free tier đủ dùng cho một đồ án tốt nghiệp.

Hạn chế:

- Phụ thuộc vào kết nối internet (backend muốn truy cập Atlas phải có mạng).
- Nếu dữ liệu lớn, cần hiệu năng cao phải nâng gói tính phí.
- Cần cấu hình bảo mật (IP, user, password) cẩn thận, tránh lộ connection string.

#### ❖ **Kết nối MongoDB Atlas trong dự án**

Quy trình sử dụng Atlas trong đề tài thường gồm các bước:

##### **1. Tạo Cluster trên MongoDB Atlas**

- Đăng ký tài khoản MongoDB Atlas.
- Tạo một cluster (free tier) và một database, ví dụ mit\_detection.
- Tạo user (username/password) dùng cho ứng dụng.

- Cấu hình Network Access (cho phép IP của máy phát triển, hoặc 0.0.0.0/0 nếu chỉ dùng tạm khi phát triển – nhưng nên hạn chế trong thực tế).

## 2. Lấy connection string (URL)

Atlas cung cấp một URL dạng , và chúng ta sẽ copy đoạn URL này dán vào file .env môi trường của dự án.

```
mongodb+srv://<username>:<password>@<cluster-name>.mongodb.net/mit_detection?retryWrites=true&w=majority
```

## 3. Cài đặt thư viện Pymongo trong môi trường ảo

```
pip install pymongo
```

## 4. Kết nối từ backend đến MongoDB Atlas

# Tải các biến môi trường từ file .env

```
load_dotenv()
```

```
def get_database():
```

```
    """
```

```
    Kết nối tới MongoDB Atlas và trả về database 'mit_detection'.
```

```
    """
```

```
    MONGO_URI = os.getenv("MONGO_URI")
```

```
    try:
```

```
        client = MongoClient(MONGO_URI)
```

```
        client.admin.command('ping')
```

```
        print("Kết nối MongoDB Atlas thành công!")
```

```
        return client["mit_detection"]
```

```
    except Exception as e:
```

```
        print("Lỗi kết nối MongoDB:", e)
```

```
    raise e
```

### 1.2.5. Ngôn ngữ Dart và Framework Flutter cho ứng dụng di động

#### 1.2.5.1. Dart là gì ?

Dart là một ngôn ngữ lập trình mới, được phát triển bởi Google dành cho việc phát triển các ứng dụng đa nền tảng. Dart thường được gắn liền với Flutter framework, một lựa chọn cho việc xây dựng mobile app. Tuy nhiên, Dart không chỉ dùng cho việc xây dựng ứng dụng di động, ngôn ngữ này còn được áp dụng cho nhiều dự án phần mềm đa lĩnh vực khác. Dart hỗ trợ lập trình hướng đối tượng (OOP) , cú pháp gần gũi với các ngôn ngữ như Java, C# , đồng thời được tối ưu để kết hợp với Flutter giúp ứng dụng có hiệu năng tốt trên thiết bị di động.



Hình 1.24. Ngôn ngữ lập trình Dart

#### 1.2.5.2. Flutter là gì ?

Flutter được tạo ra bởi ông lớn Google và được ra mắt vào năm 2017 và cho đến nay Flutter được phát hành phiên bản mới nhất vào cuối tháng 5 vừa qua. Flutter là framework UI đa nền tảng (cross-platform UI framework) cũng do Google phát triển, cho phép xây dựng ứng dụng chạy trên Android, iOS, web và desktop chỉ với một codebase duy nhất. Flutter cung cấp hệ thống widget phong phú, cơ chế hot reload và engine render riêng, giúp việc phát triển giao diện nhanh và mượt.

Trong đề tài, Dart và Flutter được sử dụng để xây dựng ứng dụng di động AgriVision Mobile với các đặc điểm chính:

- Chụp ảnh trái mít trực tiếp từ camera hoặc chọn ảnh từ thư viện.
- Gửi ảnh cho mô hình phân loại (được huấn luyện bằng Teachable Machine) để dự đoán mức độ chín của trái mít.
- Hiển thị kết quả phân loại trên giao diện trực quan và dễ hiểu.
- Lưu lại lịch sử các lần nhận dạng và đồng bộ với backend Supabase.



Hình 1.25. Nền tảng Framework Flutter

#### 1.2.6. Nền tảng Supabase backend cho ứng dụng di động

Supabase là nền tảng Backend-as-a-Service (BaaS) mã nguồn mở, được xây dựng trên PostgreSQL và cung cấp sẵn các tính năng như: cơ sở dữ liệu, authentication, REST

API, lưu trữ file (storage) và real-time. Supabase giúp lập trình viên triển khai backend nhanh chóng mà không cần tự cấu hình và quản lý server.

Trong ứng dụng AgriVision Mobile, Supabase được sử dụng với các vai trò chính:

- Lưu trữ thông tin tài khoản và phiên đăng nhập người dùng (nếu sử dụng đăng nhập trên app).
- Lưu log các lần nhận dạng từ ứng dụng di động, bao gồm: thời gian, kết quả phân loại.



Hình 1.26. Nền tảng Supabase backend Service

### 1.2.7. Công cụ hỗ trợ làm việc nhóm và quản lý mã nguồn

#### 1.2.7.1. Trello là gì ?

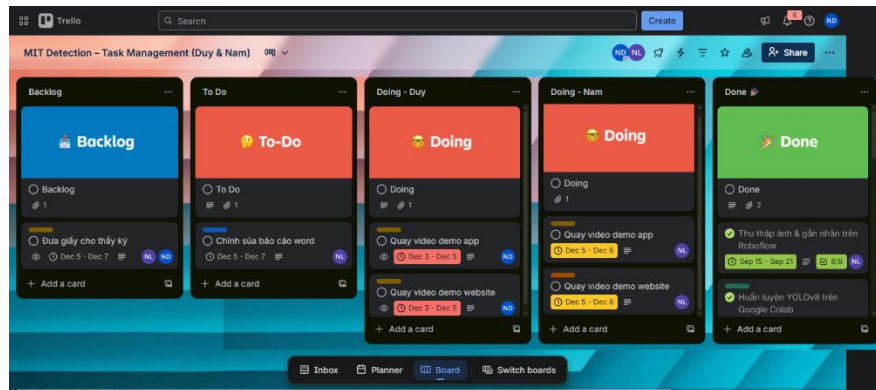
Trello là một công cụ quản lý công việc và dự án trực tuyến, hoạt động theo phương pháp Kanban: công việc được tổ chức trên các bảng (board), trong mỗi bảng có các danh sách (list) như Backlog, To Do, Doing, Done và mỗi thẻ (card) đại diện cho một nhiệm vụ cụ thể. Người dùng có thể kéo - thả thẻ giữa các cột để thể hiện trạng thái công việc, gán người thực hiện, đặt deadline, đính kèm tài liệu và trao đổi qua bình luận. Nhờ giao diện trực quan và cách sử dụng đơn giản, Trello rất phù hợp cho việc lên kế hoạch, phân công nhiệm vụ và theo dõi tiến độ trong các dự án học tập cũng như dự án phần mềm.

#### ❖ Ứng dụng Trello trong quá trình thực hiện đồ án.

Trong quá trình thực hiện đồ án “Nghiên cứu và ứng dụng YOLOv8 trong nhận dạng và phân loại độ chín trái mít phục vụ nông nghiệp thông minh”, hai sinh viên Nguyễn Văn Duy và Lê Nam đã sử dụng một bảng Trello riêng để quản lý toàn bộ công việc của nhóm.

- Nhóm tạo board với tên “MIT Detection – Task Management (Duy & Nam)”, trong đó chia thành các cột:
- **Backlog:** chứa các ý tưởng, đầu việc chung cần thực hiện (ví dụ: “Đưa giấy cho thầy ký”).
- **To Do:** những nhiệm vụ đã được thống nhất sẽ làm, chờ phân công và thực hiện (như “Chỉnh sửa báo cáo Word”).

- **Doing – Duy, Doing – Nam:** thể hiện rõ công việc mà từng thành viên đang thực hiện, ví dụ Duy phụ trách quay video demo website, Nam phụ trách quay video demo app,...
- **Done:** các công việc đã hoàn thành, như “Xây dựng Backend FastAPI + tích hợp YOLOv8”, “Kết nối Frontend – Backend và test toàn bộ hệ thống”, “Tiếp tục viết code + chỉnh sửa + hoàn thiện hệ thống”, kèm ngày bắt đầu, ngày kết thúc và nhãn màu cho từng thẻ.



Hình 1.27. Hình ảnh làm việc nhóm trên Trello

#### 1.2.7.2. Công cụ quản lý mã nguồn GitHub

GitHub là một nền tảng lưu trữ và quản lý mã nguồn trên đám mây, xây dựng dựa trên hệ thống kiểm soát phiên bản Git. Nền tảng này cho phép lập trình viên lưu trữ source code, theo dõi lịch sử thay đổi, tạo nhánh (branch) để phát triển tính năng độc lập, đồng thời hỗ trợ làm việc nhóm thông qua commit, pull request, review và thảo luận ngay trên giao diện web. Nhờ đó, GitHub trở thành công cụ gần như “bắt buộc” trong các dự án phần mềm hiện đại, đặc biệt là khi có nhiều người cùng tham gia và cần quản lý nhiều phiên bản code khác nhau.

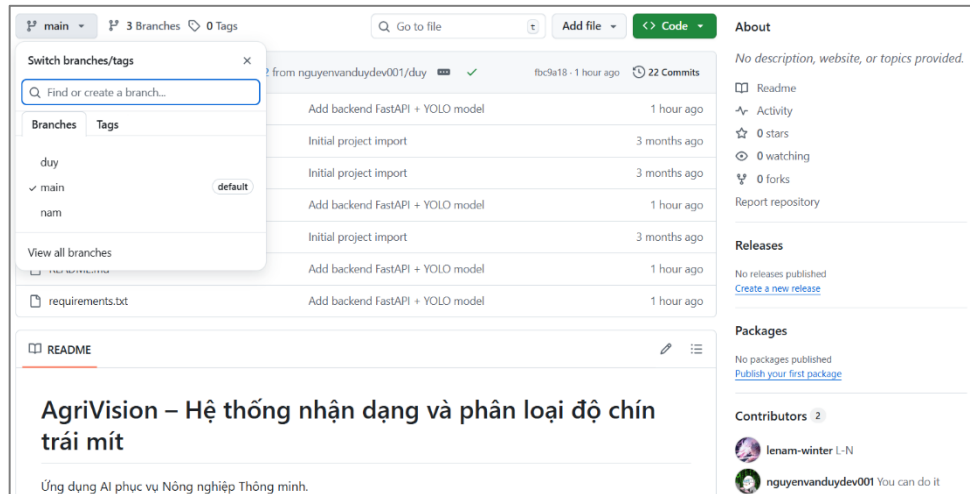
##### ❖ Ứng dụng GitHub trong quá trình phát triển hệ thống

Trong đồ án “Nghiên cứu và ứng dụng YOLOv8 trong nhận dạng và phân loại độ chín trái mít phục vụ nông nghiệp thông minh”, Nguyễn Văn Duy và Lê Nam đã sử dụng GitHub làm kho lưu trữ chính cho mã nguồn hệ thống.

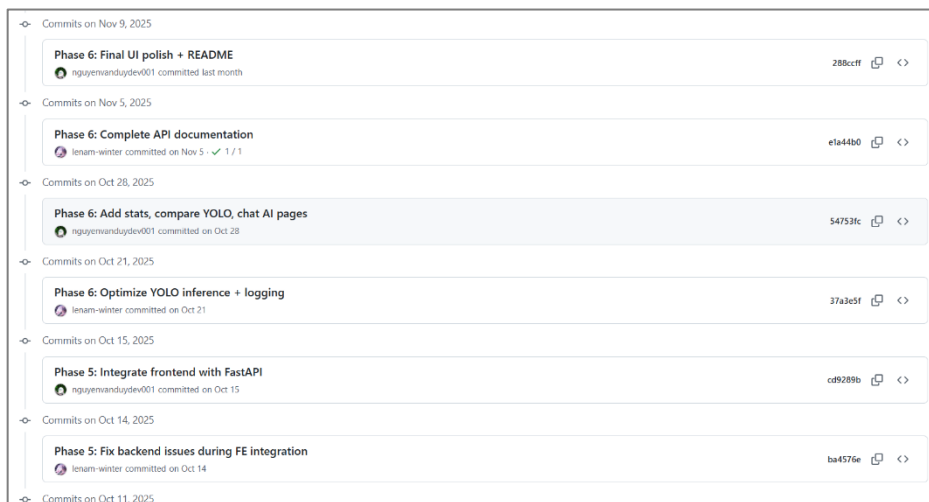
- **main:** nhánh chính, chứa phiên bản ổn định của hệ thống.
- **duy:** nhánh làm việc của duy là bên frontend
- **nam:** nhánh làm việc của nam là bên phía backend
- Lịch sử commit thể hiện rõ quá trình phát triển theo từng giai đoạn (phase).

Mỗi thành viên phát triển trên nhánh riêng, sau đó hợp nhất (merge) vào main khi chức năng đã hoàn thành và được kiểm thử, Mục Contributors hiển thị rõ hai

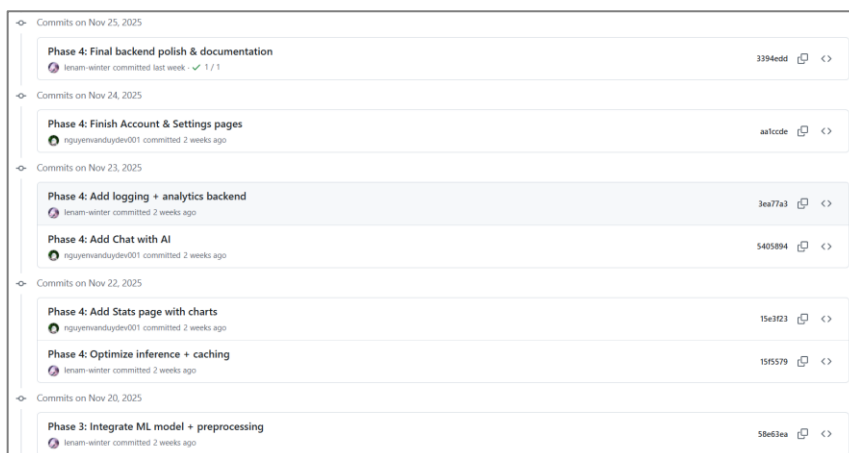
tài khoản tham gia: nguyenvanduydev001 và lenam-winter, cho thấy vai trò và đóng góp của từng thành viên trong suốt quá trình phát triển.



Hình 1.28. Repository GitHub để quản lý mã nguồn và phân chia công việc nhóm



Hình 1.29. Lịch sử commit trên Github phía bên web



Hình 1.30. Lịch sử commit trên Github phía bên App Mobile

## Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

### 2.1. Khảo sát thực tế và xác định yêu cầu

#### 2.1.1. Khảo sát thực tế

Trong thực tế sản xuất và kinh doanh trái mít tại các vườn trồng, hợp tác xã, chợ đầu mối và siêu thị, đánh giá độ chín và phát hiện các vấn đề về chất lượng (sâu bệnh, dập nát, khuyết tật bề mặt...) hiện nay chủ yếu được thực hiện bằng phương pháp thủ công. Người Nông dân, thương lái hoặc nhân viên thu mua thường dựa vào kinh nghiệm để quan sát:

- Màu sắc vỏ, độ bóng, độ nở của gai mít.
- Hình dạng trái, mức độ đồng đều của bề mặt.
- Một số trường hợp còn dựa vào mùi hương hoặc âm thanh khi gõ vào trái mít.



Hình 2.1. Hình ảnh người dân đánh giá mít dựa vào kinh nghiệm nuôi trồng

Cách đánh giá này tuy linh hoạt nhưng mang tính chủ quan, phụ thuộc vào rất nhiều kinh nghiệm cá nhân, dẫn đến các hạn chế:

- Kết quả phân loại không đồng nhất giữa người đánh giá, dễ xảy ra tranh cãi về độ chín hoặc chất lượng trái.
- Khó mở rộng khi số lượng trái cần kiểm tra lớn (ví dụ trong khâu phân loại trước khi đóng gói, xuất khẩu, gây tốn thời gian và nhân công.
- Không tạo ra được dữ liệu số để lưu trữ, phân tích và tối ưu dần quy trình sản xuất – tiêu thụ.

Cùng với sự phổ biến của smartphone, camera giám sát và các thiết bị IoT trong nông nghiệp, nhu cầu ứng dụng thị giác máy tính để hỗ trợ đánh giá chất lượng nông sản ngày càng tăng. Tuy nhiên, qua khảo sát ban đầu cho thấy:

- Các giải pháp sẵn có chủ yếu tập trung vào một số loại trái cây phổ biến như (chuối, táo, cam..), ít có hệ thống chuyên biệt cho trái mít phù hợp điều kiện Việt Nam.
- Một số hệ thống nhận dạng sử dụng ảnh tĩnh, chưa hỗ trợ đa dạng nguồn dữ liệu như webcam, video, chatbot tư vấn và không tích hợp giao diện thân thiện cho người dùng cuối.
- Hầu như chưa có giải pháp tích hợp chatbot tư vấn để người dùng đặt câu hỏi, tra cứu nhanh thông tin về hệ thống, kết quả nhận dạng hay kiến thức cơ bản liên quan đến mô hình.
- Việc thử nghiệm, so sánh nhiều mô hình YOLOv8 khác nhau (ví dụ thay đổi cấu hình dữ liệu huấn luyện) thường chỉ thực hiện ở mức mã nguồn, chưa có chức năng trực quan giúp người dùng lựa chọn và so sánh chất lượng giữa các mô hình.

Từ thực tế đó, có thể thấy nhu cầu cấp thiết xây dựng một hệ thống:

- Cho phép người dùng tải ảnh, bật webcam hoặc tải video để hệ thống tự động phát hiện và phân loại trái mít trong nhiều bối cảnh khác nhau.
- Cung cấp giao diện sử dụng đơn giản, tích hợp chatbot hỗ trợ giải đáp thắc mắc về thao tác, kết quả và các khái niệm cơ bản.
- Hỗ trợ tải và so sánh nhiều mô hình YOLOv8 khác nhau, đưa ra các chỉ số đánh giá và kết quả trực quan để lựa chọn mô hình phù hợp nhất cho bài toán thực tế.

### 2.1.2. Mục tiêu hệ thống

Dựa trên kết quả khảo sát thực tế, hệ thống được xây dựng với mục tiêu chính là hỗ trợ tự động hoá quá trình nhận dạng và phân loại độ chín, tình trạng trái mít dựa trên ảnh, video, từ đó giảm phụ thuộc vào việc đánh giá thủ công, tăng tính khách quan và tạo nền tảng cho các ứng dụng nông nghiệp thông minh.

Cụ thể, hệ thống hướng đến các mục tiêu sau:

- **Cung cấp giao diện web thân thiện, dễ sử dụng:** Cho phép người dùng thao tác trực tiếp trên trình duyệt với các chức năng rõ ràng như: đăng ký, đăng nhập, tải ảnh, bật webcam, tải video, xem kết quả... mà không cần kiến thức chuyên sâu về kỹ thuật.
- **Hỗ trợ đa dạng nguồn dữ liệu đầu vào:** Người dùng có thể. Tải ảnh trái mít từ thiết bị lên hệ thống để nhận dạng, Sử dụng webcam để nhận dạng trực tiếp các trái mít xuất hiện trong khung hình, tải video lên hệ thống phân tích và phát hiện các trái mít trong từng khung hình của video.

- **Tích hợp mô hình YOLOv8 để nhận dạng và phân loại tự động:** Ứng dụng mô hình YOLOv8 đã được huấn luyện nhằm. Phát hiện vị trí các trái mít trên ảnh/video. Phân loại độ chín/tình trạng (ví dụ: non, chín, sâu bệnh) với độ chính xác phù hợp cho thực tế.
- **Quản lý và lưu trữ dữ liệu nhận dạng một cách hệ thống:** Lưu Trữ thông tin ảnh/video, kết quả suy luận (số lượng trái, nhãn, xác suất, thời gian xử lý, mô hình sử dụng..) trong cơ sở dữ liệu để phục vụ việc tra cứu, thống kê và phân tích sau này.
- **Hỗ trợ so sánh chất lượng giữa các mô hình YOLOv8 khác nhau:** Cung cấp chức năng tải và lựa chọn mô hình, so sánh hai mô hình (ví dụ: YOLOv8n và YOLOv8s) dựa trên các chỉ số đánh giá và kết quả thực nghiệm, từ đó giúp lựa chọn mô hình phù hợp nhất cho bài toán.
- **Tích hợp chatbot hỗ trợ người dùng:** Cho phép người dùng gửi câu hỏi lên chatbot để được hỗ trợ về: Cách sử dụng hệ thống, giải thích các kết quả nhận dạng và một số khái niệm cơ bản về mô hình YOLOv8 và xử lý ảnh.

Thông qua các mục tiêu trên, hệ thống không chỉ dừng ở mức minh họa mô hình AI mà còn hướng đến một công cụ hỗ trợ thực tế, có khả năng mở rộng rất nhiều thứ và tích hợp vào giải pháp dành cho nông nghiệp thông minh cho hiện nay và trong tương lai.

Bên cạnh phiên bản web, đề tài còn hướng đến việc xây dựng ứng dụng di động demo AgriVision Mobile, cho phép người dùng chụp ảnh trái mít trực tiếp bằng điện thoại, phân loại nhanh mức độ chín và lưu lại lịch sử nhận dạng để thuận tiện khi làm việc tại vườn.

### 2.1.3. Xác định yêu cầu

Dựa trên kết quả khảo sát thực tế và các mục tiêu hệ thống đã đề ra, phần này trình bày các yêu cầu cụ thể đối với hệ thống, Trong phạm vi này, các yêu cầu được trình bày tập trung chủ yếu cho hệ thống web sử dụng YOLOv8, các yêu cầu chi tiết cho ứng dụng di động sẽ được kế thừa và triển khai ở phần thiết kế, triển khai ứng dụng AgriVision Mobile, bao gồm yêu cầu chức năng và yêu cầu phi chức năng. Đây sẽ là cơ sở quan trọng cho việc phân tích, thiết kế và triển khai hệ thống ở các chương tiếp theo.

#### 2.1.3.1. Yêu cầu chức năng

Các yêu cầu chức năng được chia theo nhóm người dùng hệ thống và quản trị viên hệ thống như sau:

##### ❖ Nhóm chức năng dành cho người dùng hệ thống.

1. Đăng ký tài khoản

- Người dùng có thể tạo tài khoản mới bằng cách cung cấp các thông tin cơ bản như (tên đăng nhập, mật khẩu, email..).
- Hệ thống kiểm tra trùng lặp tên đăng nhập, kiểm tra định dạng mật khẩu, thông báo lỗi nếu không hợp lệ.

2. Đăng nhập - đăng xuất

- Người dùng sử dụng tên đăng nhập và mật khẩu để truy cập hệ thống.
- Hệ thống kiểm tra thông tin xác thực, từ chối truy cập nếu sai.
- Cho phép người dùng đăng xuất an toàn khỏi hệ thống sau khi sử dụng.

3. Nhận dạng trái mít từ hình ảnh tải lên

- Người dùng có thể chọn và tải một hoặc nhiều ảnh trái mít từ thiết bị lên ứng dụng.
- Hệ thống gửi ảnh đến backend, sử dụng mô hình YOLOv8 để: Phát hiện các trái mít trong ảnh, phân loại độ chín/tình trạng (ví dụ: mít non, mít chín, mít sâu bệnh).
- Kết quả được hiển thị dưới dạng: Ảnh có vẽ bounding box và nhãn lớp.
- Bảng danh sách các đối tượng được phát hiện (Lớp, xác suất, vị trí..)
- Yêu cầu phân tích hình ảnh tải lên bằng AI và đưa ra kết quả giám định

4. Nhận dạng trái mít qua Webcam

- Người dùng sẽ bật webcam từ hệ thống trình duyệt
- Hệ thống nhận luồng hình ảnh từ webcam và thực hiện: Phát hiện các trái mít có trong khung hình
- Cập nhật kết quả theo thời gian thực (real-time).
- Kết quả hiển thị trực tiếp trên giao diện (khung video có bounding box + nhãn)

5. Nhận dạng trái mít từ video tải lên

- Người dùng có thể tải file video (định dạng được hỗ trợ) lên hệ thống
- Khi người dùng nhấn bắt đầu phân tích video thì video sẽ phát hiện trái mít xuất hiện trong video
- Trái mít xuất hiện trên video sẽ có bounding box trên từng trái mít , và thống kê số lượng trái mít có trong video.
- Khi phân tích video xong sẽ hiển thị ra kết quả json và từ kết quả đó AI sẽ phân tích ra kết quả cho người dùng.

6. Tương tác với Chatbot AI hỗ trợ

- Người dùng có thể gửi câu hỏi cho chatbot ở khung nhập câu hỏi ngay trên giao diện hệ thống.
- Chatbot sẽ trả lời các câu hỏi liên quan đến trang web như: Bạn có thể giúp gì cho tôi, cách sử dụng hệ thống, hướng dẫn chức năng, giải thích ý nghĩa một số thông số có trong mô hình và một số thông tin cơ bản về nhận dạng trái mít.

7. Tải tệp - chọn mô hình YOLOv8 và so sánh mô hình

- Hệ thống cho phép người dùng: Lựa chọn mô hình YOLOv8 đang được triển khai (ví dụ: YOLOv8n, YOLOv8s).
- Tải lên file trọng số mô hình theo đúng định dạng yêu cầu.
- Chức năng so sánh hai mô hình: Người dùng chọn và tải lên 2 mô hình lên hệ thống cần so sánh.
- Hệ thống đánh giá trên tập dữ liệu
- Hiển thị kết quả so sánh (mAP, Precision, Recall , và các thông số khác một cách trực quan.

8. Thống kê xem lại lịch sử

- Người dùng có thể xem lại lịch sử và kết quả vừa mới nhận diện
- Thời gian thực hiện.
- Số lượng và các đối tượng được phát hiện trước đó.

❖ **Nhóm chức năng cho quản trị viên hệ thống**

1. Quản lý tài khoản người dùng

- Xem danh sách tài khoản người dùng.
- Thêm, sửa, xoá tài khoản

2. Quản lý mô hình YOLOv8

- Thêm mới mô hình (upload trọng số, đặt tên phiên bản).
- Cập nhật cấu hình mô hình
- Gỡ bỏ các mô hình không còn sử dụng

3. Quản lý cơ sở dữ liệu

- Xem và tra cứu lịch sử các phiên nhận dạng đã được lưu trên MongoDB Atlas.
- Quản lý các dữ liệu của hệ thống và theo dõi được dữ liệu của người dùng.

4. Cấu hình hệ thống

- Cấu hình một số tham số chung: thời gian timeout, kích thước tối đa , số lượng ảnh upload.

- Cấu hình quyền truy cập đối với một số chức năng nâng cao (so sánh mô hình, upload mô hình mới..).

#### 2.1.3.2. Yêu cầu phi chức năng

Ngoài các chức năng cụ thể, hệ thống cần đáp ứng một số yêu cầu phi chức năng nhằm đảm bảo khả năng hoạt động ổn định, dễ sử dụng và có khả năng mở rộng trong tương lai.

##### 1. Hiệu năng

- Thời gian xử lý trung bình cho một ảnh ở mức chấp nhận được trên máy cấu hình trung bình (ví dụ: từ vài trăm mili giây đến vài giây, tùy kích thước của ảnh và mô hình).
- Việc nhận dạng qua webcam cần phản hồi đủ nhanh và chính xác để người dùng cảm nhận được tính “gần thời gian thực”
- Hệ thống phải xử lý được nhiều yêu cầu liên tiếp mà không bị lag, treo máy hoặc không xử lý được.

##### 2. Tính chính xác

- Mô hình YOLOv8 được huấn luyện phải đạt các chỉ số mAP, Precision, Recall ở mức đáp ứng được yêu cầu bài toán (cụ thể được trình bày ở chương trình thực nghiệm).
- Kết quả hiển thị cần rõ ràng, tránh gây hiểu lầm cho người dùng.

##### 3. Tính chất dễ sử dụng

- Giao diện sử dụng ngôn ngữ tiếng việt, bố cục đơn giản, các chức năng chính như (tải ảnh, webcam, video, chatbot..) phải thật dễ nhận biết.
- Các thông báo trong web phải rõ ràng và thân thiện, không khiến người dùng cảm thấy hoang mang.
- Chatbot hỗ trợ cơ bản để người dùng tự tìm hiểu cách sử dụng mà không cần đến tài liệu phức tạp.

##### 4. Tính tin cậy và ổn định

- Hệ thống phải hoạt động ổn định trong thời gian dài, hạn chế tối đa tình trạng xảy ra lỗi không mong muốn.
- Khi xảy ra lỗi kỹ thuật gì đó trong nội bộ, thì hệ thống cần: Hiển thị thông báo phù hợp cho người dùng yên tâm, đồng thời không làm mất toàn bộ dữ liệu đã lưu trước đó.

##### 5. Tính bảo mật

- Thông tin tài khoản (username, password) phải được bảo vệ (ví dụ: lưu mật khẩu người dùng dưới dạng hash).
- Connection string tới MongoDB Atlas phải được bảo mật, không chia sẻ công khai trên bất cứ nền tảng nào.

## 2.2. Mô hình Use Case tổng quát và chi tiết

### 2.2.1. Xác định các actor trong hệ thống

Trong hệ thống nhận dạng và phân loại độ chín trái mít, có hai actor chính tham gia tương tác với hệ thống:

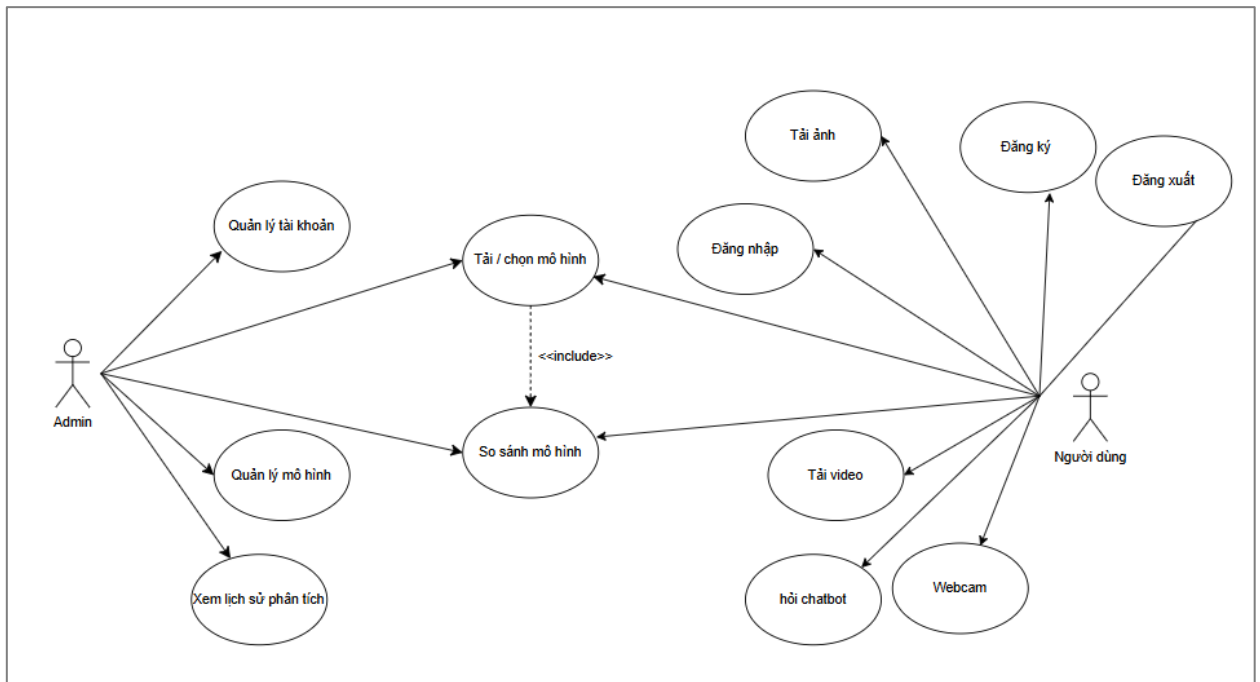
- Người dùng: Là đối tượng trực tiếp khai thác các chức năng của hệ thống như (sinh viên, nông dân, kỹ thuật viên...). Người dùng có thể đăng ký/đăng nhập, tải ảnh hoặc video, sử dụng webcam để nhận dạng trái mít, tương tác với chatbot, lựa chọn và so sánh các mô hình YOLOv8.
- Admin: Là người quản trị hệ thống, phụ trách kỹ thuật và quản lý hệ thống. Admin có quyền quản lý tài khoản người dùng, quản lý các mô hình YOLOv8 được triển khai, và xem lịch sử dữ liệu nhận dạng.

Bảng 2.1. Mô tả các Actor trong hệ thống

| Actor      | Mô tả                            | Vai trò chính                                                                      |
|------------|----------------------------------|------------------------------------------------------------------------------------|
| Người dùng | Người trực tiếp sử dụng hệ thống | Đăng ký/đăng nhập, tải ảnh-video, bật webcam, hỏi chatbot, chọn và so sánh mô hình |
| Admin      | Người quản trị hệ thống          | Quản lý tài khoản người dùng, quản lý mô hình, xem lịch sử nhận dạng               |

### 2.2.2. Sơ đồ Use Case tổng quát của hệ thống

Để mô tả các chức năng chính và mối quan hệ giữa người sử dụng với hệ thống, đề tài sử dụng mô hình use case theo chuẩn UML. Sơ đồ use case tổng quát của hệ thống được thể hiện như sau:



Hình 2.2. Hình ảnh về sơ đồ Use Case tổng quát

Trong sơ đồ Use Case, một số ký pháp UML chính được sử dụng như sau:

❖ **Actor (tác nhân)**



- Được biểu diễn bằng hình người que (biểu tượng “stick man”) với tên đặt phía dưới, ví dụ: Người dùng, Admin.
- Actor thể hiện vai trò tương tác với hệ thống, không nhất thiết là một cá nhân cụ thể mà có thể là một nhóm người hay một hệ thống bên ngoài.

❖ **Use Case (chức năng)**



- Được biểu diễn bằng hình elip (oval) bên trong khung hệ thống, kèm theo tên chức năng ở chính giữa.
- Mỗi use case mô tả một chức năng mà hệ thống cung cấp cho actor, được nhìn từ góc độ người dùng hệ thống.

❖ **Đường kết nối actor – use case**

- Là các đường thẳng nối từ actor tới use case.
- Mỗi đường nối thể hiện rằng actor đó có quyền sử dụng hoặc tham gia vào chức năng tương ứng trong hệ thống.

❖ **Quan hệ <<include>>**

- Được biểu diễn bằng đường kẻ nét đứt có mũi tên, kèm theo nhãn <<include>> hướng từ use case gọi đến use case được dùng chung
- Trong sơ đồ use case “So sánh mô hình YOLOv8 và xuất kết quả” có quan hệ <<include>> với use case “Tải/chọn mô hình YOLOv8”, thể hiện rằng mỗi lần so sánh mô hình đều phải thực hiện bước tải/chọn mô hình.

### 2.2.3. Mô tả các chức năng (use case) của hệ thống

Các chức năng trong sơ đồ use case được chia thành hai nhóm chính: nhóm dành cho người dùng và nhóm dành cho Admin.

#### 2.2.3.1. Nhóm chức năng dành cho người dùng

Bảng 2.2. Mô tả chức năng của người dùng

| STT | Chức năng                     | Actor      | Mô tả                                                                                                  |
|-----|-------------------------------|------------|--------------------------------------------------------------------------------------------------------|
| 1   | Đăng ký                       | Người dùng | Tạo tài khoản mới bằng cách đăng nhập các thông tin cần thiết như (tên tài khoản, email, mật khẩu).    |
| 2   | Đăng nhập                     | Người dùng | Xác thực tài khoản từ bên đăng ký để truy cập vào hệ thống website, để sử dụng các tính năng.          |
| 3   | Đăng xuất                     | Người dùng | Thoát khỏi phiên làm việc hiện tại, không còn thông tin người dùng đó nữa                              |
| 4   | Tải ảnh trái mít và nhận dạng | Người dùng | Tải một ảnh từ thiết bị lên hệ thống: Mô hình YOLOv8 sẽ phát hiện và phân loại độ chín/tình trạng mít. |

|   |                                  |            |                                                                                                                  |
|---|----------------------------------|------------|------------------------------------------------------------------------------------------------------------------|
| 5 | Nhận dạng trái mít qua webcam    | Người dùng | Bật webcam, hệ thống xử lý luồng hình ảnh trực tiếp và hiển thị kết quả nhận dạng (bounding box, nhãn, xác suất) |
| 6 | Phân tích video                  | Người dùng | Tải file video lên, hệ thống bắt đầu phân tích các khung hình và nhận dạng các trái mít xuất hiện.               |
| 7 | Hỏi đáp chatbot                  | Người dùng | Gửi câu hỏi cho chatbot để nhận hướng dẫn sử dụng hoặc giải thích một số khái niệm.                              |
| 8 | So sánh hai mô hình YOLOv8       | Người dùng | Chọn hai mô hình YOLOv8, hệ thống sẽ tiến hành so sánh kết quả các thông số và xuất ra kết quả.                  |
| 9 | Xem lịch sử nhận dạng / thống kê | Người dùng | Xem lại kết quả nhận dạng trước đó với kết quả tương ứng                                                         |

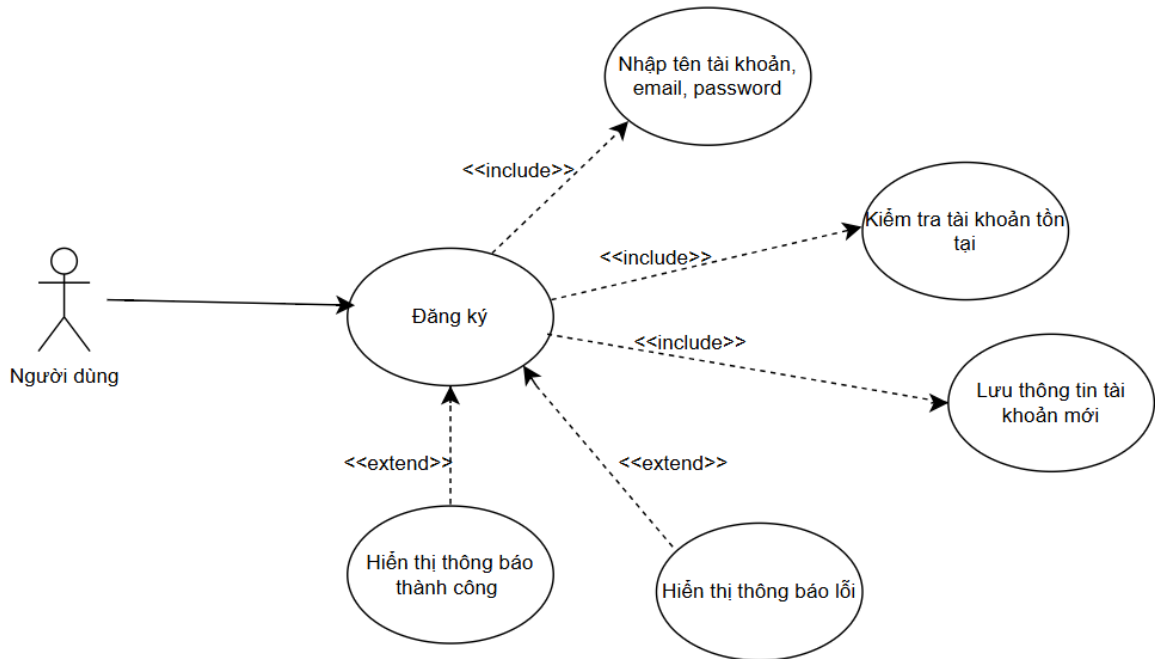
#### 2.2.3.2. Nhóm chức năng dành cho Admin

Bảng 2.3. Mô tả các chức năng của Admin

| STT | Chức năng                     | Actor | Mô tả                                                                                |
|-----|-------------------------------|-------|--------------------------------------------------------------------------------------|
| 1   | Quản lý tài khoản             | Admin | Xem danh sách tài khoản người dùng, quản lý tài khoản, thêm, sửa, xóa khi cần thiết. |
| 2   | Quản lý mô hình YOLOv8        | Admin | Quản lý cài đặt và cập nhật các mô hình YOLOv8 và các thông số khác.                 |
| 3   | Xem lịch sử dữ liệu nhận dạng | Admin | Tra cứu lịch sử nhận dạng, thống kê số lượng, quản lý dữ liệu trong database.        |

## 2.2.4. Use Case chi tiết

### 2.2.4.1. Use Case Đăng ký



Hình 2.3. Sơ đồ Use Case đăng ký chi tiết

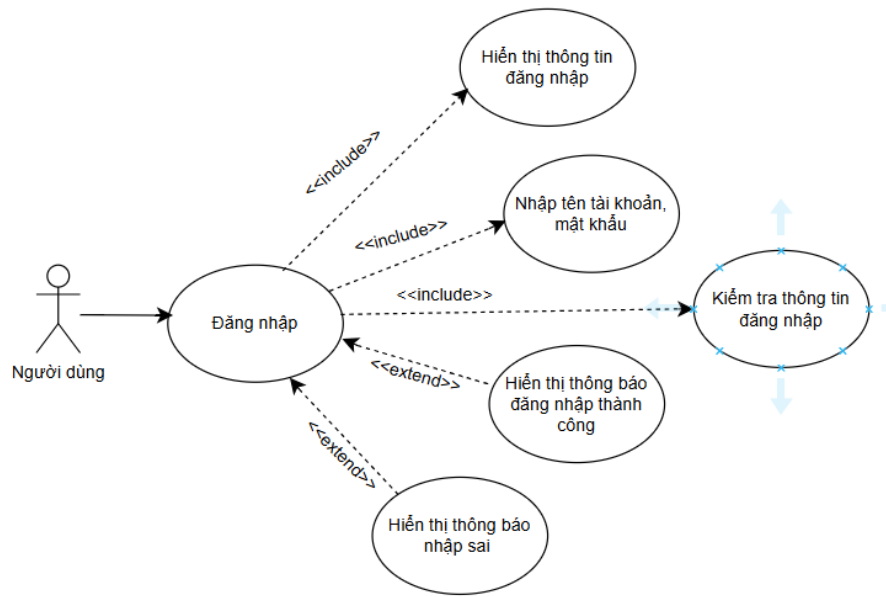
#### ❖ Đặc tả use case:

Bảng 2.4. Use Case chi tiết chức năng đăng ký

| Thành phần    | Nội dung                                                                                                                                                                                                                                                                           |
|---------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Tên Use Case  | Đăng ký tài khoản                                                                                                                                                                                                                                                                  |
| Tác nhân      | Người dùng – người chưa có tài khoản , muốn tạo tài khoản mới để sử dụng.                                                                                                                                                                                                          |
| Mô tả         | Người dùng nhập thông tin đăng ký (tên tài khoản, email, mật khẩu, xác nhận mật khẩu). Hệ thống kiểm tra tính hợp lệ và trùng lặp tên tài khoản. Nếu hợp lệ thì lưu tài khoản mới vào CSDL và thông báo đăng ký thành công; nếu không hợp lệ thì hiển thị thông báo lỗi tương ứng. |
| Điều kiện đầu | <ul style="list-style-type: none"> <li>- Người dùng chưa đăng nhập hệ thống</li> <li>- Form đăng ký sẵn sàng hiển thị</li> </ul>                                                                                                                                                   |

|                                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>Luồng sự kiện xử lý chính</p>  | <ol style="list-style-type: none"> <li>1. Người dùng mở màn hình Đăng ký.</li> <li>2. Hệ thống hiển thị form đăng ký với các trường: Tên tài khoản, Email, Mật khẩu, Xác nhận mật khẩu.</li> <li>3. Người dùng nhập đầy đủ thông tin và nhấn Đăng ký.</li> <li>4. Hệ thống kiểm tra tính hợp lệ dữ liệu (không rỗng, định dạng email, mật khẩu và xác nhận mật khẩu trùng nhau,...).</li> <li>5. Hệ thống kiểm tra trong CSDL xem tên tài khoản đã tồn tại hay chưa.</li> <li>6. Nếu dữ liệu hợp lệ và tên tài khoản chưa tồn tại, hệ thống tạo bản ghi tài khoản mới và lưu vào MongoDB Atlas (lưu username, email, mật khẩu đã mã hóa, thời gian tạo,...).</li> <li>7. Hệ thống hiển thị thông báo đăng ký thành công cho người dùng.</li> </ol> |
| <p>Luồng xử lý phụ / ngoại lệ</p> | <ul style="list-style-type: none"> <li>- Thiếu thông tin hoặc thông tin không hợp lệ (Thiếu trường dữ liệu, mật khẩu và mật khẩu xác minh không trùng khớp. Hiển thị thông báo lỗi và yêu cầu người dùng nhập lại)</li> <li>- Tên tài khoản đã tồn tại: hệ thống hiển thị thông báo “Tên tài khoản này đã tồn tại” yêu cầu tạo lại.</li> <li>- Lỗi hệ thống, không truy cập được kết nối database hoặc cơ sở dữ liệu đang bảo trì.</li> </ul>                                                                                                                                                                                                                                                                                                      |
| <p>Điều kiện cuối</p>             | <ul style="list-style-type: none"> <li>- Trường hợp thành công: một tài khoản mới được lưu trong MongoDB Atlas, người dùng có thể dùng tài khoản này để đăng nhập</li> <li>- Trường hợp thất bại: không có tài khoản nào được tạo. CSDL không bị thay đổi.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

#### 2.2.4.2. Use Case Đăng nhập



Hình 2.4. Sơ đồ use case đăng nhập chi tiết

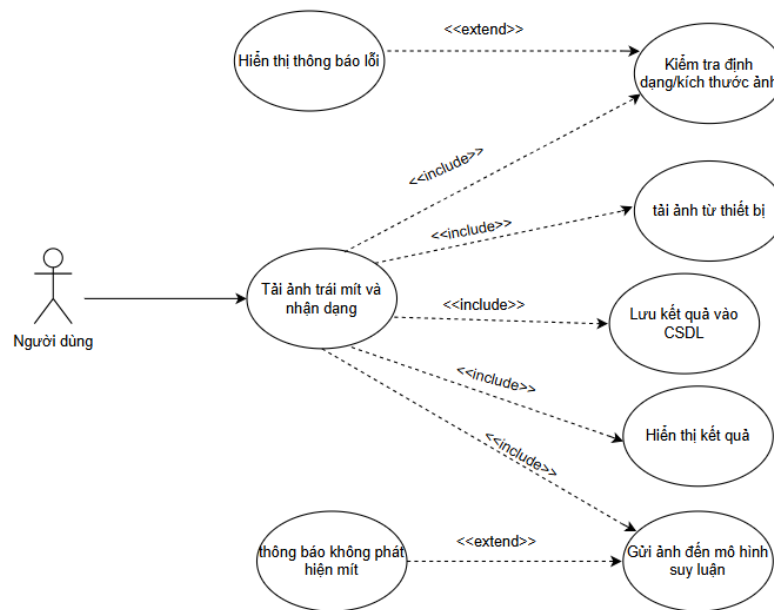
#### ❖ Đặc tả use case

Bảng 2.5. Use case chi tiết chức năng đăng nhập

| Thành phần                | Nội dung                                                                                                                                                                                                                                       |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Tên use case              | Đăng nhập hệ thống (UC_LOGIN)                                                                                                                                                                                                                  |
| Tác nhân                  | Người dùng – người đã có tài khoản trong hệ thống                                                                                                                                                                                              |
| Mô tả                     | Người dùng nhập tên tài khoản và mật khẩu. Hệ thống kiểm tra thông tin đăng nhập với dữ liệu lưu trong MongoDB Atlas. Nếu hợp lệ thì tạo phiên làm việc và chuyển người dùng vào giao diện chính; nếu không hợp lệ thì hiển thị thông báo lỗi. |
| Điều kiện đầu             | <ul style="list-style-type: none"> <li>- Hệ thống hoạt động bình thường, kết nối được đến MongoDB Atlas.</li> <li>- Người dùng đã có tài khoản (đã đăng ký trước đó).</li> <li>- Người dùng chưa đăng nhập ở phiên hiện tại.</li> </ul>        |
| Luồng sự kiện xử lý chính | <ol style="list-style-type: none"> <li>1. Người dùng mở màn hình và nhấn đăng nhập.</li> <li>2. Hệ thống hiển thị form đăng nhập với các trường: Tên tài khoản, Mật khẩu.</li> </ol>                                                           |

|                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                          | <p>3. Người dùng nhập tên tài khoản và mật khẩu, sau đó nhấn nút Đăng nhập.</p> <p>4. Hệ thống kiểm tra dữ liệu nhập vào (không để trống các trường).</p> <p>5. Hệ thống gửi thông tin đăng nhập lên backend để đối chiếu với dữ liệu tài khoản trong MongoDB Atlas (so sánh username, mật khẩu sau khi mã hóa).</p> <p>6. Nếu thông tin chính xác và tài khoản đang ở trạng thái hoạt động, hệ thống tạo phiên đăng nhập (session/token) cho người dùng.</p> <p>7. Hệ thống chuyển người dùng tới giao diện chính của ứng dụng.</p>                                             |
| Luồng xử lý phụ/ngoại lệ | <ul style="list-style-type: none"> <li>- Thiếu thông tin đăng nhập: nếu người dùng bỏ trống tên tài khoản hoặc mật khẩu, hệ thống sẽ hiển thị thông báo yêu cầu nhập đầy đủ thông tin đăng nhập.</li> <li>- Tên tài khoản, mật khẩu không chính xác: Nếu tài khoản hoặc mật khẩu không tương ứng trong CSDL, thì hệ thống sẽ thông báo “Tên tài khoản hoặc mật khẩu không chính xác” và yêu cầu người dùng nhập lại.</li> <li>- Tài khoản bị xoá bởi quản trị viên: thì người dùng không thể thực hiện đăng nhập được</li> <li>- Lỗi database khi kiểm tra tài khoản.</li> </ul> |
| Điều kiện cuối           | <ul style="list-style-type: none"> <li>- Trường hợp thành công: người dùng đăng nhập vào hệ thống, có session hợp lệ và có thể sử dụng các chức năng khác như tải ảnh, tải video, webcam, chatbot, so sánh mô hình,...</li> <li>- Trường hợp thất bại: người dùng vẫn ở màn hình đăng nhập, không tạo session, dữ liệu trong CSDL không thay đổi.</li> </ul>                                                                                                                                                                                                                     |

### 2.2.4.3. Use case tải ảnh và nhận dạng



Hình 2.5. Use case tải ảnh và nhận dạng

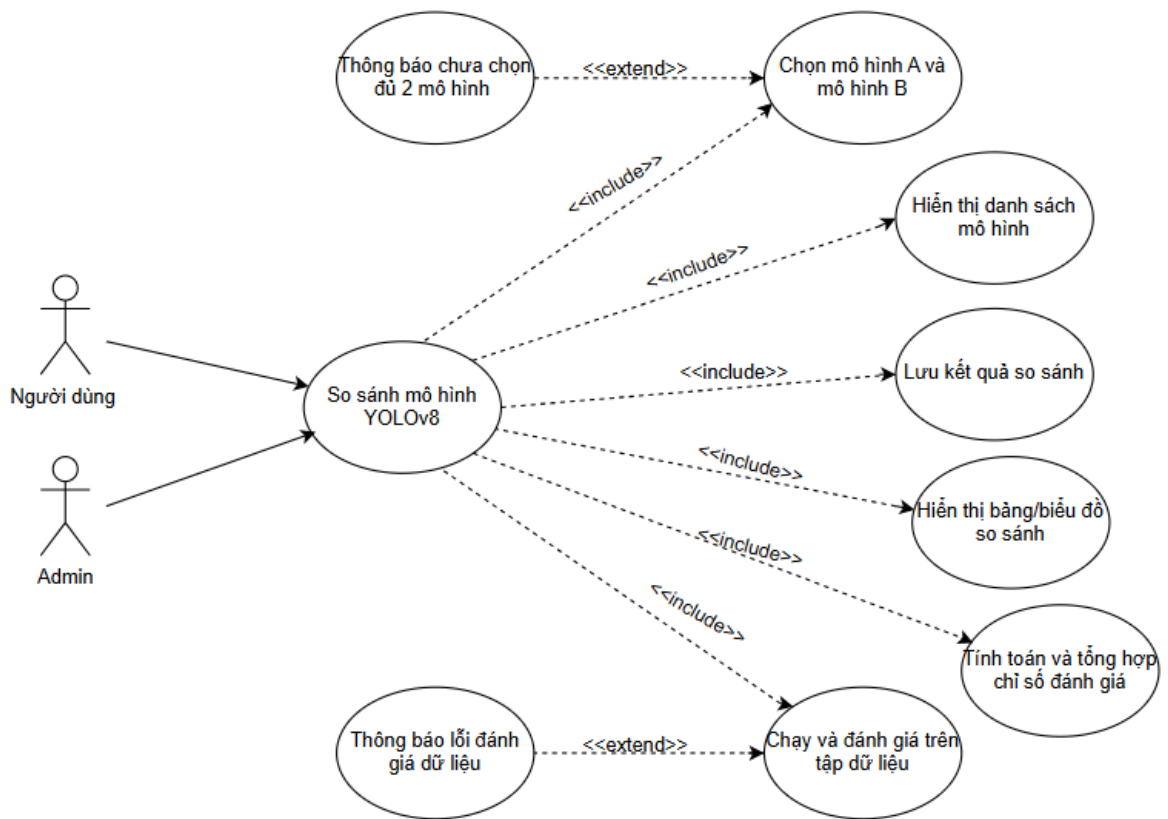
#### ❖ Đặc tả use case

Bảng 2.6. Use case chi tiết chức năng tải ảnh và nhận dạng

| Thành phần                | Nội dung                                                                                                                                                                                                                                                                |
|---------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Tên use case              | Tải ảnh trái mít và nhận dạng                                                                                                                                                                                                                                           |
| Tác nhân                  | Người dùng                                                                                                                                                                                                                                                              |
| Mô tả                     | Người dùng chọn và tải một hoặc nhiều ảnh trái mít từ thiết bị lên hệ thống. Hệ thống kiểm tra tính hợp lệ của file ảnh, gửi ảnh đến mô hình YOLOv8 để suy luận, sau đó hiển thị kết quả nhận dạng (bounding box, nhãn lớp, xác suất) và lưu kết quả vào MongoDB Atlas. |
| Điều kiện đầu             | <ul style="list-style-type: none"> <li>- Người dùng đã đăng nhập hệ thống</li> <li>- Mô hình YOLOv8 đã được cấu hình và sẵn sàng sử dụng</li> <li>- Kết nối với backend và MongoDB Atlas ổn định</li> </ul>                                                             |
| Luồng xử lý sự kiện chính | <ol style="list-style-type: none"> <li>1. Người dùng vào màn hình Nhận dạng từ ảnh.</li> <li>2. Hệ thống hiển thị nút Chọn ảnh / Upload.</li> <li>3. Người dùng chọn một hoặc nhiều file ảnh từ thiết bị và</li> </ol>                                                  |

|                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                          | <p>nhấn xác nhận tải lên.</p> <p>4. Hệ thống kiểm tra định dạng (JPG/PNG,...) và kích thước mỗi ảnh.</p> <p>5. Với các ảnh hợp lệ, frontend gửi ảnh lên backend qua API.</p> <p>6. Backend nạp mô hình YOLOv8 tương ứng và thực hiện suy luận trên từng ảnh.</p> <p>7. Backend trả về danh sách đối tượng được phát hiện trên mỗi ảnh (tọa độ bounding box, nhãn lớp, xác suất).</p> <p>8. Frontend vẽ bounding box và nhãn lên ảnh, đồng thời hiển thị bảng danh sách kết quả cho người dùng.</p> <p>9. Hệ thống lưu thông tin ảnh và kết quả nhận dạng (đường dẫn ảnh, lớp, xác suất, thời gian xử lý...) vào MongoDB Atlas.</p> |
| Luồng xử lý phụ/ngoại lệ | <ul style="list-style-type: none"> <li>- Ảnh không hợp lệ sai định dạng hoặc vượt quá dung lượng</li> <li>- Không phát hiện được trái mít có trong ảnh</li> <li>- lỗi mô hình và backend không trả về kết quả</li> <li>- lỗi kết nối CSDL không lưu được kết quả vào Database</li> </ul>                                                                                                                                                                                                                                                                                                                                           |
| Điều kiện cuối           | <ul style="list-style-type: none"> <li>- Trường hợp thành công: kết quả nhận dạng cho từng ảnh được hiển thị cho người dùng và lưu vào MongoDB Atlas.</li> <li>- Trường hợp lỗi: không nhận diện được đối tượng có trong ảnh, thông báo lỗi.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                            |

#### 2.2.4.4. Use case so sánh hai mô hình YOLOv8



Hình 2.6. Use Case chi tiết so sánh hai mô hình YOLOv8

#### ❖ Đặc tả use case

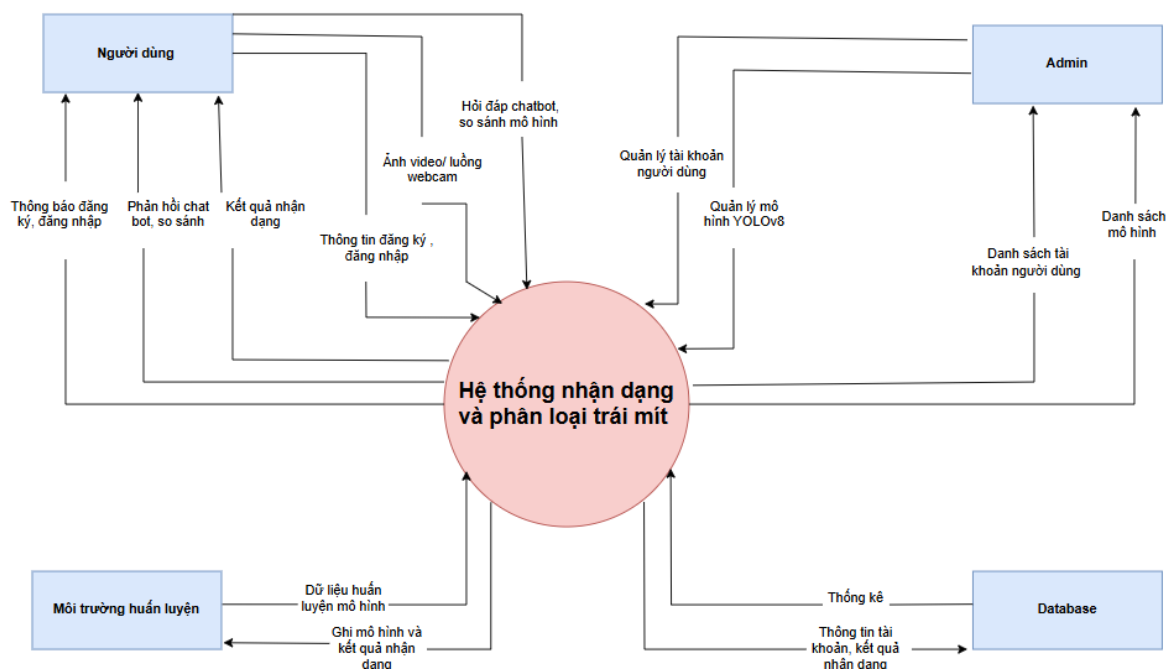
Bảng 2.7. Use case chi tiết chức năng so sánh 2 mô hình YOLOv8

| Thành phần   | Nội dung                                                                                                                                                                                                                                                                                                                            |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Tên use case | So sánh hai mô hình YOLOv8                                                                                                                                                                                                                                                                                                          |
| Tác nhân     | Người dùng, Admin                                                                                                                                                                                                                                                                                                                   |
| Mô tả        | Tác nhân chọn hai mô hình YOLOv8 (ví dụ: hai phiên bản huấn luyện khác nhau). Hệ thống tải thông tin mô hình, chạy đánh giá trên một tập dữ liệu mẫu/validation, tính toán các chỉ số (mAP, Precision, Recall, số lượng phát hiện đúng/sai,...) và hiển thị bảng/biểu đồ so sánh. Có thể lưu lại kết quả so sánh vào MongoDB Atlas. |

|                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Điều kiện đầu             | <ul style="list-style-type: none"> <li>- Tối thiểu hai mô hình YOLOv8 đã được huấn luyện và cấu hình trong hệ thống (đã lưu thông tin/đường dẫn).</li> <li>- Người dùng hoặc Admin đã đăng nhập hệ thống.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Luồng xử lý sự kiện chính | <ol style="list-style-type: none"> <li>1. Tác nhân (Người dùng hoặc Admin) truy cập màn hình So sánh mô hình.</li> <li>2. Hệ thống truy vấn danh sách mô hình YOLOv8 hiện có trong CSDL và hiển thị cho tác nhân chọn (ví dụ: YOLOv8n, YOLOv8s ...).</li> <li>3. Tác nhân chọn mô hình A và mô hình B từ danh sách.</li> <li>4. Tác nhân nhấn nút So sánh.</li> <li>5. Hệ thống tải thông tin/đường dẫn trọng số của mô hình A và mô hình B.</li> <li>6. Hệ thống sử dụng một tập dữ liệu đánh giá (validation/test hoặc bộ ảnh mẫu) và lần lượt chạy suy luận với mô hình A: <ul style="list-style-type: none"> <li>• Tính toán các chỉ số đánh giá (mAP, Precision, Recall, F1-score, ...).</li> <li>• Ghi nhận một số ví dụ dự đoán tiêu biểu (ảnh + bbox + nhãn).</li> </ul> </li> <li>7. Hệ thống thực hiện tương tự bước 6 với mô hình B trên cùng tập dữ liệu đánh giá.</li> <li>8. Hệ thống so sánh kết quả giữa hai mô hình và tổng hợp thành bảng/biểu đồ (ví dụ: so sánh mAP, Precision, Recall; số lượng TP/FP/FN).</li> <li>9. Hệ thống hiển thị kết quả so sánh cho tác nhân (bảng chỉ số + một số ảnh minh họa).</li> <li>10. Hệ thống lưu kết quả so sánh (tên mô hình, chỉ số đánh giá, thời gian thực hiện, người thực hiện) vào MongoDB Atlas.</li> </ol> |
| Luồng xử lý phụ/ngoại lệ  | <ul style="list-style-type: none"> <li>- Không chọn đủ 2 mô hình để thực hiện so sánh</li> <li>- Mô hình không tồn tại hoặc lỗi khi tải mô hình lên hệ thống</li> <li>- Lỗi trong quá trình đánh giá , hoặc mô hình đánh giá bị sai</li> <li>- lỗi ghi kết quả vào CSDL.</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Điều kiện cuối | <ul style="list-style-type: none"> <li>- Trường hợp thành công: Tác nhân xem được bảng/biểu đồ so sánh hai mô hình, có thể dựa vào đó để lựa chọn mô hình phù hợp; nếu cấu hình, kết quả so sánh được lưu vào MongoDB Atlas.</li> <li>- Trường hợp thất bại: Quá trình so sánh bị dừng ở giữa do lỗi (E1–E5), hệ thống không lưu hoặc chỉ lưu một phần log, tác nhân nhận được thông báo lỗi tương ứng.</li> </ul> |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### 2.2.5. Sơ đồ ngữ cảnh hệ thống (Context Diagram)



Hình 2.7. Sơ đồ ngữ cảnh hệ thống

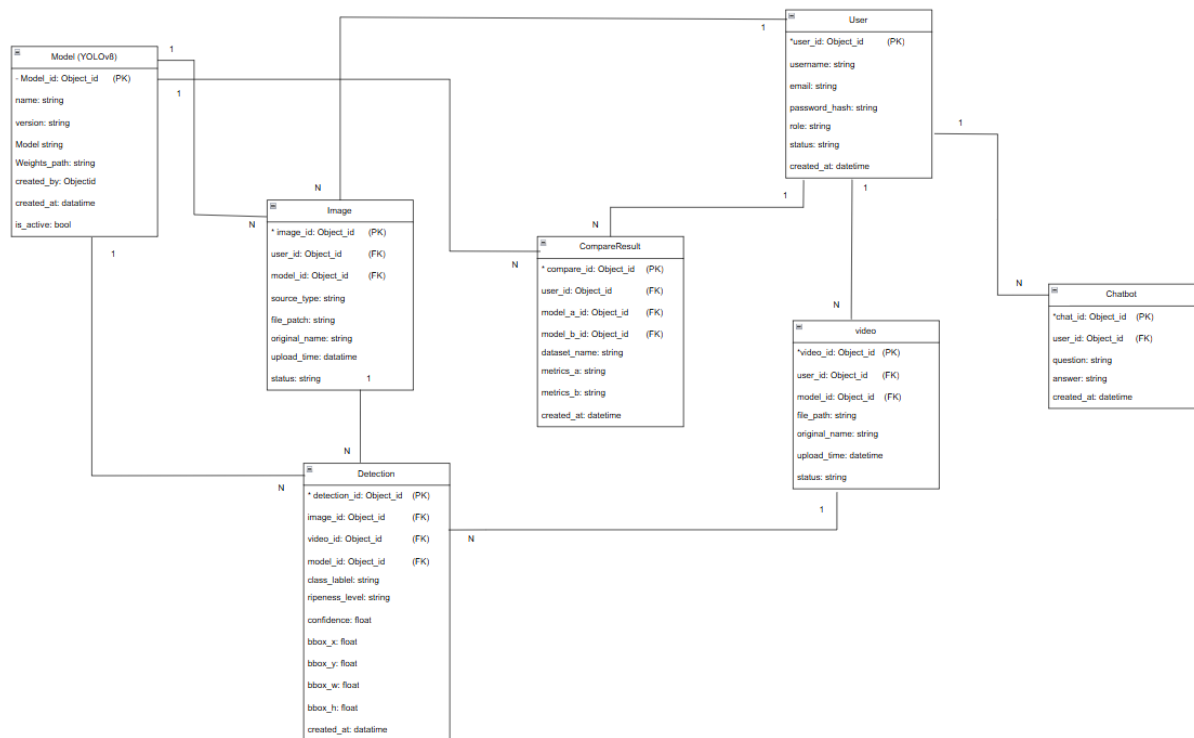
#### ❖ Mô tả sơ đồ ngữ cảnh

Bảng 2.8. Mô tả các chức năng của sơ đồ ngữ cảnh

| Thành Phần | Mô tả vai trò                                                                         | Dữ liệu trao đổi                                                                                                                                                              |
|------------|---------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Người dùng | Là đối tượng sử dụng trực tiếp hệ thống để thực hiện nhận dạng và phân loại trái mít. | <ul style="list-style-type: none"> <li>- Gửi thông tin đăng ký, đăng nhập</li> <li>- Tải ảnh, video hoặc luồng webcam</li> <li>- Nhận kết quả nhận dạng, phân loại</li> </ul> |

|                                          |                                                                                                     |                                                                                                                                                                                                                                      |
|------------------------------------------|-----------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                          |                                                                                                     | - Nhận phản hồi từ chatbot AI và kết quả so sánh mô hình                                                                                                                                                                             |
| Admin                                    | Đóng vai trò quản trị và giám sát hoạt động của hệ thống                                            | <ul style="list-style-type: none"> <li>- Quản lý tài khoản người dùng</li> <li>- Quản lý mô hình YOLOv8</li> <li>- Xem danh sách mô hình</li> <li>- Theo dõi thông tin hệ thống</li> </ul>                                           |
| Hệ thống nhận dạng và phân loại trái mít | Thành phần trung tâm, thực hiện xử lý nghiệp vụ, suy luận AI và điều phối dữ liệu giữa các tác nhân | <ul style="list-style-type: none"> <li>- Tiếp nhận dữ liệu từ người dùng và admin</li> <li>- Thực hiện nhận dạng, phân loại trái mít</li> <li>- Trả kết quả xử lý</li> <li>- Ghi nhận dữ liệu huấn luyện, thống kê và log</li> </ul> |
| Môi trường huấn luyện                    | Hỗ trợ quá trình huấn luyện và cải tiến mô hình nhận dạng                                           | <ul style="list-style-type: none"> <li>- Cung cấp dữ liệu huấn luyện cho hệ thống</li> <li>- Nhận thông tin mô hình và kết quả nhận dạng phục vụ đánh giá</li> </ul>                                                                 |
| Cơ sở dữ liệu (Database)                 | Lưu trữ thông tin và dữ liệu của toàn bộ hệ thống                                                   | <ul style="list-style-type: none"> <li>- Lưu thông tin tài khoản người dùng</li> <li>- Lưu kết quả nhận dạng, thống kê</li> <li>- Truy xuất dữ liệu phục vụ báo cáo và giám sát</li> </ul>                                           |

## 2.2.6. Sơ đồ thực thể (ERD)



Hình 2.8. Sơ đồ thực thể ERD

### ❖ Mô tả mô hình thực thể (ERD)

Mô hình ERD của hệ thống nhận dạng và phân loại độ chín trái mít được xây dựng nhằm mô tả các thực thể dữ liệu chính và mối quan hệ giữa chúng. Các thực thể trong mô hình phản ánh trực tiếp các chức năng đã phân tích ở các phần trước, bao gồm: quản lý tài khoản, quản lý mô hình YOLOv8, lưu trữ ảnh/video, kết quả nhận dạng, kết quả so sánh mô hình và lịch sử tương tác với chatbot.

#### • Thực thể user

Thực thể User lưu trữ thông tin tài khoản đăng nhập của hệ thống, gồm các thuộc tính như: user\_id, username, email, password\_hash, role, status, created\_at. Thuộc tính role dùng để phân biệt người dùng thường và người quản trị (admin). Một người dùng có thể tải lên nhiều ảnh, nhiều video, thực hiện nhiều lần so sánh mô hình và có nhiều bản ghi chat với chatbot.

#### • Thực thể Model (YOLOv8)

Thực thể Model biểu diễn các mô hình YOLOv8 được hệ thống sử dụng, với các thuộc tính tiêu biểu: model\_id, name, version, model\_type, weights\_path, description, created\_by, created\_at, is\_active. Mỗi mô hình có thể được dùng để nhận dạng cho nhiều ảnh, nhiều video và sinh ra nhiều

bản ghi kết quả nhận dạng (Detection). Ngoài ra, một mô hình cũng có thể xuất hiện trong nhiều lần so sánh mô hình khác nhau (CompareResult).

- **Thực thể Image**

Thực thể Image lưu thông tin về các ảnh được người dùng tải lên hoặc chụp từ webcam để nhận dạng, gồm các thuộc tính: image\_id, user\_id, model\_id, source\_type, file\_path, original\_name, upload\_time, status. Một người dùng (User) có thể sở hữu nhiều ảnh, và một mô hình (Model) có thể được dùng để xử lý nhiều ảnh. Mỗi ảnh có thể chứa nhiều đối tượng được phát hiện, do đó có quan hệ 1–N với thực thể Detection.

- **Thực thể video**

Thực thể Video lưu thông tin các video do người dùng tải lên để phân tích, với các thuộc tính: video\_id, user\_id, model\_id, file\_path, original\_name, upload\_time, status. Quan hệ giữa User – Video và Model – Video tương tự như với Image: một người dùng có thể tải lên nhiều video, một mô hình có thể được sử dụng để phân tích nhiều video. Mỗi video khi phân tích có thể sinh ra nhiều kết quả nhận dạng (Detection).

- **Thực thể Detection**

Thực thể Detection lưu trữ chi tiết từng đối tượng trái mít được mô hình phát hiện, với các thuộc tính như: detection\_id, image\_id, video\_id, model\_id, class\_label, ripeness\_level, confidence, bbox\_x, bbox\_y, bbox\_w, bbox\_h, created\_at. Mỗi bản ghi Detection gắn với đúng một ảnh hoặc một video, và một mô hình cụ thể. Ngược lại, một ảnh hay một video có thể có nhiều bản ghi Detection, thể hiện quan hệ 1–N giữa Image/Video với Detection.

- **Thực thể CompareResult**

Thực thể CompareResult dùng để lưu kết quả so sánh hai mô hình YOLOv8, gồm các thuộc tính: compare\_id, user\_id, model\_a\_id, model\_b\_id, dataset\_name, metrics\_a, metrics\_b, created\_at. Mỗi lần so sánh được thực hiện bởi một người dùng, nhưng có thể liên quan đến hai mô hình khác nhau (model A và model B). Về mặt logic, một mô hình có thể tham gia nhiều lần so sánh với các mô hình khác, do đó giữa Model và CompareResult là quan hệ nhiều–nhiều, được triển khai thông qua các khóa ngoại model\_a\_id và model\_b\_id trong thực thể CompareResult.

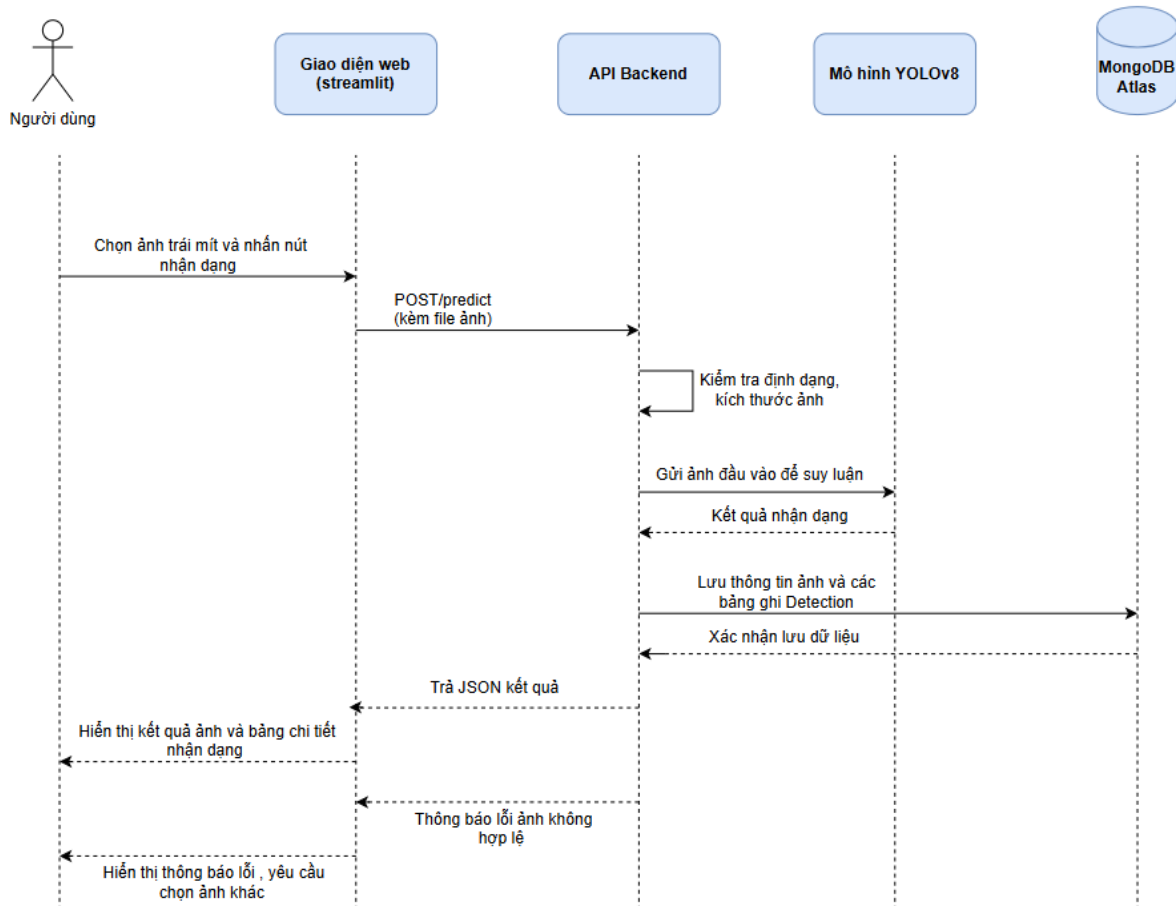
- **Thực thể chatbot**

Thực thể Chatbot (hoặc ChatHistory) lưu lịch sử tương tác giữa người dùng và chatbot, với các thuộc tính chính: chat\_id, user\_id, question,

answer, created\_at. Một người dùng có thể có nhiều bản ghi chat khác nhau, thể hiện quan hệ 1–N giữa User và Chatbot.

## 2.2.7. Sơ đồ tuần tự

### 2.2.7.1. Sơ đồ tuần tự chức năng tải ảnh trái mít và nhận dạng.



Hình 2.9. Sơ đồ tuần tự chức năng tải ảnh trái mít và nhận dạng

#### ❖ Mô tả sơ đồ tuần tự chức năng tải ảnh trái mít và nhận dạng

Sơ đồ tuần tự ở hình 3.9 mô tả chi tiết luồng tương tác giữa người dùng và các thành phần của hệ thống trong chức năng tải ảnh trái mít và nhận dạng bằng mô hình YOLOv8. Các đối tượng tham gia gồm có:

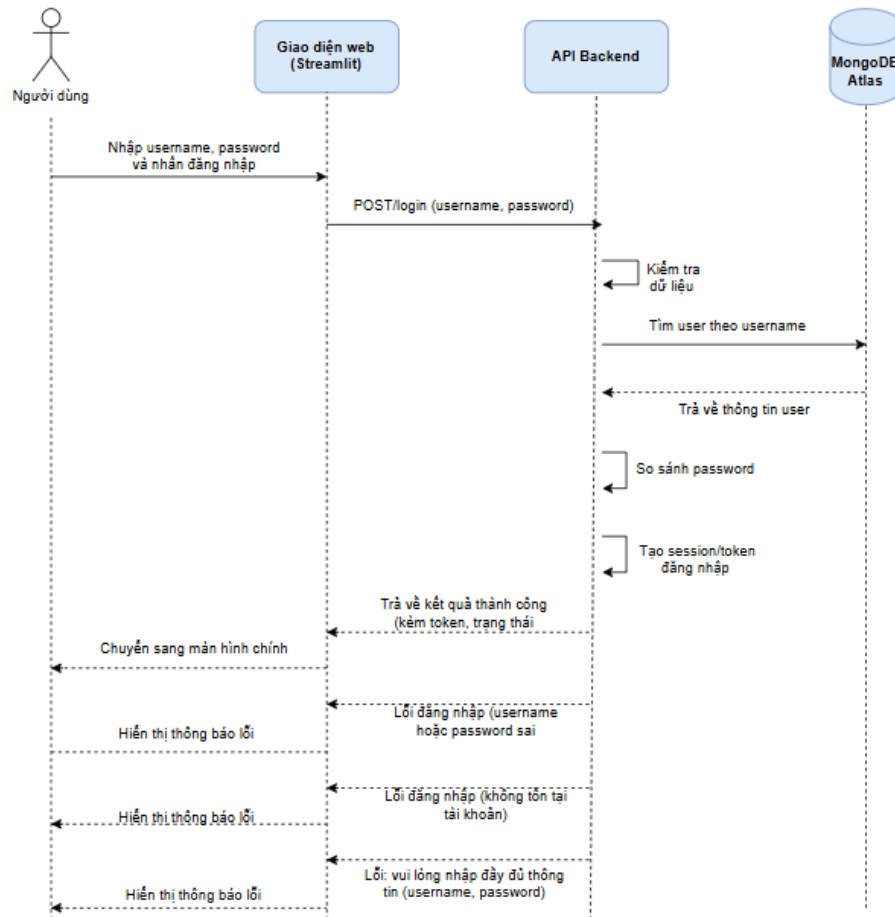
- **Người dùng:** là tác nhân khởi tạo yêu cầu nhận dạng bằng cách chọn trái mít và nhấn nút “Nhận dạng” trên giao diện web.
- **Giao diện Web (Streamlit):** thành phần frontend, tiếp nhận thao tác của người dùng, gửi yêu cầu đến backend và hiển thị kết quả trả về.
- **API Backend (FastAPI):** thành phần xử lý phía server, nhận request từ frontend, kiểm tra dữ liệu, gọi mô hình YOLOv8 để suy luận và làm việc với cơ sở dữ liệu.

- **Mô hình YOLOv8:** thành phần thực hiện suy luận (inference), nhận ảnh đầu vào và trả về danh sách các đối tượng (trái mít) được phát hiện cùng nhãn lớp và độ tin cậy.
- **MongoDB Atlas:** cơ sở dữ liệu dùng để lưu trữ thông tin ảnh và kết quả nhận dạng tương ứng.

Quy trình xử lý chính diễn ra như sau:

1. Người dùng chọn ảnh trái mít từ thiết bị và nhấn nút “Nhận dạng” trên giao diện Streamlit.
2. Giao diện Web gửi một yêu cầu HTTP (ví dụ: POST /predict) kèm file ảnh đến API Backend (FastAPI).
3. Backend kiểm tra tính hợp lệ của dữ liệu nhận được: định dạng ảnh, kích thước, các tham số cần thiết,... và thực hiện tiền xử lý ảnh nếu cần.
4. Nếu ảnh hợp lệ, backend gửi ảnh đã tiền xử lý đến mô hình YOLOv8 để thực hiện suy luận.
5. Mô hình YOLOv8 trả về cho backend danh sách các trái mít được phát hiện trên ảnh, bao gồm: nhãn lớp, độ tin cậy (confidence) và các thông số bounding box (vị trí, kích thước).
6. Backend tiến hành lưu thông tin ảnh và các bản ghi kết quả nhận dạng (Detection) tương ứng vào MongoDB Atlas.
7. Cơ sở dữ liệu xác nhận việc lưu trữ thành công, backend xây dựng cấu trúc phản hồi (JSON) chứa đường dẫn ảnh đã được vẽ bounding box kèm theo danh sách kết quả nhận dạng và gửi lại cho frontend.
8. Giao diện Web (Streamlit) nhận kết quả, hiển thị ảnh đã đánh dấu vùng trái mít cùng bảng chi tiết (nhãn, độ tin cậy, vị trí) để người dùng quan sát và đánh giá.

### 2.2.7.2. Sơ đồ tuần tự chức năng đăng nhập



Hình 2.10. Sơ đồ tuần tự chức năng đăng nhập

#### ❖ Mô tả sơ đồ tuần tự chức năng đăng nhập

Sơ đồ tuần tự ở Hình 3.10 mô tả luồng xử lý của chức năng Đăng nhập hệ thống. Các đối tượng tham gia gồm:

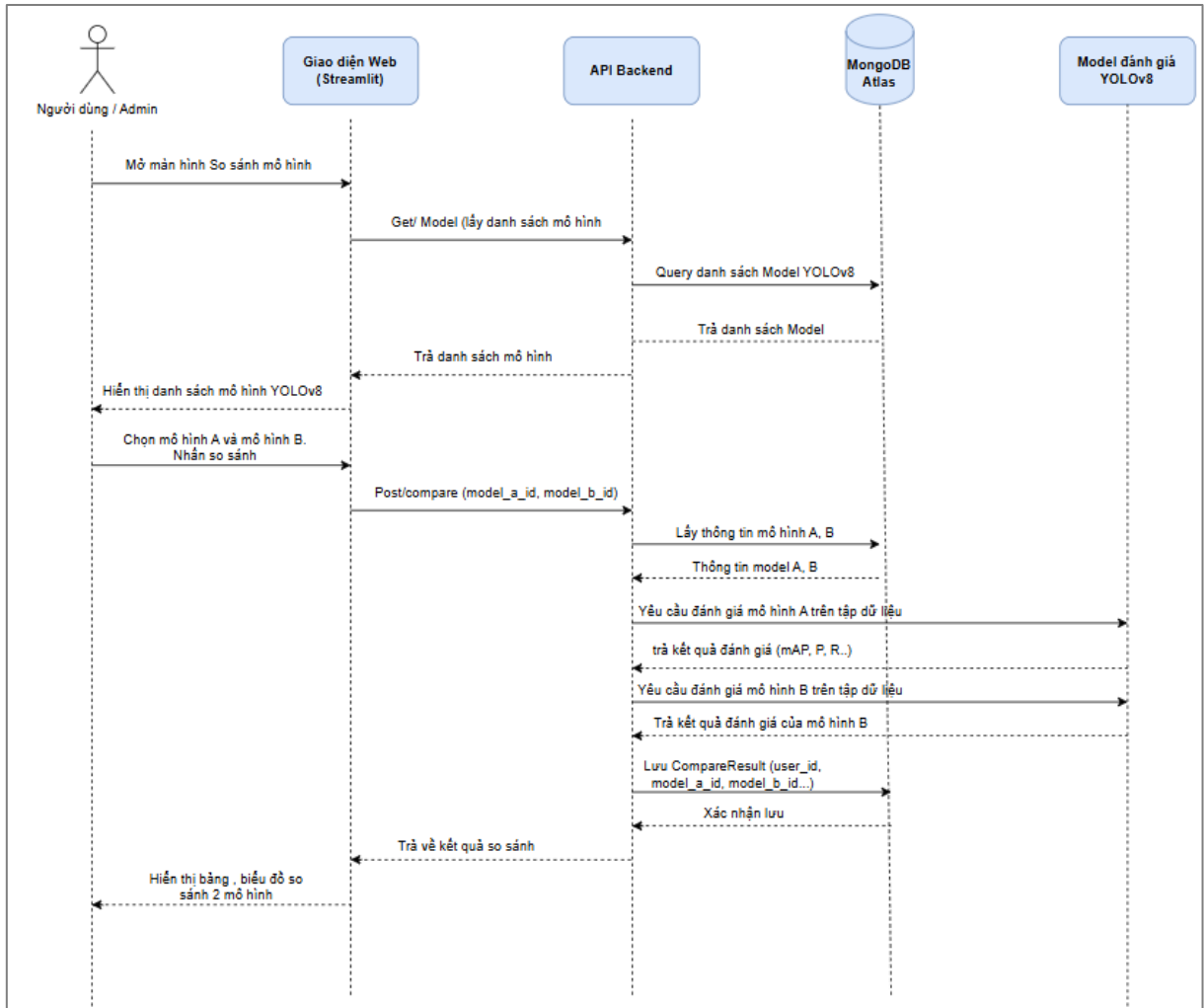
- **Người dùng:** nhập thông tin tài khoản để đăng nhập.
- **Giao diện Web (Streamlit):** nhận dữ liệu đăng nhập từ người dùng và gửi tới backend.
- **API Backend (FastAPI):** kiểm tra thông tin đăng nhập, xác thực tài khoản.
- **MongoDB Atlas:** lưu trữ và cung cấp dữ liệu tài khoản người dùng.

Quy trình xử lý chính được mô tả như sau:

1. Người dùng truy cập trang Đăng nhập, nhập tên tài khoản (username) và mật khẩu, sau đó nhấn nút “Đăng nhập” trên giao diện Streamlit.

2. Giao diện gửi yêu cầu POST /login lên API backend, kèm theo username và password mà người dùng vừa nhập.
3. Backend kiểm tra tính hợp lệ của dữ liệu đầu vào (ví dụ: không để trống username hoặc password). Nếu dữ liệu thiếu hoặc không hợp lệ, backend trả về thông báo lỗi cho giao diện, yêu cầu người dùng nhập lại.
4. Nếu dữ liệu đầu vào hợp lệ, backend gửi truy vấn đến MongoDB Atlas để tìm kiếm tài khoản có username tương ứng.
5. Cơ sở dữ liệu trả về thông tin tài khoản (nếu tồn tại) hoặc null (nếu không tìm thấy).
6. Trường hợp không tồn tại tài khoản với username đã nhập, backend trả về thông báo lỗi (ví dụ: “Tài khoản không tồn tại”) cho giao diện để hiển thị cho người dùng.
7. Nếu tài khoản tồn tại, backend tiến hành so sánh mật khẩu người dùng nhập với mật khẩu đã mã hóa (password\_hash) lưu trong cơ sở dữ liệu, đồng thời kiểm tra trạng thái tài khoản (active/locked).
8. Nếu mật khẩu đúng và tài khoản đang ở trạng thái hoạt động, backend tạo phiên đăng nhập (session/token) cho người dùng và trả về kết quả đăng nhập thành công cho giao diện. Giao diện sau đó chuyển người dùng sang màn hình chính của hệ thống (ví dụ: trang nhận dạng trái mít).
9. Nếu mật khẩu không đúng hoặc tài khoản đang bị khóa, backend trả về thông báo lỗi đăng nhập (ví dụ: “Tên tài khoản hoặc mật khẩu không chính xác” hoặc “Tài khoản đang bị khóa”), giao diện hiển thị thông báo tương ứng để người dùng biết và thao tác lại.

### 2.2.7.3. Sơ đồ tuần tự chức năng so sánh hai mô hình



Hình 2.11. Sơ đồ tuần tự chức năng so sánh hai mô hình

#### ❖ Mô tả sơ đồ tuần tự chức năng so sánh hai mô hình

Sơ đồ tuần tự ở Hình 3.x mô tả luồng tương tác giữa các thành phần của hệ thống trong chức năng so sánh hai mô hình YOLOv8. Các đối tượng tham gia gồm:

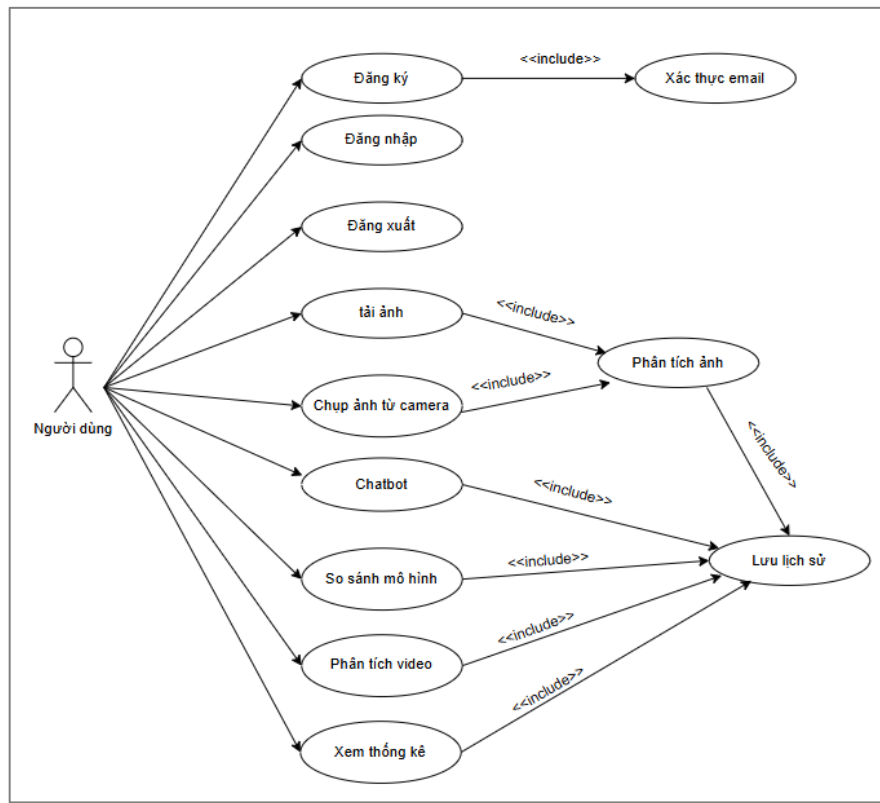
- **Người dùng / Admin:** là tác nhân khởi tạo yêu cầu so sánh.
- **Giao diện Web (Streamlit):** nhận thao tác lựa chọn mô hình từ người dùng và gửi yêu cầu đến backend.
- **API Backend (FastAPI):** xử lý logic so sánh, tương tác với cơ sở dữ liệu và module đánh giá.
- **MongoDB Atlas:** lưu trữ thông tin mô hình và lưu lại kết quả so sánh.
- **Module đánh giá YOLOv8:** thực hiện việc đánh giá chất lượng mô hình trên tập dữ liệu kiểm thử/validation.

Quy trình hoạt động chính diễn ra như sau:

1. Người dùng hoặc Admin mở màn hình “So sánh mô hình” trên giao diện web. Giao diện Streamlit gửi yêu cầu đến API backend để lấy danh sách các mô hình YOLOv8 hiện có trong hệ thống.
2. Backend truy vấn MongoDB Atlas để đọc danh sách mô hình (tên mô hình, phiên bản, loại YOLOv8,...) và trả lại cho giao diện. Giao diện hiển thị danh sách này để người dùng lựa chọn.
3. Người dùng chọn mô hình A và mô hình B, sau đó nhấn nút “So sánh”. Giao diện gửi yêu cầu POST /compare lên backend, kèm theo model\_a\_id và model\_b\_id tương ứng.
4. Backend sử dụng các ID này để truy vấn MongoDB Atlas, lấy thông tin chi tiết của hai mô hình cần so sánh.
5. Backend gọi module đánh giá YOLOv8, yêu cầu thực hiện đánh giá mô hình A trên một tập dữ liệu đánh giá (validation/test hoặc tập dữ liệu mẫu). Module đánh giá trả lại các chỉ số như mAP, Precision, Recall, F1-score,... cho mô hình A.
6. Backend tiếp tục yêu cầu module đánh giá thực hiện tương tự với mô hình B trên cùng tập dữ liệu để đảm bảo tính công bằng khi so sánh. Kết quả đánh giá của mô hình B cũng được trả về cho backend.
7. Sau khi nhận kết quả đánh giá của cả hai mô hình, backend tổng hợp các chỉ số lại, so sánh và tạo thành một bản ghi CompareResult (chứa user thực hiện, model A, model B, tên tập dữ liệu, các chỉ số đánh giá). Bản ghi này được lưu xuống MongoDB Atlas.
8. Cuối cùng, backend trả về cho giao diện web dữ liệu so sánh (bảng chỉ số và/hoặc dữ liệu để vẽ biểu đồ). Giao diện hiển thị kết quả cho Người dùng/Admin, giúp dễ dàng quan sát mô hình nào có hiệu năng tốt hơn trên tập dữ liệu đánh giá.

## 2.3. Phân tích và thiết kế ứng dụng di động AgriVision Mobile

### 2.3.1. Sơ đồ use case tổng quát cho ứng dụng di động



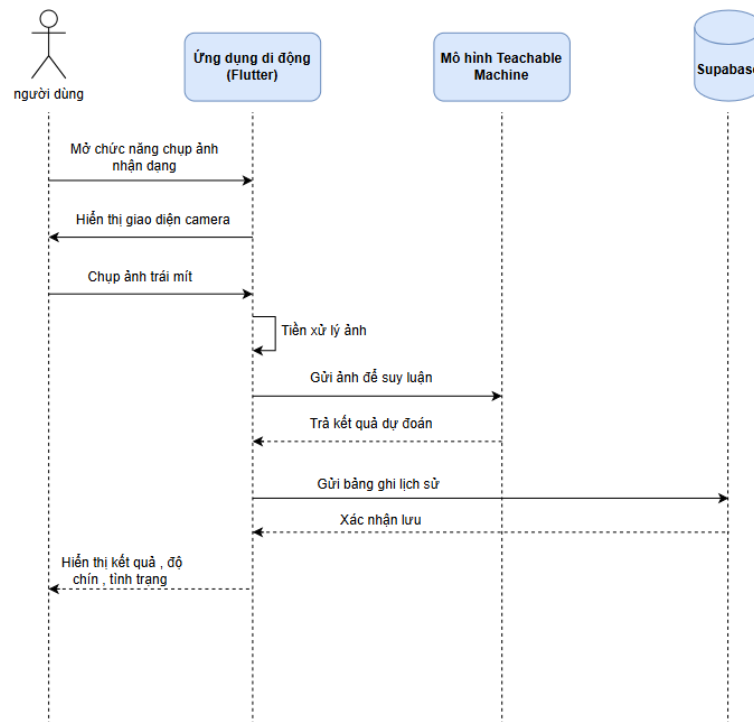
Hình 2.12. Sơ đồ Use case tổng quát ứng dụng di động Mobile App

#### ❖ Mô tả use case

- **Đăng ký:** Cho phép người dùng tạo tài khoản mới bằng cách nhập địa chỉ email, mật khẩu và xác nhận mật khẩu. Sau khi đăng ký hệ thống sẽ gửi về email để xác thực tài khoản.
- **Xác thực email:** Người dùng mở email, nhấn vào liên kết xác nhận. Hệ thống kích hoạt tài khoản sau khi xác thực thành công.
- **Đăng nhập:** Người dùng nhập tên đăng nhập và mật khẩu để truy cập vào ứng dụng. Hệ thống kiểm tra thông tin, nếu đúng sẽ cho phép vào màn hình chính.
- **Đăng xuất:** Kết thúc phiên làm việc hiện tại, xóa thông tin đăng nhập trên thiết bị và đưa người dùng về màn hình đăng nhập.
- **Tải ảnh:** Người dùng chọn ảnh trái mít từ thư viện trên điện thoại. Ảnh sau đó được gửi cho chức năng phân tích ảnh.

- **Chụp ảnh từ camera:** Người dùng mở camera trong ứng dụng và chụp ảnh trái mít. Ảnh chụp được đưa vào chức năng phân tích ảnh.
- **Phân tích ảnh:** Ứng dụng sử dụng mô hình học máy để nhận dạng và phân loại độ chín/tình trạng trái mít trên ảnh. Kết quả được hiển thị và ghi lại vào lịch sử.
- **Phân tích video:** Người dùng chọn video cần phân tích. Ứng dụng xử lý từng khung hình, phát hiện các trái mít và phân loại, đồng thời lưu kết quả vào lịch sử.
- **So sánh mô hình:** Cho phép người dùng chọn hoặc tải các mô hình khác nhau (ví dụ các phiên bản YOLOv8 trên app), thực hiện đánh giá trên cùng một tập dữ liệu và hiển thị kết quả so sánh, sau đó lưu lại vào lịch sử.
- **Chatbot:** Người dùng nhập câu hỏi vào chatbot. Hệ thống trả lời các câu hỏi liên quan đến cách sử dụng ứng dụng, giải thích kết quả phân tích hoặc kiến thức cơ bản về nhận dạng trái mít; nội dung trao đổi được lưu vào lịch sử.
- **Thống kê:** Ứng dụng tổng hợp các kết quả đã lưu (số lần nhận dạng, tỉ lệ các mức độ chín, tần suất sử dụng chức năng...) và hiển thị dưới dạng thống kê, đồ thị cơ bản. Các lần xem thống kê cũng được ghi nhận vào lịch sử.
- **Lưu lịch sử:** Sau mỗi thao tác quan trọng (phân tích ảnh, phân tích video, so sánh mô hình, dùng chatbot, xem thống kê...), hệ thống tự động tạo bản ghi lịch sử chứa thời gian, loại thao tác và kết quả tương ứng.

### 2.3.2. Sơ đồ tuần tự chức năng chụp ảnh nhận dạng trái mít



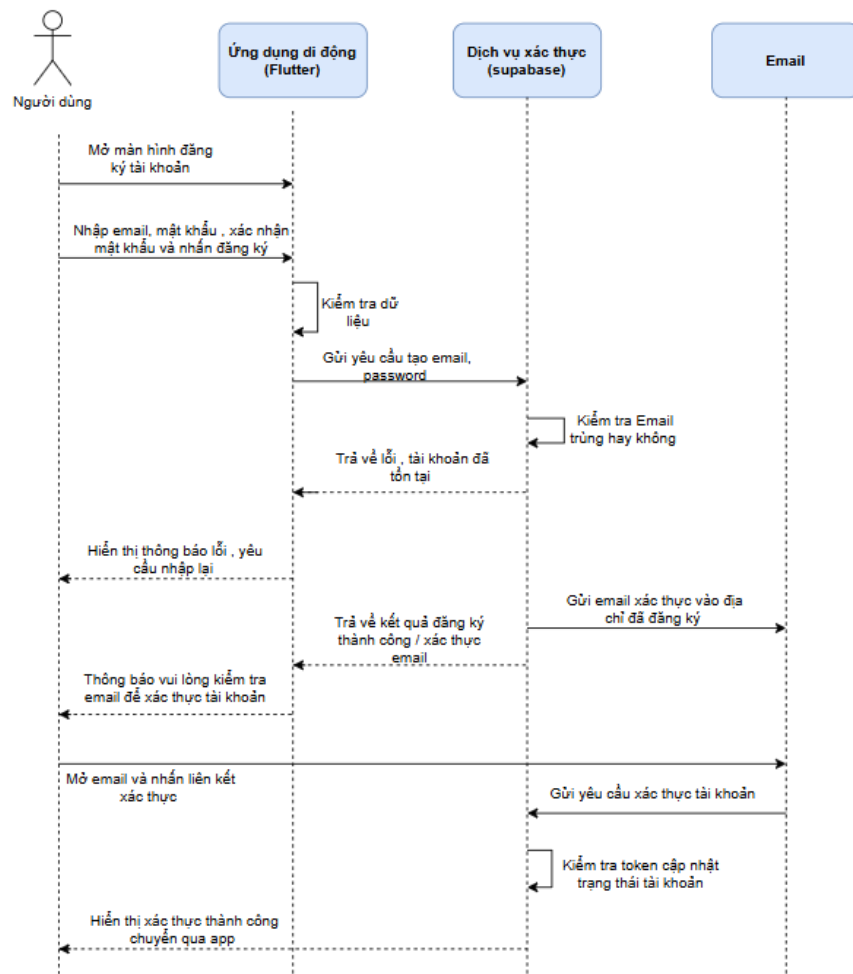
Hình 2.13. Sơ đồ tuần tự chức năng chụp ảnh nhận dạng trái mít

#### ❖ Giới thiệu về sơ đồ

Sơ đồ tuần tự ở Hình 3.13 mô tả luồng xử lý chính của chức năng chụp ảnh và phân tích trái mít trên ứng dụng di động AgriVision Mobile. Người dùng mở chức năng chụp ảnh trong app, giao diện Flutter hiển thị camera để người dùng chụp lại hình ảnh trái mít. Sau khi nhận được ảnh, ứng dụng thực hiện tiền xử lý (thay đổi kích thước, chuẩn hóa dữ liệu) và gửi ảnh đến mô hình phân loại được huấn luyện bằng Teachable Machine để suy luận.

Mô hình trả về kết quả dự đoán gồm nhãn lớp (ví dụ: mít non, mít chín, mít sâu bệnh) và độ tin cậy. Ứng dụng hiển thị kết quả này cho người dùng, đồng thời gửi thông tin ảnh, nhãn dự đoán, độ tin cậy và thời gian thực hiện lên Supabase để lưu thành một bản ghi lịch sử. Nhờ đó, người dùng có thể xem lại các lần nhận dạng đã thực hiện trên ứng dụng trong các phiên làm việc sau.

### 2.3.3. Sơ đồ tuần tự chức năng đăng ký tài khoản

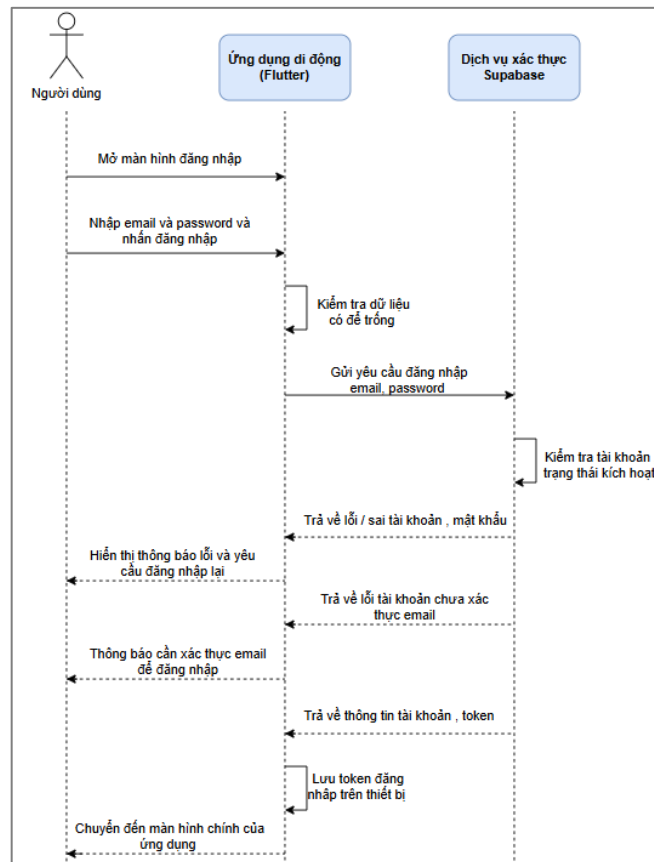


Hình 2.14. Sơ đồ tuần tự chức năng đăng ký của App Mobile

#### ❖ Giới thiệu về sơ đồ

Sơ đồ tuần tự trên mô tả luồng xử lý chức năng Đăng ký tài khoản trên ứng dụng di động AgriVision Mobile. Người dùng nhập thông tin đăng ký (tên đăng nhập, email, mật khẩu, xác nhận mật khẩu) trên ứng dụng Flutter, sau đó app gửi yêu cầu tạo tài khoản lên dịch vụ xác thực (Supabase/Auth). Dịch vụ kiểm tra trùng tên đăng nhập/email, nếu đã tồn tại thì trả về lỗi để app hiển thị thông báo cho người dùng. Nếu đăng ký hợp lệ, dịch vụ sẽ tạo tài khoản mới và gửi email xác thực thông qua mail server. Người dùng mở email, nhấn vào liên kết xác thực; dịch vụ xác thực kiểm tra token, cập nhật trạng thái tài khoản sang “đã kích hoạt” và thông báo xác thực thành công. Sau bước này, người dùng có thể quay lại app và đăng nhập bình thường.

#### 2.3.4. Sơ đồ tuần tự chức năng đăng nhập tài khoản

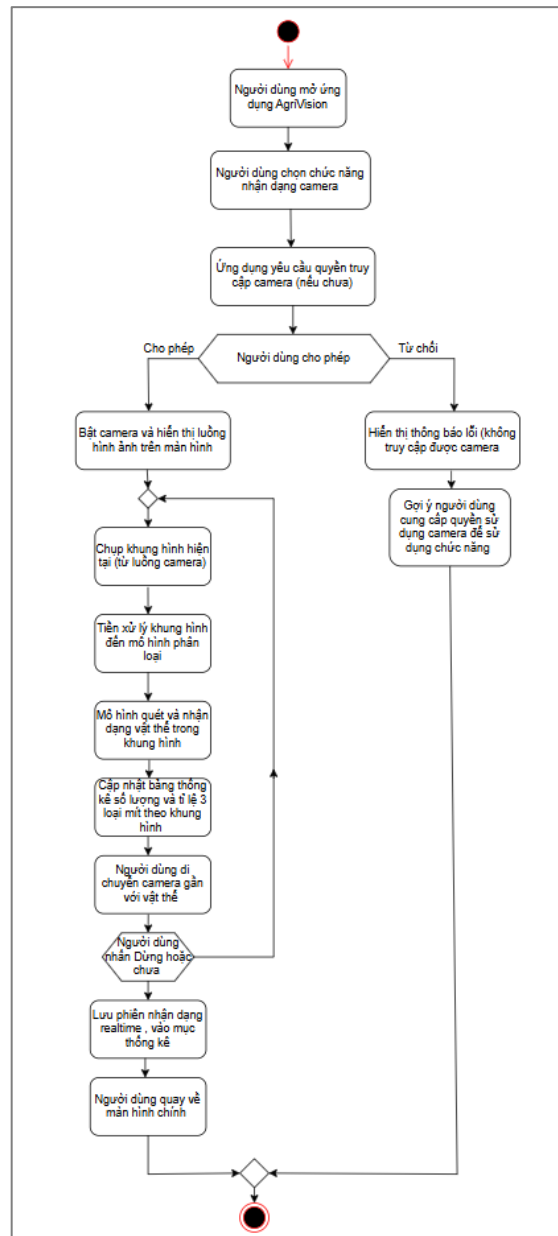


Hình 2.15. Sơ đồ tuần tự chức năng đăng nhập tài khoản của App Mobile

#### ❖ Giới thiệu về sơ đồ

Sơ đồ tuần tự ở Hình 3.15 mô tả luồng xử lý chức năng đăng nhập trên ứng dụng di động AgriVision Mobile. Người dùng mở màn hình đăng nhập, nhập tên đăng nhập (hoặc email) và mật khẩu, sau đó ứng dụng kiểm tra dữ liệu đầu vào và gửi yêu cầu xác thực tới dịch vụ Supabase/Auth. Dịch vụ kiểm tra thông tin tài khoản, trạng thái kích hoạt và mật khẩu. Nếu thông tin sai, hệ thống trả về thông báo lỗi để người dùng nhập lại; nếu tài khoản chưa xác thực email, hệ thống yêu cầu người dùng hoàn tất bước xác thực. Trường hợp đăng nhập thành công, dịch vụ trả về token phiên làm việc và thông tin người dùng, ứng dụng lưu token trên thiết bị và chuyển người dùng tới màn hình chính.

#### 2.3.5. Sơ đồ hoạt động chức năng nhận dạng qua Camera



Hình 2.16. Sơ đồ hoạt động chức năng nhận dạng qua Camera trên App Mobile

### ❖ Giới thiệu về sơ đồ

Sơ đồ hoạt động ở Hình 3.16 mô tả luồng xử lý chức năng nhận dạng trái mít qua camera trên ứng dụng di động AgriVision Mobile. Người dùng chọn chức năng “Nhận dạng qua camera”, ứng dụng yêu cầu quyền truy cập camera và bật luồng hình ảnh trực tiếp trên màn hình. Ứng dụng liên tục lấy các khung hình hiện tại, thực hiện tiền xử lý và gửi tới mô hình phân loại được huấn luyện bằng Teachable Machine, đồng thời bảng thống kê bên dưới được cập nhật liên tục theo từng khung hình. Khi người dùng nhấn dừng, ứng dụng lưu lại thông tin phiên nhận dạng vào lịch sử và cho phép quay về màn hình chính.

## **Chương 3. TRIỂN KHAI HỆ THỐNG VÀ KẾT QUẢ ĐẠT ĐƯỢC**

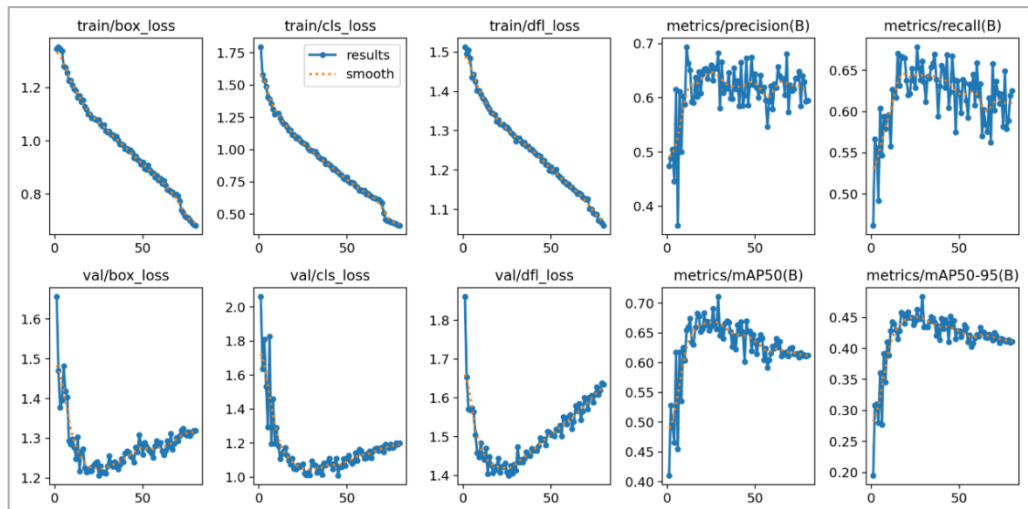
### **3.1. Cấu hình hệ thống**

#### **3.1.1. Môi trường phát triển cục bộ**

- Hệ điều hành: Windows 10/11 64-bit
- CPU: Intel Core i5 hoặc tương đương
- RAM: Tối thiểu 8GB
- Ngôn ngữ lập trình: Python version 3.11
- IDE: Visual Studio Code
- Môi trường ảo: venv
- Thư viện chính sử dụng:
  - Ultralytics (YOLOv8)
  - Opencv-python
  - Numpy, pandas
  - Streamlit (frontend)
  - Fastapi (backend API)
  - Uvicorn (chạy server FastAPI)
  - Pymongo (kết nối MongoDB Atlas)
  - Python-dotenv (quản lý biến môi trường)
- Trình duyệt web: Google Chrome

#### **3.1.2. Môi trường huấn luyện mô hình**

- Nền tảng: Google Colab
- Phần cứng: GPU trên Colab
- Dung lượng RAM: ~12–16 GB
- Bộ dữ liệu: Tập ảnh trái mít đã gắn nhãn trên Roboflow, chia thành các tập train, val, test.
- Thư viện:
  - Ultralytics
  - Torch, torchvision
  - Numpy, pandas, matplotlib, seaborn

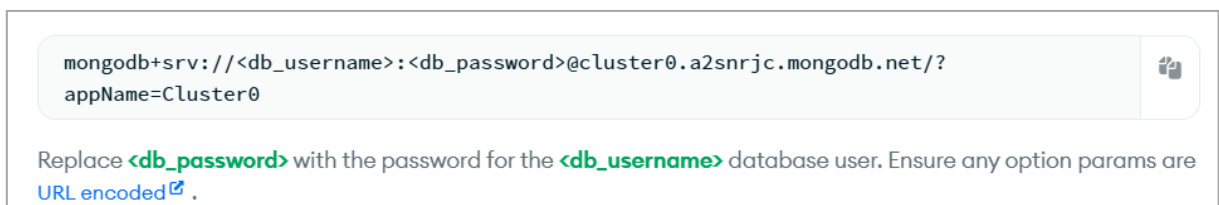


Hình 3.1. Hình ảnh Google Colab khi đã train xong mô hình YOLOv8

### 3.1.3. Cấu hình cơ sở dữ liệu

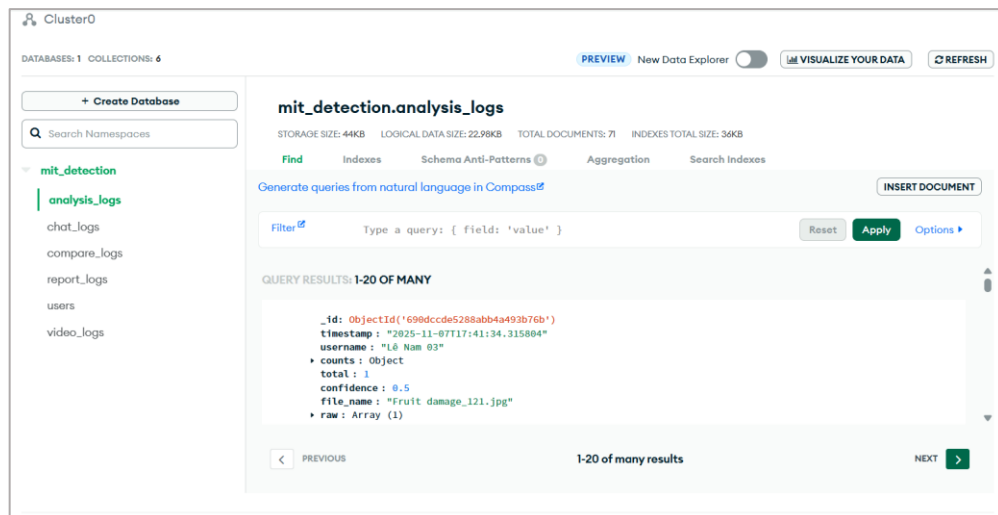
- Hệ quản trị CSDL: MongoDB Atlas
- Cluster: 1 cluster miễn phí, 1 replica primary
- Database: mit\_detection.
- Các collection chính:
  - Users: lưu thông tin tài khoản người dùng
  - Analysis\_logs: lưu kết quả phân tích ảnh
  - Chat\_logs: lưu lịch sử hội thoại với chatbot
  - Video\_logs: lưu thông tin video tải lên
  - Report\_logs: lưu kết quả so sánh model

Kết nối tới MongoDB Atlas được cấu hình thông qua chuỗi kết nối (connection string)



Hình 3.2. Chuỗi mã connection string của MongoDB Atlas

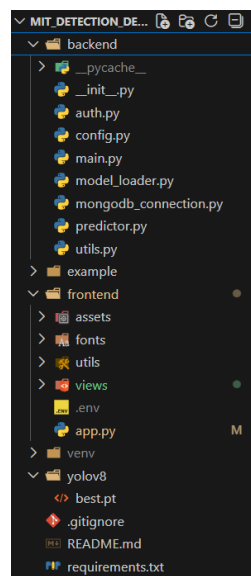
Chuỗi này được đặt trong file cấu hình .env để đảm bảo an toàn dữ liệu.



Hình 3.3. Các collection chính của hệ thống

## 3.2. Quy trình cài đặt và chạy hệ thống

### 3.2.1. Cấu trúc thư mục chương trình



Hình 3.4. Cấu trúc thư mục chương trình

### 3.2.2. Cài đặt môi trường và các thư viện

- Cài đặt Python 3.11 và Visual Studio Code
- Mở thư mục dự án bằng VS Code
- Tại thư mục gốc dự án, tạo môi trường ảo:

**python -m venv venv**

- Kích hoạt môi trường ảo:

**venv\Scripts\activate**

- Cài đặt các thư viện cần thiết (dựa trên file requirement.txt)

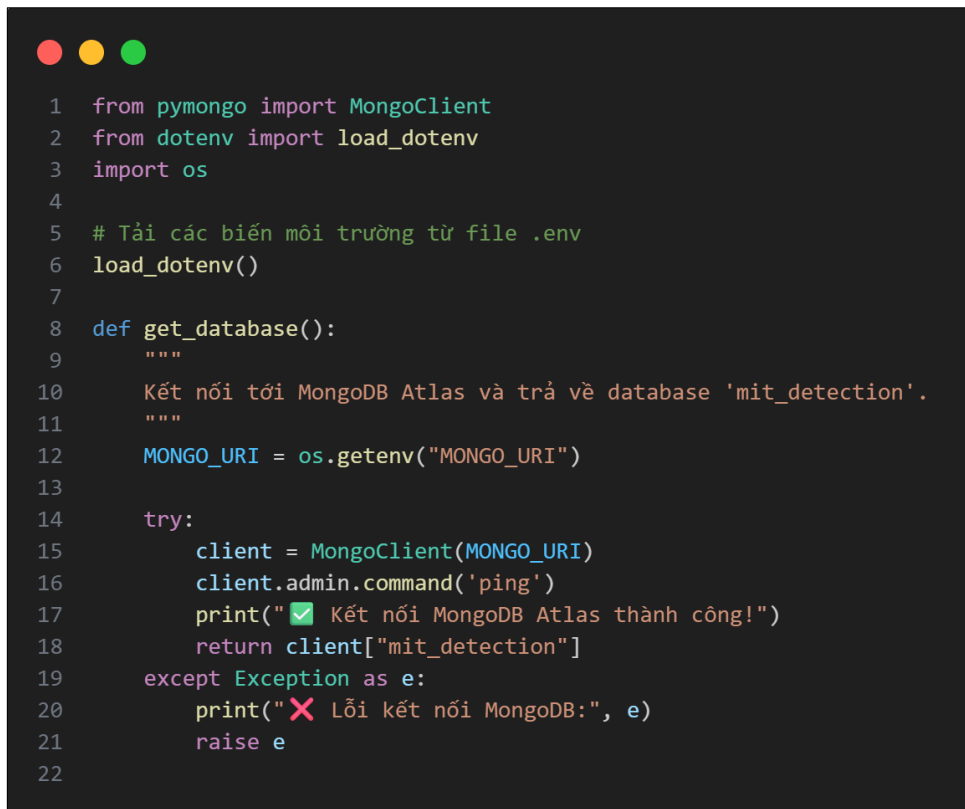
**pip install -r requirements.txt**

### 3.2.3. Cấu hình kết nối cơ sở dữ liệu và tham số hệ thống

Tại file .env của dự án, lưu các thông số của CSDL và các tham số khác:

- MONGODB\_URI=mongodb+srv://<user>:<password>@<cluster-url>/jackfruit\_db
- DB = client[mit\_detection]

Trong mã nguồn backend chương trình đọc giá trị này để kết nối MongoDB Atlas thông qua thư viện pymongo

A screenshot of a code editor with a dark background and light-colored text. The code is in Python and is used to connect to MongoDB Atlas. It imports pymongo and dotenv, loads environment variables from a .env file, and defines a function get\_database() that attempts to connect to the MongoDB Atlas instance using the MONGODB\_URI from the environment. The function prints a success message with a green checkmark or an error message with a red X. The code is numbered from 1 to 22.

```
1 from pymongo import MongoClient
2 from dotenv import load_dotenv
3 import os
4
5 # Tải các biến môi trường từ file .env
6 load_dotenv()
7
8 def get_database():
9     """
10     Kết nối tới MongoDB Atlas và trả về database 'mit_detection'.
11     """
12     MONGO_URI = os.getenv("MONGO_URI")
13
14     try:
15         client = MongoClient(MONGO_URI)
16         client.admin.command('ping')
17         print("✅ Kết nối MongoDB Atlas thành công!")
18         return client["mit_detection"]
19     except Exception as e:
20         print("❌ Lỗi kết nối MongoDB:", e)
21         raise e
22
```

Hình 3.5. Hình ảnh đoạn mã đọc biến môi trường khởi tạo kết nối MongoDB

### 3.2.4. Khởi tạo Backend

Tại thư mục gốc dự án (khi đã kích hoạt môi trường ảo), chạy lệnh:

**uvicorn backend.main:app --reload**

- Backend.main:app là đường dẫn đến đối tượng ứng dụng FastAPI.
- Tham số --reload giúp server tự động tải lại khi có thay đổi mã nguồn.

Sau khi khởi chạy thành công, backend lắng nghe tại địa chỉ:

**http://127.0.0.1:8000**

### 3.2.5. Khởi chạy frontend

Mở một cửa sổ terminal mới (vẫn ở trong môi trường ảo) , thực thi lệnh:

**streamlit run frontend/app.py**

Sau khi chạy, Streamlit sẽ tự động mở trình duyệt tại địa chỉ:

**http://localhost:8501**

## 3.3. Kết quả đạt được của website

### 3.3.1. Giao diện trang chủ của Website



Hình 3.6. Giao diện trang chủ chính của website

Hình ảnh minh họa giao diện và luồng hoạt động của hệ thống AgriVision, cho phép người dùng tải ảnh hoặc sử dụng camera để hệ thống AI tự động phân tích, nhận dạng và cung cấp thông tin hỗ trợ trong lĩnh vực nông nghiệp thông minh.

### 3.3.2. Giao diện đăng ký và đăng nhập

The screenshot shows the 'Đăng ký tài khoản mới' (New Account Registration) page of the Agri Vision system. The page has a light green and white color scheme. On the left is a sidebar menu with icons and labels for 'Trang chủ' (Home), 'Đăng nhập' (Login), 'Phân tích ảnh' (Image Analysis), 'Video/Webcam' (Video/Webcam), 'Thống kê' (Statistics), 'So sánh YOLOv8' (Compare YOLOv8), 'Chat AgriVision', and 'Tài khoản' (Account). The main content area features the title 'AGRI VISION — HỆ THỐNG NHẬN DẠNG VÀ PHÂN LOẠI ĐỘ CHÍN TRÁI MÍT' and the subtitle 'Ứng dụng AI phục vụ Nông nghiệp Thông minh'. Below this, there are tabs for 'Đăng nhập' and 'Đăng ký', with 'Đăng ký' being the active tab. The registration form includes fields for 'Tên người dùng' (Username), 'Địa chỉ email' (Email), 'Mật khẩu' (Password), and 'Xác nhận mật khẩu' (Confirm Password). A 'Tạo tài khoản' (Create Account) button is at the bottom right. An illustration of a person standing next to a green bar is on the left side of the form.

Hình 3.7. Giao diện đăng ký của website

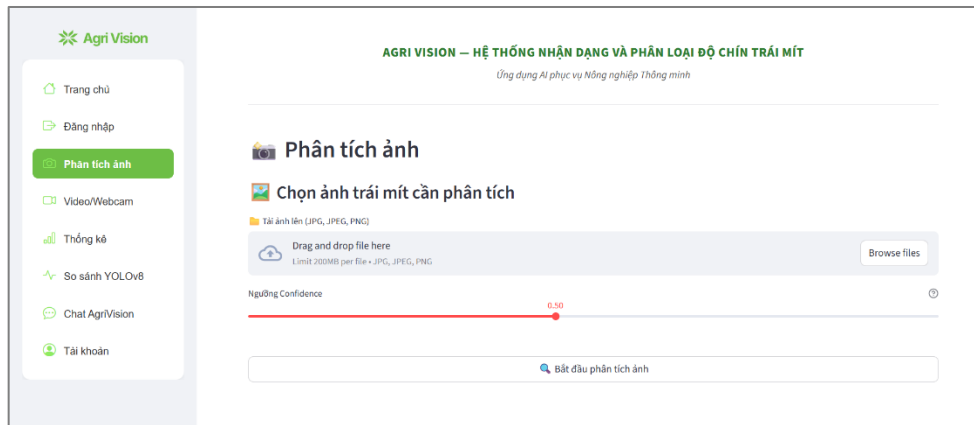
Hình ảnh minh họa giao diện đăng ký tài khoản của hệ thống AgriVision, cho phép người dùng tạo tài khoản mới bằng cách nhập thông tin cá nhân cơ bản, phục vụ việc quản lý người dùng và khai thác các chức năng nhận dạng, phân loại độ chín trái mít của hệ thống.

The screenshot shows the 'Đăng nhập tài khoản' (Login Account) page of the Agri Vision system. The layout is consistent with the registration page. The sidebar menu is on the left. The main content area has the same title and subtitle. Below the title, there are tabs for 'Đăng nhập' and 'Đăng ký', with 'Đăng nhập' being the active tab. The login form includes fields for 'Tên người dùng' (Username) and 'Mật khẩu' (Password). A 'Đăng nhập' (Login) button is at the bottom right. An illustration of a person walking towards a screen is on the left side of the form.

Hình 3.8. Giao diện đăng nhập của website

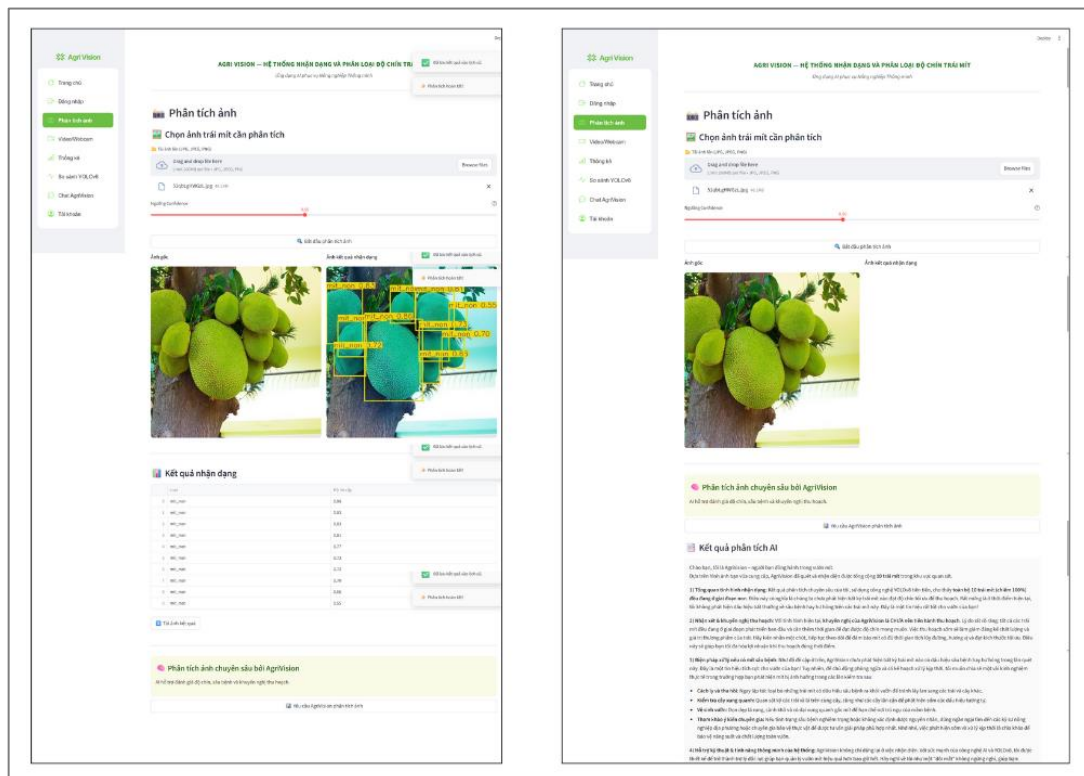
Hình ảnh minh họa giao diện đăng nhập của hệ thống AgriVision, cho phép người dùng xác thực tài khoản để truy cập và sử dụng các chức năng nhận dạng, phân loại độ chín trái mít.

### 3.3.3. Giao diện phân tích ảnh và kết quả phân tích



Hình 3.9. Giao diện phân tích ảnh của website

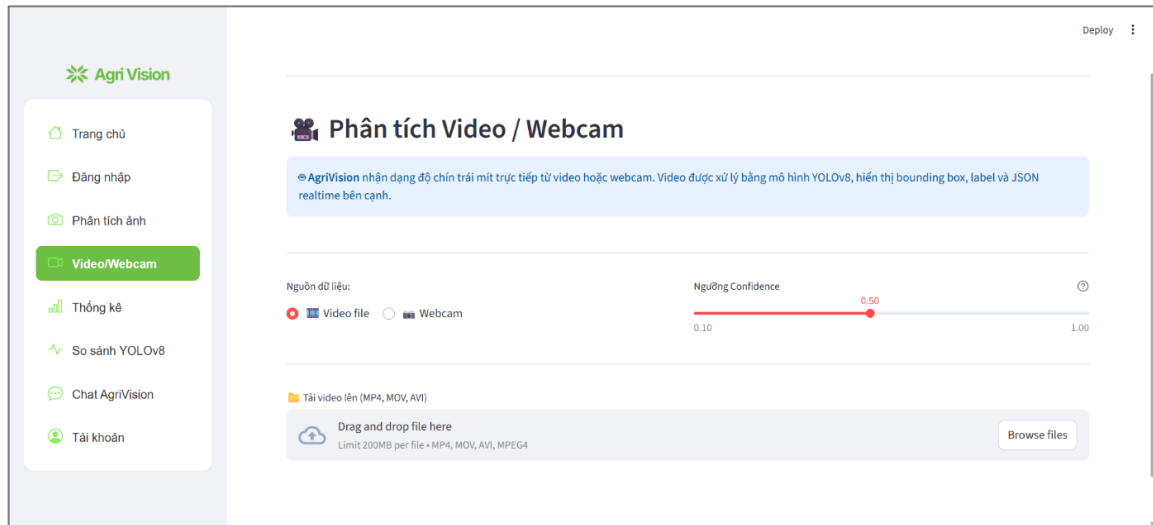
Hình ảnh minh họa giao diện phân tích ảnh của hệ thống AgriVision, cho phép người dùng tải ảnh trái mít, thiết lập ngưỡng nhận dạng và thực hiện phân tích, phân loại độ chín trái mít bằng mô hình AI.



Hình 3.10. Kết quả nhận diện trái mít và phân tích của AI

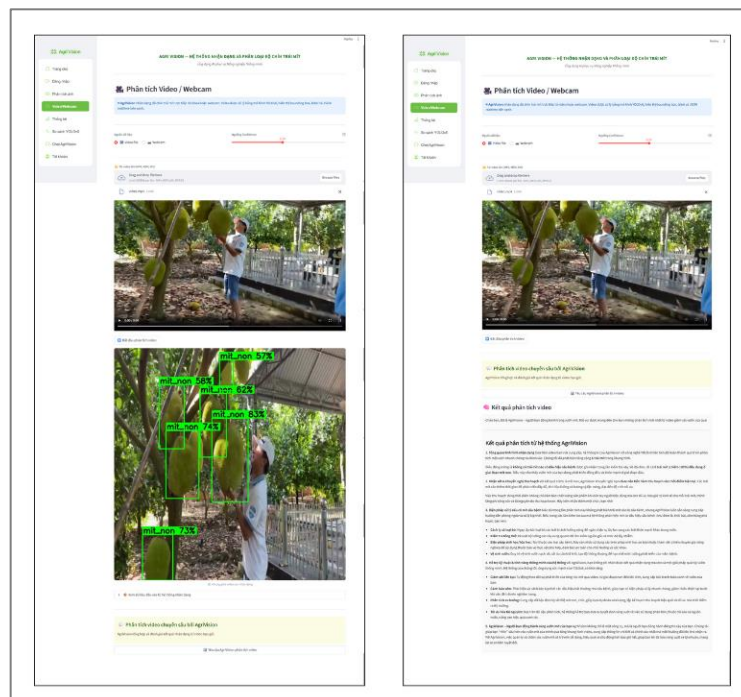
Hình ảnh minh họa kết quả phân tích ảnh trái mít của hệ thống AgriVision, trong đó hệ thống thực hiện phát hiện trái mít, hiển thị hộp bao và nhãn phân loại, đồng thời cung cấp bảng kết quả và nội dung phân tích hỗ trợ bởi AI.

### 3.3.4. Giao diện phân tích video và kết quả phân tích



Hình 3.11. Giao diện phân tích video của website

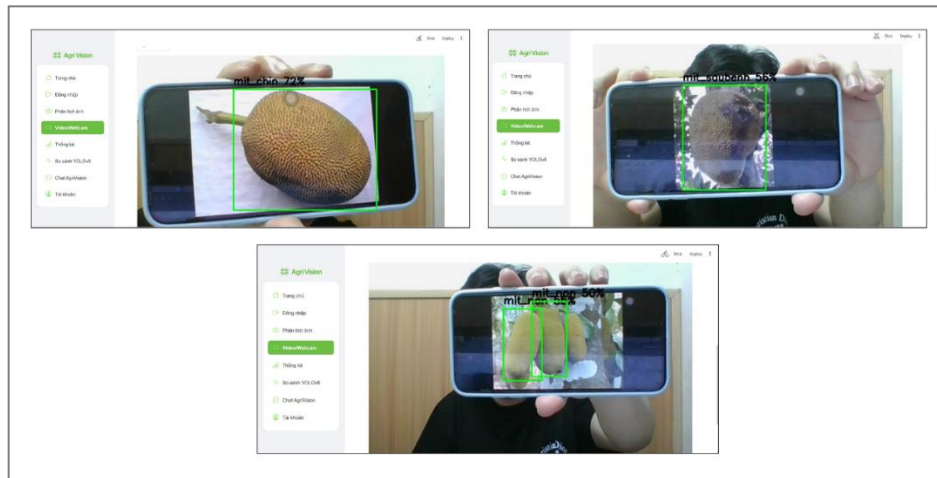
Hình ảnh minh họa giao diện phân tích video và webcam của hệ thống AgriVision, cho phép xử lý video hoặc dữ liệu camera theo thời gian gần thực nhằm nhận dạng và phân loại độ chín trái mít bằng mô hình AI.



Hình 3.12. Kết quả phân tích video và phân tích AI của video

Hình ảnh minh họa kết quả phân tích video/webcam của hệ thống AgriVision, trong đó mô hình AI thực hiện nhận dạng trái mít theo thời gian gần thực, hiển thị hộp bao, nhãn phân loại và cung cấp nội dung phân tích hỗ trợ cho người dùng.

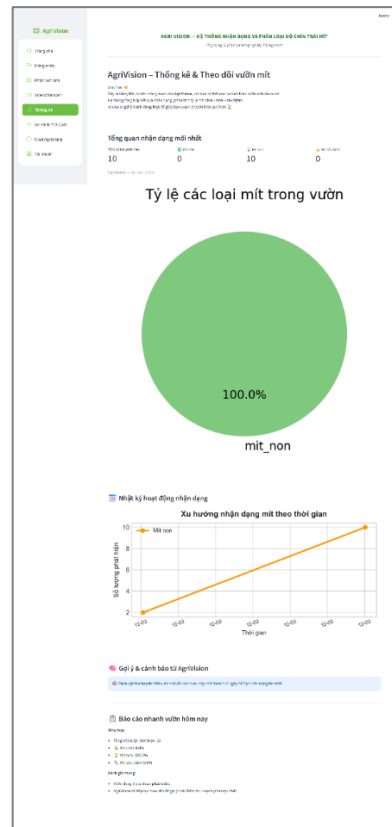
### 3.3.5. Giao diện webcam và kết quả nhận dạng webcam



Hình 3.13. Giao diện webcam và kết quả nhận diện của webcam

Hình ảnh minh họa kết quả nhận dạng độ chín trái mít bằng video/webcam trong hệ thống AgriVision, trong đó mô hình AI phát hiện trái mít và hiển thị hộp bao cùng nhãn phân loại theo thời gian gần thực.

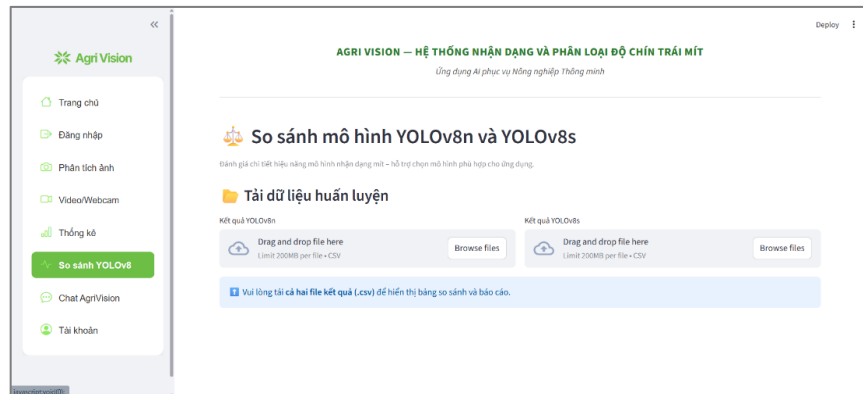
### 3.3.6. Giao diện chức năng thống kê và phân tích số liệu



Hình 3.14. Kết quả thống kê và phân tích số liệu

Hình ảnh minh họa giao diện thống kê của hệ thống AgriVision, hiển thị tỷ lệ các loại trái mít trong vườn, số lượng nhận dạng và xu hướng thay đổi theo thời gian nhằm hỗ trợ theo dõi và đánh giá tình trạng vườn cây.

### 3.3.7. Giao diện chức năng so sánh mô hình



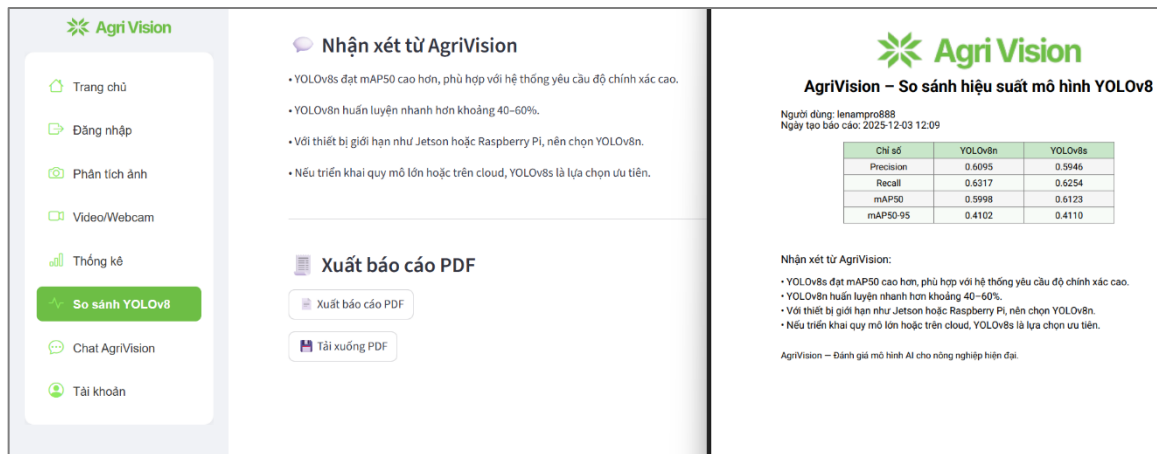
Hình 3.15. Giao diện chức năng so sánh hai mô hình

Hình ảnh minh họa giao diện so sánh hai mô hình YOLOv8n và YOLOv8s, cho phép đánh giá và đối chiếu kết quả huấn luyện nhằm lựa chọn mô hình phù hợp cho bài toán nhận dạng độ chín trái mít.



Hình 3.16. Kết quả so sánh và phân tích dữ liệu của hai mô hình YOLOv8

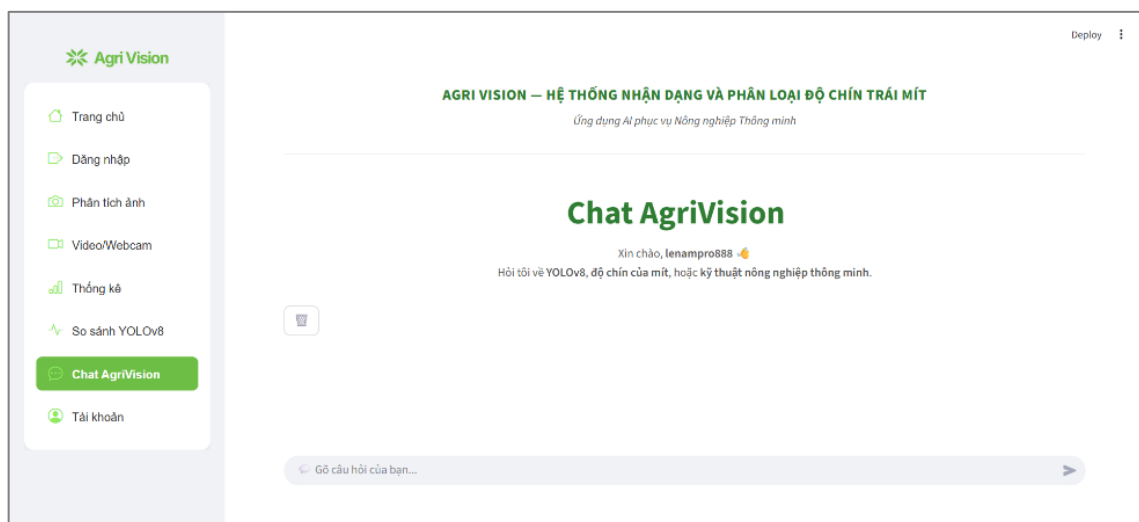
Hình ảnh minh họa kết quả so sánh hiệu năng giữa hai mô hình YOLOv8n và YOLOv8s, bao gồm các chỉ số tổng quan, biểu đồ huấn luyện theo epoch và biểu đồ hiệu năng nhằm hỗ trợ lựa chọn mô hình phù hợp cho bài toán nhận dạng độ chín trái mít.



Hình 3.17. Xuất File PDF phân tích từ hệ thống

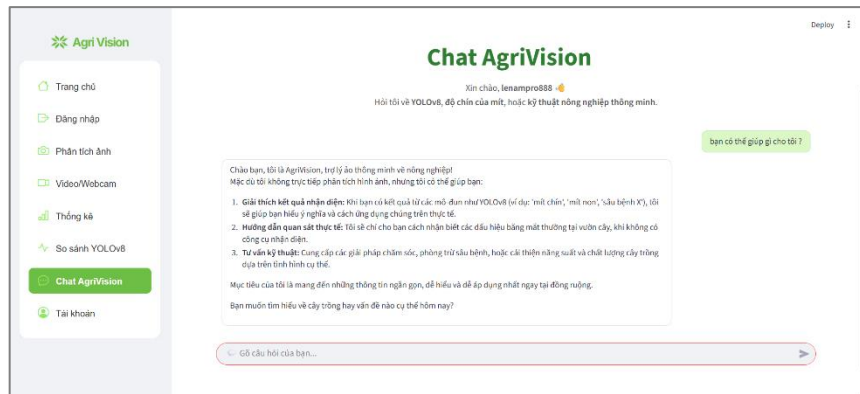
Hình ảnh minh họa chức năng nhận xét và xuất báo cáo của hệ thống AgriVision, cho phép tổng hợp, so sánh hiệu năng các mô hình YOLOv8 và sinh báo cáo kết quả dưới dạng PDF.

### 3.3.8. Giao diện AI hỗ trợ Chatbot



Hình 3.18. Giao diện Chatbot hỗ trợ AgriVision

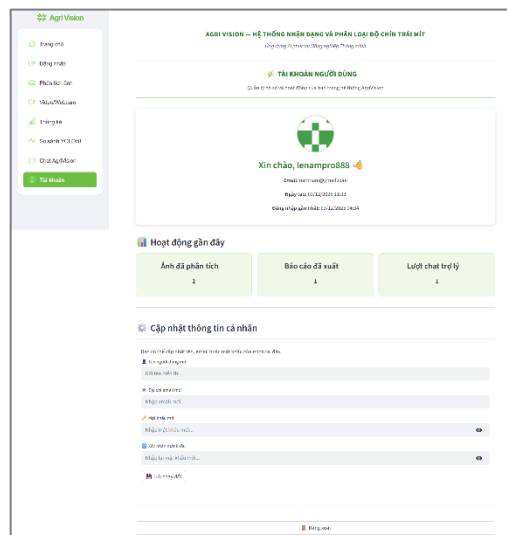
Hình ảnh minh họa giao diện Chat AgriVision, cho phép người dùng trao đổi và nhận tư vấn liên quan đến nhận dạng độ chín trái mít và ứng dụng AI trong nông nghiệp thông minh.



Hình 3.19. Kết quả trả lời của Chatbot

Hình ảnh minh họa kết quả tương tác trong chức năng Chat AgriVision, cho phép người dùng đặt câu hỏi và nhận tư vấn liên quan đến nhận dạng độ chín trái mít và ứng dụng AI trong nông nghiệp thông minh.

### 3.3.9. Giao diện thông tin người dùng



Hình 3.20. Giao diện thông tin người dùng

Hình ảnh minh họa giao diện quản lý tài khoản người dùng của hệ thống AgriVision, cho phép xem thông tin cá nhân, lịch sử hoạt động và cập nhật thông tin người dùng.

## 3.4. Quy trình cài đặt và chạy ứng dụng AgriVision Mobile

### 3.4.1. Chuẩn bị môi trường phát triển

Để cài đặt và chạy được ứng dụng di động AgriVision Mobile, cần chuẩn bị:

- Ngôn ngữ & SDK: Flutter SDK (kèm Dart) đã cài đặt trên máy tính và cấu hình biến môi trường PATH.

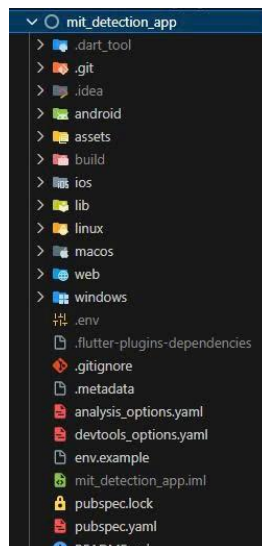
- IDE: Visual Studio Code (hoặc Android Studio) có cài các extension hỗ trợ Flutter/Dart.
- Android SDK / Emulator: Cài Android SDK và tạo sẵn một thiết bị ảo (AVD) dùng để chạy và kiểm thử ứng dụng.
- Tài khoản Supabase: dùng để cấu hình Backend (URL, ANON KEY) cho ứng dụng.

Sau khi cài Flutter, có thể chạy lệnh:

### Flutter doctor

Để kiểm tra nhanh môi trường đã đầy đủ hay chưa (SDK, thiết bị ảo, IDE...)

#### 3.4.2. Cấu trúc thư mục dự án



Hình 3.21. Cấu trúc thư mục của dự án App Mobile

#### 3.4.3. Chạy ứng dụng trên thiết bị giả lập (Android Emulator)

Quy trình chạy app trên máy ảo Android:

##### 1. Khởi động emulator

Mở Android Studio hoặc dùng AVD Manager để bật một thiết bị ảo (ví dụ Pixel 2 – Android x86 64).

##### 2. Chọn thiết bị chạy trong VS Code

Đảm bảo VS Code nhận diện được emulator ở góc bên phải sẽ hiển thị thiết bị đang kết nối.

##### 3. Chạy lệnh flutter run

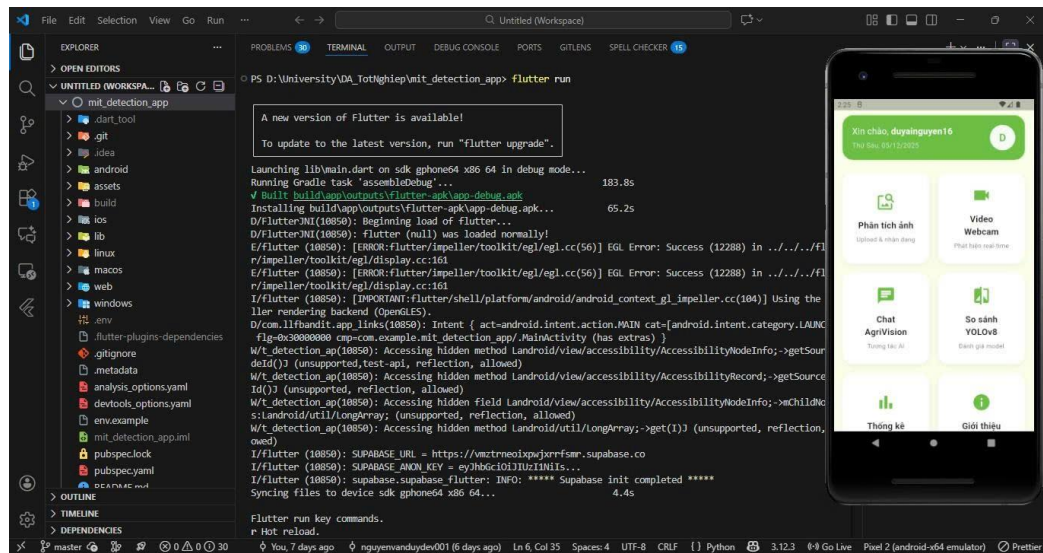
Trong terminal VS Code, tại thư mục dự án, thực hiện:

### Flutter run

Lệnh này sẽ:

- Biên dịch mã nguồn Dart sang max native Android.
- Chạy Gradle để build file app-debug.apk (thể hiện qua dòng log: Built build\app\outputs\flutter-apk\app-debug.apk).
- Cài đặt APK này vào thiết bị ảo (log: Installing build\app\outputs\flutter-apk\app-debug.apk...).
- Khởi động ứng dụng trên emulator.

## 4. Quan sát kết quả

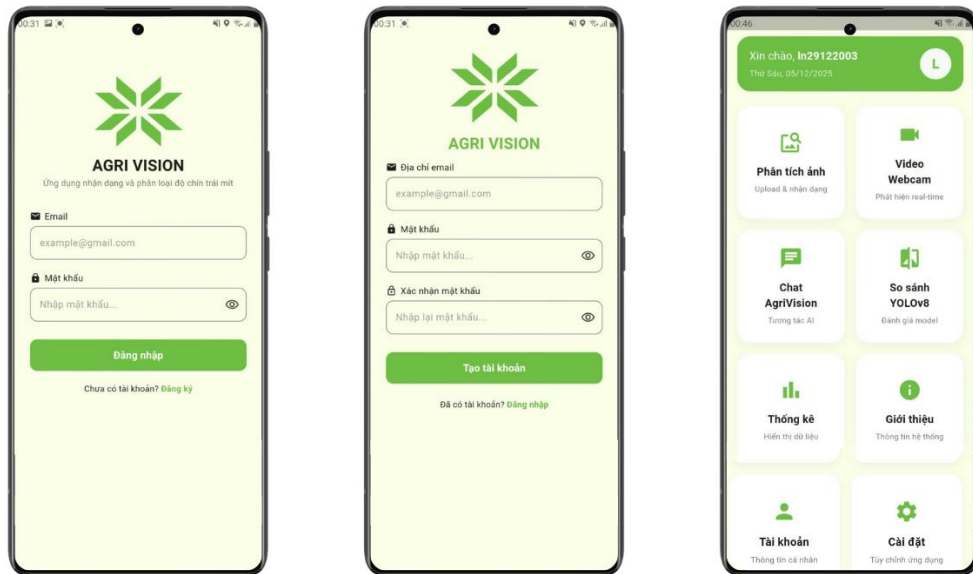


Hình 3.22. Khởi động thiết bị giả lập thành công

Sau khi build thành công, màn hình giả lập hiển thị giao diện chính của app AgriVision Mobile với các chức năng: Phân tích ảnh, Video, webcam, camera real time, Chat AgriVision, So sánh YOLOv8, Thống kê, Giới thiệu.

### 3.5. Kết quả đạt được của App Mobile

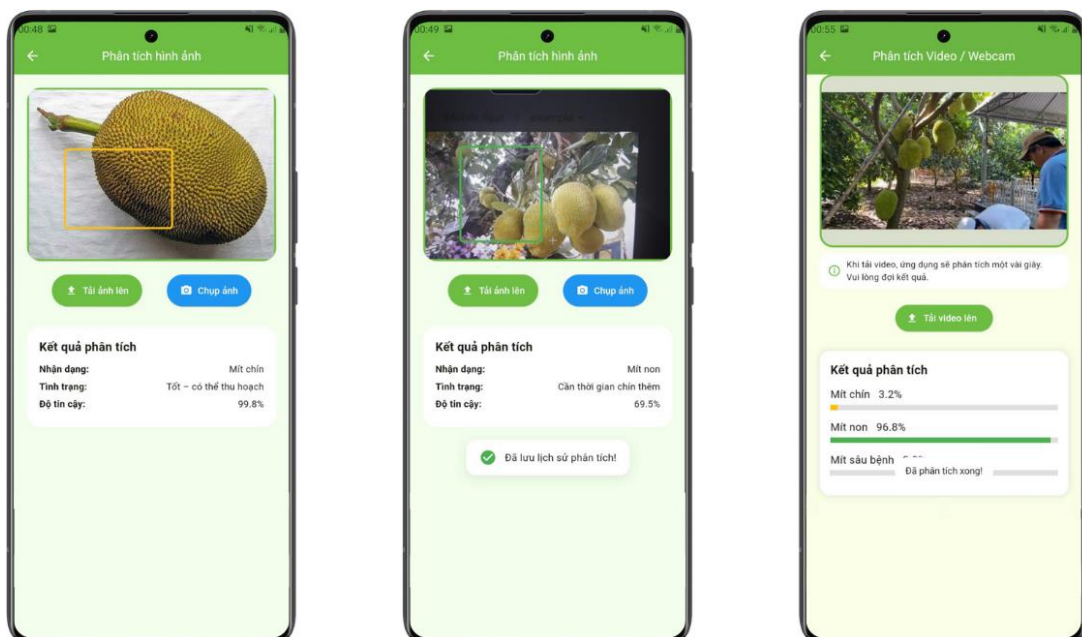
#### 3.5.1. Giao diện đăng ký, đăng nhập và trang chủ



Hình 3.23. Giao diện trang đăng ký, đăng nhập và trang chủ của App Mobile

Hình ảnh minh họa giao diện ứng dụng AgriVision trên thiết bị di động, bao gồm các chức năng đăng nhập, đăng ký tài khoản và màn hình chính cho phép người dùng truy cập nhanh các tính năng nhận dạng, phân loại độ chín trái mít và hỗ trợ AI.

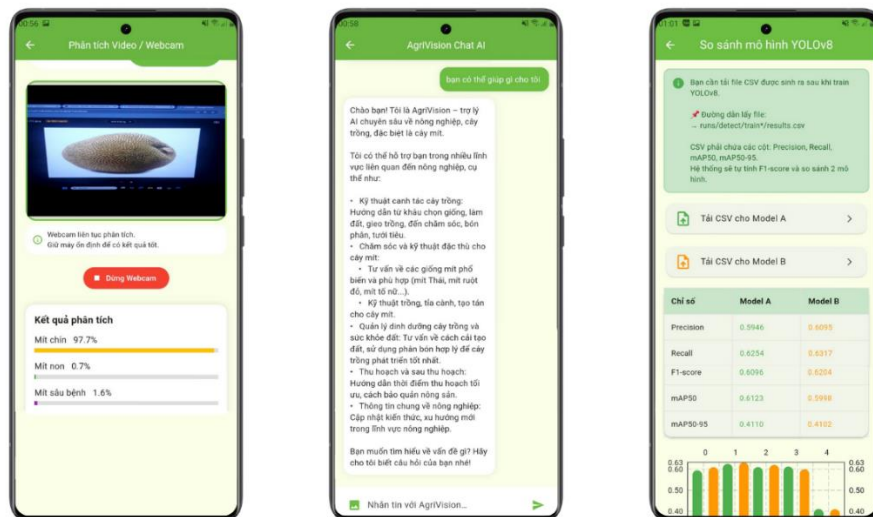
#### 3.5.2. Giao diện chức năng Phân tích ảnh, Camera, Video



Hình 3.24. Giao diện chức năng phân tích ảnh camera và video của App Mobile

Hình ảnh minh họa chức năng phân tích ảnh và video trên ứng dụng AgriVision di động, cho phép người dùng chụp ảnh hoặc tải video để hệ thống AI nhận dạng và phân loại độ chín trái mít, hiển thị kết quả trực quan.

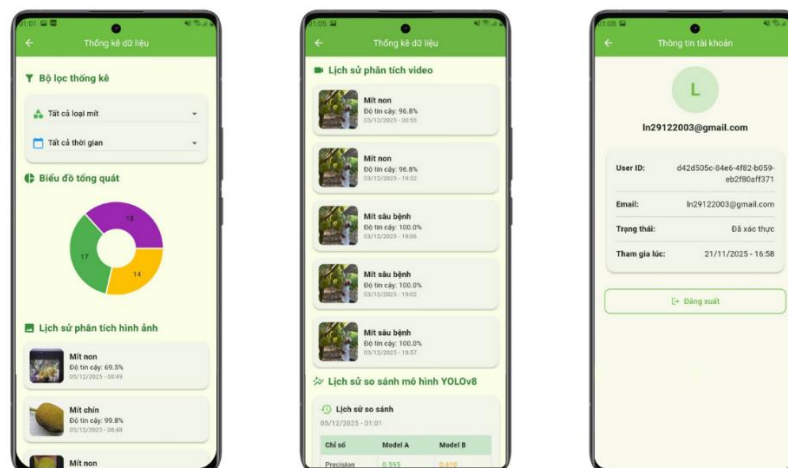
### 3.5.3. Giao diện chức năng Webcam, Chatbot AI và so sánh mô hình YOLOv8



Hình 3.25. Giao diện chức năng webcam, chatbot AI và so sánh mô hình

Hình ảnh minh họa các chức năng chính trên ứng dụng AgriVision di động, bao gồm phân tích video/webcam theo thời gian thực, tư vấn thông minh qua Chat AI và so sánh hiệu năng các mô hình YOLOv8.

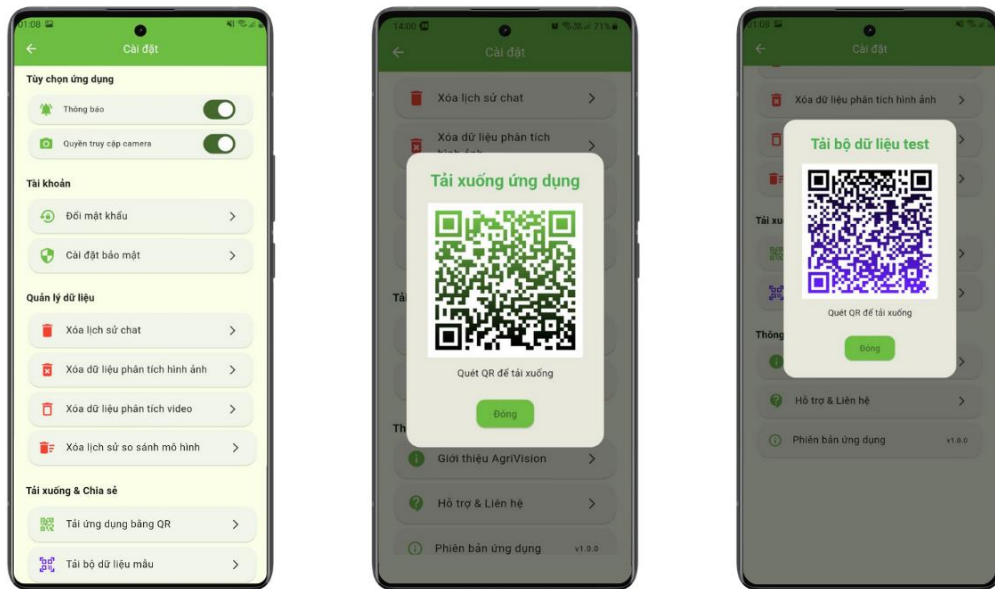
### 3.5.4. Giao diện chức năng thống kê dữ liệu và thông tin tài khoản



Hình 3.26. Giao diện chức năng thống kê dữ liệu và thông tin tài khoản người dùng

Hình ảnh minh họa các chức năng thống kê, lịch sử phân tích và quản lý tài khoản trên ứng dụng AgriVision di động, hỗ trợ người dùng theo dõi kết quả nhận dạng và quản lý thông tin cá nhân.

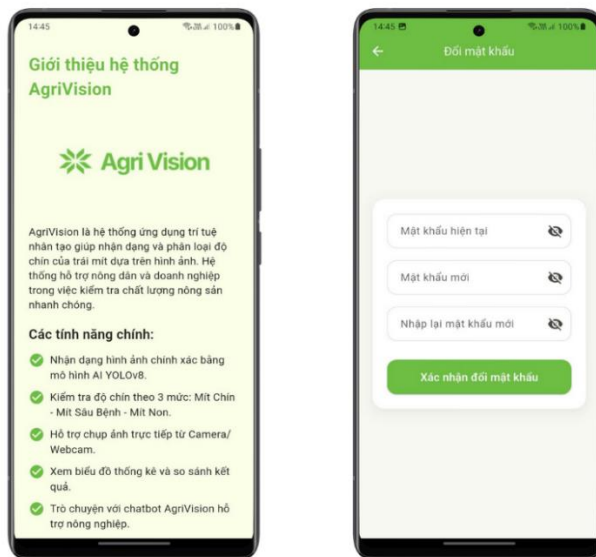
### 3.5.5. Giao diện phần cài đặt của App Mobile



Hình 3.27. Giao diện phần cài đặt và mã QR tải ứng dụng

Hình ảnh minh họa giao diện cài đặt trên ứng dụng AgriVision di động, cho phép người dùng tùy chỉnh ứng dụng, quản lý dữ liệu, tải ứng dụng và bộ dữ liệu thông qua mã QR.

### 3.5.6. Giao diện phần giới thiệu và đổi mật khẩu



Hình 3.28. Giao diện giới thiệu và đổi mật khẩu

Hình ảnh minh họa giao diện giới thiệu hệ thống và chức năng đổi mật khẩu trên ứng dụng AgriVision di động, giúp người dùng nắm được thông tin tổng quan và quản lý bảo mật tài khoản.

## KẾT LUẬN

Trong đồ án “Nghiên cứu và ứng dụng YOLOv8 trong nhận dạng và phân loại độ chín trái mít phục vụ nông nghiệp thông minh”, nhóm chúng em đã nghiên cứu các kiến thức liên quan đến thị giác máy tính, mô hình YOLOv8, quy trình xây dựng và gắn nhãn dữ liệu, huấn luyện mô hình trên Google Colab, cùng các công nghệ triển khai như FastAPI, Streamlit, MongoDB Atlas, Flutter, Supabase và Teachable Machine. Trên cơ sở đó, nhóm xây dựng được hệ thống AgriVision gồm ứng dụng web (nhận dạng trái mít từ ảnh, video, webcam, so sánh mô hình, chatbot hỗ trợ, lưu lịch sử trên MongoDB Atlas) và ứng dụng di động AgriVision Mobile (đăng nhập, nhận dạng trái mít qua ảnh/camera thời gian thực, lưu lịch sử trên Supabase).

Bên cạnh những kết quả đạt được, đồ án vẫn còn một số hạn chế như: bộ dữ liệu huấn luyện cho mô hình YOLOv8 chưa thật sự lớn và đa dạng, khả năng khái quát trong các điều kiện ánh sáng, góc chụp phức tạp còn hạn chế, hệ thống hiện mới ở mức demo, chưa tối ưu hiệu năng cho thiết bị cấu hình thấp và môi trường IoT, ứng dụng di động chủ yếu tập trung vào chức năng nhận dạng cơ bản, chưa hỗ trợ các nghiệp vụ quản lý sâu hơn phục vụ sản xuất và kinh doanh.

Trong thời gian tới, nhóm định hướng phát triển theo các hướng: mở rộng và làm giàu bộ dữ liệu trái mít, thử nghiệm thêm các biến thể YOLOv8 hoặc mô hình mới hơn, tối ưu tham số để cân bằng giữa độ chính xác và tốc độ, cải thiện hiệu năng hệ thống để có thể triển khai trên thiết bị IoT, hoàn thiện các chức năng quản lý dữ liệu, thống kê, báo cáo và đồng bộ tốt hơn giữa web và ứng dụng di động, đồng thời xem xét mở rộng mô hình sang các loại nông sản khác.

Sinh viên thực hiện Lê Nam và Nguyễn Văn Duy xin gửi lời cảm ơn chân thành đến quý thầy cô trường Đại Học Kiến Trúc Đà Nẵng Khoa Công nghệ Thông tin, đặc biệt là thầy T.S Nguyễn Hà Huy Cường là giảng viên hướng dẫn của chúng em đã tận tình hỗ trợ, góp ý trong suốt quá trình thực hiện đồ án. Nhóm em cũng xin cảm ơn những công cụ hỗ trợ hết sức tuyệt vời đã giúp đỡ một phần nào công việc và tạo điều kiện để đồ án được hoàn thành, đồng thời rất mong nhận được thêm ý kiến đóng góp của quý thầy cô và hội đồng để tiếp tục hoàn thiện đề tài trong tương lai.

## TÀI LIỆU THAM KHẢO

### Tài liệu Tiếng Anh

- [1] Ultralytics. (2023, n.d.). YOLOv8 documentation. *Ultralytics Docs*.  
<https://docs.ultralytics.com/>
- [2] Roboflow. (2023, January 10). How to train YOLOv8 object detection on a custom dataset. *Roboflow Blog*. <https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/>
- [3] Xiao, B. (2024, n.d.). Fruit ripeness identification using YOLOv8 model. *Multimedia Tools and Applications (Springer)*.  
<https://link.springer.com/article/10.1007/s11042-023-16570-9>
- [4] Google. (n.d.). Image model – Teachable Machine. *Teachable Machine*.  
<https://teachablemachine.withgoogle.com/train/image>
- [5] Roboflow. (2023, March 16). Getting started with Roboflow. *Roboflow Blog*.  
<https://blog.roboflow.com/getting-started-with-roboflow/>
- [6] Streamlit Inc. (n.d.). Streamlit documentation. *Streamlit Docs*.  
<https://docs.streamlit.io/>
- [7] Tiangolo, S. (n.d.). FastAPI documentation. *FastAPI*.  
<https://fastapi.tiangolo.com/>
- [8] Google. (n.d.). Google Colaboratory. *Google Colab*. <https://colab.google/>
- [9] Flutter Team. (n.d.). Flutter documentation. *Flutter Docs*.  
<https://docs.flutter.dev/>
- [10] Supabase. (2025, n.d.). Supabase docs – database, auth and storage. Supabase Documentation. <https://supabase.com/docs>

## Tài liệu tiếng việt

- [1] Lê Hoàng Hưng (2024, Tháng 09 27). Khám phá YOLOv8: Bước đột phá trong phát hiện đối tượng. THỊ GIÁC MÁY TÍNH. <https://thigiacmaytinh.com/kham-pha-yolov8-buoc-dot-pha-trong-phat-hien-doi-tuong/>
- [2] Phan Thị Ngọc Hồng (2025). Cải tiến mô hình nhận diện hình ảnh YOLOv8 dựa trên thay đổi kiến trúc của mô hình. Tạp chí Khoa học HUFLIT. <https://hjs.huflit.edu.vn/index.php/hjs/article/view/242>
- [3] Trịnh Trung Hải, Bùi Xuân Thiện, Nguyễn Lê Tùng Khánh & Nguyễn Hà Huy Cường (2021). Xây dựng hệ thống nhận dạng phân loại trái cây chín tiếp cận mạng Noron tích chập (CNN). Thư viện Tài nguyên số – Trường Đại học CNTT & TT Việt – Hàn. <https://elib.vku.udn.vn/handle/123456789/1882>.
- [4] Abdelrahman Elgendy (2024, Tháng 11 13). Thị giác máy tính trong nông nghiệp: Chuyển đổi việc phát hiện trái cây và canh tác chính xác. Ultralytics. <https://www.ultralytics.com/vi/blog/computer-vision-in-agriculture-transforming-fruit-detection-and-precision-farming>.