

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**



Nguyễn Quỳnh Mai

**XÂY DỰNG ỨNG DỤNG NHẬN DIỆN BỆNH
TRÊN CÂY TRỒNG**

**KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHẤT LƯỢNG CAO
Ngành: Khoa học máy tính - CLC**

HÀ NỘI – 2022

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

Nguyễn Quỳnh Mai

**XÂY DỰNG ỨNG DỤNG NHẬN DIỆN BỆNH
TRÊN CÂY TRỒNG**

**KHÓA LUẬN TỐT NGHIỆP ĐẠI HỌC HỆ CHẤT LƯỢNG CAO
Ngành: Khoa học máy tính - CLC**

Cán bộ hướng dẫn: TS. Nguyễn Thị Hậu

HÀ NỘI – 2022

**VIETNAM NATIONAL UNIVERSITY, HANOI
UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

Nguyen Quynh Mai

PLANT DISEASE CLASSIFICATION APPLICATION

Major: Computer science – High Quality

Supervisor: DR. Nguyen Thi Hau

HA NOI - 2022

LỜI CẢM ƠN

Trong suốt quá trình thực hiện khoá luận, tôi đã nhận được rất nhiều sự động viên, chỉ bảo từ gia đình, thầy cô và bạn bè. Đó là nguồn cỗ vũ to lớn giúp tôi hoàn thành khoá luận tốt nghiệp này. Tôi xin trân thành gửi lời cảm ơn tới tất cả mọi người trong suốt khoảng thời gian vừa qua.

Đầu tiên, tôi xin gửi lời cảm ơn sâu sắc đến giảng viên hướng dẫn - TS. Nguyễn Thị Hậu đã tận tình hướng dẫn và đưa ra những góp ý để tôi có thể hoàn thành khoá luận của mình. Tôi cũng xin cảm ơn Ban Giám hiệu trường Đại học Công nghệ - ĐHQGHN đã tạo cơ hội cho tôi được thực hiện và bảo vệ đề tài này. Cảm ơn tất cả quý thầy cô, đặc biệt là các thầy cô Khoa Công nghệ thông tin đã tận tình giảng dạy, truyền đạt những kiến thức bổ ích và truyền cảm hứng trong suốt những năm học tập và rèn luyện trên ghế giảng đường, giúp chúng tôi trang bị những kiến thức để theo đuổi ước mơ của mình.

Cuối cùng, tôi xin gửi lời cảm ơn đến gia đình, bạn bè, đồng nghiệp và tập thể K62 – CACLC3 đã luôn luôn đồng hành, giúp đỡ và khuyến khích và giúp tôi có thêm thật nhiều kỷ niệm quý báu trong suốt quãng thời gian sinh viên tươi đẹp của mình.

Trân trọng cảm ơn!

Hà Nội, tháng 11 năm 2022

Sinh viên

Nguyễn Quỳnh Mai

TÓM TẮT

Tóm tắt: Ngày nay, vai trò của công nghệ thông tin trong nông nghiệp đang ngày càng trở nên quan trọng. Trong đó, việc sử dụng các hệ thống phát hiện và nhận dạng sâu bệnh ở lá cây đã giúp ích rất nhiều trong việc phòng ngừa, ngăn chặn các loại bệnh, góp phần nâng cao năng suất, chất lượng sản phẩm nông sản và độ an toàn cho người nông dân. Chính vì vậy, khoá luận đã tìm hiểu và trình bày phương pháp phân loại lá cây bị bệnh, được giải quyết thông qua việc học chuyển tiếp các mô hình mạng thần kinh tích chập và xây dựng một ứng dụng di động giúp người dùng có thể dễ dàng tiếp cận và sử dụng. Ứng dụng giúp hỗ trợ người dùng xác định được bệnh cây tròng, và cung cấp thêm thông tin về một số loại bệnh cây tròng, từ đó có thể đưa ra những phương pháp phòng trị thích hợp cho cây tròng.

Từ khóa: Phân loại ảnh bệnh cây tròng, học chuyển tiếp, ứng dụng di động.

ABSTRACT

Abstract: Nowadays, the role of information technology in agriculture is becoming increasingly important. In particular, the use of pest detection and identification systems in leaves has greatly helped in the prevention and control of diseases, contributing to improving productivity, quality of agricultural products and the quality of agricultural products. safe for farmers. Therefore, the thesis has explored and presented the method of classifying diseased leaves, which is solved through forward learning of convolutional neural network models and building a mobile application that helps users to can be easily accessed and used. The application helps users to identify plant diseases, and provides more information about a number of plant diseases, from which it is possible to recommend appropriate treatment methods for plants.

Keywords: *Plant Diseases Classification, Transfer Learning and Mobile Application.*

LỜI CAM ĐOAN

Tôi xin cam đoan đây là khoá luận của tôi, do chính bản thân tôi tự tìm hiểu và nghiên cứu dưới sự hướng dẫn của giảng viên TS. Nguyễn Thị Hậu. Những tham khảo, nghiên cứu và tài liệu liên quan đến khoá luận đều được ghi rõ ràng và trích dẫn cụ thể trong phần danh mục tham khảo và nằm trong phạm vi cho phép theo đúng với quy chế của nhà trường. Tôi chịu trách nhiệm nếu những nội dung trình bày và kết quả trong khoá luận này là không trung thực.

Hà Nội, tháng 11 năm 2022

Sinh viên

Nguyễn Quỳnh Mai

MỤC LỤC

LỜI CẢM ƠN	i
TÓM TẮT	ii
ABSTRACT	iii
LỜI CAM ĐOAN	iv
DANH SÁCH KÝ HIỆU VÀ CHỮ VIẾT TẮT	3
DANH SÁCH HÌNH ẢNH	4
DANH SÁCH BẢNG BIỂU	12
MỞ ĐẦU	1
CHƯƠNG 1. GIỚI THIỆU CHUNG	2
1.1. Mô tả bài toán	3
1.1.1. Đầu vào của bài toán.....	3
1.1.2. Đầu ra của bài toán	4
1.2. Phân tích các hệ thống tương tự	4
1.2.1. Ứng dụng Plantix trên nền tảng di động Android.....	5
1.2.2. Ứng dụng Agrio trên nền tảng di động iOS và Android.....	6
CHƯƠNG 2. MÔ HÌNH NHẬN DIỆN BỆNH TRÊN LÁ CÂY DỰA TRÊN MÔ HÌNH MẠNG NO-RON TÍCH CHẬP	9
2.1. Các nghiên cứu liên quan đến nhận diện bệnh trên lá cây.....	9
2.1.1. Giới thiệu chung	9
2.1.2. Tìm hiểu MobileNet	10
2.2. Nhận dạng bệnh trên lá cây	14
2.2.1. Tập dữ liệu	14
2.2.2. Cài đặt, đánh giá mô hình và kết quả.....	20
CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG.....	24
3.1. Tổng quan hệ thống	24
3.2. Tầng CSDL	26
3.2.1. Giới thiệu tổng quan	26
3.2.2. Tìm hiểu và phân tích, thiết kế dữ liệu trên Firebase	27
3.3. Tầng giao diện	35
3.3.1. Đăng nhập/ Đăng ký	35
3.3.2. Cập nhật thông tin.....	38
3.3.3. Tìm kiếm mẫu bệnh cây	39
3.3.4. Nhận diện mẫu bệnh cây tròn.....	41
3.3.5. Quản lý nhận diện bệnh cây tròn	43

3.3.6. Tham khảo ý kiến cộng đồng.....	45
3.4. Các công nghệ sử dụng.....	46
3.4.1. Hệ điều hành iOS.....	46
3.4.2. Swift.....	48
3.4.3. TensorFlow	49
3.4.4. Keras	50
3.4.5. CoreML.....	50
3.4.6. Firebase.....	52
CHƯƠNG 4. CÀI ĐẶT VÀ KIỂM THỬ HỆ THỐNG	54
4.1. Xây dựng và cài đặt ứng dụng PlantCare	54
4.2. Ứng dụng PlantCare	55
4.2.1. Màn hình chính	55
4.2.2. Đăng nhập	56
4.2.3. Tìm kiếm mẫu bệnh cây trồng	57
4.2.4. Nhận diện cây trồng bằng hình ảnh	58
4.2.5. Nhận diện bệnh cây trồng theo thời gian thực	59
4.2.6. Tham khảo ý kiến cộng đồng.....	60
4.2.7. Quản lý bài viết.....	61
4.2.8. Quản lý thông tin người dùng	62
4.2.9. Lịch sử tra cứu	63
4.2.10. Chuyển đổi ngôn ngữ.....	64
KẾT LUẬN	66
TÀI LIỆU THAM KHẢO	67

DANH SÁCH KÝ HIỆU VÀ CHỮ VIẾT TẮT

Ký tự viết tắt	Tên đầy đủ	Giải thích
KLTN	Khoa luận tốt nghiệp	
TS	Tiến sĩ	
API	Application Programming Interface	Là một giao diện mà hệ thống hay ứng dụng cung cấp để cho phép các yêu cầu dịch vụ có thể được tạo ra từ các hệ thống, ứng dụng khác, cho phép dữ liệu được trao đổi qua lại giữa chúng
noSQL	no Structured Query Language	Ngôn ngữ truy vấn phi cấu trúc
CSDL	Cơ sở dữ liệu	
SDK	Software Development Kit	Là các công cụ và phần mềm dùng để phát triển ứng dụng thông qua một nền tảng nhất định
BN	Batch normalization	Chuẩn hóa hàng loạt
CPU	Central Processing Unit	Bộ xử lý trung tâm
GPU	Graphics Processing Unit	Đơn vị xử lý đồ họa
CNN	Convolutional Neural Network	Mạng nơ-ron tích chập
RNN	<u>Recurrent Neural Network</u>	Mạng thần kinh hồi quy
DSC	Depthwise Separable Convolution	Một cải tiến được áp dụng trong MobileNet giúp giảm thiểu số lượng tham số

DANH SÁCH HÌNH ẢNH

Hình 1.1. Ảnh bộ phận lá cây tròn trong bộ dữ liệu.....	3
Hình 1.2. Ảnh bộ phận lá cây bị bệnh do người dùng chụp.....	4
Hình 1.3. Minh họa ứng dụng Plantix	5
Hình 1.4. Minh họa ứng dụng Agrio	7
Hình 2.1. Minh họa phép tích chập.....	11
Hình 2.2. Minh họa phép tích chập.....	12
Hình 2.3. Mô hình mạng MobileNet	13
Hình 2.4. So sánh Convolution truyền thống (Trái) và Depthwise Separable Convolution (Phải).....	14
Hình 2.5. So sánh mạng MobileNet truyền thống và Depthwise Separable Convolution..	14
Hình 2.6. Tỉ lệ số lượng ảnh lá cây bị nhiễm bệnh và lá cây khoẻ mạnh xấp xỉ 3/4	15
Hình 2.7. Minh họa bộ dữ liệu PlantVillage.....	18
Hình 2.8. Các tham số, thiết lập đầu vào, đầu ra của các lớp	20
Hình 2.9. Kết quả sau khi thực hiện 5 epochs cuối.	21
Hình 2.10. Biểu đồ biểu diễn độ chính xác, mất mát của tập huấn luyện và kiểm thử	22
Hình 2.11. Kết quả nhận diện bệnh héo sớm lá khoai tây.	23
Hình 3.1. Kiến trúc tổng quan hệ thống.....	25
Hình 3.2. Cấu trúc cây JSON của Realtime Database	27
Hình 3.3. Dữ liệu users	28
Hình 3.4. Dữ liệu classifying_results.....	29
Hình 3.5. Dữ liệu relabel_disease_images.....	30
Hình 3.6. Dữ liệu posts	31
Hình 3.7. Dữ liệu nhánh comments	32
Hình 3.8. Dữ liệu likes.....	33
Hình 3.9. Dữ liệu người dùng lưu trên Firebase Authentication	34
Hình 3.10. Dữ liệu ảnh trên Storage	34
Hình 3.11. Các tính năng tầng giao diện	35
Hình 3.12. Biểu đồ hoạt động cho tính năng Đăng nhập.....	36
Hình 3.13. Biểu đồ hoạt động cho tính năng “Cập nhật thông tin”	38
Hình 3.14. Biểu đồ hoạt động cho chức năng Tìm kiếm mẫu bệnh cây tròn	40
Hình 3.15. Biểu đồ hoạt động cho chức năng “Nhận diện bệnh cây tròn theo ảnh”	41
Hình 3.16. Biểu đồ hoạt động cho chức năng “Nhận diện bệnh cây tròn theo thời gian thực”.....	42
Hình 3.17. Biểu đồ hoạt động cho chức năng “Quản lý nhận diện bệnh cây tròn”	44
Hình 3.18. Biểu đồ hoạt động cho chức năng Tham khảo ý kiến cộng đồng	45
Hình 3.19. Các thành phần cơ bản của hệ điều hành iOS.....	47
Hình 3.20. Các bước thêm một mô hình đã được huấn luyện vào ứng dụng iOS	51
Hình 3.21. Vị trí CoreML trong ứng dụng	52
Hình 4.1. Thêm mới hệ thống iOS vào Firebase	54
Hình 4.2. Thêm câu lệnh để hoàn thiện thêm Firebase SDK vào hệ thống.....	55
Hình 4.3. Giao diện màn hình chính.....	56
Hình 4.4. Giao diện màn hình đăng nhập, đăng ký và khôi phục mật khẩu	57

Hình 4.5. Giao diện tìm kiếm mẫu bệnh cây trồng.....	58
Hình 4.6. Giao diện màn chi tiết bệnh cây trồng	59
Hình 4.7. Giao diện nhận diện bệnh cây trồng theo thời gian thực	60
Hình 4.8. Giao diện chức năng tham khảo ý kiến cộng đồng.....	61
Hình 4.9. Giao diện trang cá nhân của người dùng ứng dụng	62
Hình 4.10. Giao diện thay đổi thông tin người dùng	63
Hình 4.11. Giao diện chức năng Tìm kiếm.....	64
Hình 4.12. Giao diện chức năng Chuyển đổi ngôn ngữ	65

DANH SÁCH BẢNG BIỂU

Bảng 2.1. Thống kê số lượng ảnh trong tập dữ liệu	18
Bảng 2.2. Kết quả huấn luyện của các mô hình nơ-ron học chuyển tiếp	22
Bảng 3.1. Những thuộc tính lưu trong nhánh users.	28
Bảng 3.2. Những thuộc tính lưu trong nhánh classifying_results.	29
Bảng 3.3. Những thuộc tính lưu trong nhánh relabel_disease_images.	30
Bảng 3.4. Những thuộc tính lưu trong nhánh posts.	31
Bảng 3.5. Những thuộc tính lưu trong nhánh comments.	32
Bảng 3.6. Những thuộc tính lưu trong nhánh likes.	33
Bảng 3.7. Mô tả ca sử dụng chức năng “Đăng nhập”.....	37
Bảng 3.8. Mô tả ca sử dụng chức năng “Đăng ký”.	38
Bảng 3.9. Mô tả ca sử dụng chức năng “Cập nhật thông tin”	39
Bảng 3.10. Mô tả ca sử dụng chức năng “Tìm kiếm”.	40
Bảng 3.11. Mô tả ca sử dụng chức năng “Nhận diện bệnh cây trồng theo ảnh”.	42
Bảng 3.12. Mô tả ca sử dụng chức năng “Nhận diện bệnh cây trồng theo thời gian thực”.	43
Bảng 3.13. Mô tả ca sử dụng chức năng “Quản lý nhận diện bệnh cây trồng”....	45
Bảng 3.14. Mô tả ca sử dụng chức năng “Tham khảo ý kiến cộng đồng”.	46

MỞ ĐẦU

Sâu bệnh hại cây trồng luôn là vấn đề khó khăn và không thể tránh khỏi trong trồng trọt. Việc phát hiện kịp thời bệnh của các loại lá cây giúp mọi người có thể chủ động hơn trong công tác ngăn ngừa và phòng chống sâu bệnh, đem lại nguồn kinh tế cao hơn. Tuy nhiên, với mỗi loại cây trồng khác nhau đều có nguy cơ mắc các loại bệnh khác nhau. Đôi khi, việc phát hiện bệnh cây trồng khó phát hiện bằng mắt thường, hay với những người ít kinh nghiệm trong việc trồng trọt.

Vì vậy, khoá luận đã phát triển một ứng dụng hỗ trợ người dùng nhận biết được sâu bệnh hại của cây trồng. Người dùng chỉ cần chụp ảnh, hệ thống sẽ tự động phân tích ảnh đó và đưa ra kết quả phân tích. Bên cạnh đó, hệ thống cũng cung cấp thêm thông tin và cách phòng trị cho cây trồng, giúp người dùng tham khảo, cũng như tìm cách điều trị cho cây trồng.

Việc nhận diện bệnh cây trồng đã được nghiên cứu rất nhiều, cũng đã có rất nhiều ứng dụng liên quan đến nhận diện bệnh cây trồng nhưng trên hệ điều hành iOS còn khá ít. Khoá luận tập trung xây dựng, phát triển ứng dụng nhận diện bệnh cây trồng trên hệ điều hành iOS dành riêng cho người Việt. Ứng dụng giúp người dùng nhận diện bệnh một cách nhanh chóng và thuận tiện hơn.

Khoá luận bao gồm các chương sau:

Chương 1: Trong chương 1, khoá luận sẽ trình bày tổng quan về bài toán phân tích và nhận biết một số loại bệnh phổ biến xuất hiện ở các loại lá cây trồng, đồng thời phân tích các hệ thống tương tự khác để xây dựng lại hệ thống nhận diện lá cây.

Chương 2: Trong chương này, khoá luận tập trung tìm hiểu việc học chuyển tiếp, áp dụng mô hình MobileNet vào nhận diện bệnh cây trồng. Giới thiệu các công nghệ được sử dụng để xây dựng ứng dụng di động hỗ trợ nhận diện bệnh cây trồng.

Chương 3: Với chương 3, từ kết quả của chương 2, KLTN sẽ trình bày chi tiết về việc phân tích và thiết kế hệ thống nhận diện bệnh trên lá cây trên nền tảng iOS.

Chương 4: Trong chương 4, KLTN sẽ triển khai cài đặt và đánh giá hệ thống, các kết quả đã đạt được ở chương trước.

Phần kết luận: Tóm tắt các kết quả đạt được và định hướng các nghiên cứu trong tương lai.

CHƯƠNG 1. GIỚI THIỆU CHUNG

Bệnh hại cây trồng nông nghiệp (sâu hại, bệnh hại, cỏ dại...) luôn là yếu tố gây ảnh hưởng nghiêm trọng đến năng suất, sản lượng và chất lượng nông sản. Trong điều kiện khí hậu ngày càng biến đổi phức tạp như hiện nay, góp phần làm xuất hiện các loài dịch mới; nhiều loài dịch hại thứ yếu có thể trở thành chủ yếu; vòng đời dịch hại ngắn lại, số lứa trong năm và khả năng sinh sản tăng lên, dẫn đến thiệt hại do chúng gây ra sẽ ngày càng nghiêm trọng và khó kiểm soát.

Việt Nam có thành phần sinh vật gây bệnh hại cây trồng rất đa dạng và phong phú. Chỉ tính riêng sâu hại (các loài côn trùng và nhện nhỏ hại thực vật), đến năm 2013 đã có ít nhất 1.124 loài gây hại trên 77 loại cây trồng nông nghiệp phổ biến ở nước ta. Các nhà khoa học cũng ghi nhận được hơn 540 loài nấm gây các bệnh hại thực vật khác nhau trên các cây trồng ở nước ta và gần 700 loài cỏ dại... Một số nghiên cứu gần đây cho thấy, nước ta đã xuất hiện thêm nhiều loài dịch hại mới và bùng phát trở lại nhiều đợt dịch sâu bệnh gây hại nặng nề cho nền sản xuất nông nghiệp ở nước ta như dịch vàng lùn (lùn xoăn lá), bệnh lùn sọc đen phương nam, dịch rầy nâu..., và gần đây là sâu keo mùa thu hại ngô, bệnh khóm lá săn xuất hiện tại nhiều địa phương trong cả nước. Từ thực trạng về sâu bệnh hại hiện nay, đặc biệt là sự bùng phát của các loại sâu bệnh hại mới, các sâu bệnh hại thứ yếu gần đây trở thành chủ yếu, gây hại mạnh đối với một số cây trồng chính, cần có những biện pháp bảo vệ cây trồng kịp thời và nhanh chóng [1].

Do đó, với sự bùng nổ của công nghệ thông tin như hiện nay, điện thoại di động trở nên phổ biến, việc áp dụng mô hình học máy nhận diện bệnh cây giúp hỗ trợ cho người sử dụng có thể biết và khắc phục bệnh cho cây, dựa vào ảnh chụp của lá hoặc phát hiện bệnh cây dựa vào các biểu hiện của lá cây thông qua ảnh chụp. Nếu không có những công cụ nhận dạng tự động, để nhận dạng một loài cây hoặc phát hiện một loại bệnh cây chúng ta sẽ phải tự tìm hiểu, tra cứu từ nhiều nguồn tài liệu, dữ liệu. Để có được kết quả của việc phân loại cây hoặc phát hiện bệnh cây đôi khi mất nhiều thời gian. Để hỗ trợ trong việc xác định bệnh cây, có nhiều giải pháp nhận diện bệnh cây trồng đã được các nhà nghiên cứu đề xuất, như là các phương pháp học sâu như sử dụng xử lý ảnh, các mô hình học máy... Trong đó, tiêu biểu là sử dụng mạng nơ-ron tích chập. Nhận dạng các loại bệnh trên lá cây qua phương pháp sử dụng mạng nơ-ron tích chập là sử dụng tập dữ liệu có sẵn, đã qua xử lý hình ảnh để huấn luyện một mô hình để dự đoán hình ảnh của lá cây, từ đó xác định được tên bệnh của lá cây trồng.

1.1. Mô tả bài toán

1.1.1. Đầu vào của bài toán

KLTN sẽ xây dựng lại hệ thống nhận diện bệnh trên lá cây với các yêu cầu đầu vào bao gồm:

- Ảnh bộ phận của cây trồng từ bộ dữ liệu



Hình 1.1. Ảnh bộ phận lá cây trồng trong bộ dữ liệu.

- Hình ảnh của người dùng tự chụp.



Hình 1.2. Ảnh bộ phận lá cây bị bệnh do người dùng chụp.

1.1.2. Đầu ra của bài toán

Hình ảnh sau khi người dùng gửi lên sẽ được hệ thống phân tích và trả về kết quả là tên bệnh của cây trồng mà hệ thống nhận dạng được và cung cấp thêm một số thông tin về bệnh đó. Ứng dụng nhận diện được 38 loại cây trồng của 14 loài cây khác nhau gồm cây táo, việt quất, anh đào, ngô, nho, cam, đào, ót chuông, khoai tây, bí đỏ, cà chua, cam, mâm xôi và dâu tây sẽ được liệt kê chi tiết ở chương 2.

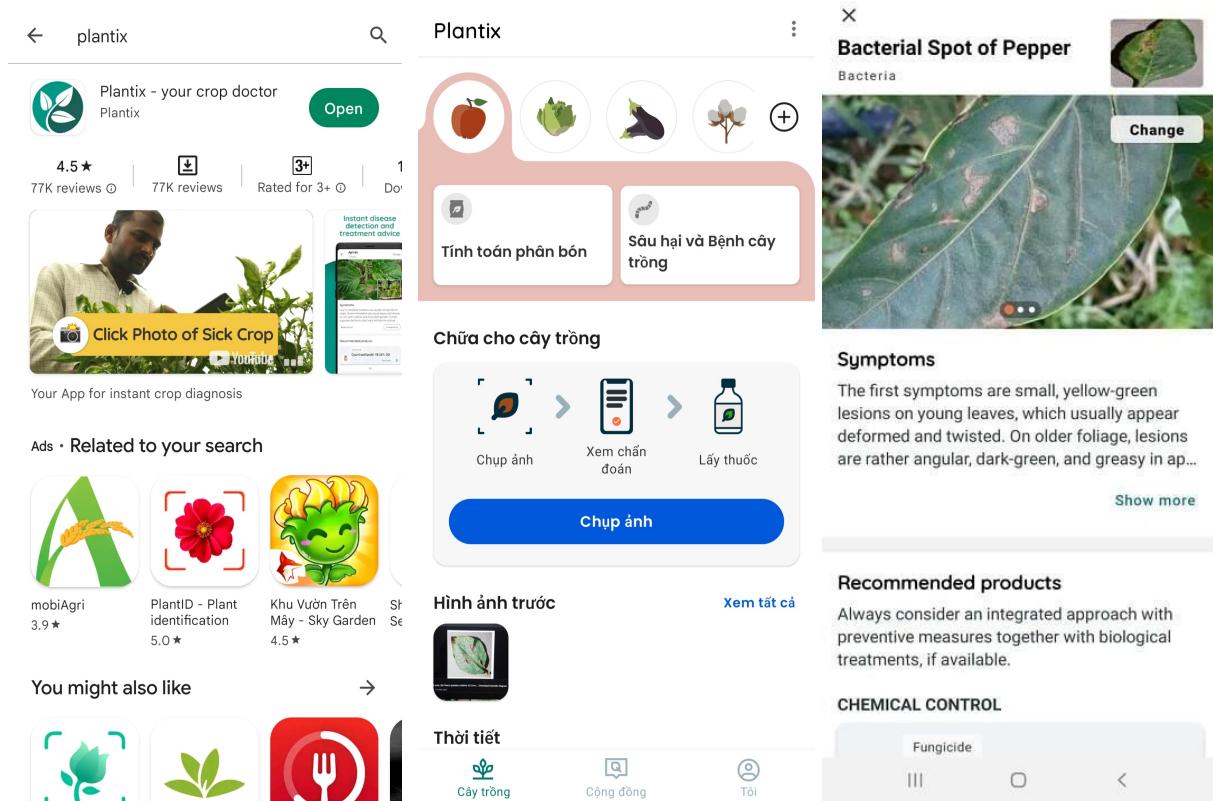
Ngoài ra, người dùng còn có thể tìm kiếm và tham khảo một số loại bệnh cây trồng khác, và tham khảo ý kiến của mọi người để có kết quả chính xác hơn.

Để xây dựng hệ thống phù hợp với người Việt Nam, tiếp theo đây KLTN sẽ đưa ra hai hệ thống tương tự để tiến hành phân tích, so sánh, cũng như đánh giá được ưu, nhược điểm của từng hệ thống tương ứng, qua đó có những góc nhìn đa chiều về hệ thống đã xây dựng, cũng như học hỏi và áp dụng vào hệ thống này.

1.2. Phân tích các hệ thống tương tự

Với sự phát triển công nghệ thông tin như hiện nay, có rất nhiều hệ thống lớn giúp người dân nhận biết được bệnh của cây trồng trên nền tảng di động. Dưới đây là những tìm hiểu và phân tích về những ứng dụng này.

1.2.1. Ứng dụng Plantix trên nền tảng di động Android



Hình 1.3. Minh họa ứng dụng Plantix

Progressive Environmental & Agricultural Technologies (PEAT), một công ty khởi nghiệp của Đức đã phát triển Plantix là một ứng dụng di động Android chuẩn đoán bệnh ở cây trồng đa ngôn ngữ. Ứng dụng hỗ trợ xác định thiệt hại của thực vật với sự trợ giúp của học máy và trí tuệ nhân tạo. Hình ảnh do người dùng chụp hoặc từ thư viện của người dùng được tải trực tiếp lên máy chủ và tự động phân tích bằng các thuật toán của PEAT để xác định bệnh và trả về kết quả cho người dùng trong vài giây. Thông tin quan trọng về các triệu chứng, các tác nhân gây bệnh, hóa chất nông nghiệp cũng như phương pháp điều trị sinh học được cung cấp cho người dùng thông qua hình ảnh hoặc thời gian thực. Hơn nữa, ứng dụng bao gồm một hệ thống thông tin thời tiết cụ thể cho vị trí của người dùng, tính toán phân bón dựa trên diện tích đất và một tính năng cộng đồng tạo điều kiện tương tác với các bên liên quan khác, quan tâm đến dịch vụ bảo vệ thực vật. Hệ thống còn cung cấp thêm hướng dẫn về cây trồng trong toàn bộ chu kỳ.

Điểm nổi bật của ứng dụng:

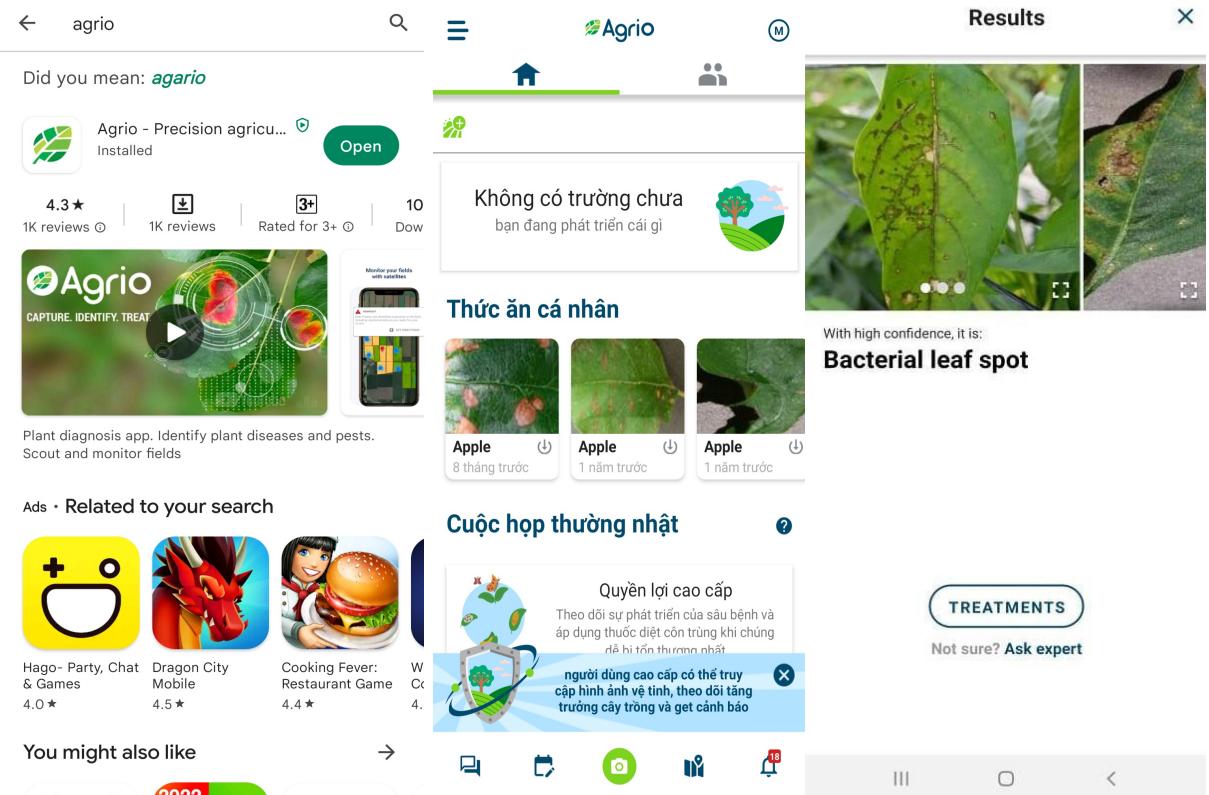
- Bộ cục giao diện hợp lý, thân thiện với người dùng.

- Hỗ trợ nhiều ngôn ngữ, trong đó có cả Tiếng Việt.
- Độ nhận diện bệnh chính xác cao, kết quả trả về nhanh.
- Nhận diện được cả cây tròng khỏe mạnh và cây tròng bị bệnh.
- Bên cạnh đó, ứng dụng vẫn hiển thị các kết quả bệnh có khả năng cao xảy ra với những ảnh không thể nhận diện được chắc chắn tình trạng bệnh của cây tròng đó.

Tuy nhiên, phải có kết nối mạng mới sử dụng được tính năng chẩn đoán bệnh của cây tròng. Thêm vào đó nữa, hệ thống chưa ổn định khi sử dụng ngôn ngữ Tiếng Việt, gây khó khăn cho người dùng Việt Nam khi sử dụng ứng dụng này và hệ thống hiện chỉ hỗ trợ cho Android.

1.2.2. Ứng dụng Agrio trên nền tảng di động iOS và Android

Ứng dụng Agrio do công ty Saillog phát triển, được phát hành vào năm 2017 và vẫn đang được cập nhật. Ứng dụng có sẵn trên App Store và CH Play. Agrio là một ứng dụng xác định bệnh thực vật dựa trên trí tuệ nhân tạo và thị giác máy tính. Ứng dụng sử dụng hình ảnh do người dùng tải lên để khớp với dữ liệu có sẵn trong cơ sở dữ liệu của ứng dụng và sau đó cung cấp chi tiết chính xác về phương pháp điều trị bệnh cây tròng. Agrio đã cách mạng hóa công nghệ giám sát cây tròng bằng cách cung cấp các thay đổi trên điện thoại của người dùng bằng cách sử dụng cảm biến hình ảnh từ xa khi có bất kỳ thay đổi nào trong ruộng cây tròng. Ứng dụng này cũng giúp nông dân, người kiểm tra cây tròng ghi lại dữ liệu cây tròng khi đang di chuyển.



Hình 1.4. Minh họa ứng dụng Agrio

Điểm nổi bật của ứng dụng:

- Hỗ trợ nhiều ngôn ngữ trong đó có cả Tiếng Việt.
- Giao diện thân thiện với người dùng.
- Độ phát hiện bệnh cao và hiển thị theo 3 mức cao, trung bình và thấp giúp người dùng biết được độ tin cậy của kết quả nhận được.
- Luôn có đội ngũ chuyên gia để hỗ trợ người dùng xác định bệnh của cây trồng.
- Tính năng làm việc nhóm giúp quản lý cây trồng làm theo nhóm.

Tuy nhiên, để tăng độ chính xác trong phát hiện bệnh, ứng dụng vẫn yêu cầu người dùng chọn cây trồng. Trong phần mục bài đăng không có tính năng xóa bài đã đăng. Và ứng dụng phải có kết nối mạng mới dùng được.

Kết luận: Với những phân tích, đánh giá từ hai ứng dụng tham khảo, hệ thống trong khoá luận sẽ tập trung xây dựng lại một hệ thống giúp người dùng nhận diện được bệnh cây trồng, có thể sử dụng nhận diện ngay cả khi ngoại tuyến, tìm kiếm thông tin một cách nhanh chóng hơn trên hệ điều hành iOS. Hệ thống sẽ chuyển đổi linh hoạt giữa 2 ngôn ngữ tiếng Anh và

tiếng Việt, giúp trải nghiệm người dùng tốt hơn. Và xây dựng diễn đàn trao đổi để người dùng ứng dụng có thể trao đổi, giải đáp và hỗ trợ lẫn nhau liên quan đến cây trồng. Trong chương này, khoá luận đã giới thiệu qua về tình trạng bệnh cây trồng hiện nay và cũng giới thiệu hệ thống nhận diện bệnh cây trồng. Bên cạnh đó, chương một cũng trình bày ưu điểm và nhược điểm của hai hệ thống tương tự với mục đích nghiên cứu, triển khai và xây dựng hệ thống này. Chương tiếp theo, KLTN sẽ trình bày chi tiết về việc áp dụng mô hình học chuyển tiếp vào hệ thống.

CHƯƠNG 2. MÔ HÌNH NHẬN DIỆN BỆNH TRÊN LÁ CÂY DỰA TRÊN MÔ HÌNH MẠNG NO-RON TÍCH CHẬP

Trong phạm vi chương này, khoá luận sẽ tập trung vào việc giới thiệu và trình bày các công trình, các nghiên cứu khoa học liên quan đến nhận diện bệnh trên lá cây trồng. Thứ hai, KLTN sẽ giới thiệu mạng MobileNet, và áp dụng việc học chuyển tiếp của mạng này vào bài toán nhận diện bệnh cây trồng.

2.1. Các nghiên cứu liên quan đến nhận diện bệnh trên lá cây

Trên thế giới đã có một số công trình nghiên cứu nhận dạng sâu bệnh trên trái cà chua, dưa chuột... bằng cách vận dụng kỹ thuật xử lý ảnh và nhận dạng đã đạt được một số kết quả khả quan như sử dụng các kỹ thuật xử lý ảnh để trích chọn đặc trưng, các phương pháp học máy như máy véc-tơ hỗ trợ... và tiêu biểu là sử dụng mạng nơ-ron tích chập. Trong phạm vi khoá luận này sẽ giới thiệu một phương pháp được áp dụng phổ biến giúp cải thiện độ chính xác và tiết kiệm chi phí thời gian huấn luyện. Phương pháp được xây dựng dựa trên ý tưởng chuyển giao tri thức đã được học từ những mô hình tốt trước đó. Đó chính là học chuyển tiếp.

2.1.1. Giới thiệu chung

Nhận diện bệnh lá cây trồng đã được nghiên cứu rất nhiều sử dụng mạng học sâu và thị giác máy tính. Các phương pháp học máy bao gồm các thuật toán thị giác máy tính cổ điển như haar, hog, sift, surt, phân đoạn ảnh, máy véc-tơ hỗ trợ (SVM), K-Nearest Neighbours (KNN), K-means và mạng thần kinh nhân tạo (ANN). Mô hình học sâu phân loại bệnh cây trồng gồm rất nhiều mô hình mạng nơ-ron tích chập như AlexNet, GoogleNet, VGGNet... Đôi khi, bộ dữ liệu không đủ lớn, hay phân loại nhiều lớp yêu cầu phải điều chỉnh tham số sao cho phù hợp để tránh việc học quá khớp.

Năm 2008, Phadikar và Sil sử dụng hình ảnh của những cây lúa bị nhiễm bệnh bằng máy ảnh kỹ thuật số. Kỹ thuật phân đoạn hình ảnh để phát hiện các bộ phận bị nhiễm bệnh của cây đã được sử dụng. Tiếp đó, năm 2012, Revathi và Hemalatha đã sử dụng kỹ thuật trích xuất đặc trưng RGB để xác định các bệnh mà hình ảnh đã chụp được thông qua việc xử lý ảnh. Sau khi phân đoạn ảnh màu này được thực hiện để có được các vùng mục tiêu xung quanh ảnh gốc. Các kỹ thuật tiếp theo bao gồm các bộ lọc Sobel và Canny đã được áp dụng để xác định các cạnh. Các đặc điểm cạnh được trích xuất này được sử dụng trong phân loại để xác định các đốm bệnh hiện diện trong hình ảnh. Cuối cùng, phần bị nhiễm bệnh của lá đã được sử dụng cho mục đích phân loại bằng cách sử dụng một mạng học sâu [2]. Hay Guan Wang và những người cộng sự của mình đã áp dụng việc học chuyển tiếp và tinh

chỉnh kiến trúc mô hình mạng VGG16 để phân loại bệnh của cây táo với độ chính xác là 90.16%. Tương tự như vậy, Ferentinos (2018) đã sử dụng AlexNet và GoogleNet để huấn luyện mô hình sử dụng tập dữ liệu gồm 87.848 ảnh, bao gồm 25 loại cây khác nhau, chia thành 58 lớp đạt độ chính xác là 99.53% [3]. Việc học chuyển tiếp mô hình mạng đã được huấn luyện trên một tập dữ liệu để giải quyết một bài toán khác giúp rút ngắn thời gian đào tạo mô hình và có thể đạt được những kết quả tốt hơn.

Mục tiêu của khoá luận là xây dựng được ứng dụng di động hỗ trợ nhận diện bệnh cây tròng, vì vậy trong phần tiếp theo KLTN sẽ trình bày về mô hình mạng phù hợp với các thiết bị di động, cũng như phân tích, đánh giá mô hình trên tập dữ liệu huấn luyện.

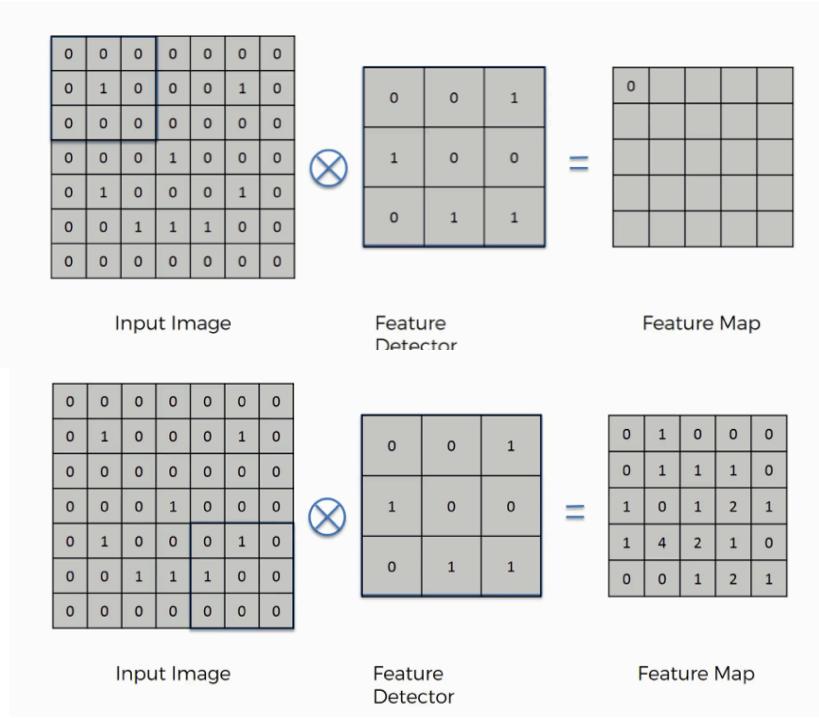
2.1.2. Tìm hiểu MobileNet

Với sự phát triển lĩnh vực học máy hiện nay, càng có nhiều mô hình nhận diện với độ chính xác cao. Nhưng không phải toàn bộ trong số những mô hình này đều có thể sử dụng được trên các thiết bị gặp hạn chế về tài nguyên tính toán. Để phát triển những mô hình học máy trên các thiết bị điện thoại thì thường sử dụng những mô hình có số lượng tính toán ít và có độ chính xác cao. Và một trong những mô hình tiêu biểu là MobileNet.

MobileNets được giới thiệu lần đầu vào năm 2017 được phát triển bởi đội ngũ Google, mô hình sử dụng cách tính chập tích mang tên Depthwise Separable Convolution nhằm giảm kích thước mô hình và giảm độ phức tạp tính toán [3].

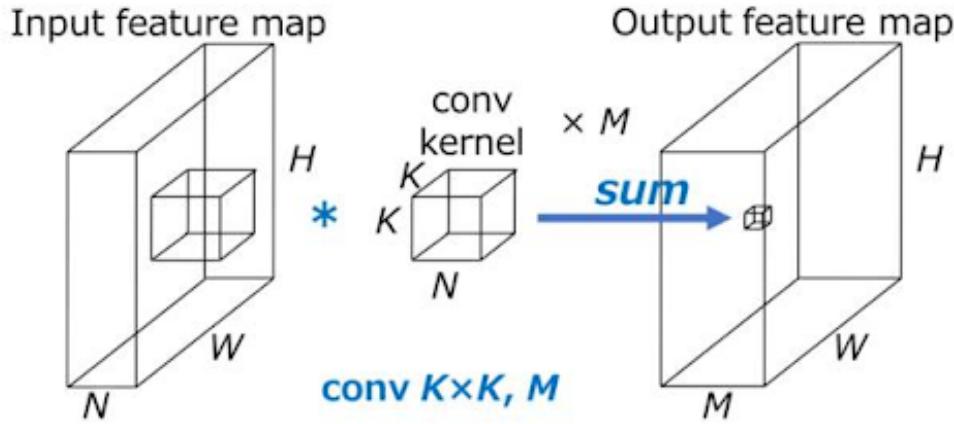
2.1.2.1 Depthwise Separable Convolution

Muốn hiểu rõ hơn về Depthwise Separable Convolution, cần phải hiểu rõ phép tích chập. Có một ma trận là Kernel, khi chiếu nó lên một vùng của bức ảnh. Lấy từng phần tử tương ứng của ảnh, nhân nó với phần tử tương ứng của Kernel. Sau đó lấy tổng. Mục tiêu của các lớp tích chập là trích chọn các đặc trưng của ảnh đầu vào. Như hình 2.1, một ma trận kích thước 3×3 quét qua ảnh đầu vào có kích thước là 7×7 , bằng cách nhân nó với từng phần tử tương ứng, được một bức ảnh mới có kích thước 5×5 , vẫn giữ nguyên đặc trưng của ảnh và gọi đó là feature map.



Hình 2.1. Minh họa phép tích chập

Tuy nhiên, xét từ khía cạnh độ phức tạp tính toán của một phép tích chập. Như mô tả ở hình 2.2, để tính một bức ảnh có số chiều là (H, W, N) với H là chiều cao, W là chiều rộng và N là chiều sâu dùng một kernel gồm (K, K, N) với K là độ lớn của kernel và N là chiều sâu của kernel (cùng với chiều sâu của ảnh), được một feature map có D chiều, cần $D \times D \times K \times K \times N$ phép tính nhân. Và thực tế, có nhiều kernel, nên số phép nhân cần tính là $D \times D \times K \times K \times N \times M$ với M là số kernel.



Hình 2.2. Minh họa phép tích chập

Như vậy, việc tính toán trở nên phức tạp. MobileNet đã giảm bớt việc tính toán bằng cách sử dụng Depthwise Separable Convolution, chia phép tích chập thành hai phần gồm: Depthwise Convolution và Pointwise Convolution.

- Depthwise Convolution: Là phép tích chập áp dụng một bộ lọc duy nhất cho mỗi kênh đầu vào. Độ phức tạp của một kênh là $1*D*D*K*K$, vậy bức ảnh có N chiều (N là chiều sâu của ảnh) cần $D*D*K*K*N$ phép tính nhân, như phép tích chập thông thường.
- Pointwise Convolution: Sau khi sử dụng Depthwise Convolution, thực hiện phép tích chập $1*1$ để trích M đặc trưng ra từ N lớp phép tích chập đã tính ở trên. Duyệt tất cả các ô trong ma trận, cần $D*D*N*M$ phép tính.

Vậy tổng độ phức tạp của Depthwise Separable Convolution là:

$$D*D*K*K*N + D*D*N*M$$

Độ phức tạp tính toán giảm đi khá nhiều. Tỷ lệ giữa phép tích chập truyền thống và Depthwise Separable Convolution là:

$$\frac{D*D*K*K*N + D*D*M*N}{D*D*K*K*N*M} = \frac{1}{M} + \frac{1}{K^2}$$

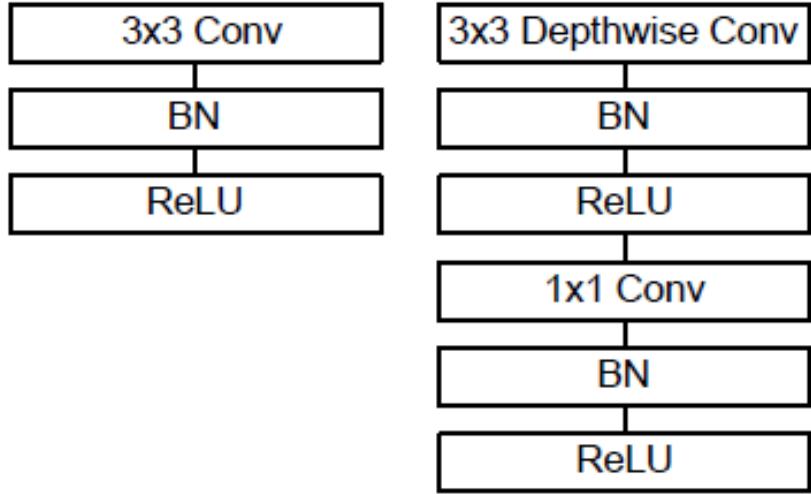
2.1.2.2. Kiến trúc mạng MobileNet

Mạng ngày gồm có 30 lớp, với lớp đầu tiên gồm một lớp tích chập với stride bằng 2. Sau đó đến lớp Depthwise và 1 lớp Pointwise và tiếp tục thêm một lớp depthwise với stride bằng 2 nữa. Và sau đó thêm một lớp Pointwise. Cuối cùng, là lớp Softmax để phân lớp.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Hình 2.3. Mô hình mạng MobileNet [3]

MobileNet dùng rất nhiều lớp Depthwise Separable Convolution để giảm số lượng parameter đi nhiều lần khoảng 9 lần. Một chú ý nhỏ về kiến trúc ở đây, là sau mỗi convolution MobileNet sẽ sử dụng Batch Normalization (BN) và ReLU như hình bên dưới:



Hình 2.4. So sánh Convolution truyền thống (Trái) và Depthwise Separable Convolution (Phải) [3]

So sánh kết quả của việc sử dụng mạng 30 lớp sử dụng convolution truyền thống và mạng 30 lớp sử dụng Depthwise Separable Convolution (MobileNet) trên tập dữ liệu ImageNet, chúng ta có kết quả như sau:

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
Conv MobileNet	71.7%	4866	29.3
MobileNet	70.6%	569	4.2

Hình 2.5. So sánh mạng MobileNet truyền thống và Depthwise Separable Convolution [3]

Qua hình 2.5 có thể thấy, MobileNet chỉ giảm 1% về độ chính xác nhưng lại giảm lượng parameter của mô hình và số lượng phép tính toán giảm đi rất nhiều lần.

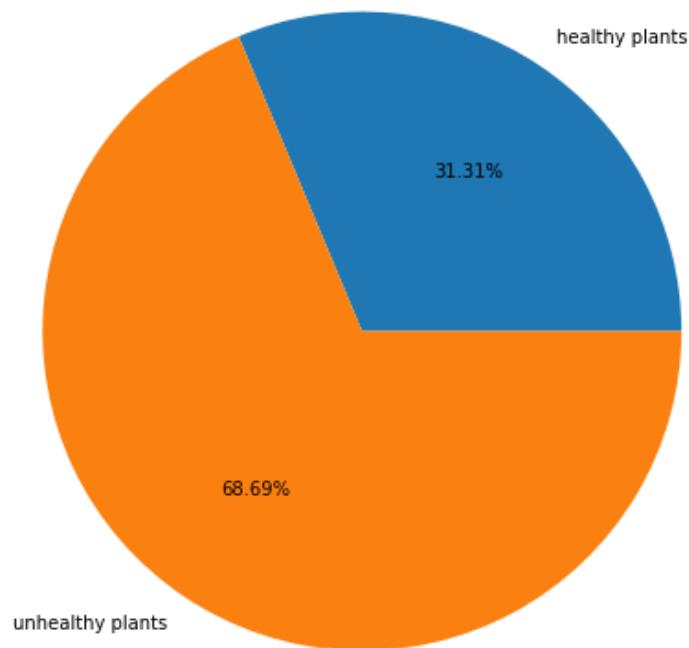
2.2. Nhận dạng bệnh trên lá cây

2.2.1. Tập dữ liệu

Bộ dữ liệu được lấy từ Kaggle (nguồn: <https://www.kaggle.com/vipoooool/new-plant-diseases-dataset>), từ bộ dữ liệu gốc Plant Village được thu thập bởi đội nghiên cứu và phát triển của trường Đại học Penn State, được Samir Bhattacharai xử lý. Gồm hơn 87.000 ảnh màu, gồm lá cây bị bệnh và khoẻ mạnh của 14 loài cây khác nhau gồm cây táo, việt quất, anh đào, ngô, nho, cam, đào, ót chuông, khoai tây, bí đỏ, cà chua, cam, mâm xôi và dâu tây. Bộ

dữ liệu gồm 17 loại lá cây bệnh do nấm, 4 loại lá cây bệnh do vi khuẩn, 2 loại lá cây bệnh do vi-rút, 2 loại lá cây bệnh do nấm mốc và 1 loại lá cây bệnh bị gây ra bởi côn trùng, và 12 loại lá cây của cây khoẻ mạnh, chia thành 38 lớp được gán theo tên khoa học bệnh cây trồng.

Bộ dữ liệu được chụp trong điều kiện phòng thí nghiệm và đã sử dụng sáu kỹ thuật nâng cao khác nhau để tăng kích thước tập dữ liệu. Các kỹ thuật này là lật ảnh, chỉnh gamma, chèn nhiễu, tăng màu PCA. Tỉ lệ tập dữ liệu được thể hiện ở hình 2.6, gồm 68,69% lá cây trồng bị bệnh và 31,31% lá cây trồng khoẻ mạnh. Trong đó, gồm 70.295 ảnh cho tập huấn luyện, và 17.572 ảnh cho tập xác thực. Ngoài ra, bộ dữ liệu còn cung cấp thêm 33 ảnh khác dùng cho việc dự đoán.



Hình 2.6. Tỉ lệ số lượng ảnh lá cây bị nhiễm bệnh và lá cây khoẻ mạnh xấp xỉ 3/4

Tập dữ liệu PlantVillage được thống kê như trong bảng 2.1 như sau:

STT	Loại cây	Tên bệnh	Tên khoa học	Nguyên nhân gây bệnh	Số lượng ảnh
1	Táo	Khoẻ mạnh			2510
2	Táo	Ghẻ	Apple scab	Nấm	2520

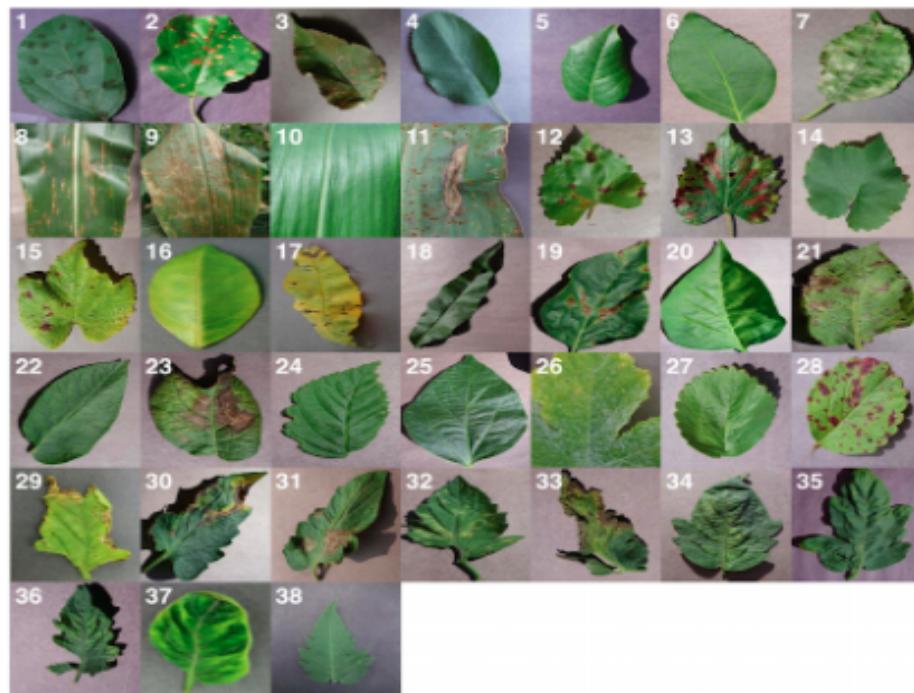
STT	Loại cây	Tên bệnh	Tên khoa học	Nguyên nhân gây bệnh	Số lượng ảnh
3	Táo	Thối	Black rot	Nấm	2484
4	Táo	Nấm cedar	Cedar apple rust	Nấm	2200
5	Việt quất	Khoẻ mạnh			2270
6	Anh đào	Khoẻ mạnh			2282
7	Anh đào	Phấn trắng	Powdery mildew	Nấm	2104
8	Ngô	Khoẻ mạnh			2324
9	Ngô	Đốm lá nhỏ	Cercospora leaf spot	Nấm	2052
10	Ngô	Gỉ sắt	Common rust	Nấm	2384
11	Ngô	Đốm lá phượng Bắc	Northern Leaf Blight	Nấm	2385
12	Nho	Khoẻ mạnh			2385
13	Nho	Thối	Black rot	Nấm	2360
14	Nho	Esca	Esca (Black measles)	Nấm	2400
15	Nho	Cháy lá sớm	Leaf blight (Isariopsis)	Nấm	2152
16	Cam	Vàng lá, gân xanh	Haunglongbing (Citrus greening)	Vi khuẩn	2513

STT	Loại cây	Tên bệnh	Tên khoa học	Nguyên nhân gây bệnh	Số lượng ảnh
17	Đào	Khoẻ mạnh			2160
18	Đào	Đốm vi khuẩn	Bacterial spot	Vi khuẩn	2297
19	Ót chuông	Khoẻ mạnh			2485
20	Ót chuông	Đốm vi khuẩn	Bacterial spot	Vi khuẩn	2391
21	Khoai tây	Khoẻ mạnh			2280
22	Khoai tây	Cháy lá sớm	Early blight	Nấm	2424
23	Khoai tây	Cháy lá muộn	Late blight	Nấm	2424
24	Mâm xôi	Khoẻ mạnh			2226
25	Đậu tương	Khoẻ mạnh			2527
26	Bí đỏ	Phấn trắng	Powdery mildew	Nấm	2170
27	Dâu tây	Khoẻ mạnh			2280
28	Dâu tây	Cháy lá	Leaf scorch	Nấm	2218
29	Cà chua	Khoẻ mạnh			2407
30	Cà chua	Đốm vi khuẩn	Bacterial spot	Vi khuẩn	2127
31	Cà chua	Héo lá sớm	Early blight	Nấm	2400
32	Cà chua	Mốc sương	Late blight	Nấm	2314

STT	Loại cây	Tên bệnh	Tên khoa học	Nguyên nhân gây bệnh	Số lượng ảnh
33	Cà chua	Bệnh đốm xám	Leaf mold	Nấm	2352
34	Cà chua	Đốm lá Septoria	Septoria leaf spot	Nấm mốc	2181
35	Cà chua	Ve nhện	Spider mites	Côn trùng	2176
36	Cà chua	Đốm nâu	Target spot	Nấm	2284
37	Cà chua	Hoa lá cà chua	Tomato mosaic virus	Vi - rút	2238
38	Cà chua	Héo vàng (Héo rũ)	Tomato yellow leaf curl virus	Vi - rút	2451

Bảng 2.1. Thống kê số lượng ảnh trong tập dữ liệu

Mô tả cái nhìn tổng thể về tập dữ liệu qua hình 2.7. Với những loại cây khác nhau, hình dáng lá, màu sắc cũng khác nhau. Những vết bệnh của từng loại cũng khác nhau.



Hình 2.7. Minh họa bộ dữ liệu PlantVillage [4]

Hình 2.7 thể hiện bộ dữ liệu PlantVillage, trong đó bao gồm:

- 1) Apple Scab - Bệnh ghẻ ở lá cây táo
- 2) Apple Black Rot - Bệnh thối đen ở lá cây táo
- 3) Apple Cedar Rust - Bệnh gỉ sét tuyết tùng ở lá cây táo
- 4) Apple healthy - Lá táo khoẻ mạnh
- 5) Blueberry healthy - Lá cây việt quất khoẻ mạnh
- 6) Cherry healthy - Lá cây anh đào khoẻ mạnh
- 7) Cherry Powdery Mildew - Bệnh phấn trắng ở lá cây anh đào
- 8) Corn Gray Leaf Spot - Bệnh đốm lá nhỏ ở lá cây ngô
- 9) Corn Common Rust - Bệnh gỉ sét ở lá cây ngô
- 10) Corn healthy - Lá cây ngô khoẻ mạnh
- 11) Corn Northern Leaf Blight - Bệnh đốm lá lớn ở lá cây ngô
- 12) Grape Black Rot - Bệnh thối đen ở lá cây nho
- 13) Grape Black Measles (Esca) - Bệnh Sởi đen hoặc Sởi Tây Ban Nha ở lá cây nho
- 14) Grape Healthy - Lá cây nho khoẻ mạnh
- 15) Grape Leaf Blight - Bệnh đốm lá ở lá cây nho
- 16) Orange Haunglongbing (Citrus Greening) - Bệnh vàng lá ở lá cây cam
- 17) Peach Bacterial Spot - Bệnh đốm vi khuẩn ở lá cây đào
- 18) Peach healthy - Lá cây đào khoẻ mạnh
- 19) Bell Pepper Bacterial Spot - Bệnh đốm lá ở lá cây ớt chuông
- 20) Bell Pepper healthy - Lá cây ớt chuông khoẻ mạnh
- 21) Potato Early Blight - Bệnh héo sớm ở lá cây khoai tây
- 22) Potato healthy - Lá cây khoai tây khoẻ mạnh
- 23) Potato Late Blight - Bệnh héo muộn ở lá cây khoai tây
- 24) Raspberry healthy - Lá cây mâm xôi khoẻ mạnh
- 25) Soybean healthy - Lá cây đậu nành khoẻ mạnh
- 26) Squash Powdery Mildew - Bệnh phấn trắng ở lá bí đỏ
- 27) Strawberry Healthy - Lá cây dâu tây khoẻ mạnh
- 28) Strawberry Leaf Scorch - Bệnh cháy lá ở lá cây dâu tây
- 29) Tomato Bacterial Spot - Bệnh đốm vi khuẩn ở lá cây cà chua
- 30) Tomato Early Blight - Bệnh héo sớm ở lá cây cà chua
- 31) Tomato Late Blight - Bệnh héo muộn ở lá cây cà chua
- 32) Tomato Leaf Mold - Bệnh mốc lá ở lá cây cà chua
- 33) Tomato Septoria Leaf Spot - Bệnh đốm lá Septoria ở cây cà chua
- 34) Tomato Two Spotted Spider Mite - Bệnh đốm do ve nhện cắn ở lá cây cà chua

- 35) Tomato Target Spot - Bệnh đốm nâu ở lá cây cà chua
- 36) Tomato Mosaic Virus - Bệnh khăm ở lá cây cà chua
- 37) Tomato Yellow Leaf Curl Virus - Bệnh héo vàng (héo rũ) ở lá cây cà chua
- 38) Tomato healthy - Lá cây cà chua khoẻ mạnh

2.2.2. Cài đặt, đánh giá mô hình và kết quả

Trong phần này, KLTN trình bày về việc cài đặt, kiểm thử và đánh giá việc học chuyển tiếp của mô hình học máy đã tìm hiểu ở phần trước vào tập dữ liệu PlantVillage. Nhận thấy mô hình MobileNet phân loại khá tốt trên tập ImageNet. Vì vậy, thay vì xây dựng lại mô hình từ đầu, KLTN sẽ sử dụng lại mô hình đã được huấn luyện trên tập dữ liệu đó, giữ lại những trọng số để huấn luyện lại bộ dữ liệu lá cây trồng bị bệnh nhanh hơn và chuẩn xác hơn.

Mô hình được huấn luyện sử dụng Google Coblab, sử dụng mô hình MobileNet đã được huấn luyện sẵn với tập huấn luyện là bộ dữ liệu ImageNet. Mô hình giữ nguyên các tham số đã được huấn luyện. Với đầu vào là ảnh màu có kích thước 224x224 với 3 kênh là đỏ, xanh lam và xanh lục. Hình 2.8 đã mô tả việc thiết lập các giá trị đầu vào và đầu ra khi thực hiện xử lý từng lớp.

```

✓  ⏎ model.summary()

↳ Model: "LeafDisease_MobileNet"

+----+-----+-----+
| Layer (type)        | Output Shape | Param # |
+----+-----+-----+
| input_2 (InputLayer) | [(None, 224, 224, 3)] | 0         |
| mobilenet_1.00_224 (Functional) | (None, 7, 7, 1024) | 3228864   |
| global_average_pooling2d (GlobalAveragePooling2D) | (None, 1024) | 0         |
| dropout (Dropout)    | (None, 1024) | 0         |
| dense (Dense)       | (None, 38)   | 38950     |
+----+-----+-----+
Total params: 3,267,814
Trainable params: 38,950
Non-trainable params: 3,228,864

```

Hình 2.8. Các tham số, thiết lập đầu vào, đầu ra của các lớp

Mô hình nhận các tham số như bảng 2.2 dưới đây:

Tham số mô hình	Giá trị
Tổng số mẫu huấn luyện (Training)	70.295
Tổng số mẫu xác thực (Validation)	17.57
Epochs	25
Batch_size	32
Step_per_epoch	150
Validation_steps	100
Learning_rate (Tốc độ học)	0.001

Bảng 2.2. Các tham số cho mô hình mạng nơ-ron tích chập.

Qua từng epoch, độ chính xác được tăng dần. Hình 2.9 thể hiện kết quả của 5 lần epochs cuối. mô hình đang thể hiện khá tốt và tổng quát hơn cho cả quá trình mô hình được huấn luyện được thể hiện qua hình 2.10 biểu thị độ chính xác và độ mất mát trên bộ dữ liệu được huấn luyện.

```

Epoch 20/25
150/150 [=====] - 59s 396ms/step - loss: 0.2271 - categorical_accuracy: 0.9258
Epoch 21/25
150/150 [=====] - 59s 395ms/step - loss: 0.2134 - categorical_accuracy: 0.9292
Epoch 22/25
150/150 [=====] - 59s 396ms/step - loss: 0.2187 - categorical_accuracy: 0.9242
Epoch 23/25
150/150 [=====] - 61s 406ms/step - loss: 0.2155 - categorical_accuracy: 0.9271
Epoch 24/25
150/150 [=====] - 60s 397ms/step - loss: 0.2133 - categorical_accuracy: 0.9298
Epoch 25/25
150/150 [=====] - 59s 396ms/step - loss: 0.2078 - categorical_accuracy: 0.9344

```

Hình 2.9. Kết quả sau khi thực hiện 5 epochs cuối



Hình 2.10. Biểu đồ biểu diễn độ chính xác, mất mát của tập huấn luyện và kiểm thử

Cùng với những tham số đó, bộ dữ liệu cũng huấn luyện trên những mô hình học máy khác để so sánh. Qua bảng , nhận thấy kết quả học chuyển tiếp của MobileNet trên tập dữ liệu này tốt hơn so với những mô hình còn lại do tham số chưa được tối ưu hoá ở những mô hình mạng này.

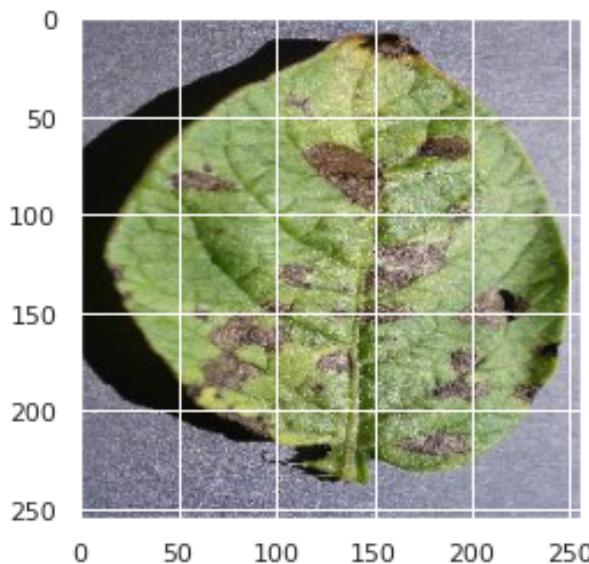
Mô hình	MobileNet	VGG16	InceptionV3
Độ chính xác trên tập validation	95.22%	79.54%	86.4%
Độ mất mát trên tập validation	14.42%	88.62%	42.56%

Bảng 2.2. Kết quả huấn luyện của các mô hình no-ron học chuyển tiếp

Kết quả nhận diện tương đối tốt trên mẫu tập kiểm thử.

```
▶ predict_disease('/content/test/test/PotatoEarlyBlight2.JPG')
```

```
□ 1/1 [=====] - 0s 19ms/step
Actual class : /content/test/test/PotatoEarlyBlight2.JPG
Predicted class name: Potato_Early_blight
Label index: 20
Confidence: 98.66%
```



Hình 2.11. Kết quả nhận diện bệnh héo sóm lá khoai tây

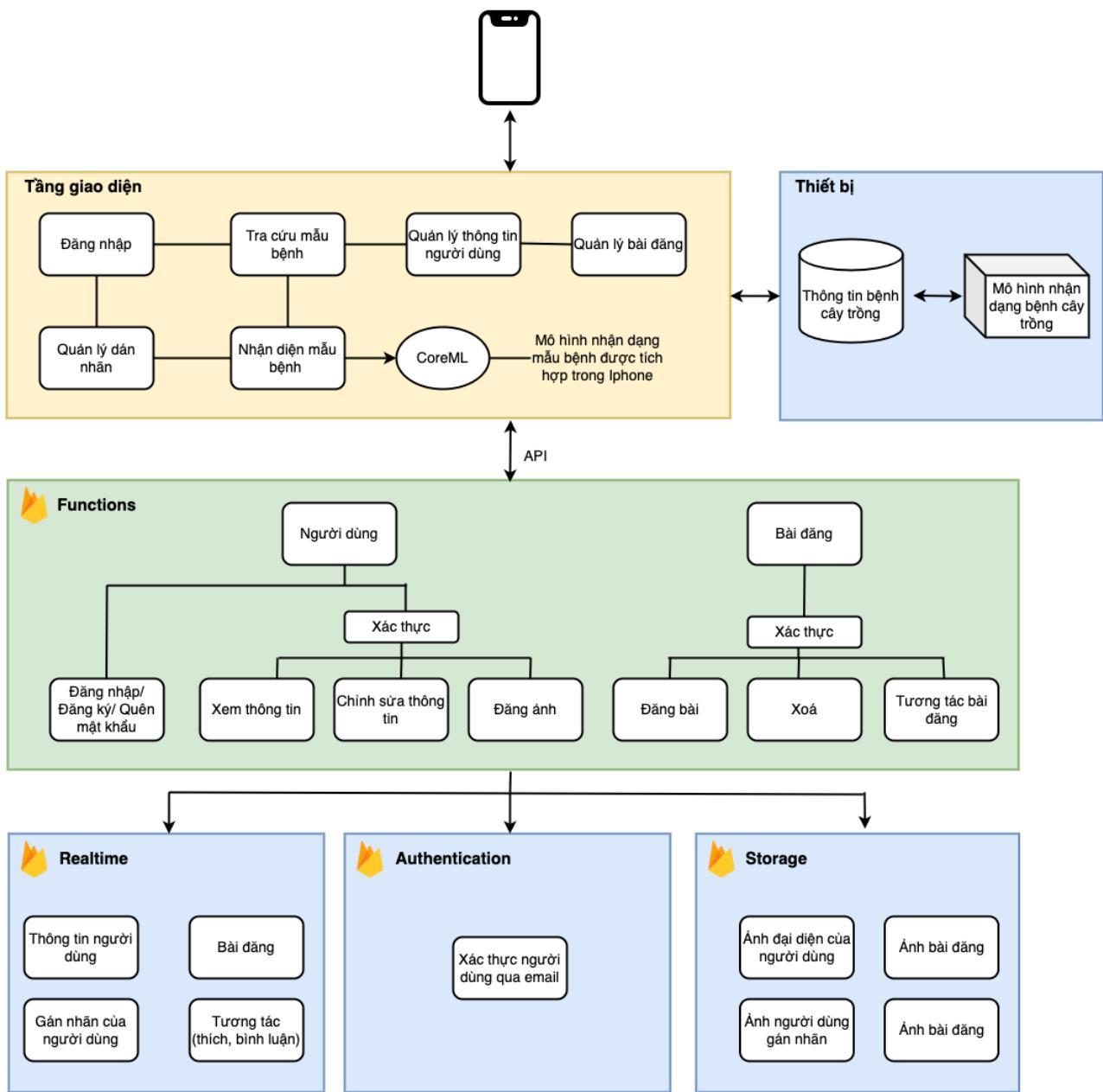
Trong chương này, khoá luận đã giới thiệu sơ lược các phương pháp học máy nhận diện bệnh cây trồng. Khoá luận cũng tập trung tìm hiểu mô hình mạng MobileNet và áp dụng học chuyển tiếp mô hình mạng này vào tập dữ liệu PlantVillage để áp dụng xây dựng ứng dụng nhận diện 38 loại bệnh cây trồng. Trong chương tiếp theo, KLTN sẽ trình bày việc thiết kế, phân tích và xây dựng ứng dụng, cũng như tích hợp mô hình học máy đã được huấn luyện vào ứng dụng điện thoại hệ điều hành iOS.

CHƯƠNG 3. PHÂN TÍCH THIẾT KẾ HỆ THỐNG

Với chương 3, khoá luận sẽ tập trung vào việc nghiên cứu và xây dựng hệ thống nhận diện bệnh lá cây tròng. KLTN cũng sẽ trình bày chi tiết cách đưa một mô hình học sâu nhận diện bệnh cây tròng vào ứng dụng di động iOS cũng như là quy trình vận hành, trao đổi giữa các mô đun, các thành phần trong hệ thống giao tiếp với nhau trong chương này.

3.1. Tổng quan hệ thống

Sau khi phân tích và đánh giá những yêu cầu chức năng của hệ thống, KLTN đã chọn nền tảng Firebase để phát triển ứng dụng. Firebase là một dịch vụ hệ thống backend được Google cung cấp sẵn cho ứng dụng, giúp cho người dùng phát triển, triển khai và mở rộng ứng dụng di động nhanh hơn. Những SDK dành cho client sẽ tương tác trực tiếp với các dịch vụ này. Vì vậy, nó khác mô hình truyền thống quen thuộc, yêu cầu phải viết chương trình cho cả frontend lẫn backend. Với mô hình truyền thống, frontend tương tác với backend thông qua API, backend sẽ chịu trách nhiệm xử lý dữ liệu. Tuy nhiên với ứng dụng sử dụng Firebase, thì có thể không cần backend nữa, phần việc xử lý sẽ giao cho client truy vấn trực tiếp đến cơ sở dữ liệu (được cung cấp bởi Firebase). Theo đó, kiến trúc hệ thống sẽ bao quát như sau:



Hình 3.1. Kiến trúc tổng quan hệ thống

Về cơ bản, hệ thống cũng chia thành 3 tầng chính:

- Tầng 1: Tầng CSDL

Tầng CSDL là tầng quan trọng và không thể thiếu trong bất kỳ hệ thống nào. Tầng này chịu trách nhiệm lưu trữ, ghi, đọc, sửa, xoá và cập nhật dữ liệu. Với những yêu cầu chức năng của hệ thống, khoá luận đã tách cơ sở dữ liệu thành hai phần chính. Phần thứ nhất, dữ liệu liên quan đến thông tin bệnh cây trồng, sẽ được lưu trực tiếp tại thiết bị, giúp cho việc truy xuất và tìm kiếm thông tin về bệnh cây trồng trở nên nhanh hơn. Phần thứ hai, liên quan đến

dữ liệu người dùng, cần hiển thị theo thời gian thực, hệ thống đã chọn một vài dịch vụ của Firebase để lưu trữ, quản lý hình ảnh, thông tin của người dùng. Hệ thống đã sử dụng Authentication để xác thực, lưu trữ thông tin người dùng qua địa chỉ email. Bên cạnh đó, hệ thống còn sử dụng Realtime Database để lưu trữ, cũng như cho phép người dùng cập nhật lại thông tin theo thời gian thực, quản lý bài đăng và tương tác của người dùng, quản lý lịch sử tra cứu bệnh của người dùng, cũng như quản lý gán nhãn dữ liệu của người dùng cho mỗi loại bệnh cây trồng. Mỗi khi người dùng đăng ảnh lên hệ thống, cũng được lưu trữ tại Firebase Storage.

- Tầng 2: Kết nối với các dịch vụ của Firebase

Với mô hình truyền thống, thì tầng 2 thường là tầng máy chủ, nhưng hệ thống trong khoá luận sử dụng các dịch vụ của Firebase nên tầng này được thay thế bằng việc tương tác với cơ sở dữ liệu trên Firebase thông qua các API mà Firebase cung cấp. Và được thực hiện ở phía client.

- Tầng 3: Tầng giao diện

Tầng này bao gồm các thiết bị người dùng cuối, cụ thể ở đây là các thiết bị chạy hệ điều hành iOS. Tầng này chịu trách nhiệm quản lý, thực hiện việc hiển thị các thông tin đến người dùng. Người dùng có thể thực hiện các chức năng đăng nhập, đăng ký, tra cứu bệnh, tham khảo ý kiến cộng đồng, tìm các bài viết về bệnh cây trồng...

3.2. Tầng CSDL

Ở phần này, KLTN sẽ trình bày về việc phân tích và xây dựng tầng CSDL. Cùng với đó là, cách kết nối với Firebase để lưu trữ, đọc ghi các dữ liệu và hình ảnh của người dùng. Hệ thống sẽ tách cơ sở dữ liệu thành hai phần, được thiết kế và lưu trữ tại các phần khác nhau.

3.2.1. Giới thiệu tổng quan

Đầu tiên, với phần dữ liệu về bệnh cây trồng. KL đã thu thập và tổng hợp những thông tin về bệnh cây trồng, bao gồm tên bệnh, loại bệnh và mức độ bệnh, cũng như cách phòng trị bệnh cây trồng từ nhiều nguồn khác nhau theo từng loại bệnh của tập dữ liệu Plant Village. Vì dữ liệu không quá lớn, cũng như để hạn chế độ trễ hiển thị thông tin bệnh cây trồng, tăng trải nghiệm của người dùng, dữ liệu thông tin về bệnh cây trồng đã được lưu tại ứng dụng. Về dữ liệu người dùng, hệ thống đã chọn sử dụng ba dịch vụ, gồm Realtime Database, Authentication và Storage của Firebase để lưu trữ, đọc, ghi, sửa và xoá dữ liệu và xác thực người dùng.

3.2.2. Tìm hiểu và phân tích, thiết kế dữ liệu trên Firebase

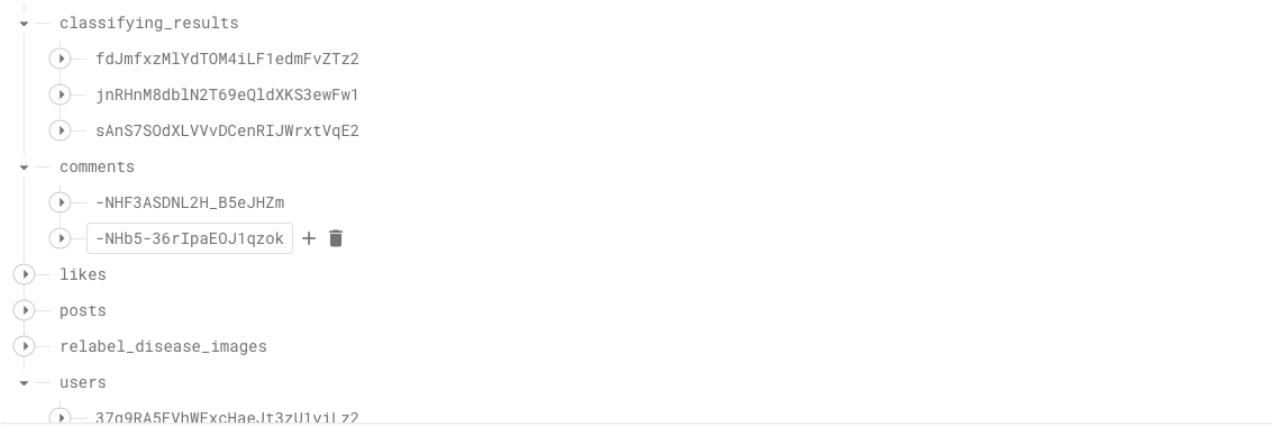
Hệ thống sử dụng 3 dịch vụ của Firebase gồm: Authentication, Storage và Realtime Database:

- Authentication: Dịch vụ này hỗ trợ ứng dụng một số phương pháp xác thực tài khoản, trong đó hệ thống sử dụng xác thực email của dịch vụ này.
- Storage: Cung cấp một không gian lưu trữ dữ liệu, trong đó ứng dụng dùng Storage để lưu trữ ảnh liên quan đến người dùng, ảnh dữ liệu bệnh người dùng tra cứu và ảnh người dùng gán nhãn lại cho bệnh cây trồng.
- Realtime Database: Đây là một cơ sở dữ liệu NoSQL. Mỗi mục trong cơ sở dữ liệu được lưu trữ dưới dạng tên khoá cùng với giá trị của nó, và lưu dưới dạng JSON, không giống với MySQL, hệ cơ sở dữ liệu quan hệ. Dữ liệu được lưu trữ và đồng bộ hoá theo thời gian thực với mỗi client được kết nối. Khi dữ liệu được thay đổi, mọi thiết bị được kết nối có thể nhận thông tin một cách nhanh chóng.

3.2.2.2. Thiết kế cơ sở dữ liệu

Realtime Database là một cơ sở dữ liệu NoSQL. Mỗi mục trong cơ sở dữ liệu được lưu trữ dưới dạng tên khoá cùng với giá trị của nó, dưới dạng JSON. Dữ liệu được lưu trữ và đồng bộ hoá theo thời gian thực với mỗi client được kết nối.

Kiến trúc tổng quát cơ sở dữ liệu Realtime Database được thiết kế như hình dưới đây:



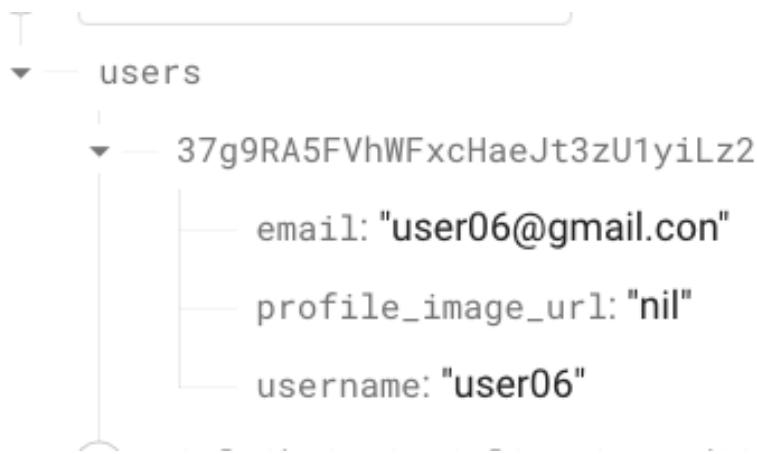
Hình 3.2. Cấu trúc cây JSON của Realtime Database

Cơ sở dữ liệu bao gồm:

- users: lưu trữ thông tin người dùng bao gồm tên người dùng, địa chỉ email và ảnh đại diện người dùng.

Trường	Mô tả	Kiểu dữ liệu
user_name	Tên người dùng	string
email	Địa chỉ mail đăng nhập	string
password	Mật khẩu đăng nhập	string

Bảng 3.1. Những thuộc tính lưu trong nhánh users



Hình 3.3. Dữ liệu users

- classifying_results: lưu trữ thông tin tra cứu bệnh mà người dùng muốn lưu lại gồm có như sau:

Trường	Mô tả	Kiểu dữ liệu
disease_name	Tên bệnh cây	string
plant_name	Tên loài cây	string
image_width	Chiều rộng ảnh bệnh	integer
image_height	Chiều dài ảnh bệnh	integer
certainty	Phần trăm dự đoán	double

creation_date	Thời gian tạo bản ghi	date
---------------	-----------------------	------

Bảng 3.2. Những thuộc tính lưu trong nhánh *classifying_results*

```

- classifying_results
  - jnRHnM8db1N2T69eQldXKS3ewFw1
    - -NHgIAbziqM8IbVMSDUs
      certainty: 0.8500000238418579
      creation_date: 1669338022.41496
      disease_name: "Bệnh đốm lá nhỏ"
      image_height: 200
      image_url: "https://firebasestorage.googleapis.com:443/<...>
      image_width: 300
      plant_name: "Ngô"
      uid: "-NHgIAbziqM8IbVMSDUs"

```

Hình 3.4. Dữ liệu *classifying_results*

- relabel_disease_images: dữ liệu người dùng gán nhãn lại được lưu tại đây

Trường	Mô tả	Kiểu dữ liệu
uid	Mã bản ghi (Khoá chính, duy nhất)	string
disease_name	Tên bệnh cây hệ thống nhận diện	string
plant_name	Tên loài cây hệ thống nhận diện	string
relabel_disease_name	Tên bệnh cây người dùng gán nhãn lại	string
relabel_plant_name	Tên loài cây người dùng gán nhãn lại	string

image_url	Đường dẫn ảnh	string
image_height	Chiều dài ảnh bệnh	int
image_width	Chiều rộng ảnh bệnh	int
creation_date	Thời gian tạo bản ghi	date

Bảng 3.3. Những thuộc tính lưu trong nhánh relabel_disease_images

```

- relabel_disease_images
  - jnRHnM8dblN2T69eQldXKS3ewFw1
    - -NHgI8vrlRHe4C6X08Tz
      creation_date: 1669338015.511856
      disease_name: "Bệnh đốm lá nhỏ"
      image_height: 200
      image_url: "https://firebasestorage.googleapis.com:443/v0/b/farm-app-5f3d0/o/images%2F1669338015.511856?alt=media&token=543a5a2a-1a20-4a20-8a20-1a201a201a20"
      image_width: 300
      plant_name: "Ngô"
      relabel_disease_name: "Sọc lá"
      relabel_plant_name: "Ngô"
      uid: "-NHgI8vrlRHe4C6X08Tz"

```

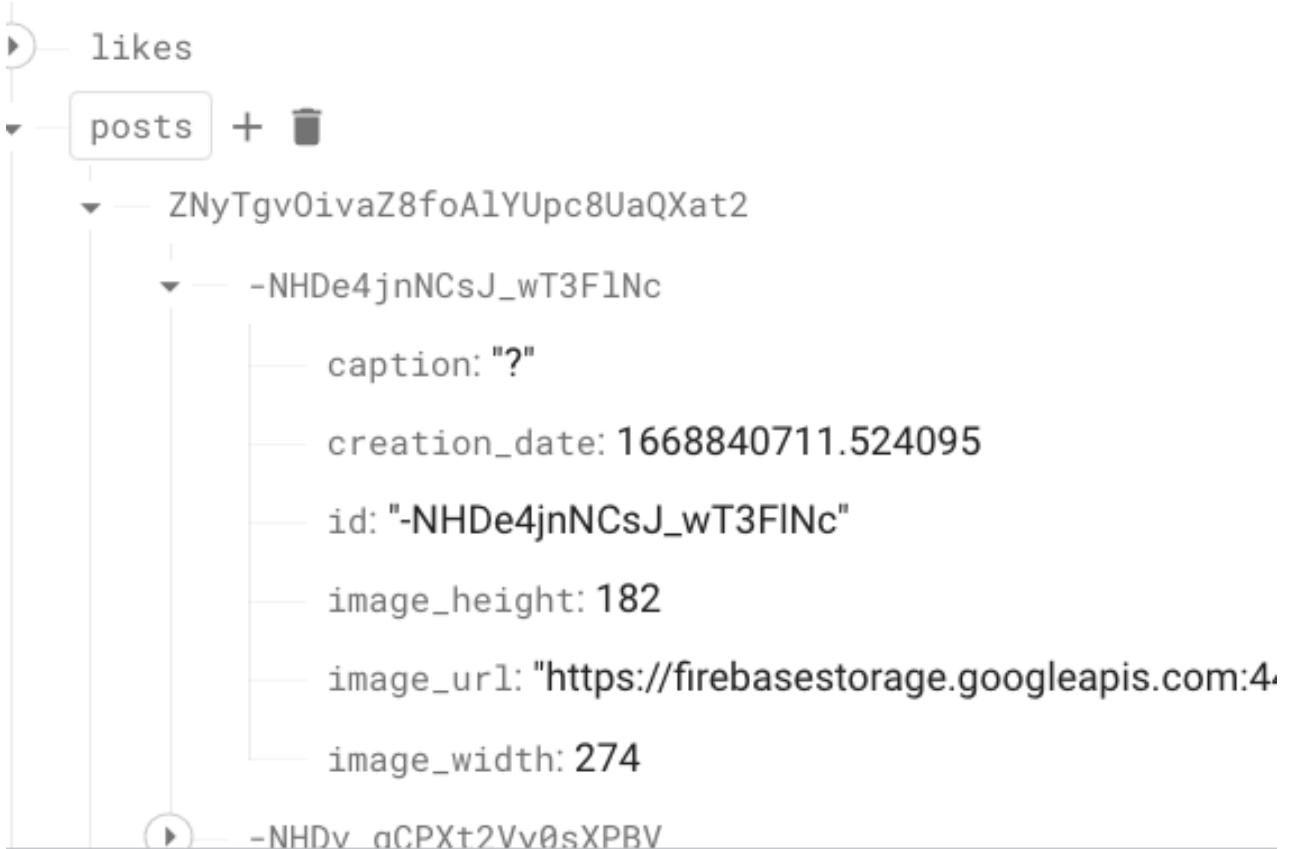
Hình 3.5. Dữ liệu relabel_disease_images

- posts: lưu thông tin bài đăng của người dùng, bao gồm:

Trường	Mô tả	Kiểu dữ liệu
uid	Mã bản ghi (Khoá chính, duy nhất)	string

Trường	Mô tả	Kiểu dữ liệu
caption	Mô tả của người dùng	string
image_url	Đường dẫn ảnh	string
image_height	Chiều dài ảnh	int
image_width	Chiều rộng ảnh	int
creation_date	Thời gian tạo bản ghi	date

Bảng 3.4. Những thuộc tính lưu trong nhánh posts.



Hình 3.6. Dữ liệu posts

- comments: lưu trữ dữ liệu bình luận bài viết của người dùng, lưu theo khoá của từng bài đăng.

Trường	Mô tả	Kiểu dữ liệu

uid	Mã bản ghi (Khoá chính, duy nhất)	string
text	Nội dung người bình luận	string
timestamp	Thời gian tạo bản ghi bài đăng	long
creation_date	Thời gian tạo bản ghi	date

Bảng 3.5. Những thuộc tính lưu trong nhánh comments.



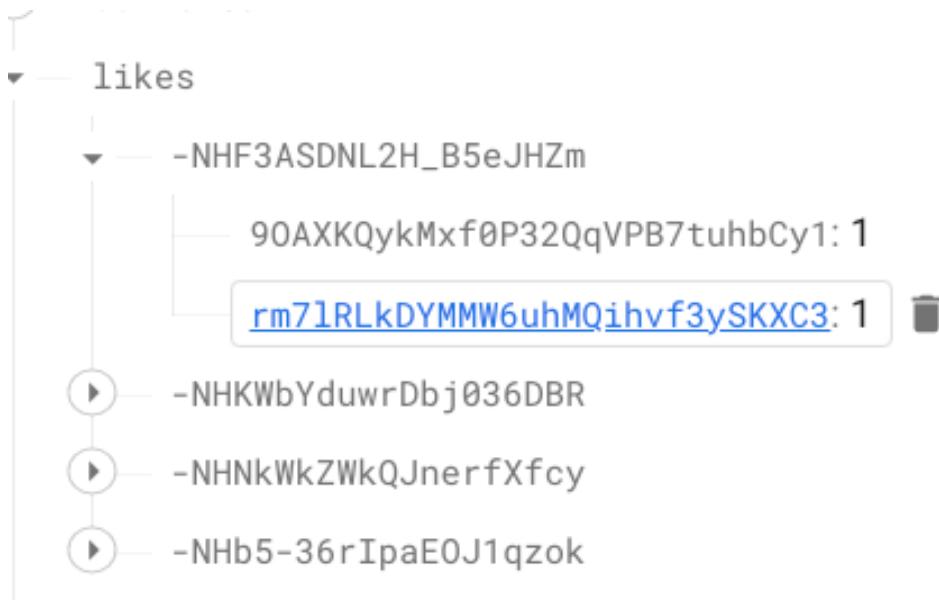
Hình 3.7. Dữ liệu nhánh comments

- likes: lưu trữ lượng thích của từng bài đăng theo từng user, nếu người dùng ấn thích lưu là 1, còn người dùng không thích, bản ghi sẽ được xoá.

Trường	Mô tả	Kiểu dữ liệu
--------	-------	--------------

user_uid	Mã người dùng ấn thích	int
----------	------------------------	-----

Bảng 3.6. Những thuộc tính lưu trong nhánh likes



Hình 3.8. Dữ liệu likes

Như đã đề cập phía trên, hệ thống sử dụng Firebase Authentication để xác thực, lưu trữ dữ liệu người dùng qua địa chỉ email, rồi cập nhật thông tin lên Realtime Database.

Identifier	Providers	Created	Signed In	User UID
user20@gmail.com	✉	Nov 24, 2022	Nov 24, 2022	jnRHnM8dblN2T69eQldXKS3ewFw1
user19@gmail.com	✉	Nov 21, 2022	Nov 21, 2022	o6L1PFQYoleVa1x5PR3B6GevXhC2
user18@gmail.com	✉	Nov 21, 2022	Nov 21, 2022	rm7IRLkDYMMW6uhMQihvf3ySKX...
user18@gmail.com	✉	Nov 21, 2022	Nov 21, 2022	D60q1M42SUNMivH3Ffb9ksUOjL...
user16@gmail.com	✉	Nov 21, 2022	Nov 21, 2022	fgOWOQvL9MTgYyKdEG0flAEgF8g2
user14@gmail.com	✉	Nov 21, 2022	Nov 21, 2022	x0OU900v4udMxQo0tTZkR1xnqD...
user10@gmail.com	✉	Nov 21, 2022	Nov 21, 2022	gesm3HzcV4WFee1Ss1GSb7qsDj...
user16@gmail.com	✉	Nov 20, 2022	Nov 20, 2022	90AXKQykMxf0P32QqVPB7tuhbC...
user15@gmail.com	✉	Nov 20, 2022	Nov 20, 2022	VwPOdk9MDtYt1fH8qL2EO9Ai2W...
user11@gmail.com	✉	Nov 19, 2022	Nov 20, 2022	49NOH3yB1bNGtuuxnlBaAMFDZ...
user09@gmail.com	✉	Nov 19, 2022	Nov 19, 2022	GfDadmmD2hagVCWd60U4K4skh...
user07@gmail.com	✉	Nov 19, 2022	Nov 19, 2022	ZNyTgvOivaZ8foAlYUpc8UaQXat2
user06@gmail.com	✉	Nov 19, 2022	Nov 19, 2022	3IIK0hqQPnQotQlISqTI3ssDqdi2
user05@gmail.com	✉	Nov 18, 2022	Nov 19, 2022	UAEnt3gLsef5PD3qLJs4s33mNPY2

Hình 3.9. Dữ liệu người dùng lưu trên Firebase Authentication

Còn Firebase Storage dùng để lưu trữ ảnh của người dùng mỗi khi thay đổi, cập nhật ảnh giao diện, bài đăng, quản lý dán nhãn và lịch sử tra cứu bệnh (Hình 3.1).

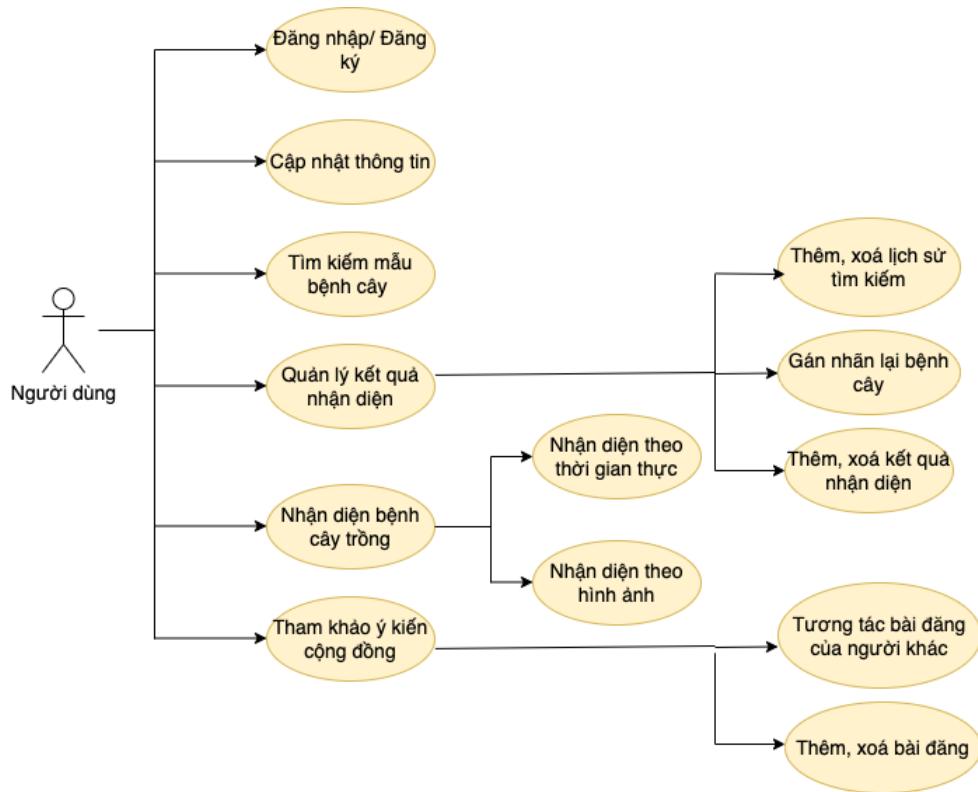
gs://plantcare-30477.appspot.com		Upload file	⋮
<input type="checkbox"/>	Name	Size	Type
<input type="checkbox"/>	classifying_results/	—	Folder
<input type="checkbox"/>	post_images/	—	Folder
<input type="checkbox"/>	profile_images/	—	Folder
<input type="checkbox"/>	relabeled_diseases_images/	—	Folder

Hình 3.10. Dữ liệu ảnh trên Storage

3.3. Tầng giao diện

Trong phân này, khoá luận sẽ trình bày các tính năng, việc phát triển và triển khai các tính năng ở tầng giao diện. Gồm có các tính năng sau:

- Tính năng đăng ký, đăng nhập.
- Tính năng Tìm kiếm bệnh cây trồng.
- Tính năng Xem chi tiết mẫu bệnh cây trồng.
- Tính năng Nhận diện mẫu bệnh bằng hình ảnh tải lên hoặc camera.
- Tính năng Nhận diện mẫu bệnh cây trồng theo thời gian thực.
- Tính năng xem, xoá lịch sử nhận diện bệnh.
- Tính năng Tham khảo ý kiến cộng đồng, quản lý lưu trữ bài đăng.
- Tính năng Thêm, xoá nhãn dán.

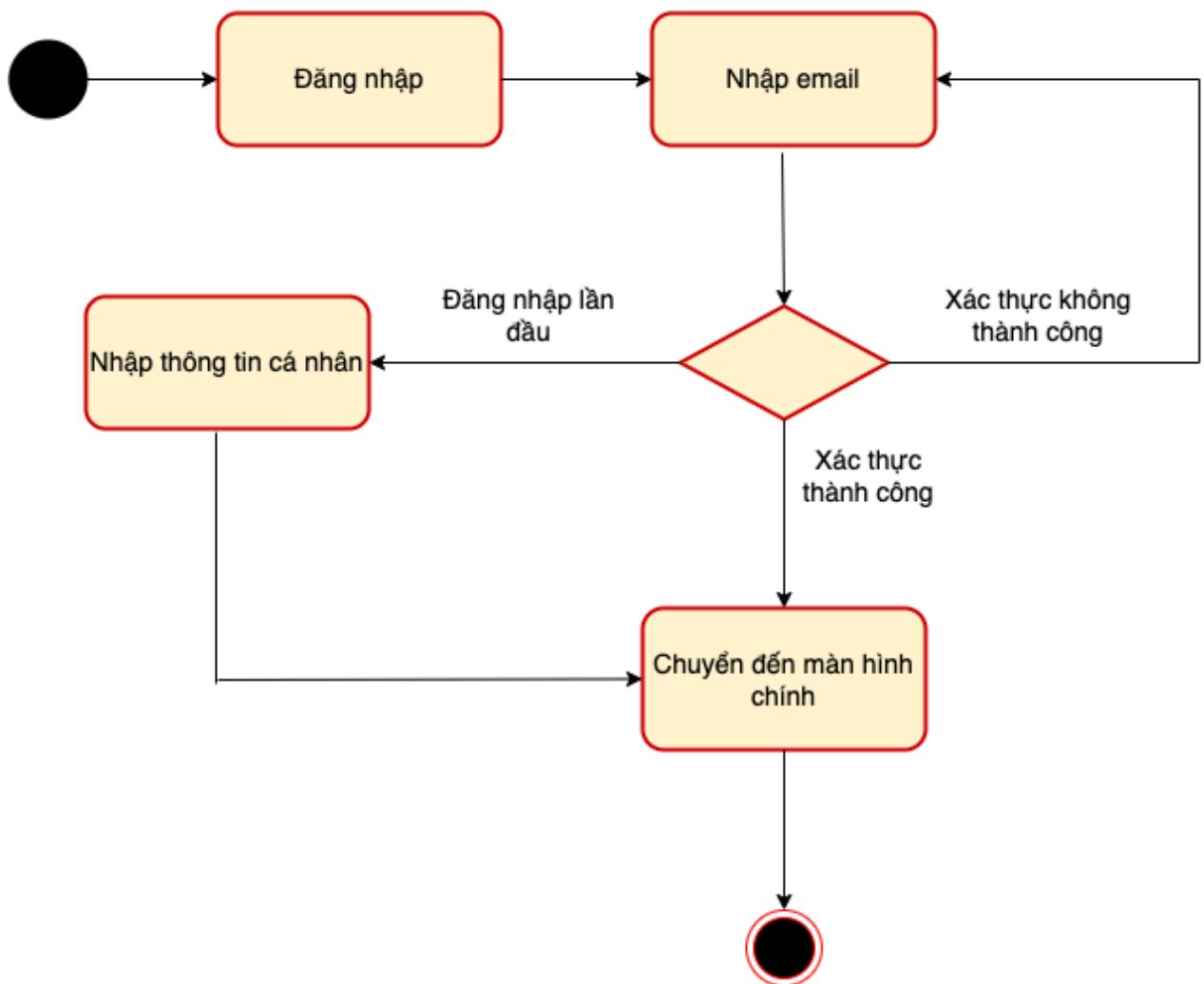


Hình 3.11. Các tính năng tầng giao diện

3.3.1. Đăng nhập/ Đăng ký

Mỗi khi người dùng có yêu cầu đăng nhập, hệ thống sẽ thực hiện xác thực sử dụng Firebase Authentication, sau khi xác thực thành công, ngay lập tức hệ thống sẽ thông báo về người

dùng và thực hiện đăng nhập. Nếu người dùng đăng nhập lần đầu, do chưa có bản ghi trong CSDL, người dùng cần nhập thông tin để tiến hành đăng nhập bằng số điện thoại hoặc email.



Hình 3.12. Biểu đồ hoạt động cho tính năng Đăng nhập

Hình 3.4 mô tả việc thực hiện đăng nhập vào ứng dụng, đầu vào là địa chỉ email của người dùng. Sau khi đăng nhập thành công, ứng dụng cho phép người dùng sử dụng các tính năng khác như lưu lịch sử tìm kiếm, bình luận bài viết cũng như đăng câu hỏi và quản lý các nhãn dán.

Tên ca sử dụng	Đăng nhập hệ thống
Tác nhân	Tất cả mọi người
Mô tả	Người dùng đăng nhập vào hệ thống bằng email đã đăng ký

Điều kiện đầu	<ol style="list-style-type: none"> Người dùng đã đăng ký tài khoản Ứng dụng đang hiển thị màn hình đăng nhập
Điều kiện cuối	Khi đăng nhập thành công, ứng dụng chuyển tới màn hình chính
Luồng cơ bản	<ol style="list-style-type: none"> Người dùng nhập thông tin đăng nhập Ứng dụng xác thực đăng nhập và cho phép người dùng truy cập và sử dụng ứng dụng
Luồng phụ	<ol style="list-style-type: none"> Người dùng điền sai email, hiển thị cho người dùng biết nhập sai và yêu cầu nhập lại Trường thông tin bị thiếu, không cho phép người dùng đăng nhập
Điểm mở rộng	Không có

Bảng 3.7. Mô tả ca sử dụng chức năng “Đăng nhập”

Nếu người dùng chưa đăng nhập, người dùng có thể điều hướng đến màn đăng ký, người dùng phải nhập đủ tên người dùng, địa chỉ email và mật khẩu.

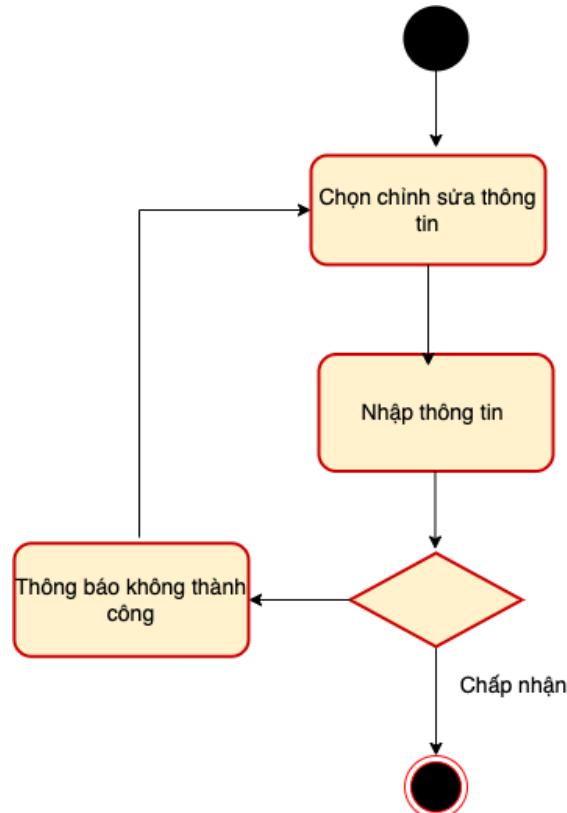
Tên ca sử dụng	Đăng ký tài khoản
Tác nhân	Tất cả mọi người
Mô tả	Người dùng đăng ký tài khoản sử dụng bằng email
Điều kiện đầu	<ol style="list-style-type: none"> Người dùng đã có email Ứng dụng đang hiển thị màn hình đăng ký
Điều kiện cuối	Khi đăng ký thành công, ứng dụng chuyển tới màn hình đăng nhập
Luồng cơ bản	<ol style="list-style-type: none"> Người dùng chọn đăng ký bằng tài khoản bằng email hoặc số điện thoại ở màn hình đăng nhập, nhập tên, mật khẩu và nhập lại mật khẩu

Tên ca sử dụng	Đăng ký tài khoản
	2. Ứng dụng hiển thị đăng ký thành công và hiển thị trang đăng nhập
Luồng phụ	1. Nếu email/số điện thoại người dùng tồn tại thì hiển thị với nội dung người dùng đã tồn tại 2. Nếu trường thông tin nào còn thiếu, yêu cầu người dùng nhập thêm
Điểm mở rộng	Không có

Bảng 3.8. Mô tả ca sử dụng chức năng “Đăng ký”

3.3.2. Cập nhật thông tin

Chức năng này cho phép người dùng cập nhật lại thông tin bao gồm tên người dùng, địa chỉ email, ảnh đại diện và mật khẩu.



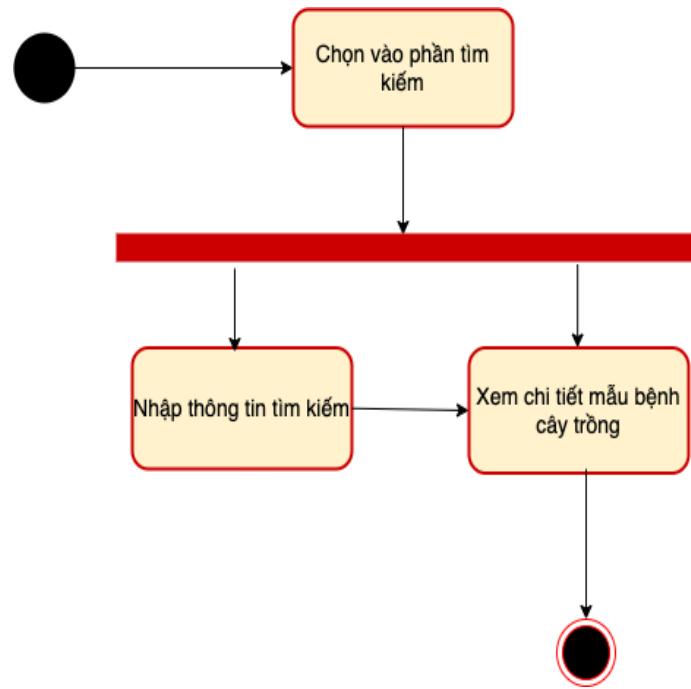
Hình 3.13. Biểu đồ hoạt động cho tính năng “Cập nhật thông tin”

Tên ca sử dụng	Cập nhật thông tin
Tác nhân	Tất cả mọi người
Mô tả	Người dùng sửa thông tin cá nhân
Điều kiện đầu	Người dùng đã đăng nhập thành công
Điều kiện cuối	Ứng dụng hiển thị được thông tin cá nhân sau khi sửa
Luồng cơ bản	1. Người dùng chọn vào phần sửa thông tin cá nhân: tên, số điện thoại, email, mật khẩu 2. Nhập vào những thông tin người dùng muốn thay đổi 3. Sau khi ấn nút “Cập nhật”, hiển thị ra màn hình những thông tin đã được thay đổi
Luồng phụ	Không có
Điểm mở rộng	Không có

Bảng 3.9. Mô tả ca sử dụng chức năng “Cập nhật thông tin”

3.3.3. Tìm kiếm mẫu bệnh cây

Chức năng này cho phép người dùng tìm kiếm, tham khảo các loại bệnh cây trồng. Người dùng không thể thao tác chỉnh sửa, xoá trên các bản ghi này, việc cập nhật dữ liệu cũng được thực hiện một cách thủ công. Khi người dùng ấn tìm kiếm, hệ thống sẽ trả lại cho người dùng ảnh và các thông tin liên quan đến mẫu bệnh cây trồng.



Hình 3.14. Biểu đồ hoạt động cho chức năng Tìm kiếm mẫu bệnh cây tròng

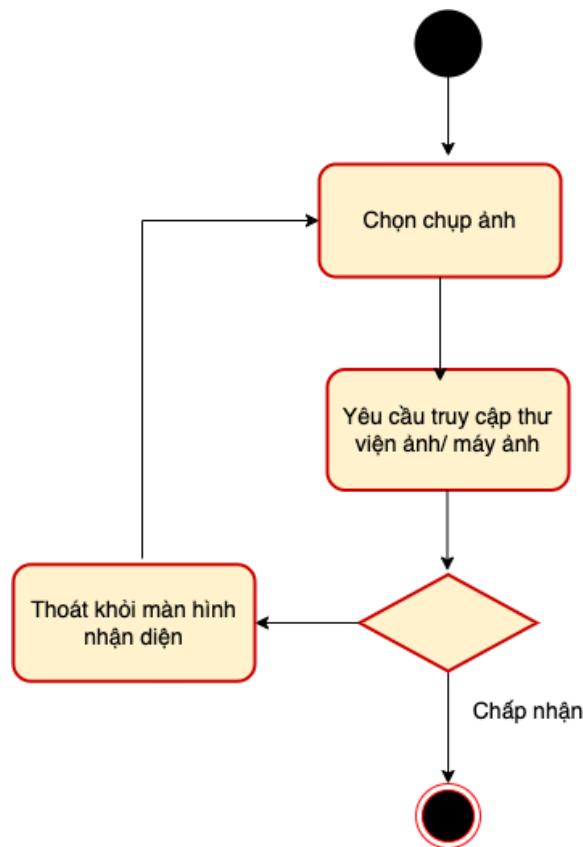
Tên ca sử dụng	Tìm kiếm mẫu bệnh cây tròng
Tác nhân	Tất cả mọi người
Mô tả	Người dùng tìm kiếm, tham khảo các bài viết liên quan đến bệnh cây tròng
Điều kiện đầu	Không có
Điều kiện cuối	Ứng dụng hiển thị được thông tin bệnh cây tròng theo bài viết được chọn
Luồng cơ bản	1. Người dùng chọn phần tìm kiếm, nhập thông tin cần tìm kiếm 2. Sau đó, hiển thị một danh sách các bệnh liên quan
Luồng phụ	Không có
Điểm mở rộng	Không có

Bảng 3.10. Mô tả ca sử dụng chức năng “Tìm kiếm”

3.3.4. Nhận diện mẫu bệnh cây trồng

Hoạt động của chức năng này được thể hiện qua hình. Với chức năng này, người dùng sẽ chụp ảnh, hoặc chọn ảnh và ứng dụng sẽ hiển thị được thông tin liên quan đến mẫu bệnh. Mô hình nhận diện đã được đề cập trong chương 2 trong KLTN này. Mỗi khi người dùng gán lại nhãn, hình ảnh sẽ được lưu trên Firebase Storage để phục vụ việc tăng cường huấn luyện mô hình nhận dạng sau này. Mô hình nhận diện sẽ được lưu trữ trên thiết bị của người dùng.

3.3.4.1. Theo ảnh



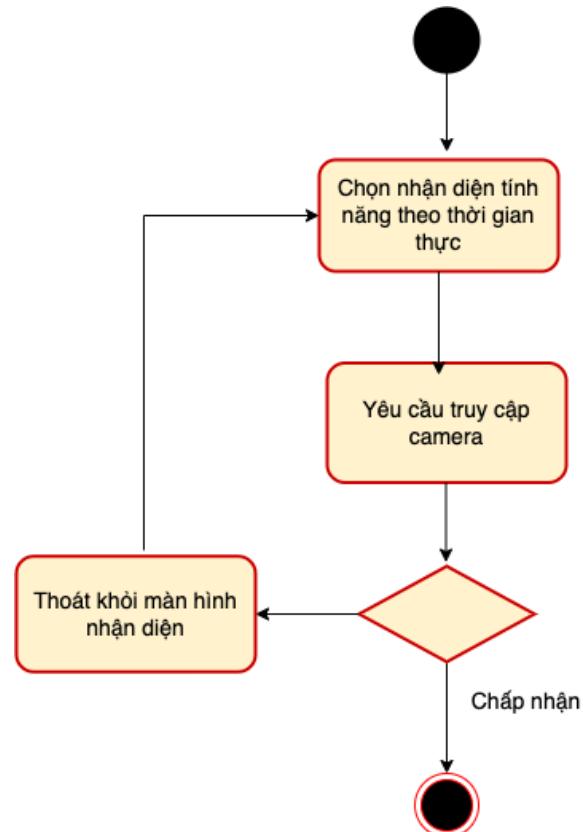
Hình 3.15. Biểu đồ hoạt động cho chức năng “Nhận diện bệnh cây trồng theo ảnh”

Tên ca sử dụng	Nhận diện bệnh cây trồng theo ảnh
Tác nhân	Tất cả mọi người
Mô tả	Người dùng chọn phần chụp ảnh
Điều kiện đầu	Không có

Tên ca sử dụng	Nhận diện bệnh cây trồng theo ảnh
Điều kiện cuối	Ứng dụng hiển thị được thông tin được kết quả và thông tin theo bệnh cây nhận diện được
Luồng cơ bản	1. Người dùng chọn phần chụp ảnh 2. Sau đó, hiển thị được thông tin được kết quả và thông tin theo bệnh cây nhận diện được
Luồng phụ	Không có
Điểm mở rộng	Không có

Bảng 3.11. Mô tả ca sử dụng chức năng “Nhận diện bệnh cây trồng theo ảnh”

3.3.4.2. Theo thời gian thực



Hình 3.16. Biểu đồ hoạt động cho chức năng “Nhận diện bệnh cây trồng theo thời gian thực”

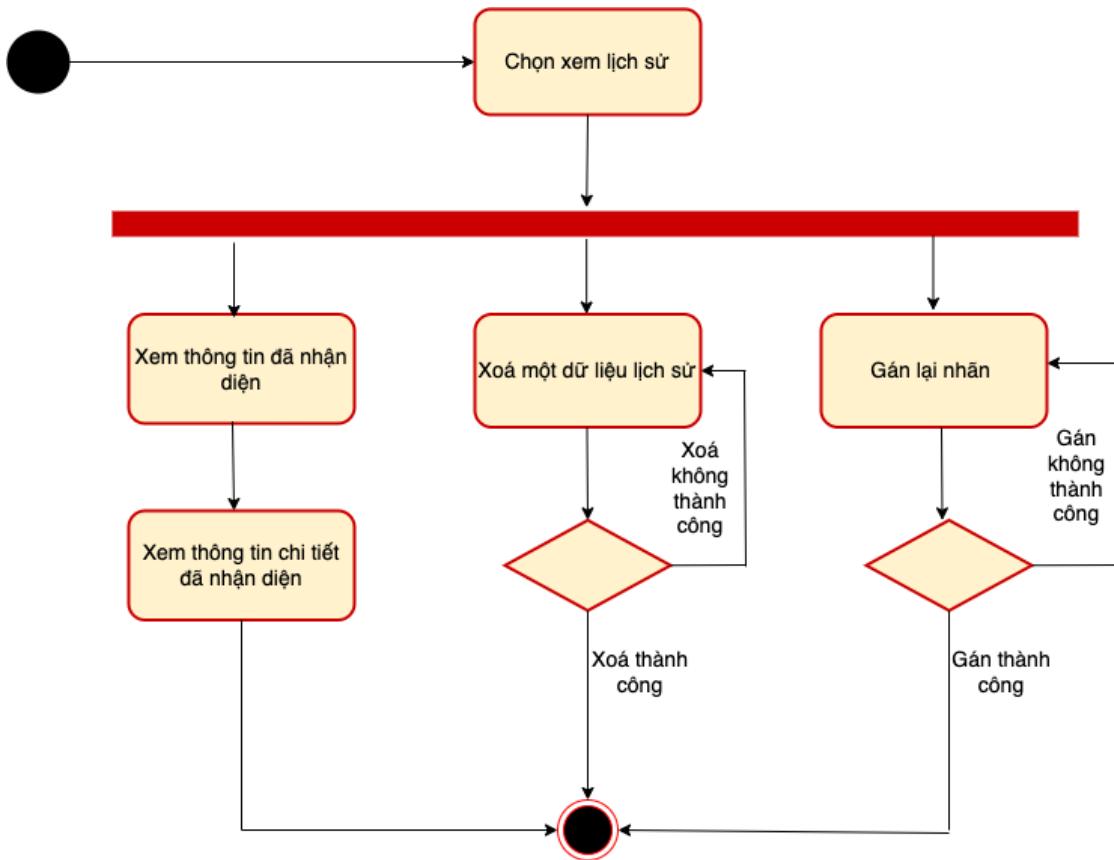
Tên ca sử dụng	Nhận diện bệnh cây trồng theo thời gian thực
Tác nhân	Tất cả mọi người
Mô tả	Người dùng xem được kết quả nhận diện bệnh khi dùng camera quay
Điều kiện đầu	Không có
Điều kiện cuối	Ứng dụng hiển thị được tên bệnh và độ chính xác nhận diện được
Luồng cơ bản	1. Người dùng chọn phần tìm kiếm, nhập thông tin cần tìm kiếm 2. Sau đó, hiển thị được tên bệnh và độ chính xác nhận diện được
Luồng phụ	Không có
Điểm mở rộng	Không có

Bảng 3.12. Mô tả ca sử dụng chức năng “Nhận diện bệnh cây trồng theo thời gian thực”

3.3.5. Quản lý nhận diện bệnh cây trồng

Lịch sử nhận diện bệnh cây trồng của người dùng sẽ được lưu trữ trên Firebase Realtime. Khi người dùng chọn vào xem lịch sử, ứng dụng hiển thị được lịch sử người dùng thao tác nhận diện, và lịch sử bệnh cây trồng.

Bên cạnh đó, để phục vụ cho việc cải thiện ứng dụng cũng như mô hình nhận diện sau này, ứng dụng cho phép người dùng gán nhãn lại mẫu bệnh. Hình ảnh sẽ được quản lý trên Firebase Storage và thông tin gán nhãn sẽ được lưu trên Firebase Realtime. Và cho phép người dùng gán lại một mẫu bệnh nhiều lần.



Hình 3.17. Biểu đồ hoạt động cho chức năng “Quản lý nhận diện bệnh cây trồng”

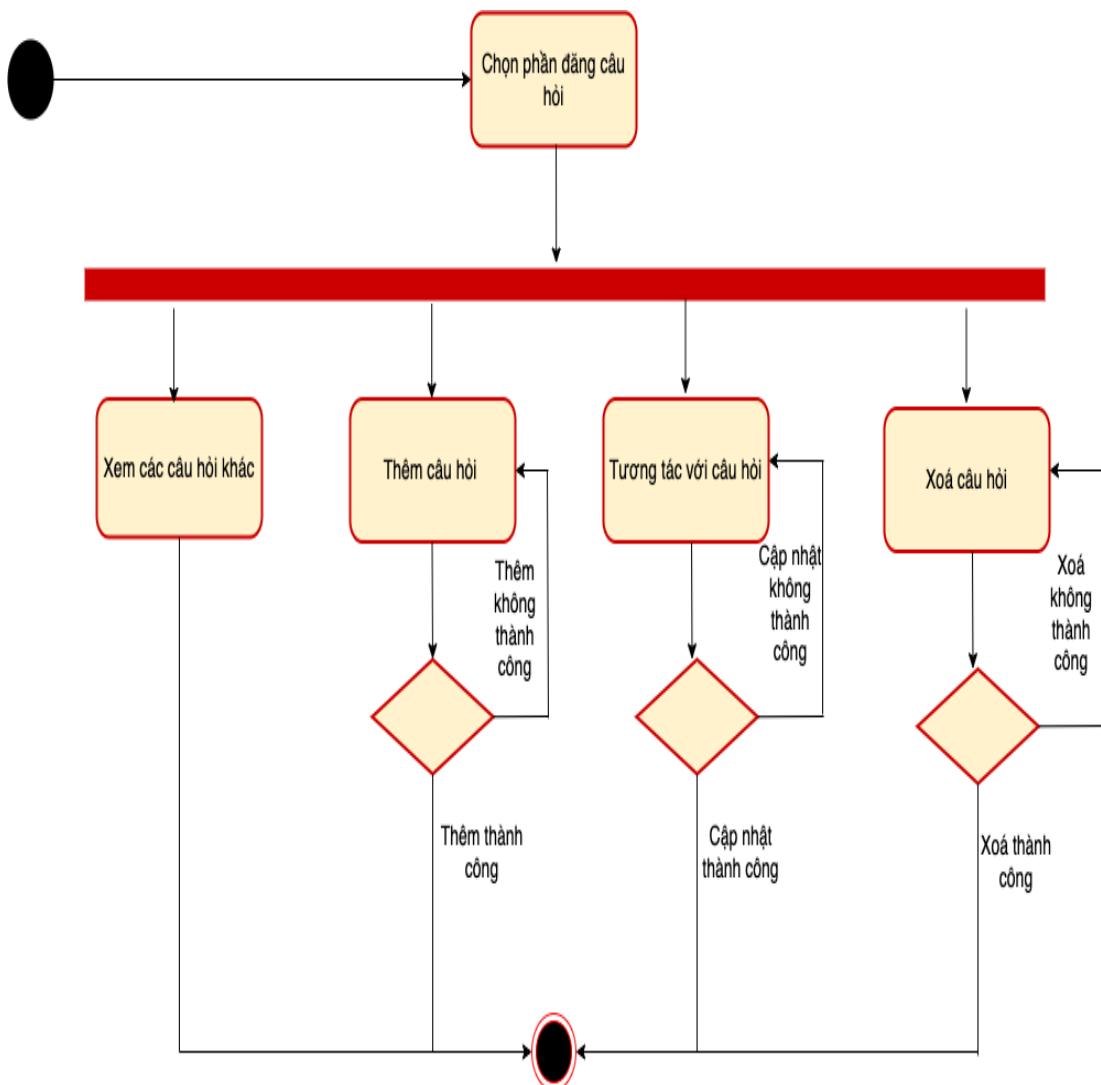
Tên ca sử dụng	Quản lý nhận diện bệnh cây trồng
Tác nhân	Tất cả mọi người
Mô tả	Người dùng xem lại kết quả nhận diện bệnh cây trồng
Điều kiện đầu	Người dùng đã đăng nhập
Điều kiện cuối	Ứng dụng hiển thị lịch sử bệnh cây đã nhận diện
Luồng cơ bản	1. Người dùng chọn Xem lịch sử tra cứu 2. Sau đó, hiển thị được lịch sử tra cứu, và cho phép người dùng xem chi tiết lại
Luồng phụ	1. Người dùng có thể gán lại tên bệnh cây trồng 2. Người dùng có thể xoá đi lịch sử tra cứu

Tên ca sử dụng	Quản lý nhận diện bệnh cây tròng
Điểm mở rộng	Không có

Bảng 3.13. Mô tả ca sử dụng chức năng “Quản lý nhận diện bệnh cây tròng”

3.3.6. Tham khảo ý kiến cộng đồng

Chức năng này cho phép người dùng tải một ảnh về mẫu bệnh cây tròng và câu hỏi để tham khảo ý kiến mọi người dùng khác. Bài đăng sẽ cũng được hiển thị theo thời gian thực. Hình ảnh được lưu trên Firebase Storage, còn câu hỏi của người dùng sẽ được lưu trên Firebase Realtime. Hoạt động của chức năng này sẽ được thể hiện ở hình 3.8. Chức năng này yêu cầu người dùng phải đăng nhập mới thực hiện được chức năng này.



Hình 3.18. Biểu đồ hoạt động cho chức năng Tham khảo ý kiến cộng đồng

Tên ca sử dụng	Tham khảo ý kiến cộng đồng
Tác nhân	Tất cả mọi người
Mô tả	Người dùng tham gia đăng bài viết, tương tác với người dùng khác
Điều kiện đầu	Người dùng đã đăng nhập
Điều kiện cuối	Ứng dụng hiển thị lịch sử bài viết của toàn bộ người dùng
Luồng cơ bản	1. Người dùng chọn tab Diễn đàn 2. Sau đó, hiển thị được toàn bộ bài đăng
Luồng phụ	1. Người dùng có thể thêm mới bài viết 2. Người dùng có thể xoá đi bài viết của chính mình 3. Người dùng có thể điều hướng đến trang cá nhân, để chỉ xem các bài đăng của mình
Điểm mở rộng	Không có

Bảng 3.14. Mô tả ca sử dụng chức năng “Tham khảo ý kiến cộng đồng”

3.4. Các công nghệ sử dụng

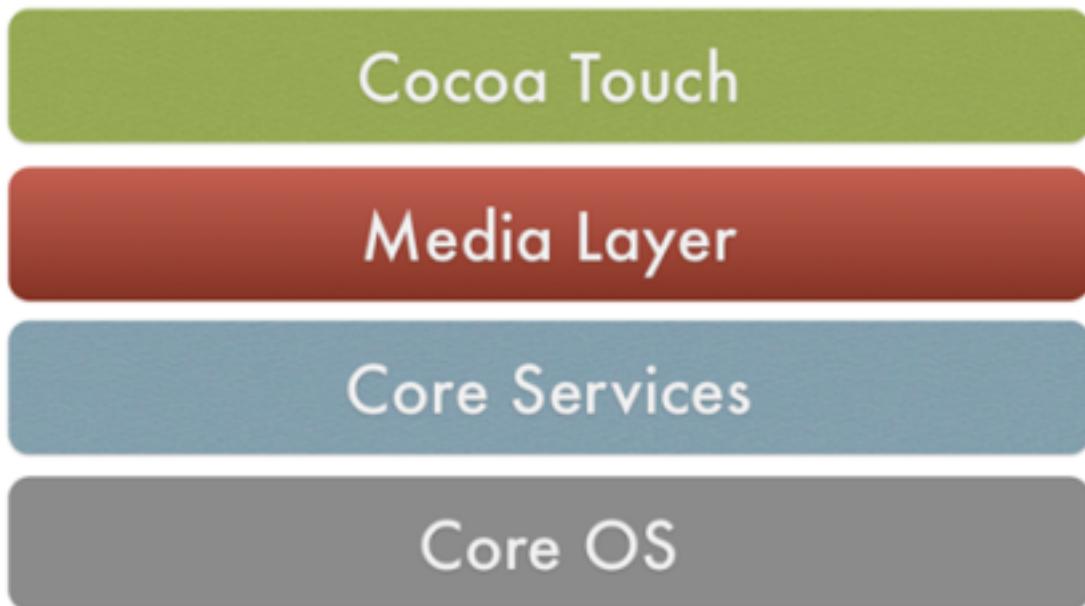
Công nghệ ngày càng phát triển, việc phát triển một ứng dụng di động ngày càng trở nên thuận tiện hơn. Trong phạm vi khoá luận, hệ thống xây dựng trên hệ điều hành iOS từ phiên bản 13.0, với ngôn ngữ sử dụng là Swift. Hệ thống lựa chọn Firebase như một backend, giúp phát triển ứng dụng nhanh hơn mà không cần xây dựng tầng máy chủ. Việc cập nhật, sửa và xoá dữ liệu cũng được thực hiện tại chính ứng dụng, và thông qua các API mà Firebase cung cấp. Và hệ thống đã sử dụng thư viện CoreML của Apple để thêm mô hình huấn luyện vào ứng dụng di động iOS.

3.4.1. Hệ điều hành iOS

iOS là một trong những hệ điều hành phổ biến hiện nay, được tạo ra và phát triển bởi công ty Apple, dành riêng cho phần cứng của các sản phẩm công ty này. Hệ điều hành bắt đầu

được công bố vào năm 2007 cho thế hệ iPhone đầu tiên của họ và sau đó nó ngày càng được mở rộng, phát triển để tương thích với các thiết bị khác của Apple như iPad và iPod Touch. Để cài đặt các ứng dụng trên iOS người dùng tải về từ AppStore – kho ứng dụng của iOS, ngoài ra cũng có những kho ứng dụng không chính thống khác khi jailbreak máy. Các phần mềm phát triển có thể sử dụng ngôn ngữ lập trình Objective-C hoặc Swift, là hai ngôn ngữ chính để phát triển các ứng dụng iOS. Ngoài ra, còn một số nền tảng hỗ trợ xây dựng ứng dụng đa nền tảng như React Native, Flutter. Phiên bản mới nhất của hệ điều hành là phiên bản 16.0. Hệ thống xây dựng ứng dụng từ phiên bản iOS 13.0 để phù hợp với cho cả những phiên bản thấp hơn.

iOS có kiến trúc phân lớp, được thể hiện như sơ đồ kiến trúc bên dưới [2]:



Hình 3.19. Các thành phần cơ bản của hệ điều hành iOS

- Core OS Layer: Những tính năng cấp thấp như Core Bluetooth Framework, Accelerate Framework, External Accessory Framework, Security Services Framework, Local Authentication Framework đều ở lớp này.
- CoreServices Layer: Lớp này cũng cấp sẵn một số thư viện quan trọng, gồm:
 - Address Book Framework: Cung cấp quyền truy cập vào cơ sở dữ liệu danh bạ của người dùng.
 - CloudKit Framework: Hỗ trợ di chuyển dữ liệu giữa ứng dụng và iCloud.

- Core Data Framework: Quản lý mô hình dữ liệu của ứng dụng Model View Controller.
- Core Location Framework: Cung cấp thông tin vị trí và tiêu đề cho các ứng dụng.
- Media Layer: Xử lý đồ họa, âm thanh và video ở lớp này.
- Cocoa Touch Layer: Lớp này sẽ chịu trách nhiệm chính về sự xuất hiện của các ứng dụng và khả năng phản hồi của chúng đối với các hành động của người dùng. Ngoài ra, nhiều tính năng xác định trải nghiệm người dùng chẳng hạn như thông báo, chế độ toàn màn hình và tự động lưu... cũng được thực hiện tại đây.

3.4.2. Swift

Swift là một ngôn ngữ lập trình hướng đối tượng dành cho việc phát triển các ứng dụng liên quan đến sản phẩm của Apple. Swift chính thức được ra mắt vào tháng 9 năm 2014 [3], và được Apple nâng cấp hàng năm qua các phiên bản. Swift được thiết kế để phù hợp với thư viện Cocoa và Cocoa Touch của Apple và phần lớn mã Objective-C hiện có được viết cho các sản phẩm của công ty này. Nó được đưa vào Xcode kể từ phiên bản 6.0 [4].

Một số đặc điểm của ngôn ngữ này:

- Swift là một ngôn ngữ hiện đại. Swift là sự kế thừa của ngôn ngữ C và Objective-C. Với ngôn ngữ này, tham số được đặt tên và thể hiện bằng một cú pháp rõ ràng giúp dễ đọc và bảo trì hơn. Hơn nữa, ngôn ngữ này cũng không yêu cầu dấu chấm, dấu phẩy khi viết hết một câu lệnh. Cấu trúc hàm rõ ràng, các mô-đun được loại bỏ tiêu đề và cung cấp không gian tên nên nhìn mã trông gọn gàng hơn. Các ký tự trong Swift là Unicode chuẩn và sử dụng mã hóa dựa trên UTF-8 để tối ưu các ngôn ngữ quốc tế và biểu tượng cảm xúc. Bộ nhớ được quản lý tự động một cách chặt chẽ và được xác định, giữ cho mức sử dụng bộ nhớ ở mức tối thiểu.
- Swift được sử dụng miễn phí và là mã nguồn mở: Swift có cộng đồng các nhà phát triển rất lớn, có rất nhiều tài liệu để tham khảo, nghiên cứu.
- An toàn: Swift loại bỏ toàn bộ các lỗ hổng an toàn. Các biến luôn được khởi tạo trước khi sử dụng, các kiểu dữ liệu như mảng và số nguyên được kiểm tra xem có bị tràn không. Bên cạnh đó, bộ nhớ được quản lý một cách tự động và thực thi quyền truy cập độc quyền vào bộ bảo vệ bộ nhớ để chống lại nhiều lỗi lập trình.

Cú pháp cũng được điều chỉnh để giúp dễ xác định ý định của người lập trình, ví dụ như từ khoá xác định một biến là var còn một hằng là let.

Ngoài ra, một tính năng nữa là giá trị mặc định của các đối tượng trong Swift không bao giờ được rỗng. Trình biên dịch Swift sẽ ngăn chặn việc cố hoặc tạo ra một đối tượng rỗng bằng cách báo lỗi trình biên dịch. Điều này giúp cho mã nguồn trở nên sạch sẽ và an toàn hơn. Và Swift cũng cung cấp thêm việc xử lý rỗng nếu trường hợp rỗng là hợp lệ sử dụng biến guard let... else hoặc if let... else hoặc sử dụng ?? để cho trình biên dịch biết.

- Nhanh và mạnh: Swift được tối ưu hóa để tận dụng tối đa phần cứng. Swift cũng gồm cả các thành phần nguyên thuỷ cấp thấp như kiểu, luồng xử lý và toán tử.
- Swift cũng cung cấp các tính năng hướng đối tượng như các lớp, giao thức, biến chung, mang lại hiệu suất cao.
- Dễ học đối với người bắt đầu lập trình iOS: cú pháp Swift đã được tinh chỉnh để giảm sự phức tạp hơn so với Objective-C, để phù hợp hơn với những người muốn học lập trình những ứng dụng trên các sản phẩm của Apple.

3.4.3. TensorFlow

Tensorflow là một thư viện phần mềm mã nguồn mở hỗ trợ mạnh mẽ các phép toán học để tính toán trong machine learning và deep learning được phát triển bởi Google.

Kiến trúc của Tensorflow cơ bản bao gồm 3 phần chính là:

- Tiền xử lý dữ liệu
- Dựng model
- Train và ước tính model

Khi Tensorflow hoạt động sẽ cho phép các lập trình viên có thể tạo ra dataflow graph, cũng như cấu trúc mô tả làm sao để cho dữ liệu có thể di chuyển qua 1 biểu đồ; hoặc di chuyển qua 1 seri mà các node đang xử lý. Mỗi một node có trong đồ thị thường đại diện cho 1 phép toán hoặc và mỗi kết nối thường hay cạnh giữa các node với nhau. Từ đó, mỗi kết nối hoặc edge giữa các node được xem là mảng dữ liệu đa chiều. Tensorflow sẽ cung cấp tất cả mọi điều đến cho lập trình viên dựa theo phương thức của ngôn ngữ Python. Ngôn ngữ này sẽ cung cấp nhiều cách tiện lợi để ta có thể hiểu được nên làm thế nào cho các high-level abstractions có thể kết hợp được với nhau. Node cũng như tensor có trong Tensorflow chính là đối tượng của Python. Và, mọi ứng dụng Tensorflow bản thân chúng chính là một ứng dụng Python.

Sử dụng TensorFlow sẽ mang lại nhiều lợi ích quan trọng cho việc lập trình machine learning chính là abstraction. Thay vì phải đối phó với các tình huống rườm rà từ việc phải thực hiện triển khai các thuật toán hay tìm ra biện pháp hợp lý nhất để có thể chuyển output của một chức năng sang kiểu input đối với một chức năng khác.

Lúc này thì bạn có thể tập trung vào phần logic tổng thể của ứng dụng hơn; lúc này thì TensorFlow sẽ chăm sóc những phần còn lại thay cho bạn. Ngoài ra, TensorFlow còn cung cấp những tiện ích giúp bổ sung cho các lập trình viên cần phải debug cũng như đảm bảo cho bạn có thể suy xét được các ứng dụng TensorFlow.

Không những thế, chế độ eager execution cho phép bạn có thể đánh giá cũng như sửa đổi được operation của biểu đồ theo cách riêng biệt nhất và minh bạch. Bởi vậy, thay vì phải dựng toàn bộ biểu đồ dưới dạng đối tượng độc lập vốn khá mơ hồ hoặc cần phải đưa ra đánh giá chung tổng thể. Ngoài ra, 1 trong những tính năng đặc đáo khác của TensorFlow là TensorBoard; nó sẽ cho bạn quan sát 1 cách trực tiếp liên quan đến những gì mà TensorFlow đang làm.

3.4.4. Keras

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 2015 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Một số ưu điểm của Keras như:

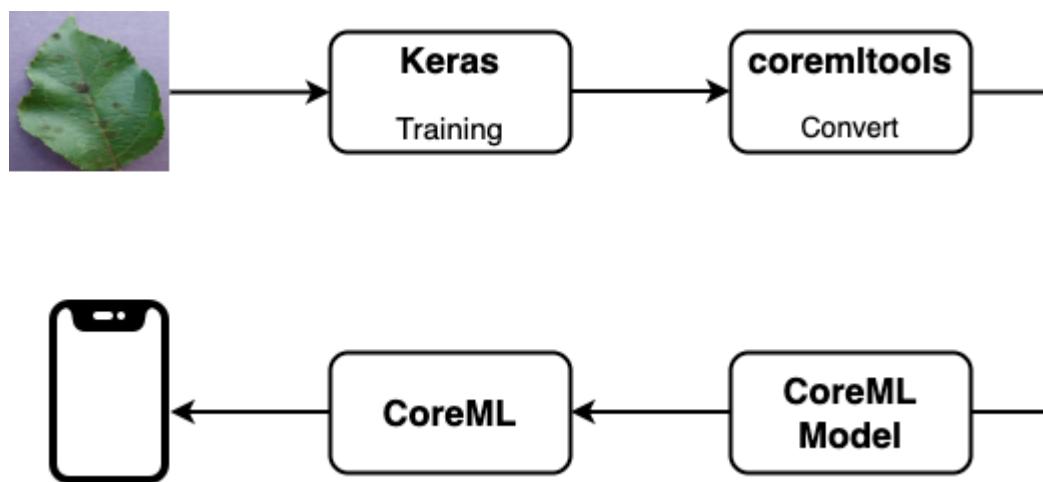
- Dễ sử dụng, dùng đơn giản hơn Tensor, xây dựng model nhanh.
- Run được trên cả CPU và GPU.
- Hỗ trợ xây dựng CNN, RNN hoặc cả hai. Với những người mới tiếp cận đến Deep như mình thì mình chọn sử dụng Keras để build model vì nó đơn giản, dễ nắm bắt hơn các thư viện khác. Dưới đây mình xin giới thiệu một chút về API này.

3.4.5. CoreML

Core ML (Core Machine Learning) là bộ thư viện được Apple thêm vào trong iOS phiên bản 11.0. Core ML cung cấp cho các nhà phát triển một cách để đưa các mô hình học máy vào ứng dụng của họ. Điều này cho phép xây dựng các tính năng thông minh trên thiết bị như phát hiện đối tượng.

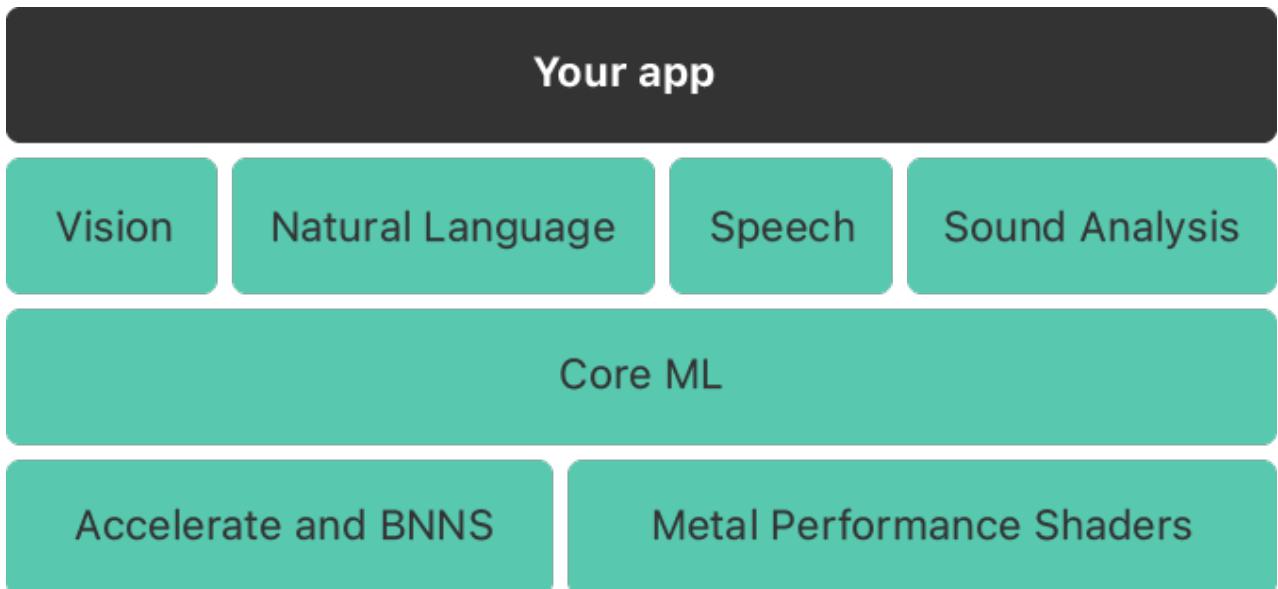
Để tích hợp với Core ML, thì cần một mô hình ở định dạng mô hình Core ML. Apple cung cấp các mô hình được đào tạo trước mà có thể sử dụng cho các tác vụ như phân loại hình ảnh. CoreML cũng hỗ trợ một số các thư viện như: Kaffe, Keras, scikit-learn.

Hệ thống đã huấn luyện dữ liệu trên Keras và sử dụng Core ML Tools để chuyển đổi model thu được thành mô hình CoreML từ đó có thể thêm mô hình huấn luyện được vào thư viện CoreML và sử dụng cho ứng dụng.



Hình 3.20. Các bước thêm một mô hình đã được huấn luyện vào ứng dụng iOS

Core ML có độ trễ thấp và kết quả gần với thời gian thực, không cần thực hiện gọi API. CoreML giống như một cái khung thư viện, nó cung cấp thêm các thư viện khác như: Vision cho việc phân tích hình ảnh, Natural Language cho xử lý đoạn văn, Speech cho việc chuyển đổi audio thành chữ, và cuối cùng là Sound Analysis cho việc nhận dạng âm thanh trong tệp âm thanh. Bản thân CoreML được xây dựng phía trên các tính năng cấp thấp như Accelerate mô tả như hình 3.21 dưới đây:



Hình 3.21. Vị trí CoreML trong ứng dụng

3.4.6. Firebase

Firebase là một nền tảng ứng dụng di động và web có các công cụ và hạ tầng được thiết kế để hỗ trợ các lập trình viên xây dựng hệ thống một cách nhanh chóng, thuận tiện.

Firebase (tiền thân là Evolve) trước đây là một công ty khởi nghiệp được thành lập bởi Andrew Lee và James Tamplin vào năm 2011. Năm 2012, Firebase ra đời với sản phẩm cung cấp là dịch vụ Backend-as-a-Service, sau này Firebase gia nhập vào công ty Google vào năm 2014 và phát triển nó thành một dịch vụ đa chức năng như hiện nay.

Firebase cung cấp rất nhiều dịch vụ hỗ trợ giúp các lập trình viên phát triển hệ thống một cách thuận tiện, dễ dàng. Giúp lập trình viên tập trung vào phát triển giao diện người dùng. Sử dụng Firebase giúp cho chuẩn hóa môi trường back-end theo một công nghệ duy nhất và cũng rất dễ học.

Tuy nhiên, bên cạnh những ưu điểm, Firebase cũng có nhiều khuyết điểm. Chẳng hạn như, Firebase không phải là mã nguồn mở, người dùng không có quyền truy cập mã nguồn. Firebase chỉ hoạt động ở một số quốc gia. Firebase cũng chỉ hoạt động với cơ sở dữ liệu Nosql, truy vấn còn chậm. Và không phải tất cả dịch vụ Firebase là miễn phí.

Firebase cung cấp bốn thao tác không thể thiếu với mọi ứng dụng có tương tác với cơ sở dữ liệu là CRUD, tức gồm:

- C (create - thêm mới)
- R (retrieve - truy vấn xem dữ liệu)
- U (update - cập nhật dữ liệu)

- D (delete - xoá dữ liệu)

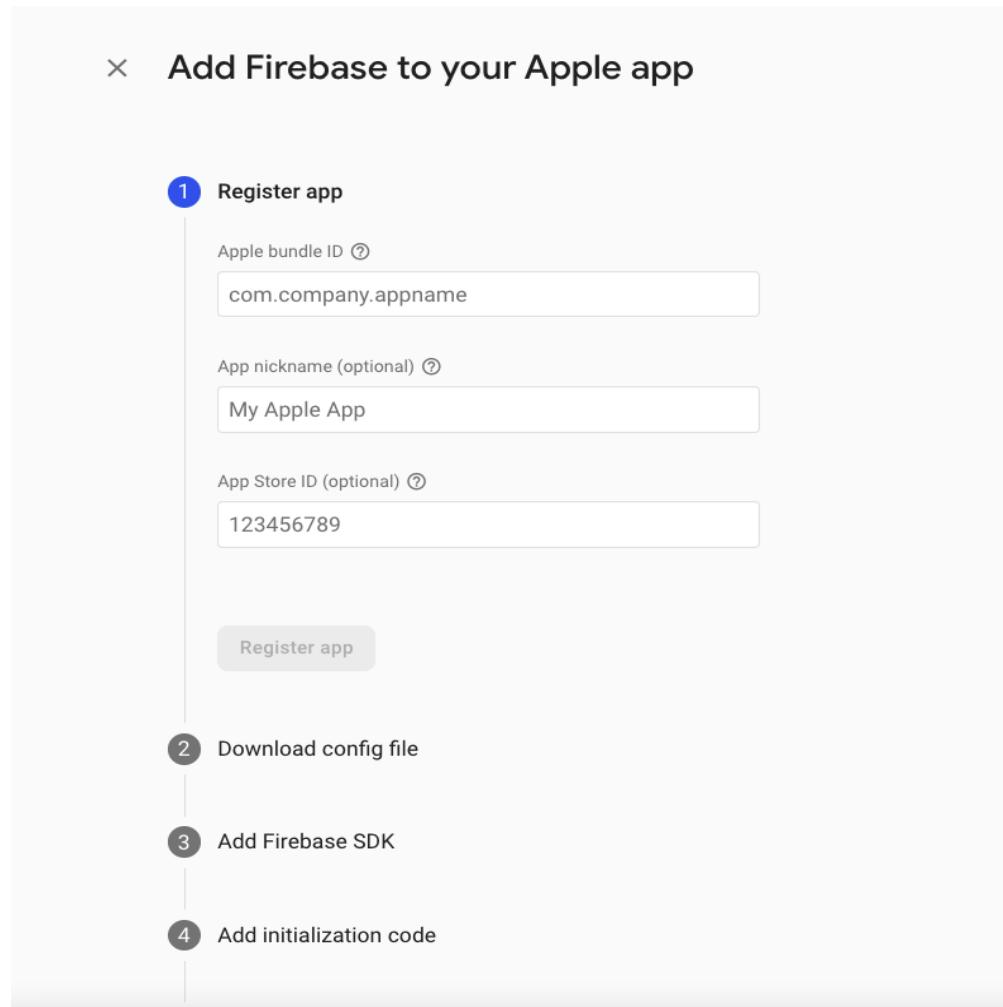
Với chương này, KLTN đã trình bày chi tiết cách phân tích, thiết kế và xây dựng ứng dụng hỗ trợ nhận diện bệnh cây trồng trên hệ điều hành iOS, cũng như các công nghệ được sử dụng trong hệ thống. Ở chương tiếp theo, KLTN sẽ trình bày về việc cài đặt ứng dụng hỗ trợ nhận diện bệnh cây trồng trên hệ điều hành iOS.

CHƯƠNG 4. CÀI ĐẶT VÀ KIỂM THỬ HỆ THỐNG

Trong chương này, khoá luận tập trung trình bày cách cài đặt và triển khai bản demo những chức năng của ứng dụng nhận diện bệnh cây trồng. Hệ thống được triển khai trên hệ điều hành iOS từ phiên bản 13.0. Ứng dụng sử dụng ngôn ngữ Swift và trình biên dịch XCode phiên bản 15.0.

4.1. Xây dựng và cài đặt ứng dụng PlantCare

Đầu tiên, ta cần thêm FireSDK vào hệ thống.



Hình 4.1. Thêm mới hệ thống iOS vào Firebase

Sau khi đã thêm được tệp config vào hệ thống, có hai cách để cài SDK của Firebase.

- Sử dụng Add packages
- Sử dụng lệnh pod

Sau đó, thêm câu lệnh ở trong tệp App Delegate của hệ thống để hoàn thành việc thêm Firebase SDK vào hệ thống.

```
import UIKit
import Firebase

@main
class AppDelegate: UIResponder, UIApplicationDelegate {
    func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        // Override point for customization after application launch.
        FirebaseApp.configure()
        let userDefaults = UserDefaults.standard
        if userDefaults.value(forKey: "appFirstTimeOpen") == nil {
            //if app is first time opened then it will be nil
            userDefaults.setValue(true, forKey: "appFirstTimeOpen")
            // signOut from Auth
            do {
                try Auth.auth().signOut()
            } catch {
                print("Error signing out: \(error.localizedDescription)")
            }
        }
        return true
    }
}
```

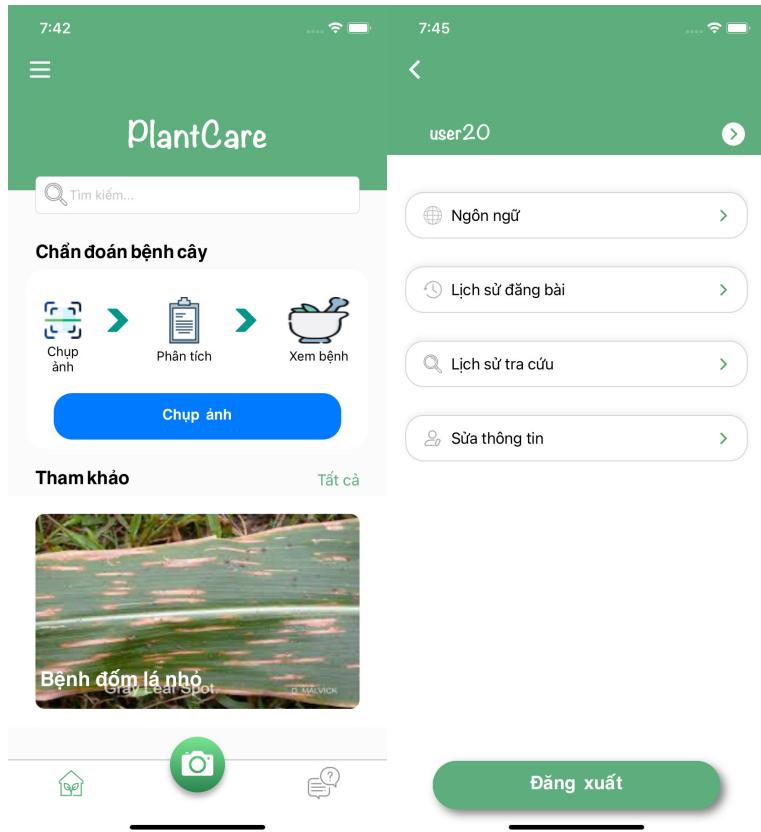
Hình 4.2. Thêm câu lệnh để hoàn thiện thêm Firebase SDK vào hệ thống

4.2. Ứng dụng PlantCare

Sau đây là những hình ảnh demo cho các chức năng hiện có của ứng dụng PlantCare.

4.2.1. Màn hình chính

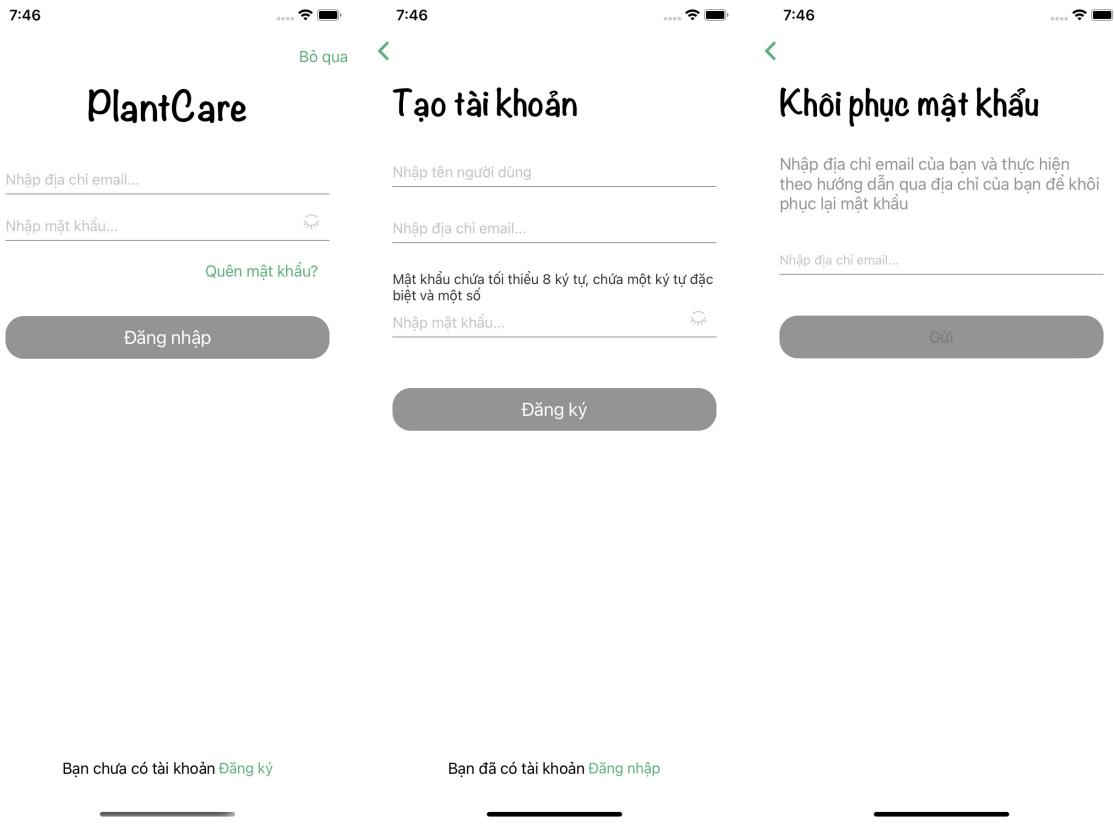
Màn hình chính của ứng dụng hiển thị toàn bộ các chức năng chính hệ thống gồm nhận dạng mẫu bệnh cây thông sử dụng ảnh từ thư viện hoặc camera điện thoại, tìm kiếm bệnh cây thông. Để tăng trải nghiệm của người dùng, ứng dụng không yêu cầu đăng nhập ở màn hình này. Mọi chức năng tra cứu, tìm kiếm đều có thể thực hiện ngoại tuyến, giúp người dùng thuận tiện nhận dạng, tìm kiếm bệnh cây thông hơn.



Hình 4.3. Giao diện màn hình chính

4.2.2. Đăng nhập

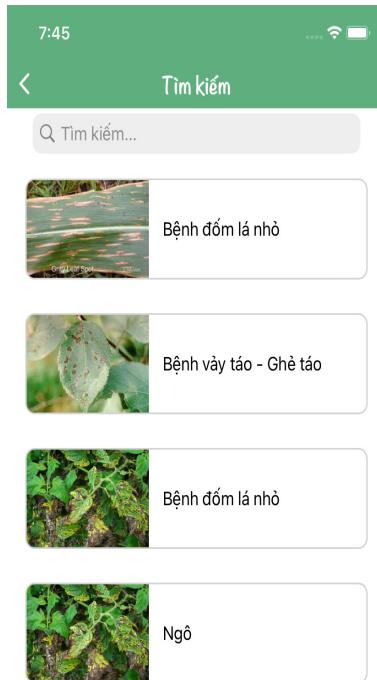
Người dùng đăng nhập hệ thống sử dụng email. Sau khi xác thực thành công, người dùng sẽ được điều hướng đến màn hình chính. Ngoài ra, nếu người dùng chưa có tài khoản, thì sẽ hiển thị thông báo email chưa tồn tại, và người dùng có thể điều hướng đến trang đăng ký hoặc án bở qua để đến màn hình chính, sử dụng những chức năng chính của hệ thống. Trường hợp người dùng quên mật khẩu, người dùng có thể điều hướng đến màn quên mật khẩu để lấy mật khẩu mới.



Hình 4.4. Giao diện màn hình đăng nhập, đăng ký và khôi phục mật khẩu

4.2.3. Tìm kiếm mẫu bệnh cây trồng

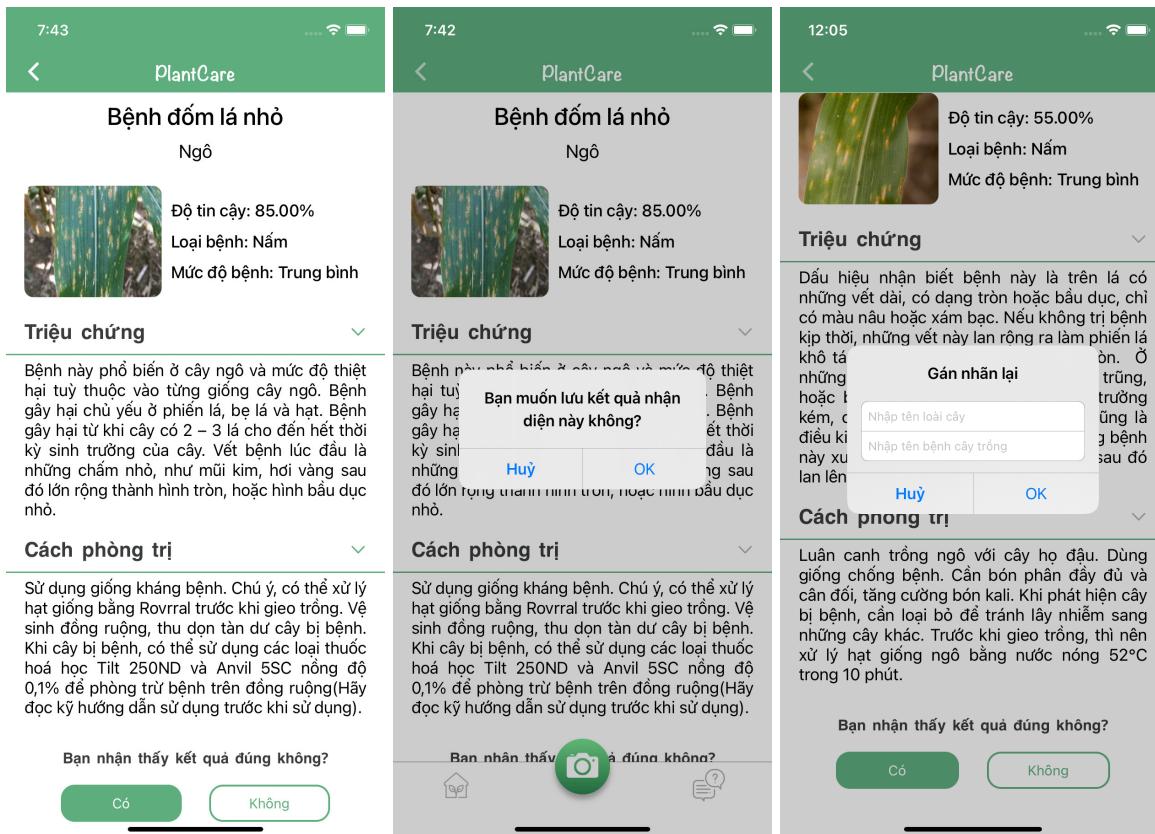
Ứng dụng hỗ trợ người dùng tìm kiếm bệnh cây trồng theo tên bệnh cây. Dữ liệu, thông tin về bệnh cây trồng được thu thập và tổng hợp từ nhiều nguồn khác nhau. Ứng dụng cũng cho phép người dùng xem thông tin chi tiết về bệnh cây trồng.



Hình 4.5. Giao diện tìm kiếm mẫu bệnh cây trồng

4.2.4. Nhận diện cây trồng bằng hình ảnh

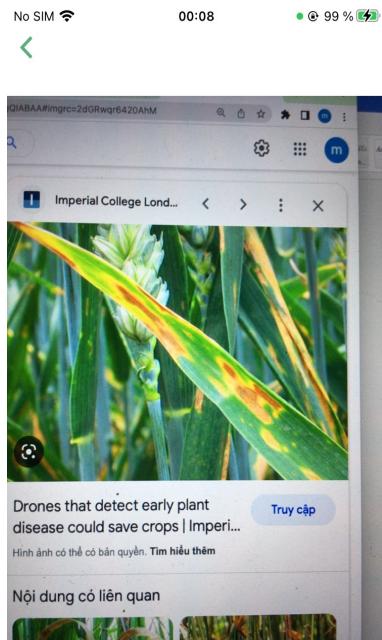
Ứng dụng cũng cung cấp cho người dùng tính năng nhận diện bệnh cây trồng thông qua hình ảnh tải lên ngoài các tính năng được kể trên. Áp dụng mô hình nhận diện mẫu bệnh ở chương 2, ứng dụng cho phép nhận diện các loại bệnh cây trồng, gồm 38 loại bệnh khác nhau. Sau khi đã tải ảnh lên từ thư viện hoặc chụp bằng máy ảnh, ứng dụng sẽ yêu cầu cắt ảnh trước khi nhận dạng. Sau khi nhận dạng thành công, ứng dụng hỗ trợ người dùng nhận diện bệnh, và đưa ra các thông tin liên quan đến bệnh cây trồng đó, bao gồm độ chính xác, tên bệnh, thông tin chung về bệnh và cách phòng trị. Người dùng có thể chọn lưu lại kết quả để lần sau xem lại trong phần lịch sử tra cứu, hoặc có thể gán nhãn lại tên bệnh khi thấy bệnh này chưa được nhận diện chính xác. Để cải thiện mô hình nhận diện mẫu bệnh cây trồng và do kiến thức về bệnh cây trồng cũng còn hạn chế. Vì vậy, ứng dụng cung cấp thêm một tính năng cho phép người dùng gán nhãn lại dữ liệu bệnh cây trồng. Từ đó, phục vụ cải thiện mô hình cũng như bổ sung thêm thông tin về bệnh cây trồng. Những dữ liệu này sẽ tạm thời được lưu lại. Trong tương lai, ứng dụng sẽ có thể mở rộng và tái sử dụng lại những dữ liệu gán nhãn này. Tính năng lưu lại kết quả nhận diện và gán nhãn lại dữ liệu bệnh yêu cầu người dùng phải đăng nhập.



Hình 4.6. Giao diện màn chi tiết bệnh cây trồng

4.2.5. Nhận diện bệnh cây trồng theo thời gian thực

Cũng gần tương tự như tính năng nhận diện bệnh cây trồng bằng hình ảnh, ứng dụng cũng cho phép người dùng nhận diện mẫu bệnh cây trồng theo thời gian thực bằng camera. Sau khi người dùng cho phép ứng dụng truy cập vào camera, ứng dụng sẽ liên tục nhận dạng và đưa ra các thông số về hình ảnh xuất hiện trên camera. Mặc dù, độ chính xác không thể cao bằng việc chẩn đoán dựa trên hình ảnh nhưng tính năng này cũng đem lại cho người dùng ứng dụng nhiều trải nghiệm hơn.

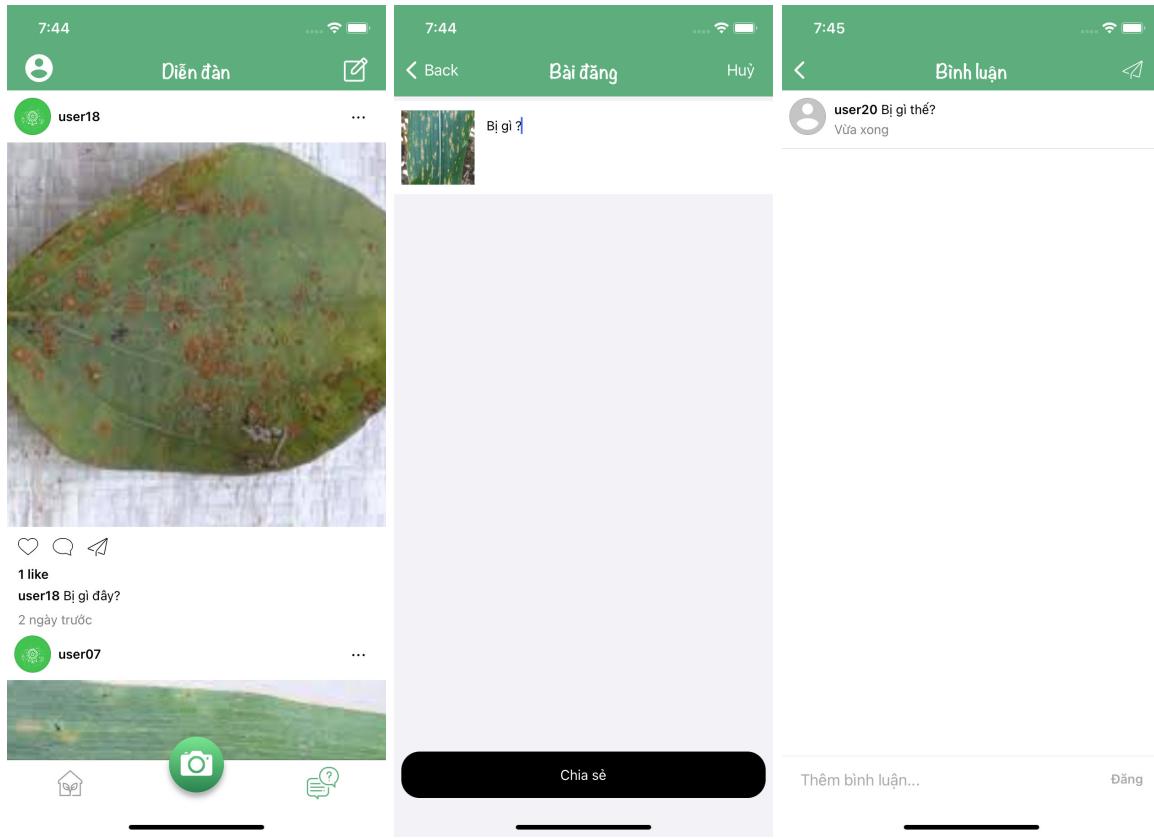


Bệnh đốm lá nhò: 54.60%

Hình 4.7. Giao diện nhận diện bệnh cây trồng theo thời gian thực

4.2.6. Tham khảo ý kiến cộng đồng

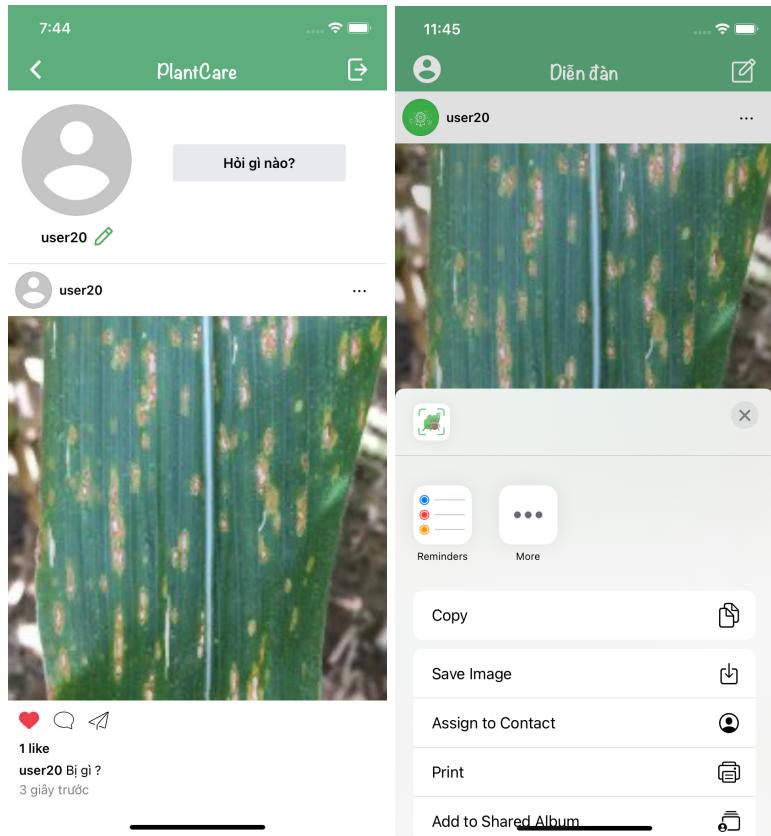
Bên cạnh việc nhận diện mẫu bệnh qua hình ảnh hoặc camera, người dùng có thể hỏi ý kiến với người dùng khác bằng việc đăng hình ảnh về mẫu bệnh cây trồng, cũng như tương tác với người dùng khác thông qua các tương tác như bình luận và thích. Người dùng cũng có thể lưu lại bài viết về thiết bị của mình. Chức năng này yêu cầu người dùng cần đăng nhập để có thể sử dụng. Và những bài đăng được cập nhật theo thời gian thực.



Hình 4.8. Giao diện chức năng tham khảo ý kiến cộng đồng

4.2.7. Quản lý bài viết

Ứng dụng cho phép quản lý bài viết của mỗi người dùng. Hệ thống cho phép người dùng quản lý những bài đăng người dùng tham khảo ý kiến cộng đồng. Những tính năng của bài viết cũng tương tự như trong tính năng tham khảo ý kiến cộng đồng. Người dùng cũng có thể xoá, bình luận, thích hay chia sẻ bài viết của mình. Và để sử dụng được tính năng này, người dùng cũng cần đăng nhập để có thể sử dụng.



Hình 4.9. Giao diện trang cá nhân của người dùng ứng dụng

4.2.8. Quản lý thông tin người dùng

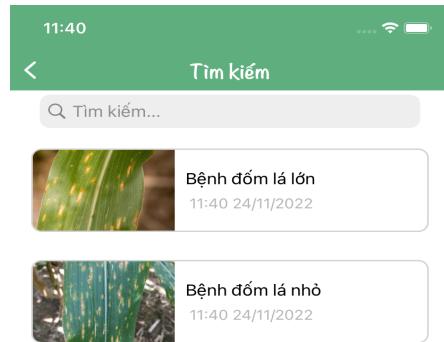
Bên cạnh những tính năng trên, người dùng cũng có thể cập nhật lại ảnh đại diện, thông tin người dùng ứng dụng theo thời gian thực.



Hình 4.10. Giao diện thay đổi thông tin người dùng

4.2.9. Lịch sử tra cứu

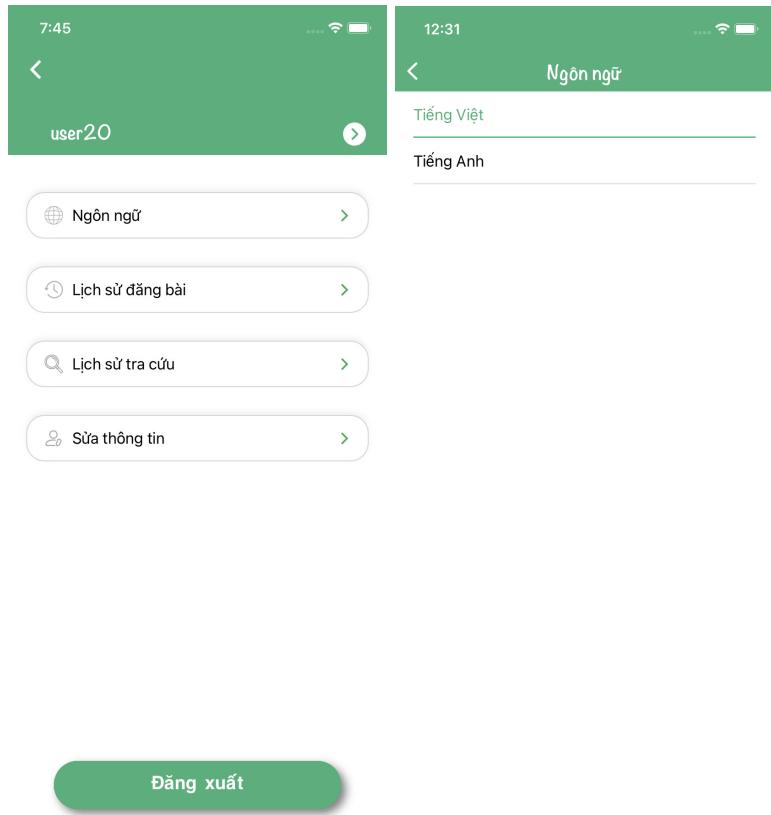
Ứng dụng cho phép xem lại lịch sử tra cứu bệnh cây trồng. Người dùng cũng cần đăng nhập vào hệ thống để có thể sử dụng tính năng này. Ngoài ra, hệ thống cho phép xoá từng dữ liệu tìm kiếm nếu người dùng không muốn lưu lại nữa.



Hình 4.11. Giao diện chức năng Tìm kiếm

4.2.10. Chuyển đổi ngôn ngữ

Ứng dụng cho phép chuyển đổi ngôn ngữ linh hoạt giữa 2 ngôn ngữ tiếng Anh và tiếng Việt, giúp người dùng thuận tiện sử dụng ứng dụng hơn. Tính năng này không yêu cầu người dùng đăng nhập.



Hình 4.12. Giao diện chức năng Chuyển đổi ngôn ngữ

KẾT LUẬN

Với sự bùng nổ của công nghệ thông tin như hiện nay, việc áp dụng khoa học công nghệ kỹ thuật đã giúp ích rất nhiều trong cuộc sống. Việc phân loại bệnh cây trồng bằng những cách thủ công chưa khách quan, chủ yếu thông qua phán đoán và kinh nghiệm cá nhân. Đó cũng chính là những nguyên nhân dẫn đến việc giảm năng suất, chất lượng cây trồng, tốn nhiều chi phí cho việc phòng và trị các loại bệnh cây trồng. Vì vậy, KLTN đã lên ý tưởng và xây dựng một hệ thống nhận diện bệnh cây trồng thông qua ảnh chụp giúp giảm chi phí, mang lại nhiều lợi ích cho người làm vườn. Thông qua những thông tin mà ứng dụng cung cấp, người làm vườn sẽ có thêm nhiều kinh nghiệm hơn. Bên cạnh đó, ứng dụng sẽ là nơi trung gian để kết nối những người làm vườn với nhau, hỗ trợ lẫn nhau trong việc chăm sóc và phòng trị bệnh cây trồng.

Khoá luận đã đạt được các kết quả sau: Tái sử dụng lại được mô hình MobileNet trong tập PlantVillage để nhận diện bệnh cây trồng. Cuối cùng, khoá luận cũng đã tích hợp được mô hình và xây dựng ứng dụng PlantCare nhận diện bệnh cây trồng và đạt được một số kết quả đáng mong đợi.

Ngoài những kết quả đạt được ở phía trên, do kiến thức và kinh nghiệp của bản thân còn ít, vẫn còn không ít thiếu sót như sau: Mô hình nhận dạng chưa tinh chỉnh để cho ra kết quả tốt hơn và cần nhiều thông tin hơn về một số loại bệnh. Trong tương lai, hệ thống cần phát triển và cải tiến tính năng nhận diện mẫu bệnh cây trồng. Và hệ thống cũng sẽ mở rộng bộ dữ liệu cho những cây đặc thù ở Việt Nam, cũng như tăng cường việc dán nhãn và quản lý lại chất lượng dữ liệu người dán nhãn. Cuối cùng, với những nỗ lực và cố gắng trong KLTN này hy vọng sẽ đóng góp một phần sức lực nhỏ bé giúp cho nền nông nghiệp nước nhà cũng như có thể triển khai thành một hệ thống đến nhiều người làm vườn.

TÀI LIỆU THAM KHẢO

Tiếng Việt

- [1] C.M, *Tạp chí khoa học công nghệ Việt Nam*, 2020.
- [2] Đặng Thị Hằng, Phạm Duy Tùng, *Tìm Hiểu Mạng MobileNetV1*, 2019
- [3] Tensorflow, *Tại sao sử dụng Tensorflow*, Google, 2021.

Tiếng Anh

- [4] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*, 2017.
- [5] R. B.K. and S. Santhosh, *Diseases Detection of Various Plant Leaf Using Image Processing Techniques: A Review*, IEEE, 2019.
- [6] J. Keerthi, S. Maloji and P. Krishna, *An approach of tomato leaf disease detection based on SVM classifier*, 2019.
- [7] M. Bhange and H. Hingoliwala, *Smart Farming: Pomegranate Disease Detection Using Image Processing*, 2015.
- [8] M. A. Noyan, *Uncovering bias in the PlantVillage dataset*, 2022.
- [9] D. P. Hughes and M. Salathé, *An open access repository of images on plant health to enable the development of mobile disease diagnostics*, 2015.
- [10] A. Bruno, D. Moroni, R. Dainelli, L. Rocchi, S. Morelli, E. Ferrari, P. Toscano and M. Martinelli, *Improving plant disease classification by adaptive minimal ensembling*, 2022.
- [11] Naveen, *Architecture of IOS Operating System*, 2021.