TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP
Xây dựng hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm tại Bệnh viện Da liễu Trung ương

TRẦN PHƯƠNG THẢO
thao.tp173383@sis.hust.edu.vn

Ngành Công nghệ thông tin
Chuyên ngành Kỹ thuật máy tính

Giảng viên hướng dẫn: TS. Nguyễn Hồng Quang _____

Bộ môn: Kỹ thuật máy tính
Viện: Công nghệ thông tin – Truyền thông

HÀ NỘI, 6/2021

Phiếu giao nhiệm vụ đồ án tốt nghiệp

Thông tin về sinh viên

Họ và tên sinh viên: Trần Phương Thảo

Điện thoại liên lạc: 0964086794 Email: thao.tp173383@sis.hust.edu.vn

Lớp: KTMT.06 – K62 Hệ đào tạo: Đại học chính quy

Thông tin về giáo viên hướng dẫn

Họ và tên GVHD : TS. Nguyễn Hồng Quang

Đồ án tốt nghiệp được thực hiện tại: Bộ môn Kỹ thuật máy tính, Viện CNTT&TT, Đại học Bách Khoa Hà Nội.

Thời gian làm ĐATN: Từ ngày 22/2/2021 đến 7/6/2021.

1. Tên đề tài: Xây dựng hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm tại Bệnh viện Da liễu Trung ương.

2. Mục đích nội dung của ĐATN:

- Tìm hiểu nghiệp vụ và quy trình thực tế tại bệnh viện
- Phân tích thiết kế và xây dựng hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm hoàn chỉnh
- Kiểm thử và triển khai thử nghiệm thực tế tại bệnh viện

3. Các nhiệm vụ cụ thể của ĐATN và kế hoạch thực hiện

- Tuần 1, 2, 3, 4: Khảo sát thực tế tại bệnh viện, tìm hiểu các công nghệ dùng để xây dựng và phát triển hệ thống.
- Tuần 5, 6: Tìm hiểu kiến trúc hệ thống, phân tích thiết kế hệ thống, tổng hợp các chức năng chính của hệ thống.
- Tuần 7, 8, 9, 11, 12: Xây dựng hệ thống.
- Tuần 13 đến hết: Kiểm thử và viết quyển Đồ án tốt nghiệp.

4. Lời cam kết của sinh viên

Tôi – *Trần Phương Thảo* – cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS. Nguyễn Hồng Quang.

Các kết quả nêu trong ĐATN là trung thực, không phải sao chép toàn văn của bất kỳ công trình nào khác.

Hà Nội, ngày tháng năm 2021

Tác giả ĐATN

Trần Phương Thảo

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

Hà Nội, ngày tháng năm 2021

Giáo viên hướng dẫn

TS. Nguyễn Hồng Quang

Lời cam kết

Họ và tên sinh viên: Trần Phương Thảo

Điện thoại liên lạc: 0964086794 Email: thao.tp173383@sis.hust.edu.vn

Lớp: KTMT.06 – K62 Hệ đào tạo: Đại học chính quy

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

Hà Nội, ngày tháng năm

Tác giả ĐATN

Trần Phương Thảo

Lời cảm ơn

Đầu tiên, em muốn gửi lời cảm ơn chân thành nhất tới TS. Nguyễn Hồng Quang. Bằng sự tận tâm và nhiệt tình của mình, thầy giúp em tìm ra định hướng đề tài phù hợp với bản thân, đưa ra những gợi ý, chỉ dẫn chi tiết để em có thể hoàn thành đồ án tốt nghiệp lần này. Không những thế, thầy còn hết sức tạo điều kiện và đưa ra những lời khuyên kịp thời mang tính định hướng cho em trong quá trình làm đồ án tốt nghiệp.

Em xin gửi lời cảm ơn đến ban lãnh đạo nhà trường nói chung và Viện Công nghệ thông tin và Truyền thông nói riêng vì đã tạo điều kiện về giáo dục, vật chất và cả tinh thần để cho em được học tập trong một môi trường nghiêm túc, kỷ luật nhưng cũng đầy những kỷ niệm đáng nhớ của trường Đại học Bách Khoa Hà Nội.

Em cũng xin gửi lời cảm ơn đến những thầy cô giáo, đã luôn tâm huyết, tận tụy với công việc, luôn hết mình khi chia sẻ vốn kiến thức của mình với sinh viên.

Em cũng muốn gửi lời cảm ơn đến những người bạn của em, những người bạn mà vừa là tấm gương vừa là người đồng hành trong quá trình em học tập tại trường. Chính sự xuất sắc và nhiệt huyết đam mê của các bạn đã thôi thúc em cố gắng, nỗ lực và nghiêm khắc hơn với bản thân mình, đặc biệt là các bạn Lê Trọng Nhân, Trần Văn Hoàng, Trần Văn Điệp, Phạm Văn Hùng, Nguyễn Doãn Nam. Cảm ơn người bạn thân của em là Vũ Ngọc Trường, bạn ấy luôn là người động viên em, luôn tin rằng em sẽ làm tốt công việc học tập của mình. Em cũng muốn gửi lời cảm ơn đến tập thể lớp KTMT.06 K62 và các bạn khác mà em được tiếp xúc và làm quen trong quá trình học tập, cảm ơn các cậu vì tất cả, vì đã đồng hành với mình trong suốt những năm tháng sinh viên đáng nhớ này.

Sau cùng, em muốn gửi lời cảm ơn tới gia đình em. Con cảm ơn cả gia đình mình vì đã là chỗ dựa vững chắc cho con, luôn ủng hộ và yêu thương con vô điều kiện.

Trong quá trình làm đồ án tốt nghiệp, em đã cố gắng hết sức những cũng không thể tránh khỏi những thiếu sót, do đó, hệ thống có thể chưa được hoàn chỉnh. Em rất mong nhận được những góp ý của thầy cô và các bạn, cũng như của các bệnh viện để hoàn thiện hơn đồ án của mình.

Lời cảm ơn là không đủ để em nói ra hết sự biết ơn của mình đối với mọi người, nhưng một lần nữa, em xin chân thành cảm ơn.

Tóm tắt

Hiện nay, các bệnh viện công tuyến Trung ương tại Hà Nội nói chung và trên toàn Việt Nam nói riêng hàng ngày đều tiếp nhận rất nhiều bệnh nhân. Tình trạng quá tải tại bệnh viện xảy ra thường xuyên và liên tục. Việc ứng dụng công nghệ thông tin vào y tế, cụ thể là các hệ thống quản lý dữ liệu bệnh nhân, đã giúp rút ngắn quá trình khám chữa bệnh và tăng hiệu suất khám chữa bệnh tại các bệnh viện. Tuy nhiên, một số bước khám chữa bệnh trung gian cần sử dụng đến các máy chụp chẩn đoán và các máy xét nghiệm hiện đại đang còn hạn chế, hoặc chưa kết hợp với các kết quả kết xuất từ mô hình chẩn đoán bệnh thông minh.

Để giải quyết bài toán trên, đồ án của em phát triển hệ thống trung gian giao tiếp giữa các thiết bị máy và nhân viên bệnh viện (gồm các bác sĩ và các nhân viên xét nghiệm) giúp cải thiện quá trình này. Dựa trên khảo sát thực tế tại Bệnh viện Da liễu Trung ương, đồ án thực hiện đọc dữ liệu trung gian nhận được từ máy chụp chẩn đoán hình ảnh, kết hợp với mô hình dự đoán xem bệnh nhân có bị ung thư hắc tố da (Melanoma) hay không của bạn Nguyễn Trí Hùng (dưới sự hướng dẫn của TS. Nguyễn Hồng Quang). Đồng thời, đồ án cũng xử lý những bước trung gian trong quá trình khám chữa bệnh nhằm đưa dữ liệu ảnh và kết quả xét nghiệm lên hệ thống, giúp trực quan hóa và tăng khả năng chẩn đoán của bác sĩ, giảm thời gian chờ phía bác sĩ, giảm thiểu sự không chính xác khi in thông tin ra giấy cũng như tiết kiệm lượng tài nguyên giấy nhằm bảo vệ môi trường.

Để dễ dàng bảo trì và mở rộng trong tương lai, hệ thống được thiết kế theo mô hình kiến trúc Microservices, sử dụng công nghệ ReactJS kết hợp cùng Redux để thiết kế và xây dựng giao diện, phần xử lý logic phía server và lưu trữ dữ liệu sử dụng NodeJS và MongoDB.

Đồ án này nếu phát triển hoàn chỉnh và được triển khai thực tế tại bệnh viện cùng mô hình dự đoán của bạn Nguyễn Trí Hùng sẽ giúp rút ngắn thời gian khám chữa bệnh, tăng độ chính xác trong quá trình chẩn đoán bệnh của bác sĩ, giảm thiểu lượng tài nguyên giấy cần dùng bệnh viện.

Mục lục

Danh mục hình vẽ

Danh mục bảng

Danh mục các từ viết tắt

API	Application Programming Interface Giao diện lập trình ứng dụng
EUD	End-User Development Phát triển ứng dụng người dùng cuối
GWT	Google Web Toolkit Công cụ lập trình Javascript bằng Java của Google
HTML	HyperText Markup Language Ngôn ngữ đánh dấu siêu văn bản
CNTT	Công nghệ thông tin
ĐATN	Đồ án tốt nghiệp
SV	Sinh viên

Danh mục thuật ngữ

Browser	Trình duyệt
Cache memory	Bộ nhớ đệm
Component	Thành phần
Client	Máy khách
Interpreter	Trình thông dịch
Compiler	Trình biên dịch
Server	Máy chủ
DOM	Mô hình các đối tượng trong tài liệu HTML
Render	Biểu diễn dữ liệu thành đồ họa
State	Trạng thái

Chương 1 Giới thiệu đề tài

Chương 1 giới thiệu những vấn đề thực tiễn khiến em lựa chọn đề tài này, đồng thời chương 1 cũng trình bày tổng quan về hệ thống, phạm vi, mục tiêu, định hướng giải pháp và bố cục trình bày của đồ án.

1.1 Đặt vấn đề

Những năm gần đây, việc ứng dụng Công nghệ thông tin vào y tế đang được Đảng và Nhà nước quan tâm, đẩy mạnh. Bộ Y tế đã triển khai đề án "Ứng dụng, phát triển công nghệ thông tin y tế thông minh giai đoạn 2019-2025" với 3 mục tiêu chính. Trong đó, mục tiêu thứ hai nêu rõ [1]: *"Đẩy mạnh ứng dụng công nghệ thông tin tại các cơ sở khám, chữa bệnh góp phần cải cách hành chính và giảm quá tải bệnh viện; sử dụng hồ sơ bệnh án điện tử tiến tới không sử dụng bệnh án giấy, thanh toán viện phí điện tử, hình thành các bệnh viện thông minh"*.

Hiện nay, sau hơn một năm thực hiện đề án, ứng dụng công nghệ thông tin trong các bệnh viện tuyến Trung ương và tuyến tỉnh đã đạt được nhiều kết quả tích cực theo thống kê của Bộ Y tế. Các bệnh viện đã được triển khai hệ thống thông tin quản lý bệnh viện, góp phần tăng cao hiệu suất khám chữa bệnh, giảm quá tải cho bệnh viện, đồng thời, bệnh nhân đến khám cũng không cần chờ quá lâu hay gặp quá nhiều rắc rối khi muốn đăng ký khám, chữa bệnh.

Tuy nhiên, các hệ thống quản lý thông tin vẫn còn rất nhiều hạn chế. Nhiều bệnh viện chưa triển khai phần mềm quản lý bệnh viện với đầy đủ các chức năng phục vụ quy trình khám chữa bệnh mà vẫn còn đang sử dụng những mô hình quản lý truyền thống hoặc sử dụng các phần mềm rời rạc chưa thống nhất (phần mềm đính kèm với thiết bị công nghệ, phần mềm chỉ có chức năng cơ bản chưa hoàn thiện). Chỉ một số bệnh viện tuyến trung ương được áp dụng và triển khai hệ thống phần mềm quản lý bệnh viện với đầy đủ các chức năng. Ngoài ra, mỗi bệnh viện sẽ có một hệ thống các máy chụp chẩn đoán hình ảnh, máy xét nghiệm, máy đo… khác nhau, tuy nhiên chưa có hoặc có rất ít các hệ thống liên kết trung gian giữa các loại máy đặc thù này với hệ thống cơ sở dữ liệu của bệnh viện, khiến cho quy trình khám chữa bệnh tồn tại sự tắc nghẽn khi bệnh nhân cần chụp chẩn đoán, xét nghiệm, đo chỉ số cơ thể. Đến bước này, thông tin bệnh nhân thường sẽ được nhập liệu lại từ đầu, các kết quả về hình ảnh được in thủ công ra giấy hoặc một số vật liệu đặc thù khác (phim chụp X-quang), khiến chất lượng hình ảnh bị giảm thiểu đáng kể, hoặc yêu cầu bác sĩ cần di chuyển đến các

khoa làm việc liên quan để xem hình ảnh, thông số xét nghiệm, dẫn đến hiệu suất khám chữa bệnh của bác sĩ không được cao như mong đợi, gây phiền hà cho cả bác sĩ lẫn người bệnh.

Do đó, em đã chọn đề tài "Xây dựng hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm tại Bệnh viện Da liễu Trung ương". Đề tài tập trung phát triển hệ thống liên quan đến các bệnh viện da liễu hoặc khoa da liễu tại các bệnh viện hiện nay. Trong khuôn khổ đồ án, em đã phát triển một hệ thống trung gian liên kết giữa máy chụp chẩn đoán hình ảnh, kết hợp cùng mô hình dự đoán bệnh ung thư hắc tố da Melanoma của bạn Nguyễn Trí Hùng. Hệ thống cung cấp quy trình làm việc trực quan, dễ hiểu từ khi tiếp nhận bệnh nhân, dựa trên chẩn đoán sơ bộ ban đầu của bác sĩ để xác định xem bệnh nhân cần làm những xét nghiệm gì và có cần chụp chẩn đoán hình ảnh không. Phần hình ảnh kết xuất được từ máy chụp chẩn đoán hình ảnh và mô hình chẩn đoán ung thư hắc tố da sẽ được kiểm tra và tải lên hệ thống tương ứng với thông tin bệnh nhân, cho phép bác sĩ có thể xem thông tin và xem các kết quả hình ảnh cũng như kết quả xét nghiệm tương ứng của bệnh nhân.

1.2 Mục tiêu và phạm vi đề tài

Dựa trên những phân tích về vấn đề được nêu ra ở mục 1.1, đồ án tốt nghiệp của em tập trung vào các mục tiêu chính sau đây:

Thứ nhất, xây dựng hệ thống xác thực và phân quyền rõ ràng cho các vai trò tương ứng với chức năng và nhiệm vụ của bác sĩ, nhân viên bệnh viện (nhân viên chụp chẩn đoán hình ảnh, nhân viên xét nghiệm, nhân viên bệnh viện,…) với giao diện dễ nhìn, đẹp mắt.

Thứ hai, cung cấp các chức năng giúp trực quan hóa hình ảnh giúp cho nhân viên chụp chẩn đoán hình ảnh và bác sĩ dễ theo dõi và chẩn đoán, đồng thời liên kết với các file kết quả (file .csv và file mask) của mô hình chẩn đoán ung thư hắc tố da, từ đó tiến hành xử lý để cho ra những điểm khác điểm khác biệt so với hình ảnh chụp chẩn đoán ban đầu. Ngoài ra, hệ thống còn liên kết với các phiếu thông tin xét nghiệm của bệnh nhân, giúp nâng cao hiệu suất khám chữa bệnh của bác sĩ.

Thứ ba, cung cấp thông tin về hồ sơ bệnh nhân kèm các thông tin khám chữa bệnh và các kết quả xét nghiệm, chụp chẩn đoán hình ảnh tương ứng liên quan của bệnh nhân.

Hệ thống hướng tới triển khai tại các bệnh viện da liễu hoặc khoa da liễu tại các bệnh viện. Do triển khai và ứng dụng tại các bệnh viện nên hệ thống phải đảm bảo an toàn, có hiệu năng tốt, giao diện phải đơn giản và dễ dùng, đồng thời hệ thống cũng cần đảm bảo khả năng bảo trì và mở rộng sau này. Các mẫu xét nghiệm và chụp chẩn đoán hình ảnh cần tuân theo các mẫu [2] đã quy định bởi Cục quản lý khám chữa bệnh trực thuộc Bộ Y tế.

1.3 Định hướng giải pháp

Do hệ thống hoạt động trên nền tảng máy tính và phải đảm bảo tính bảo mật nên đồ án của em hướng đến xây dựng một hệ thống trên nền tảng web chạy trên một server cục bộ của từng bệnh viện, không kết nối đến Internet bên ngoài. Hệ thống được phát triển kiến trúc Microservices. Hệ thống được chia thành hai thành phần tách biệt là backend và frontend, tương tác và giao tiếp với nhau thông qua REST API, giúp cho việc mở rộng hệ thống trong tương lai dễ dàng hơn.

Phía backend, em sử dụng NodeJS cùng framework ExpressJS, dữ liệu được lưu trữ bởi MongoDB. Phía frontend, em sử dụng thư viện ReactJS, kết hợp cùng Redux để quản lý trạng thái của ứng dụng, đồng thời để giao diện được đẹp mắt và thân thiện với người dùng, em đã sử dụng framework Bootstrap. Nhìn tổng thể, hệ thống của em tuân theo MERN stack – là sự kết hợp của MongoDB, ExpressJS, ReactJS và NodeJS. Đây là một stack công nghệ phổ biến được các lập trình viên ưa dùng vì độ hiệu quả và mang tính thống nhất cao do dùng chung một hệ sinh thái dựa trên ngôn ngữ lập trình Javascript.

1.4 Bố cục đồ án

Các chương còn lại của đồ án được trình bày với thứ tự như sau.

Trong Chương 2, em sẽ khảo sát hệ thống hiện có và quy trình làm việc cụ thể tại bệnh viện Da liễu Trung ương. Qua khảo sát thực tế, em sẽ phân tích yêu cầu, trình bày tổng quan các chức năng và đi vào phân tích, làm rõ từng chức năng của hệ thống.

Ở chương 3, sau khi đã phân tích được các yêu cầu và làm rõ các chức năng chính của hệ thống, em tiến hành lựa chọn các công nghệ phù hợp để xây dựng và phát triển hệ thống. Chương này bao gồm mô tả về các công nghệ và ưu điểm của chúng khi áp dụng trong hệ thống.

Tiếp theo, chương 4 mô tả chi tiết việc thiết kế và triển khai kiến trúc, cơ sở dữ liệu, quá trình xây dựng ứng dụng, kiểm thử và triển khai.

Chương 5 trình bày những đóng góp, giải pháp tâm đắc mà em đưa ra trong quá trình làm đồ án tốt nghiệp về đề tài này.

Cuối cùng, trong chương 6, em tổng kết về các đóng góp chính của đồ án, những ưu điểm, nhược điểm, đồng thời, đề ra hướng phát triển trong tương lai cho hệ thống.

Chương 2 Khảo sát và phân tích yêu cầu

Sau khi giới thiệu tổng quan về đề tại ở chương 1, trong chương 2, em sẽ trình bày về việc khảo sát hiện trạng tại bệnh viện, các phần mềm hoặc hệ thống trên thị trường hiện nay và phân tích chi tiết yêu cầu của hệ thống.

2.1 Khảo sát hiện trạng

Qua khảo sát thực tế tại Bệnh viện Da liễu Trung ương, hiện tại bệnh viện chưa có phần mềm nào liên kết trung gian quy trình chẩn đoán hình ảnh và xét nghiệm với quy trình khám chữa bệnh của bác sĩ. Cụ thể, bệnh viện đang sử dụng phần mềm Fotofinder [1] và máy chụp chẩn đoán đi kèm cho việc chẩn đoán hình ảnh. Nhược điểm của phần mềm này là nó chạy hoàn toàn độc lập với hệ thống cơ sở dữ liệu của bệnh viện, chỉ kết nối với máy chụp chẩn đoán và đọc các file hình ảnh từ máy. Mỗi lần có bệnh nhân cần chụp chẩn đoán hình ảnh thì sẽ có một nhân viên phụ trách nhập liệu thông tin bệnh nhân trên cơ sở dữ liệu của bệnh viện, đồng thời nhân viên chụp chẩn đoán hình ảnh cũng phải nhập lại thông tin bệnh nhân vào phần mềm trước khi tiến hành chụp chẩn đoán hình ảnh để có thể kết xuất ra báo cáo lúc hoàn thành. Các file hình ảnh được phân loại tay và không được liên kết với thông tin bệnh nhân nên rất khó để quản lý nếu số lượng file quá lớn.

Hiện tại, trên thị trường đã có khá nhiều phần mềm quản lý bệnh viện, phòng khám và vật tư y tế, tuy nhiên chỉ có một vài phần mềm hỗ trợ tính năng liên quan đến chẩn đoán hình ảnh và kết nối thông tin trực tiếp đến bác sĩ. Cụ thể, các phần mềm nổi bật bao gồm: Phần mềm chẩn đoán hình ảnh Ehis (1), Phần mềm NANO HOSPITAL (2), Phần mềm của FPT.eHOSPITAL (3), Hệ thống quản lý bệnh viện Viettel – HIS (4).

Phần mềm chẩn đoán hình ảnh Ehis là một phần của hệ sinh thái phần mềm quản lý bệnh viện Ehis. Phần mềm có các chức năng như: Quản lý yêu cầu chẩn đoán hình ảnh, Quản lý kết quả chẩn đoán hình ảnh, Trả kết quả tự động về máy tính, Báo cáo kết quả chẩn đoán hình ảnh. Đây là phần mềm cung cấp khá đầy đủ các chức năng liên quan đến chẩn đoán hình ảnh. Tuy nhiên do gói phần mềm đầy đủ các chức năng có chi phí khá cao nên các bệnh viện thường chỉ mua gói cơ bản với các chức năng cơ bản mà không có các chức năng hỗ trợ quản lý chẩn đoán hình ảnh.

---

[1] https://www.fotofinder.de/en/

Phần mềm NANO HOSPITAL có tính năng quản lý chẩn đoán hình ảnh – thăm dò chức năng. Cụ thể tính năng này giúp các phòng nhận được danh sách bệnh nhân chờ, xem thông tin bệnh nhân, thông tin chỉ định, nhập kết quả và trả kết quả về cho bác sỹ.

Phần mềm FPT.eHOSPITAL có tổng cộng 12 nhóm phần mềm với các phân hệ chức năng khác nhau, trong đó có hệ thống Quản lý chẩn đoán hình ảnh và thăm dò chức năng, kết nối trực tiếp với một số loại máy cơ bản để nhận được hình ảnh, quản lý và lưu trữ hình ảnh, kết xuất báo cáo thống kê, hỗ trợ kết nối thông tin bệnh nhân tới bác sĩ. Gói phần mềm này có chi phí khá cao so với mặt bằng trung và hiện đang được triển khai tại một số bệnh viện Trung ương tuyến đầu, chưa triển khai nhiều tại các bệnh viện.

Hệ thống quản lý bệnh viện Viettel – HIS hiện đang triển khai các chức năng liên quan đến quản lý chẩn đoán hình ảnh và thăm dò chức năng trong gói các chức năng nâng cao. Trong gói các tính năng cơ bản do hệ thống HIS liệt kê chưa thấy xuất hiện các tính năng này.

Trong các phần mềm trên, có phần mềm chẩn đoán hình ảnh Ehis và phần mềm FPT.eHOSPITAL cung cấp khá nhiều tính năng hiện đại và phân tích, kết nối được các kết quả chẩn đoán hình ảnh và xét nghiệm của bệnh nhân với bác sĩ, giảm bớt thời gian chờ giữa các khâu khám chữa bệnh và nâng cao hiệu suất của bác sĩ.

Hệ thống em làm trong đồ án lần này cũng có các chức năng và luồng quy trình làm việc giống với các phần mềm trên, tuy nhiên, em tập trung vào xây dựng một hệ thống trung gian mà không tác động trực tiếp đến những hệ thống phần cứng và phần mềm chuyên biệt mà bệnh viện đã có. Hệ thống sẽ đọc và lưu trữ những file ảnh chụp chẩn đoán hình ảnh đầu ra từ các máy và các phiếu xét nghiệm liên quan đến bệnh nhân, ngoài ra còn hỗ trợ đọc và phân tích các file kết quả nhận được từ mô hình chẩn đoán ung thư hắc tố da (Melanoma) của bạn Nguyễn Trí Hùng.

2.2 Tổng quan chức năng

2.2.1 Biểu đồ use case tổng quan của hệ thống

Biểu đồ use case tổng quan với các chức năng chính của hệ thống được mô tả trong **Hình 1**.

Hệ thống gồm 5 tác nhân, bao gồm: Admin, nhân viên bệnh viện, bác sĩ, nhân viên chẩn đoán hình ảnh và nhân viên xét nghiệm. Mỗi tác nhân có vai trò nhất định trong hệ thống, cụ thể:

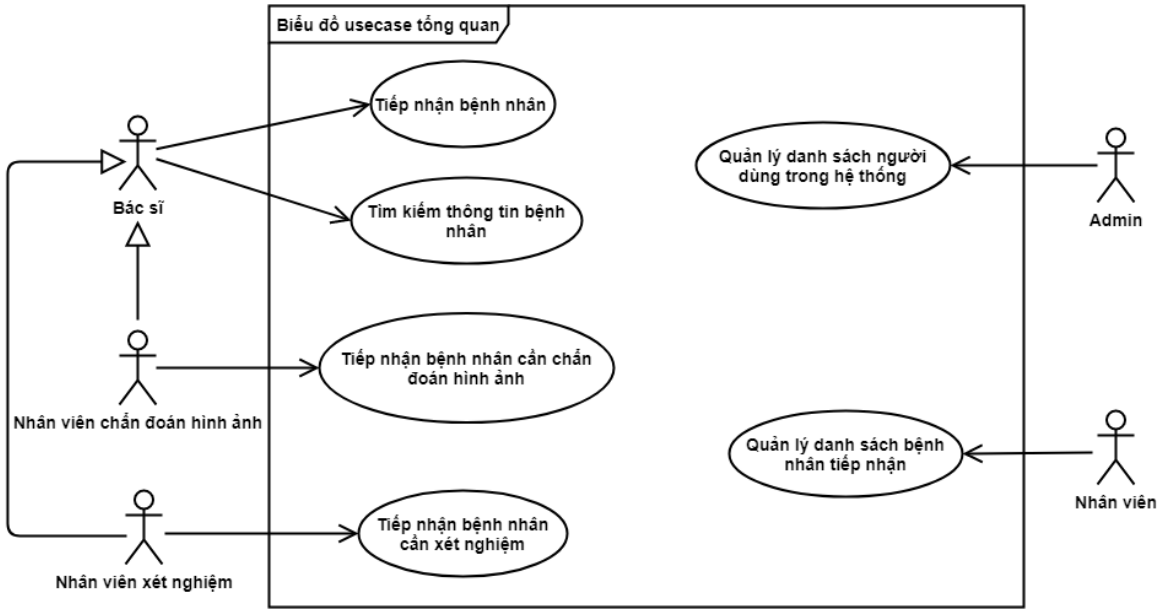
Admin là người quản lý toàn bộ danh sách người dùng trong hệ thống, có quyền thêm mới tài khoản người dùng, cập nhật thông tin tài khoản hoặc xóa một tài khoản người dùng khỏi hệ thống.

Nhân viên bệnh viện là người quản lý danh sách bệnh nhân tiếp nhận, cụ thể là nhập các thông tin bệnh nhân và điều phối các bệnh nhân tới phòng khám phù hợp, đồng thời có quyền xem và cập nhật lại thông tin bệnh nhân.

Bác sĩ sẽ tiếp nhận và xem được danh sách những bệnh nhân được điều phối từ nhân viên tại phòng khám của mình, tiến hành chẩn đoán bệnh và quyết định điều phối bệnh nhân làm những xét nghiệm cần thiết và chụp chẩn đoán hình ảnh. Bác sĩ có quyền xem toàn bộ thông tin liên quan đến quá trình khám bệnh của bệnh nhân như phiếu xét nghiệm, kết quả chụp chẩn đoán hình ảnh. Bác sĩ có thể tìm kiếm bệnh nhân dựa trên mã bệnh nhân hoặc dựa trên các chẩn đoán về bệnh.

Nhân viên chẩn đoán hình ảnh sẽ nhận được danh sách những bệnh nhân cần chụp chẩn đoán hình ảnh và tiến hành xử lý. Sau khi hoàn thành quá trình chụp, tải các ảnh chụp cần thiết lên hệ thống để xác nhận đã xử lý xong bệnh nhân.

Nhân viên xét nghiệm cũng sẽ nhận được danh sách những bệnh nhân cần làm xét nghiệm và tiến hành các xét nghiệm liên quan. Sau khi bệnh nhân hoàn thành các xét nghiệm, nhân viên xét nghiệm cần nhập dầy đủ thông tin vào các mẫu xét nghiệm có sẵn để xác nhận đã xét nghiệm xong bệnh nhân.

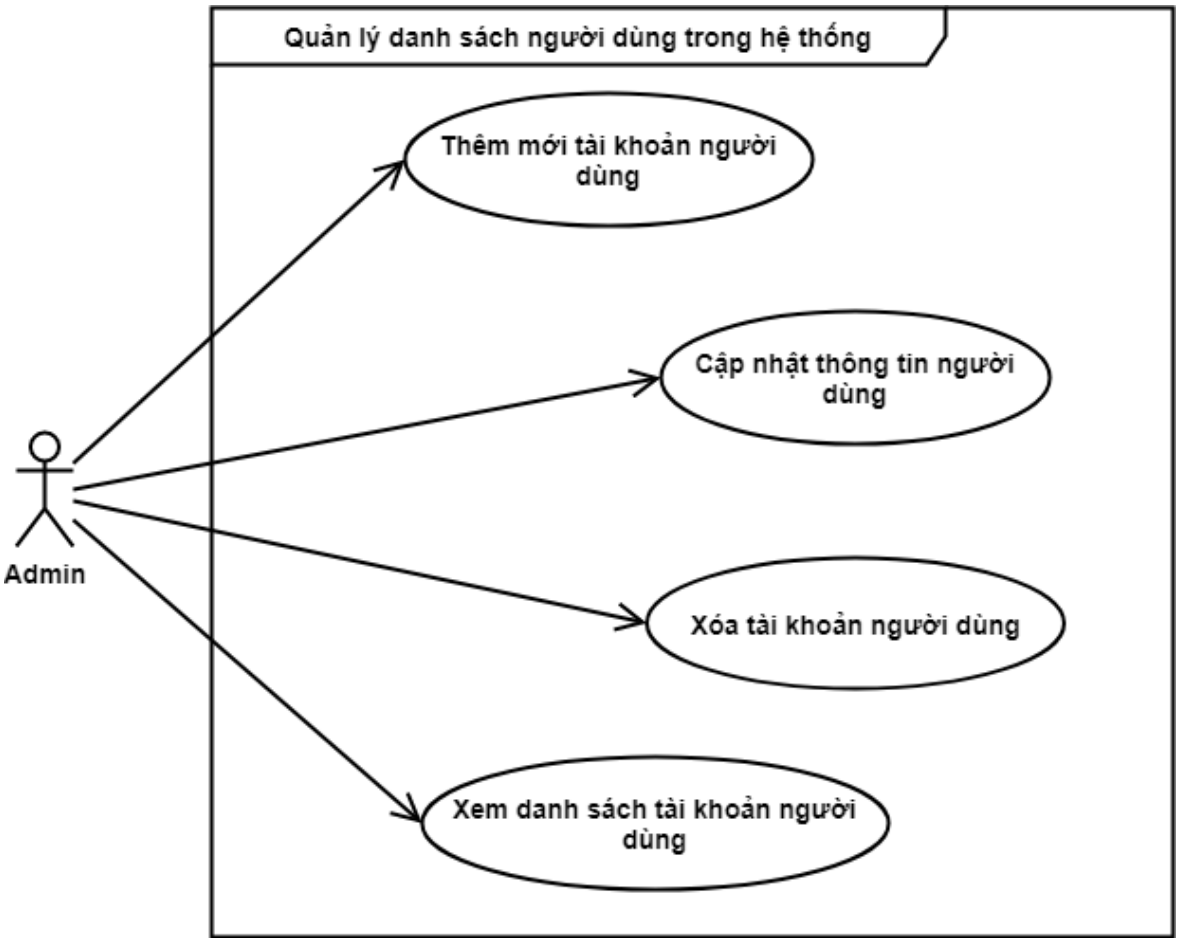


Hình 1 Biểu đồ use case tổng quan

2.2.2 Biểu đồ use case phân rã Quản lý danh sách người dùng trong hệ thống

Hình 2 mô tả use case phân rã Quản lý danh sách người dùng trong hệ thống. Để đảm bảo hệ thống không có bất kỳ tài khoản không hợp lệ nào thuộc người dùng bên ngoài hệ thống,

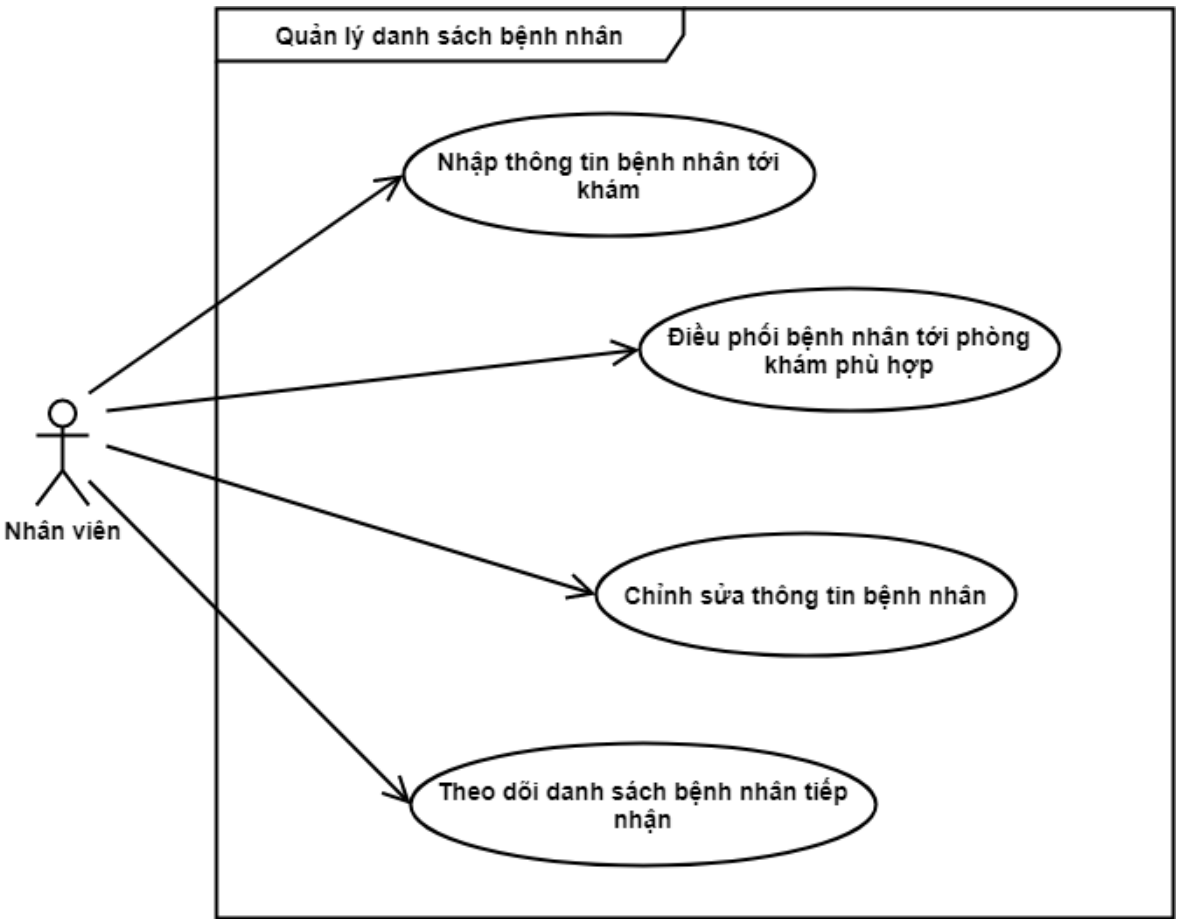
admin sẽ là người tạo và thêm mới các tài khoản người dùng ứng với các bác sĩ và nhân viên bệnh viện. Đồng thời, admin cũng có thể xóa một tài khoản nếu nhân viên ứng với tài khoản đó không còn làm việc ở bệnh viện nữa và xem danh sách các tài khoản có trong hệ thống.



Hình 2 Biểu đồ use case phân rã Quản lý danh sách người dùng trong hệ thống

2.2.3 Biểu đồ use case phân rã Quản lý danh sách bệnh nhân

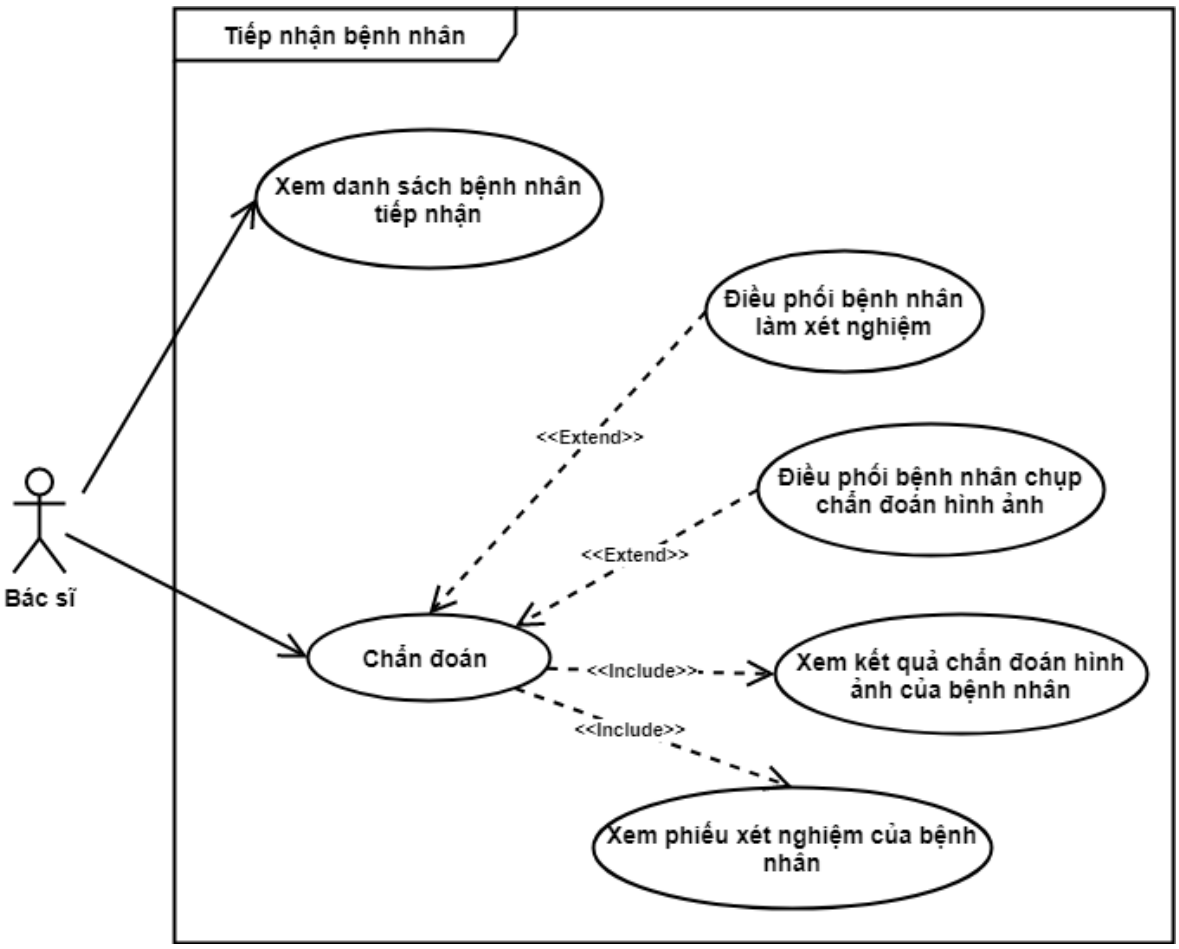
Hình 3 mô tả biểu đồ use case phân rã Quản lý danh sách bệnh nhân của nhân viên. Nhân viên bệnh viện sẽ tiếp nhận bệnh nhân và nhập thông tin bệnh nhân đến khám, sau đó điều phối bệnh nhân đến phòng khám phù hợp. Nhân viên có thể theo dõi và xem lại danh sách bệnh nhân tiếp nhận. Đồng thời, nhân viên cũng có thể chỉnh sửa thông tin bệnh nhân nếu xảy ra sai sót hoặc dư thừa thông tin.



Hình 3 Biểu đồ use case phân rã Quản lý danh sách bệnh nhân

2.2.4 Biểu đồ use case phân rã Tiếp nhận bệnh nhân

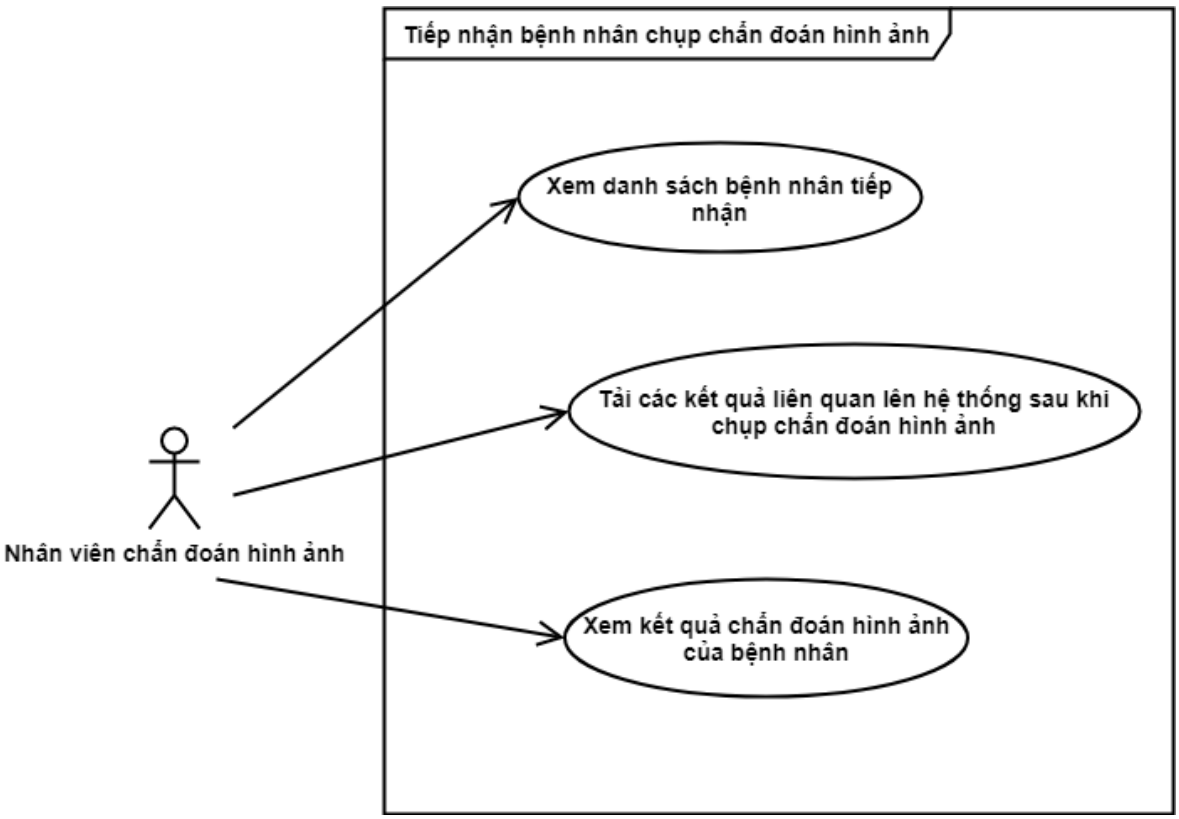
Hình 4 mô tả biểu đồ use case phân rã Tiếp nhận bệnh nhân của bác sĩ. Sau khi nhân viên bệnh viện điều phối bệnh nhân, bác sĩ ở từng phòng khám sẽ nhận được danh sách bệnh nhân tương ứng. Bác sĩ sẽ tiến hành chẩn đoán và xác nhận xem bệnh nhân có cần chụp chẩn đoán hình ảnh và các xét nghiệm liên quan không, sau đó điều phối công việc cho các bộ phận tương ứng tiếp nhận. Sau khi có kết quả, bác sĩ có thể trực tiếp xem phiếu xét nghiệm và kết quả chẩn đoán hình ảnh của từng bệnh nhân, sau đó đưa ra chẩn đoán cuối cùng.



Hình 4 Biểu đồ use case phân rã Tiếp nhận bệnh nhân

2.2.5 Biểu đồ use case phân rã Tiếp nhận bệnh nhân cần chẩn đoán hình ảnh

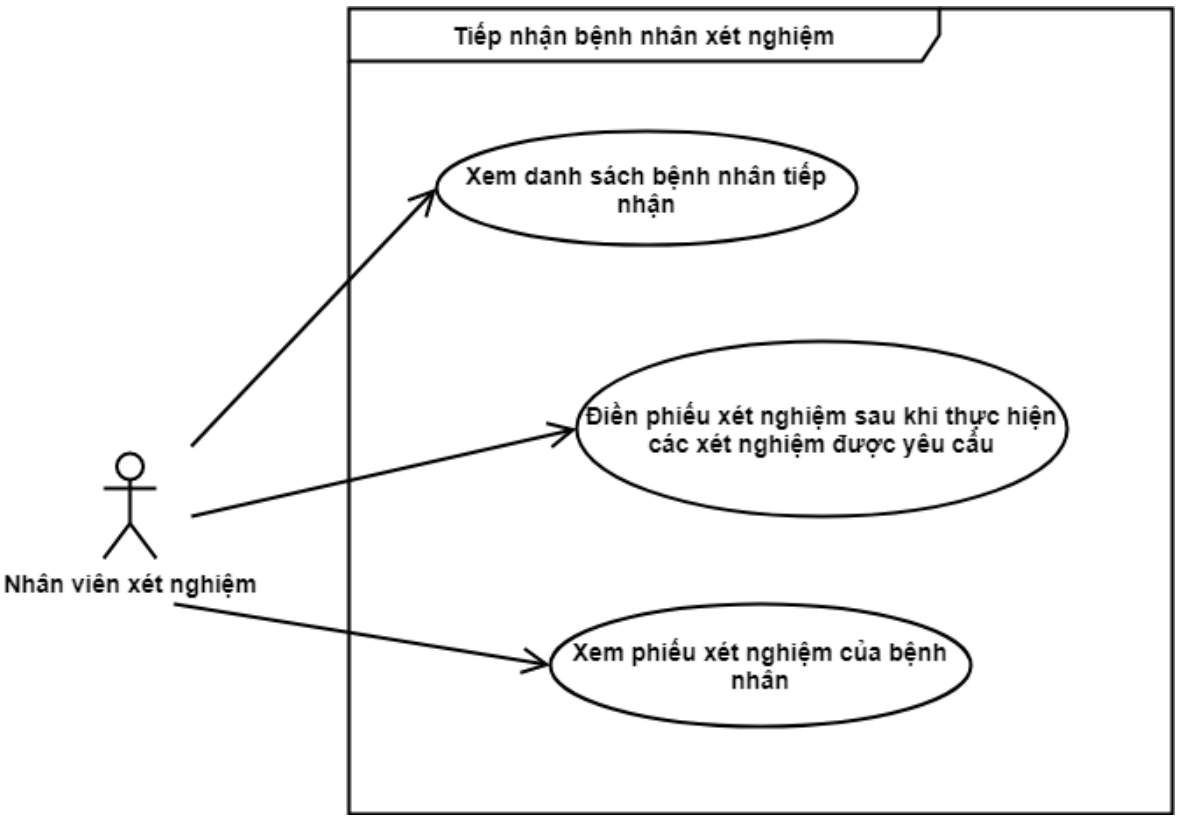
Hình 5 mô tả biểu đồ use case phân rã Tiếp nhận bệnh nhân cần chẩn đoán hình ảnh. Nhân viên chẩn đoán hình ảnh sẽ nhận được danh sách bệnh nhân cần chụp chẩn đoán hình ảnh từ bác sĩ, từ đó tiến hành chụp chẩn đoán hình ảnh và tải các hình ảnh liên quan để xác nhận là bệnh nhân đã chụp chẩn đoán xong. Các hình ảnh sẽ được hệ thống chuyển tới mô hình chẩn đoán ung thư hắc tố da và đưa ra các kết quả tương ứng kèm báo cáo được kết xuất. Cả nhân viên chẩn đoán hình ảnh và bác sĩ đều có thể xem được báo cáo này.



Hình 5 Biểu đồ use case phân rã Tiếp nhận bệnh nhân cần chẩn đoán hình ảnh

2.2.6 Biểu đồ use case phân rã Tiếp nhận bệnh nhân cần xét nghiệm

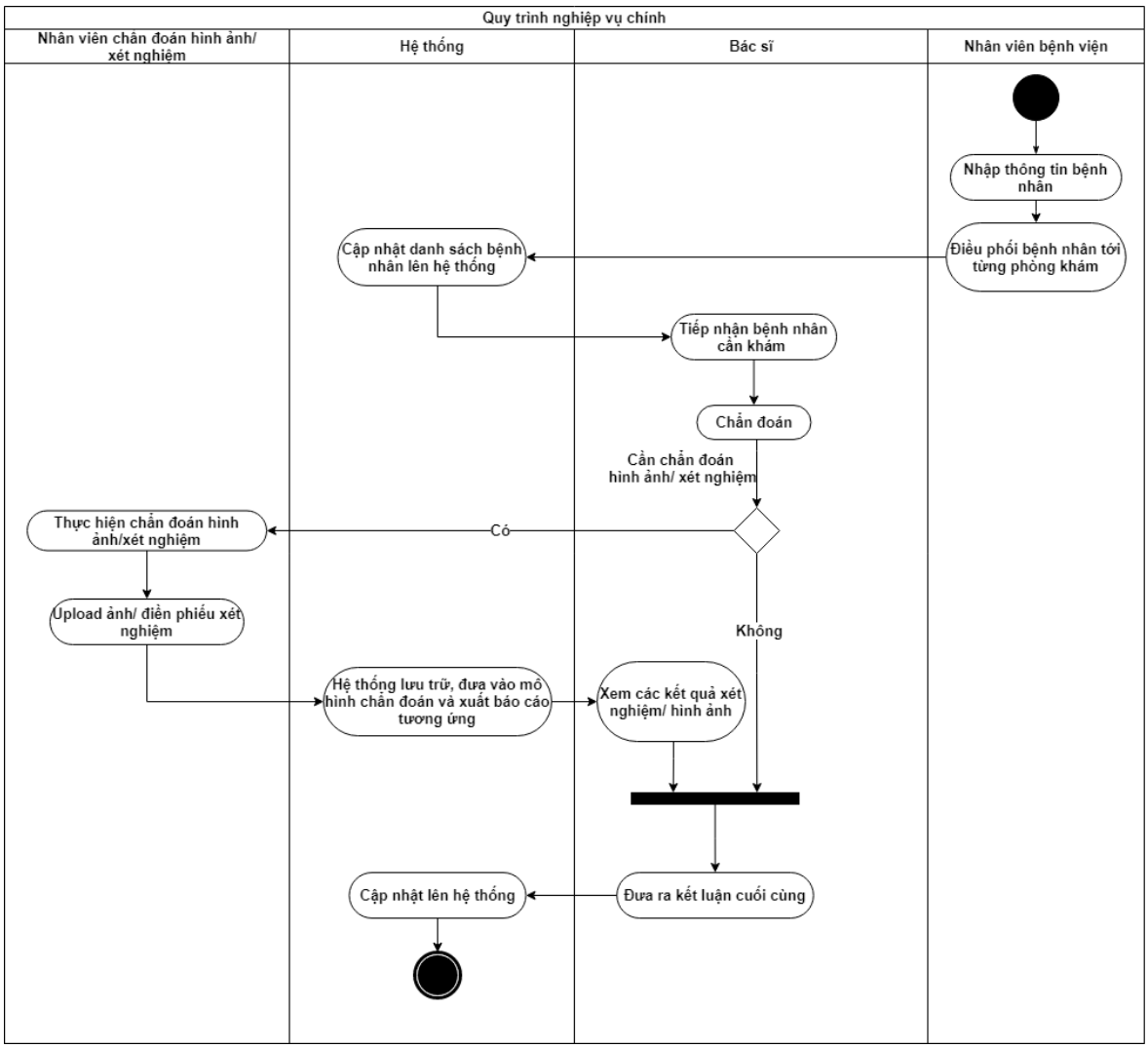
Hình 6 mô tả biểu đồ use case phân rã Tiếp nhận bệnh nhân cần xét nghiệm. Nhân viên xét nghiệm sẽ nhận được danh sách bệnh nhân cần xét nghiệm từ bác sĩ, từ đó tiến hành các xét nghiệm cần thiết và điền phiếu kết quả xét nghiệm tương ứng để xác nhận là bệnh nhân đã xét nghiệm xong. Các phiếu xét nghiệm sẽ được lưu trên hệ thống. Cả nhân viên xét nghiệm và bác sĩ đều có thể xem được báo cáo này.



Hình 6 Biểu đồ use case phân rã Tiếp nhận bệnh nhân cần xét nghiệm

2.2.7 Quy trình nghiệp vụ

Trong phần này, em xin trình bày quy trình nghiệp vụ chính của hệ thống là quy trình tiếp nhận và khám chữa bệnh được mô tả ở **Hình 7**. Nhân viên bệnh viện sẽ là người tiếp nhận và nhập thông tin bệnh nhân, sau đó điều phối bệnh nhân đến phòng khám phù hợp. Tại phòng khám, bác sĩ sẽ khám cho bệnh nhân và quyết định xem bệnh nhân có cần chụp chẩn đoán hình ảnh hoặc làm xét nghiệm không. Nếu có, nhân viên các phòng chẩn đoán hình ảnh và xét nghiệm sẽ tiếp nhận bệnh nhân và trả lại kết quả tương ứng cho bác sĩ. Sau khi nhận được kết quả cuối cùng, bác sĩ có thể tiến hành đưa ra chẩn đoán cuối cùng cho bệnh nhân.



Hình 7 Quy trình tiếp nhận và khám chữa bệnh

2.3 Đặc tả chức năng

Trong phần này, em sẽ đặc tả một số chức năng chính của hệ thống. **Bảng 1** liệt kê 20 use case được sử dụng trong đồ án. Do khuôn khổ đồ án có hạn nên em sẽ tập trung đặc tả chi tiết 4 use case, bao gồm: (1) Xem danh sách bệnh nhân tiếp nhận, (2) Chẩn đoán bệnh nhân, (3) Kết xuất báo cáo chẩn đoán hình ảnh, (4) Tìm kiếm thông tin bệnh nhân.

Bảng 1 Danh sách các use case

Mã use case	Tên use case	Mã use case	Tên use case
UC001	Nhập thông tin bệnh nhân	UC002	Xem danh sách bệnh nhân tiếp nhận

Mã use case	Tên use case	Mã use case	Tên use case
UC003	Chỉnh sửa thông tin bệnh nhân	UC004	Chỉnh sửa thông tin người dùng
UC005	Thêm mới tài khoản người dùng	UC006	Xóa tài khoản người dùng
UC007	Xem danh sách tài khoản người dùng	UC008	Chẩn đoán bệnh nhân
UC009	Xem danh sách bệnh nhân cần chẩn đoán	UC010	Tải lên kết quả chụp chẩn đoán
UC011	Xem danh sách bệnh nhân cần chẩn đoán hình ảnh	UC012	Xem báo cáo chẩn đoán hình ảnh
UC013	Kết xuất báo cáo chẩn đoán hình ảnh	UC014	Điền phiếu xét nghiệm sau khi thực hiện xét nghiệm
UC015	Xem danh sách bệnh nhân cần xét nghiệm	UC016	Xem phiếu xét nghiệm
UC017	Xuất phiếu xét nghiệm	UC018	Xem thông tin bệnh nhân
UC019	Tìm kiếm thông tin bệnh nhân	UC020	Xem kết quả chẩn đoán ung thư hắc tố da

2.3.1 Đặc tả use case Xem danh sách bệnh nhân cần tiếp nhận

Bảng 2 Đặc tả use case Xem danh sách bệnh nhân tiếp nhận

Mã use case	UC002	Tên use case	Use case Xem danh sách bệnh nhân tiếp nhận
Tác nhân	Nhân viên bệnh viện		

<table>
<tr><td>Tiền điều kiện</td><td colspan="3">Người dùng đăng nhập vào hệ thống với vai trò là Nhân viên</td></tr>
<tr><td>Luồng sự kiện chính (Thành công)</td><td colspan="3"><table>
<tr><td>STT</td><td>Thực hiện bởi</td><td>Hành động</td></tr>
<tr><td>1.</td><td>Nhân viên</td><td>Chọn mục xem danh sách bệnh nhân tiếp nhận</td></tr>
<tr><td>2.</td><td>Hệ thống</td><td>Truy vấn cơ sở dữ liệu để lấy thông tin về danh sách bệnh nhân</td></tr>
<tr><td>3.</td><td>Hệ thống</td><td>Hiển thị danh sách bệnh nhân tiếp nhận</td></tr>
<tr><td>4.</td><td>Nhân viên</td><td>Xem danh sách bệnh nhân tiếp nhận</td></tr>
</table></td></tr>
<tr><td>Luồng sự kiện thay thế</td><td colspan="3"><table>
<tr><td>STT</td><td>Thực hiện bởi</td><td>Hành động</td></tr>
<tr><td>2a.</td><td>Hệ thống</td><td>Truy vấn cơ sở dữ liệu không thành công</td></tr>
<tr><td>3a.</td><td>Hệ thống</td><td>Thông báo lỗi: Không thể tải được danh sách bệnh nhân</td></tr>
</table></td></tr>
<tr><td>Hậu điều kiện</td><td colspan="3">Không</td></tr>
</table>

2.3.2 Đặc tả use case Chẩn đoán bệnh nhân

Bảng 3 Đặc tả use case Chẩn đoán bệnh nhân

Mã use case	UC010	Tên use case	Use case Chẩn đoán bệnh nhân
Tác nhân	Bác sĩ		
Tiền điều kiện	+ Người dùng đăng nhập vào hệ thống với vai trò là Bác sĩ. + Danh sách tiếp nhận bệnh nhân có bệnh nhân cần chẩn đoán.		

<table>
<tr><td>Luồng sự kiện chính (Thành công)</td><td>
<table>
<tr><th>STT</th><th>Thực hiện bởi</th><th>Hành động</th></tr>
<tr><td>1.</td><td>Bác sĩ</td><td>Ấn vào bệnh nhân cần chẩn đoán</td></tr>
<tr><td>2.</td><td>Hệ thống</td><td>Hiển thị giao diện chẩn đoán bệnh nhân</td></tr>
<tr><td>3.</td><td>Bác sĩ</td><td>Nhập các chẩn đoán và các trường dữ liệu cần thiết (mô tả dưới *)</td></tr>
<tr><td>4.</td><td>Hệ thống</td><td>Kiểm tra thông tin các trường dữ liệu mà bác sĩ nhập</td></tr>
<tr><td>5.</td><td>Hệ thống</td><td>Cập nhật cơ sở dữ liệu và điều phối cho các bên liên quan</td></tr>
<tr><td>6.</td><td>Hệ thống</td><td>Hiển thị giao diện danh sách bệnh nhân với các trường dữ liệu cập nhật</td></tr>
</table>
</td></tr>
<tr><td>Luồng sự kiện thay thế</td><td>
<table>
<tr><th>STT</th><th>Thực hiện bởi</th><th>Hành động</th></tr>
<tr><td>3a</td><td>Bác sĩ</td><td>Chưa nhập đủ thông tin các trường dữ liệu được yêu cầu</td></tr>
<tr><td>4a</td><td>Hệ thống</td><td>Thông báo lỗi: Yêu cầu nhập đủ thông tin</td></tr>
<tr><td>5a</td><td>Hệ thống</td><td>Cập nhật cơ sở dữ liệu bị lỗi</td></tr>
<tr><td>6a</td><td>Hệ thống</td><td>Thông báo lỗi: Chưa thể lưu thông tin do có lỗi phía máy chủ, vui lòng thử lại</td></tr>
</table>
</td></tr>
<tr><td>Hậu điều kiện</td><td>Không</td></tr>
</table>

* Dữ liệu đầu vào use case Chẩn đoán bệnh nhân

Bảng 4 Dữ liệu đầu vào use case Chẩn đoán bệnh nhân

STT	Trường dữ liệu	Mô tả	Bắt buộc
1.	Phòng khám	Đã được điền sẵn do lấy thông tin tự động từ cơ sở dữ liệu	Có
2.	Bác sĩ điều trị	Đã được điền sẵn do lấy thông tin tự động từ cơ sở dữ liệu	Có
3.	Chẩn đoán ban đầu	Chẩn đoán ban đầu của bác sĩ	Có
4.	Các chẩn đoán và xét nghiệm cần thiết	Chọn chẩn đoán hình ảnh hoặc các xét nghiệm cần thiết mà bệnh nhân cần làm	Không

2.3.3 Đặc tả use case Kết xuất báo cáo chẩn đoán hình ảnh

Bảng 5 Đặc tả use case Kết xuất báo cáo chẩn đoán hình ảnh

<table>
<tr><td>Mã use case</td><td>UC0013</td><td>Tên use case</td><td>Use case Kết xuất báo cáo chẩn đoán hình ảnh</td></tr>
<tr><td>Tác nhân</td><td colspan="3">Nhân viên chẩn đoán hình ảnh</td></tr>
<tr><td>Tiền điều kiện</td><td colspan="3">+ Người dùng đăng nhập vào hệ thống với vai trò là Nhân viên chẩn đoán hình ảnh
+ Bệnh nhân đã hoàn thành chụp chẩn đoán hình ảnh
+ Nhân viên chẩn đoán hình ảnh đã tải kết quả lên hệ thống</td></tr>
<tr><td>Luồng sự kiện chính (Thành công)</td><td colspan="3"><table><tr><td>STT</td><td>Thực hiện bởi</td><td>Hành động</td></tr><tr><td>1.</td><td>Nhân viên chẩn đoán hình ảnh</td><td>Yêu cầu kết xuất báo cáo chẩn đoán hình ảnh</td></tr></table></td></tr>
</table>

	2.	Hệ thống	Lấy dữ liệu từ cơ sở dữ liệu	
	3.	Hệ thống	Đưa các file hình ảnh vào mô hình chẩn đoán ung thư hắc tố da và lấy ra kết quả	
	4.	Hệ thống	Hiển thị báo cáo chẩn đoán hình ảnh dưới định dạng mẫu	
	5.	Nhân viên chẩn đoán hình ảnh	Xem báo cáo chẩn đoán hình ảnh	
	6.	Nhân viên chẩn đoán hình ảnh	Yêu cầu xuất file báo cáo dưới dạng pdf	
	7.	Hệ thống	Thực hiện download file báo cáo dưới dạng pdf về máy người dùng	
Luồng sự kiện thay thế	STT	Thực hiện bởi	Hành động	
	2a.	Hệ thống	Lấy dữ liệu không thành công	
	4a1.	Hệ thống	Thông báo lỗi: Lỗi máy chủ, vui lòng thử lại	
	3a.	Hệ thống	Không kết nối được với mô hình chẩn đoán ung thư hắc tố da	
	4a2.	Hệ thống	Thông báo lỗi: Không kết nối được hệ thống phân tích chẩn đoán ung thư hắc tố da	
Hậu điều kiện	Không			

2.3.4 Đặc tả use case Tìm kiếm thông tin bệnh nhân

Bảng 6 Đặc tả use case Tìm kiếm thông tin bệnh nhân

Mã use case	UC019	Tên use case	Use case Tìm kiếm thông tin bệnh nhân

<table>
<tr><td>Tác nhân</td><td>Bác sĩ</td></tr>
<tr><td>Tiền điều kiện</td><td>Người dùng đăng nhập vào hệ thống với vai trò là Bác sĩ</td></tr>
<tr><td>Luồng sự kiện chính (Thành công)</td><td>
<table>
<tr><th>STT</th><th>Thực hiện bởi</th><th>Hành động</th></tr>
<tr><td>1.</td><td>Bác sĩ</td><td>Ấn vào mục Tìm kiếm thông tin bệnh nhân</td></tr>
<tr><td>2.</td><td>Hệ thống</td><td>Hiển thị giao diện tìm kiếm tương ứng với các trường dữ liệu mà bác sĩ nhập</td></tr>
<tr><td>3.</td><td>Bác sĩ</td><td>Nhập các trường dữ liệu được yêu cầu (mô tả dưới *) và ấn Tìm kiếm</td></tr>
<tr><td>4.</td><td>Hệ thống</td><td>Kiểm tra các trường dữ liệu người dùng nhập</td></tr>
<tr><td>5.</td><td>Hệ thống</td><td>Hiển thị kết quả tìm kiếm dựa trên các trường dữ liệu được nhập tương ứng</td></tr>
<tr><td>6.</td><td>Bác sĩ</td><td>Xem kết quả tìm kiếm</td></tr>
</table>
</td></tr>
<tr><td>Luồng sự kiện thay thế</td><td>
<table>
<tr><th>STT</th><th>Thực hiện bởi</th><th>Hành động</th></tr>
<tr><td>3a.</td><td>Bác sĩ</td><td>Nhập không đủ các trường dữ liệu được yêu cầu hoặc không nhập</td></tr>
<tr><td>4a.</td><td>Hệ thống</td><td>Thông báo lỗi: Vui lòng nhập đủ các trường dữ liệu được yêu cầu</td></tr>
</table>
</td></tr>
<tr><td>Hậu điều kiện</td><td>Không</td></tr>
</table>

* Dữ liệu đầu vào use case Tìm kiếm thông tin bệnh nhân

Bảng 7 Dữ liệu đầu vào use case Tìm kiếm thông tin bệnh nhân

STT	Trường dữ liệu	Mô tả	Bắt buộc
1.	Mã bệnh nhân		Không (**)
2.	Chẩn đoán bệnh	Điền chẩn đoán bệnh của bệnh nhân từ lần chẩn đoán cuối để tìm kiếm thông tin về bệnh nhân	Không (**)

(**): Phải nhập một trong hai trường dữ liệu để bắt đầu tìm kiếm

2.4 Yêu cầu phi chức năng

Ngoài sự đầy đủ về mặt chức năng, để hệ thống hoạt động một cách hiệu quả, các yêu cầu phi chức năng cũng là những yếu tố có vai trò quan trọng cần được xem xét kỹ.

2.4.1 Tính dễ dùng

Do người sử dụng hệ thống là những người dùng không chuyên và không có nhiều nền tảng về phần mềm nói riêng và công nghệ thông tin nói chung, nên hệ thống cần phải có giao diện trực quan, dễ hiểu và dễ sử dụng. Ngoài ra, cần có các mục hướng dẫn về các thao tác nghiệp vụ khi tham gia vào hệ thống dành riêng cho các nhóm người dùng khác nhau của hệ thống.

2.4.2 Tính mở rộng

Hệ thống trong tương lai có thể còn thay đổi về quy trình nghiệp vụ, quy mô dữ liệu và tính chất dữ liệu. Vậy nên, hệ thống cần được xây dựng hướng tới các tiêu chí dễ bảo trì, dễ mở rộng và nâng cấp.

2.5 Kết chương

Trong chương 2, sau khi phân tích ở mục khảo sát hiện trạng, em đã đi vào giới thiệu về hệ thống và phân tích một số chức năng chính của hệ thống qua biểu đồ use case và các đặc tả use case. Ngoài các yêu cầu về chức năng, em cũng phân tích thêm về các yêu cầu phi chức năng mà hệ thống cần đạt được. Để xây dựng được hệ thống như đã phân tích, trong chương 3, em sẽ tiến hành phân tích các công nghệ được lựa chọn để phát triển hệ thống.

Chương 3 Công nghệ sử dụng

Sau khi đã khảo sát và phân tích yêu cầu hệ thống, trong chương 3, em sẽ giới thiệu và trình bày chi tiết về các công nghệ được sử dụng để phát triển hệ thống. Hai thành phần chính của hệ thống là Frontend và Backend. Backend và Frontend giao tiếp với nhau thông qua REST API. Frontend sử dụng thư viện ReactJS, Redux, Bootstrap để thiết kế và xây dựng giao diện người dùng. Backend sử dụng NodeJS kết hợp với ExpressJS và MongoDB để xử lý logic phía server và lưu trữ dữ liệu hệ thống, ngoài ra em còn sử dụng Docker nhằm phục vụ cho việc triển khai hệ thống.

3.1 Frontend

3.1.1 ReactJS

ReactJS [3] là một trong những thư viện Javascript phổ biến nhất hiện nay dùng để xây dựng giao diện người dùng (UI). ReactJS được phát triển bởi Facebook và được áp dụng để phát triển giao diện cho nhiều phần mềm nổi tiếng hiện nay như Netflix, Facebook, Instagram, Whatsapp,… Dựa theo khảo sát State of Javascript Survey 2020 [4] với 23765 lập trình viên, ReactJS đang là thư viện được biết đến và sử dụng nhiều nhất do có tốc độ nhanh, hiệu suất cao, mượt mà, giao diện thiết kế đẹp mắt, có cộng đồng lập trình viên lớn và được hỗ trợ Facebook. Một số đặc điểm nổi bật và đặc trưng của ReactJS:

JSX (Javascript XML) – là cú pháp đặc trưng của ReactJS theo kiểu XML, cho phép lập trình viên lập trình mã HTML bên trong mã nguồn Javascript, giúp tối ưu hóa mã nguồn khi biên dịch, do vậy có tốc độ nhanh hơn so với mã nguồn Javascript tương ứng, các lỗi cũng sẽ được phát hiện ngay trong quá trình biên dịch, giúp việc gỡ lỗi dễ dàng và hiệu quả hơn.

Component-Based – React cho phép phát triển giao diện dưới dạng component – thành phần. Giao diện được chia thành nhiều phần nhỏ có quan hệ chặt chẽ với nhau, là các component, mỗi thành phần đó quản lý và cập nhật giao diện theo các trạng thái và logic riêng, nhưng vẫn đảm bảo giao diện chung hoạt động. Mỗi component có 2 khái niệm chính về state và props. Props là các thuộc tính được sử dụng để giao tiếp giữa các component, còn state là các thuộc tính của riêng một component nào đó, mỗi lần state thay đổi, component đó sẽ được cập nhật lại. Việc phát triển giao diện người dùng dựa trên component khiến mã nguồn trở nên trong sáng, dễ hiểu, giúp gỡ lỗi hiệu quả hơn, đồng thời tăng khả năng tái sử dụng mã nguồn.

Luồng dữ liệu một chiều (One-way data flow) – Luồng dữ liệu trong ReactJS là một chiều xuyên suốt từ component cha đến component con nhỏ nhất, cho phép kiểm soát xuyên suốt toàn bộ luồng dữ liệu, cho lập trình viên cái nhìn trong sáng về hệ thống cần phát triển. Tuy nhiên, việc cập nhật các state và props của các component tuân theo nguyên lý luồng dữ liệu một chiều sẽ có một chút khó khăn nếu phát triển một hệ thống lớn và phức tạp.

Virtual DOM (Virtual Document Object Model) – ReactJS duy trì View song song trên cả DOM thực và DOM ảo. Mỗi khi giao diện cập nhật hoặc thay đổi, DOM ảo sẽ thực hiện so sánh trạng thái snapshot của DOM ảo khi chưa diễn ra cập nhật, sau đó React sử dụng thuật toán Diffing để so sánh và đối chiếu xem cập nhật diễn ra ở đâu và bỏ qua những element không liên quan. Đồng thời, cập nhật những phần thay đổi đó vào DOM thực, tránh việc truy cập và duyệt DOM Tree trực tiếp và nhiều lần, giúp tăng hiệu năng của ứng dụng, đem lại trải nghiệm mượt mà cho người dùng.

3.1.2 Redux

Redux [5] – Là một thư viện Javascript giúp quản lý trạng thái của ứng dụng. Redux được xây dựng dựa trên nền tảng ngôn ngữ Elm và kiến trúc Flux do Facebook giới thiệu. Redux kết hợp tốt với React. Redux có các đặc điểm:

Predictable (Dễ đoán) – Redux giúp lập trình viên viết các ứng dụng hoạt động một cách ổn định trên nhiều môi trường khác nhau (client, server và native), đồng thời, ứng dụng cũng rất dễ để kiểm thử.

Centralized (Tập trung hóa) – Tập trung hóa các trạng thái (state) và logic của ứng dụng, khiến việc chỉnh sửa, xử lý và vận hành dễ dàng hơn.

Debuggable (Dễ gỡ lỗi) – Redux cung cấp Redux Devtools, là một extension giúp lập trình viên dễ dàng theo dõi những cập nhật và thay đổi trong trạng thái (state) của ứng dụng theo trình tự thời gian.

Flexible (Linh hoạt) – Redux có thể hoạt động tốt với Front-end framework hoặc Front-end Libary bất kỳ, trong đó có React.

Trong hệ thống của em, Redux giúp quản lý các state trong React và hạn chế tối đa những nhược điểm về Component-based và One-way data flow của React, nhằm nâng cao hiệu suất của hệ thống.

3.1.3 Bootstrap

Bootstrap [6] - là một Front-end Framework mã nguồn mở miễn phí, được phát triển bởi Mark Otto và Jacob Thornton để lập trình web front-end nhanh hơn và dễ dàng hơn. Bootstrap bao gồm một tập hợp các mẫu thiết kế HTML và CSS có sẵn, lập trình viên chỉ

cần nhúng vào mã nguồn và sử dụng chúng một cách hợp lý để có một giao diện người dùng đẹp mắt. Các ưu điểm đặc trưng của Bootstrap:

Dễ sử dụng - Chỉ cần kiến thức cơ bản về HTML và CSS, thư viện dễ cài đặt và sử dụng.

Responsive – Bootstrap giúp giao diện tương thích với mọi thiết bị như điện thoại, máy tính bảng, PC, laptop,…

Khả năng tương thích với các trình duyệt cao - Bootstrap tương thích với tất cả các trình duyệt hiện đại (Chrome, Firefox, Internet Explorer, Edge, Safari và Opera).

3.2 Backend

3.2.1 NodeJS

NodeJS [7] là nền tảng phát triển ứng dụng phía back-end được xây dựng dựa trên V8 Javascript Engine - trình thông dịch thực thi mã Javascript. NodeJS hỗ trợ chạy trên nhiều nền tảng hệ điều hành khác nhau: Linux, Windows, MacOS,… NodeJS có nhiều ưu điểm thích hợp cho việc phát triển các ứng dụng:

NodeJS sử dụng kiến trúc lập trình bất đồng bộ (non-blocking) và hướng sự kiện (event-driven) nhằm tạo ra các ứng dụng web thời gian thực nhẹ tải, hiệu suất cao đồng thời có thể chạy trên nhiều thiết bị khác nhau. NodeJS thực sự mạnh ở các ứng dụng cần tốc độ và khả năng mở rộng vì điểm mạnh của nó là khả năng xử lí một lượng lớn các connection với thông lượng cao, tương đương với khả năng mở rộng lớn. Tuy nhiên NodeJS không thích hợp với các ứng dụng yêu cầu mức độ tính toán phức tạp.

NodeJS được viết bằng Javascript nên có cộng đồng người dùng lớn mạnh với các tài liệu hướng dẫn, thư viện built-in phong phú, đa dạng.

3.2.2 ExpressJS

ExpressJS [8] là một framework được xây dựng trên nền tảng của NodeJS, nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile, tương thích và làm việc hiệu quả với NodeJS. Express hỗ trợ các phương thức HTTP và midleware tạo ra API hiệu quả và dễ sử dụng.

3.2.3 MongoDB

MongoDB [9] là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là một dạng NoSQL database. MongoDB dễ dùng và mang lại nhiều lợi ích. Một số lợi ích của MongoDB như:

Mongo DB là phần mềm mã nguồn mở miễn phí, có cộng đồng phát triển rất lớn.

Tốc độ truy vấn (find, update, insert, delete) của MongoDB nhanh hơn hệ quản trị cơ sở dữ liệu quan hệ RDBMS do: MongoDB lưu dữ liệu dạng JSON, khi insert nhiều dữ liệu sẽ insert dưới dạng mảng các JSON, gần tương tự với insert dữ liệu của một đối tượng; MongoDB không có sự ràng buộc dữ liệu giữa các bảng như RDBMS, nên tốc độ insert, update hoặc delete dữ liệu nhanh hơn do không cần kiểm tra xem có thỏa mãn ràng buộc giữa các dữ liệu; MongoDB có đánh chỉ mục (index) cho dữ liệu nên khi thực hiện truy vấn thì tìm dữ liệu rất nhanh; MongoDB thực hiện khóa các thao tác khi truy vấn, tức là khi đang trong quá trình thực hiện thao tác find, nếu có thêm thao tác update hoặc insert mà find chưa thực hiện xong thì phải đợi thao tác find thực hiện xong mới thực hiện thêm thao tác truy vấn mới được yêu cầu; Ngoài ra, MongoDB còn dễ mở rộng theo chiều ngang và có tính sẵn sàng cao, còn được sử dụng trên cloud.

Tuy nhiên khi sử dụng cần cân nhắc vì:

MongoDB không có các tính chất ràng buộc như trong RDBMS nên dễ bị làm sai dữ liệu.

Dữ liệu thuộc cơ sở dữ liệu MongoDB có thể sử dụng nhiều bộ nhớ hơn do dữ liệu trong các document được lưu trữ dưới dạng key-value và các key có khả năng bị lặp lại. Không hỗ trợ join nên sẽ bị dư thừa dữ liệu (trong RDBMS thì ta chỉ cần lưu 1 bản ghi rồi các bản ghi khác tham chiếu tới còn trong MongoDB thì không).

Không hỗ trợ join giống như RDBMS nên khi viết function join trong code ta phải làm bằng tay khiến cho tốc độ truy vấn bị giảm.

3.2.4 REST API

API là một trong những phần bắt buộc khi xây dựng và phát triển các ứng dụng web. Có rất nhiều tiêu chuẩn được dùng để thiết kế và xây dựng các API, nhưng một trong những tiêu chuẩn phổ biến và được tin dùng nhất hiện nay là RESTful API (Representational State Tranfer Application Programming Interface). Rest API thường chú trọng vào các tài nguyên của hệ thống bao gồm: ảnh, văn bản, âm thanh, video hoặc các dữ liệu di động,… Nó thường bao gồm các trạng thái tài nguyên đã được định dạng sẵn và được truyền tải thông qua HTTP. Chức năng quan trọng nhất của REST là: quy định các cách sử dụng HTTP method chẳng hạn như: POST, GET, PUT, DELETE,… và cách có thể định dạng các URL cho ứng dụng web để có thể quản lý được các resource. RESTful có thể sử dụng JSON, XML, plain text, HTML,.. làm cấu trúc dữ liệu trả về, có quy ước rõ ràng về cách viết URL và phương thức HTTP. RESTful không quy định logic code và không bị giới hạn bởi bất kỳ ngôn ngữ lập trình nào, có nghĩa là ta có thể sử dụng một ngôn ngữ, framework hoặc thư viện bất kỳ để thiết kế RESTful API.

3.3 Triển khai hệ thống

3.3.1 Docker

Docker [10] là một nền tảng cho phép lập trình viên lập trình, triển khai và chạy các ứng dụng của mình với container (các container ảo tạo ra các môi trường độc lập và tách biệt để triển khai và chạy ứng dụng). Khi cần deploy lên một server bất kỳ, chỉ cần run container thì ứng dụng sẽ chạy ngay lập tức. Docker giải quyết vấn đề về việc setup và deploy ứng dụng lên một hoặc nhiều server, giảm thiểu việc phải cài đặt các công cụ, môi trường đồng bộ để chạy ứng dụng. Đồng thời, các container của Docker cũng tận dụng được tài nguyên sẵn có của hệ thống chứ không tiêu tốn nhiều tài nguyên như việc khởi chạy các máy ảo vật lý, do đó việc chạy và sử dụng các container nhanh gọn và linh hoạt hơn các máy ảo rất nhiều. Do đó để triển khai hệ thống một cách nhanh chóng, em đã sử dụng Docker.

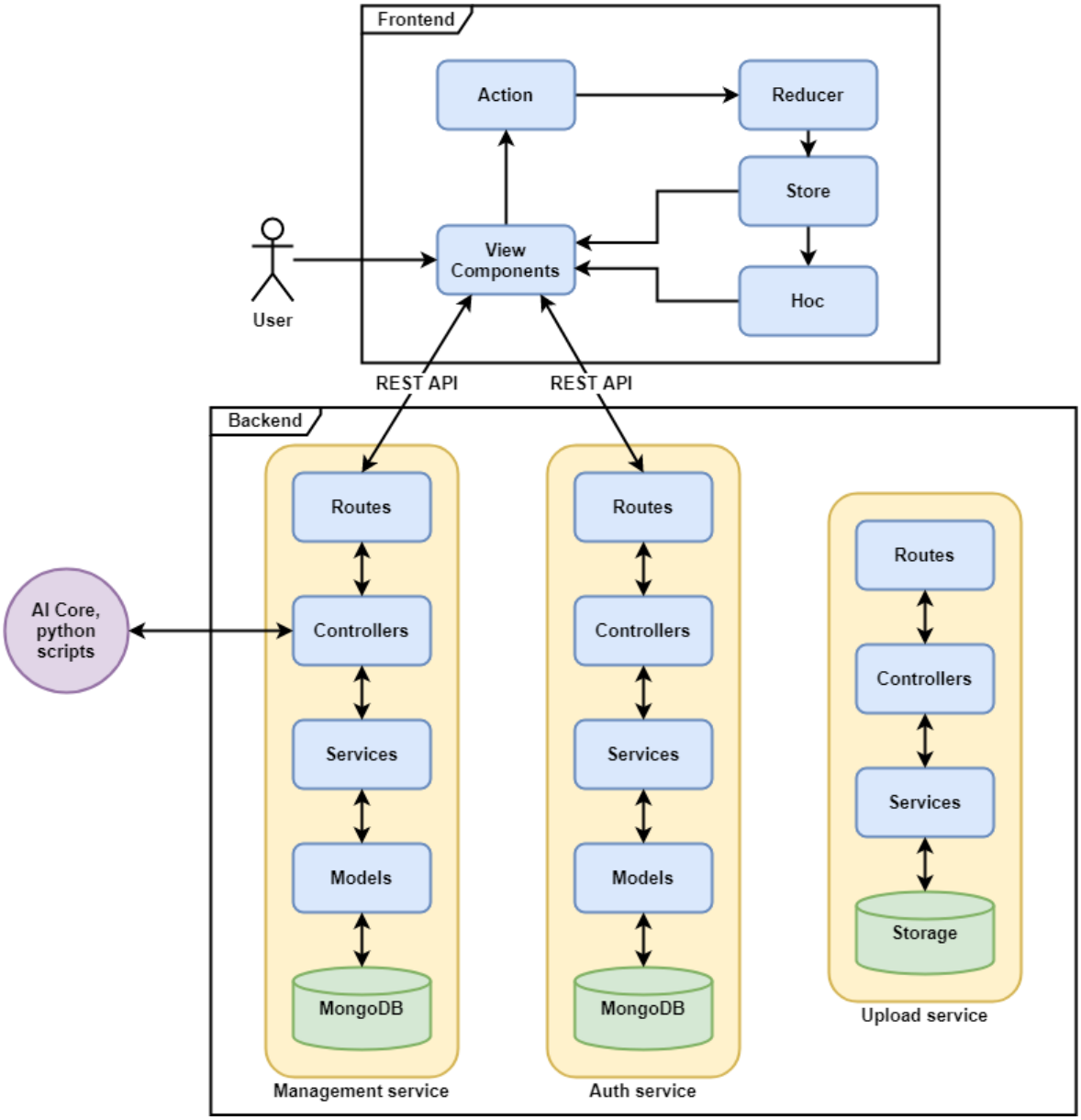
3.4 Kết chương

Em đã giới thiệu tổng quan và phân tích về những công nghệ mà em lựa chọn để xây dựng và phát triển hệ thống của mình trong chương 3. Để có cái nhìn chi tiết hơn về hệ thống, em sẽ đi sâu phân tích ở chương 4. Đồng thời, em cũng sẽ trình bày rõ cách áp dụng những công nghệ trên vào hệ thống của mình.

Chương 4 Phát triển và triển khai ứng dụng

Trong chương 4, em sẽ trình bày chi tiết về thiết kế kiến trúc hệ thống, thiết kế giao diện và cơ sở dữ liệu. Đồng thời, chương 4 cũng mô tả chi tiết về quá trình xây dựng, kiểm thử và triển khai hệ thống.

4.1 Thiết kế kiến trúc



Hình 8 Kiến trúc tổng quan của hệ thống

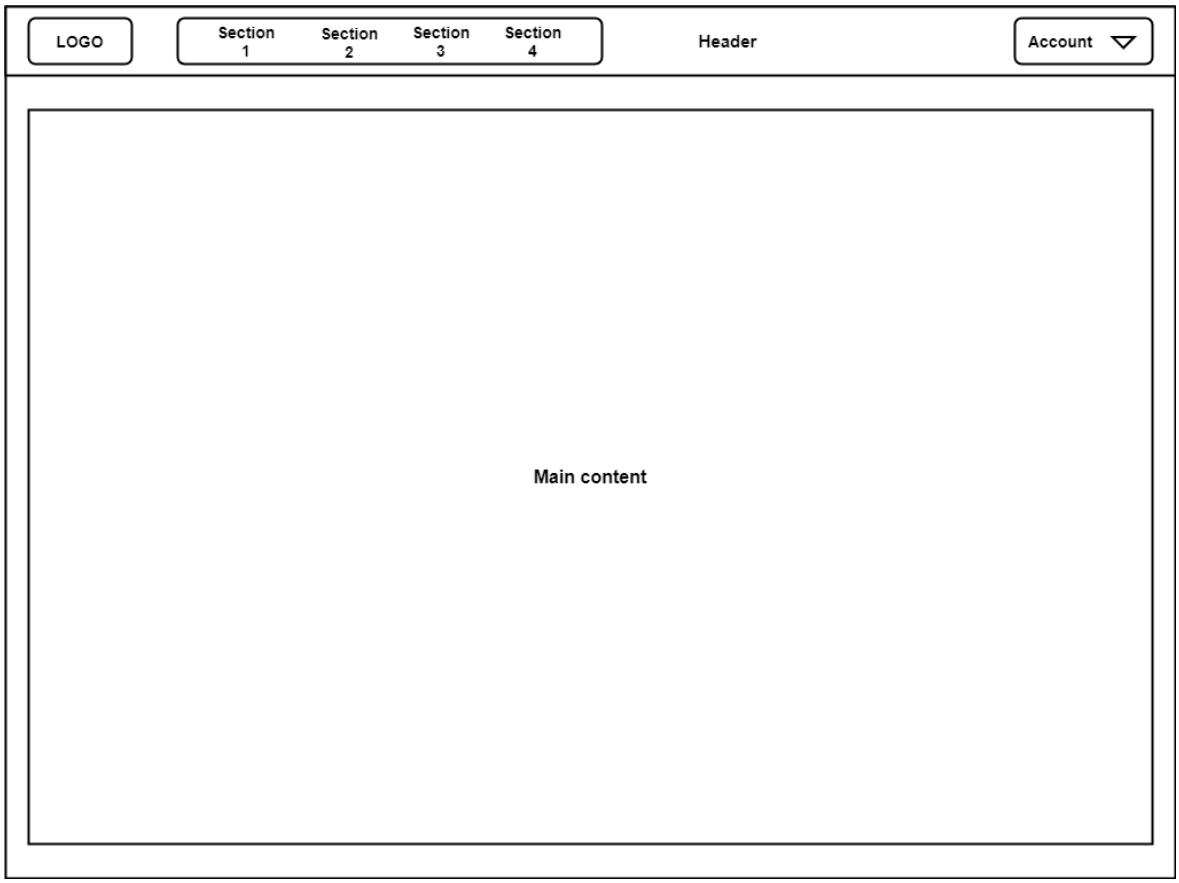
Hình 8 mô tả kiến trúc tổng quan của hệ thống. Như đã phân tích và giới thiệu ở các chương trước, hệ thống được xây dựng trong khuôn khổ đồ án gồm hai thành phần chính là Frontend và Backend. Hai thành phần giao tiếp với nhau thông qua REST API, cụ thể là thông qua giao thức HTTP.

Frontend xây dựng giao diện bằng các thành phần giao diện nhỏ hơn dựa trên thư viện ReactJS. Đồng thời dùng Redux để quản lý các trạng thái của hệ thống. Thiết kế kiến trúc các thành phần giao diện sẽ được trình bày chi tiết ở mục **5.2**.

Backend được thiết kế theo mô hình kiến trúc Microservices, bao gồm hai dịch vụ chính là dịch vụ quản lý (Managament service), dịch vụ xác thực phân quyền người dùng (Auth service) và dịch vụ lưu trữ (Upload service). Chi tiết về kiến trúc Microservices và các dịch vụ trong hệ thống sẽ được trình bày chi tiết ở mục **5.1.2**. AI Core là một dịch vụ bên ngoài được đóng gói để xử lý riêng, không nằm trong phạm vi của đồ án này. Ngoài ra còn có một số python scripts được chạy dưới nền NodeJS ở Backend nhằm xử lý các kết quả hình ảnh.

4.2 Thiết kế chi tiết Frontend

4.2.1 Thiết kế mockup



Hình 9 Thiết kế mockup của hệ thống

Giao diện hệ thống được xây dựng dựa trên thư viện ReactJS nên được chia thành các thành phần giao diện nhỏ hơn là các Component. Các giao diện website hiện nay có khá nhiều lựa chọn về độ phân giải, nhưng độ phân giải phổ biến phù hợp với màn hình máy tính ở hầu hết các cơ quan làm việc là độ phân giải 1920x1080. Do đó, em cũng lựa chọn độ phân giải này cho giao diện của hệ thống. **Hình 9** mô tả bố cục trang web của hệ thống, được chia làm 2 phần chính là Header và Main content. Tất cả các màn hình trong hệ thống đều được xây dựng dựa trên bố cục này để đảm bảo sự thống nhất về giao diện. Phần Header là phần cố định luôn có trong tất cả các màn hình, menu của phần Header bao gồm các Section tùy chỉnh để người sử dụng có thể dễ dàng điều hướng đến màn hình mong muốn. Phần Main content thay đổi tùy biến để hiển thị nội dung tương ứng của màn hình.

Các thành phần thuộc tính như màu sắc, font chữ, thuộc tính của các nút hay cách hiển thị thông báo của hệ thống được trình bày ở **Bảng 8** để đảm bảo tính nhất quán, trực quan và dễ nhìn.

Bảng 8 Cấu hình các thành phần thuộc tính của giao diện

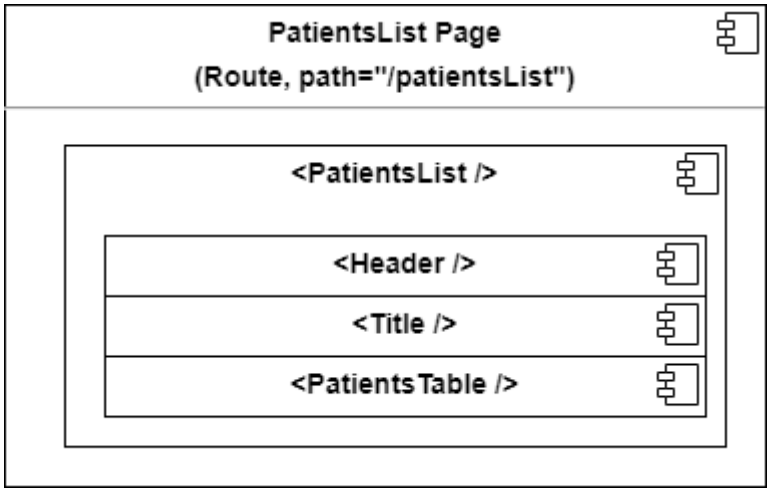
Thuộc tính	Cấu hình
Màu sắc	Màu chính: #ffffff, #24292e, #007bff Một số màu khác: #337ab7, #ff0000, #dddddd
Font chữ	Kiểu chữ: -apple-system, BlinkMacSystemFont, Segoe UI Cỡ chữ phần nội dung: 14px
Bo góc	10px
Vị trí hiển thị thông báo	Chính giữa màn hình

Từ màn hình chính của hệ thống, người dùng có thể điều hướng sang các màn hình khác tùy vào vai trò được phân quyền trong hệ thống. Các màn hình chính của hệ thống bao gồm: Màn hình nhập thông tin bệnh nhân/ người dùng, Màn hình quản lý danh sách bệnh nhân/ người dùng, Màn hình hiển thị danh sách bệnh nhân tiếp nhận, Màn hình kết xuất báo cáo, Màn hình công việc chuyên môn tùy vào vai trò trong hệ thống. Các màn hình vừa liệt kê ở trên là các màn hình thường xuyên được sử dụng và là các màn hình hiển thị nhiều chức năng chính của hệ thống. Để có hình dung rõ hơn về các màn hình này, chi tiết sẽ được đề cập ngay trong mục Thiết kế thành phần giao diện dưới đây.

4.2.2 Thiết kế thành phần giao diện

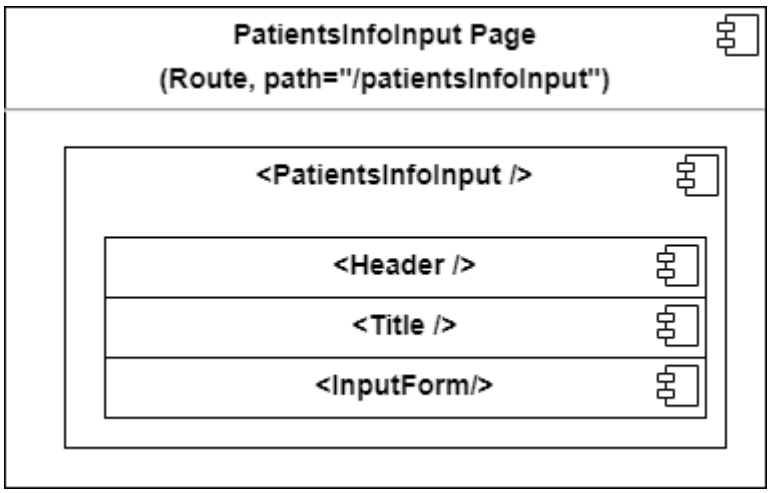
Trong phần này, do khuôn khổ đồ án có hạn, em xin trình bày chi tiết thiết kế thành phần giao diện của màn hình chính, bao gồm: Danh sách bệnh nhân tiếp nhận, Nhập thông tin bệnh nhân và Kết xuất báo cáo chẩn đoán hình ảnh.

Hình 10 mô tả màn hình Danh sách tiếp nhận bệnh nhân, gồm các thành phần chính là Header, Title và PatientsTable. Trong đó, Header là phần chứa Logo và các mục chuyển hướng tới các trang khác. Title là tiêu đề của trang. Thành phần PatientsTable là thành phần chính hiển thị danh sách bệnh nhân tiếp nhận dưới dạng bảng.



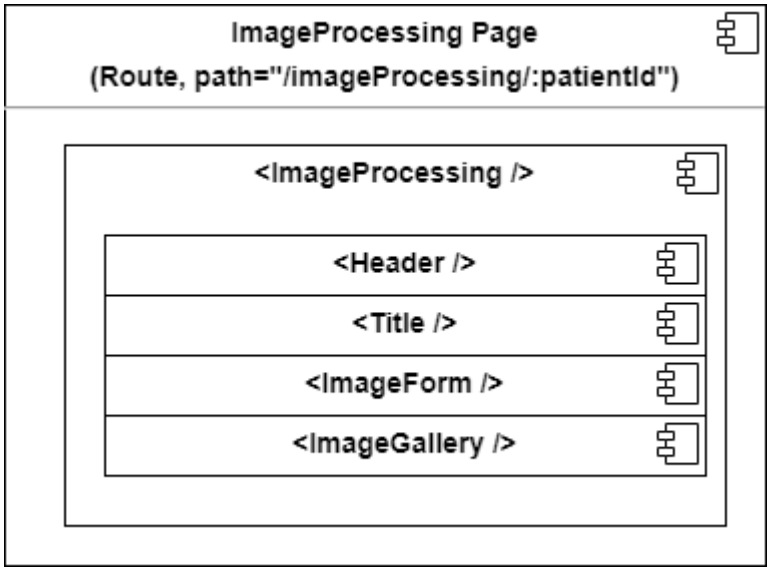
Hình 10 Thiết kế thành phần giao diện màn hình Danh sách bệnh nhân tiếp nhận

Màn hình Nhập thông tin bệnh nhân gồm 3 thành phần chính được mô tả ở **Hình 11** bên dưới. Các thành phần chính là Header, Title và InputForm. Trong đó, Header là phần chứa Logo và các mục chuyển hướng tới các trang khác. Title là tiêu đề của trang. Thành phần InputForm là thành phần chính hiển thị các trường dữ liệu dưới dạng một form để nhân viên bệnh viện thực hiện nhập thông tin bệnh nhân.



Hình 11 Thiết kế thành phần giao diện màn hình Nhập thông tin bệnh nhân

Hình 12 trình bày màn hình Kết xuất báo cáo chẩn đoán hình ảnh. Màn hình này có 4 thành phần chính là Header, Title, ImageForm và ImageGallery. Trong đó, Header là phần chứa Logo và các mục chuyển hướng tới các trang khác. Title là tiêu đề của trang. ImageForm là thành phần chính hiển thị báo cáo chẩn đoán hình ảnh và là phần được chọn để kết xuất báo cáo dưới dạng pdf. Ngoài ra còn có thành phần ImageGallery dùng để hiển thị các ảnh kết quả, cho phép người dùng có thể phóng to hoặc thu nhỏ hình ảnh tùy ý để tiện theo dõi.



Hình 12 Thiết kế thành phần giao diện màn hình Kết xuất báo cáo chẩn đoán hình ảnh

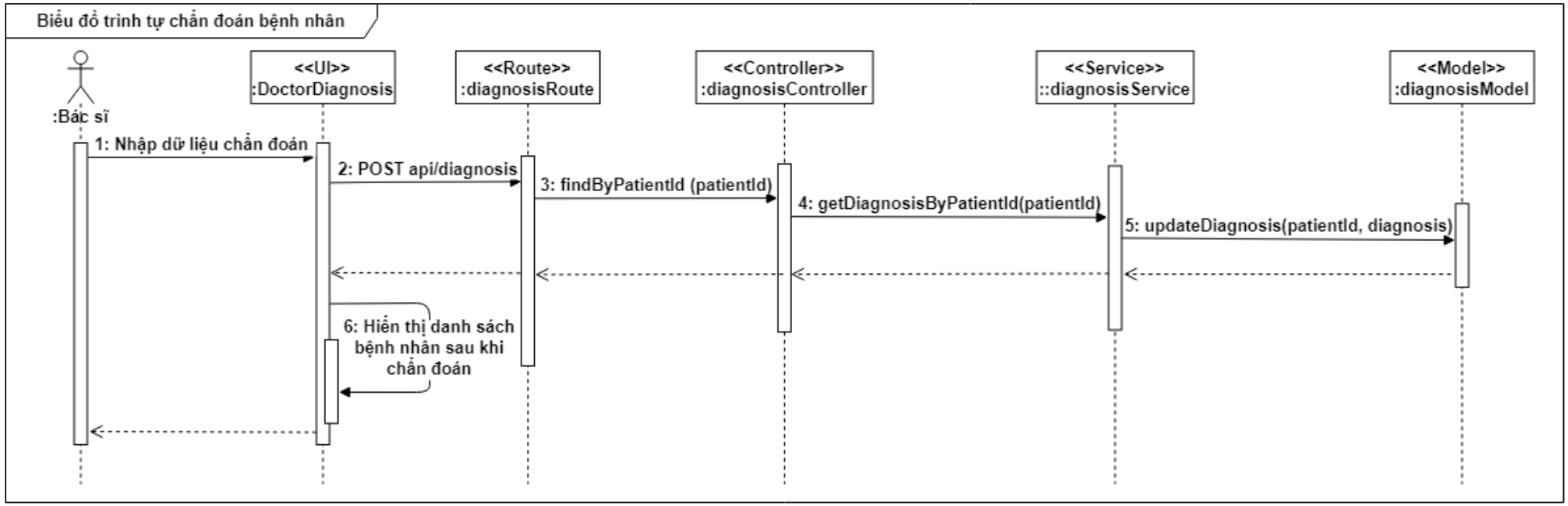
4.3 Thiết kế chi tiết Backend

4.3.1 Luồng hoạt động

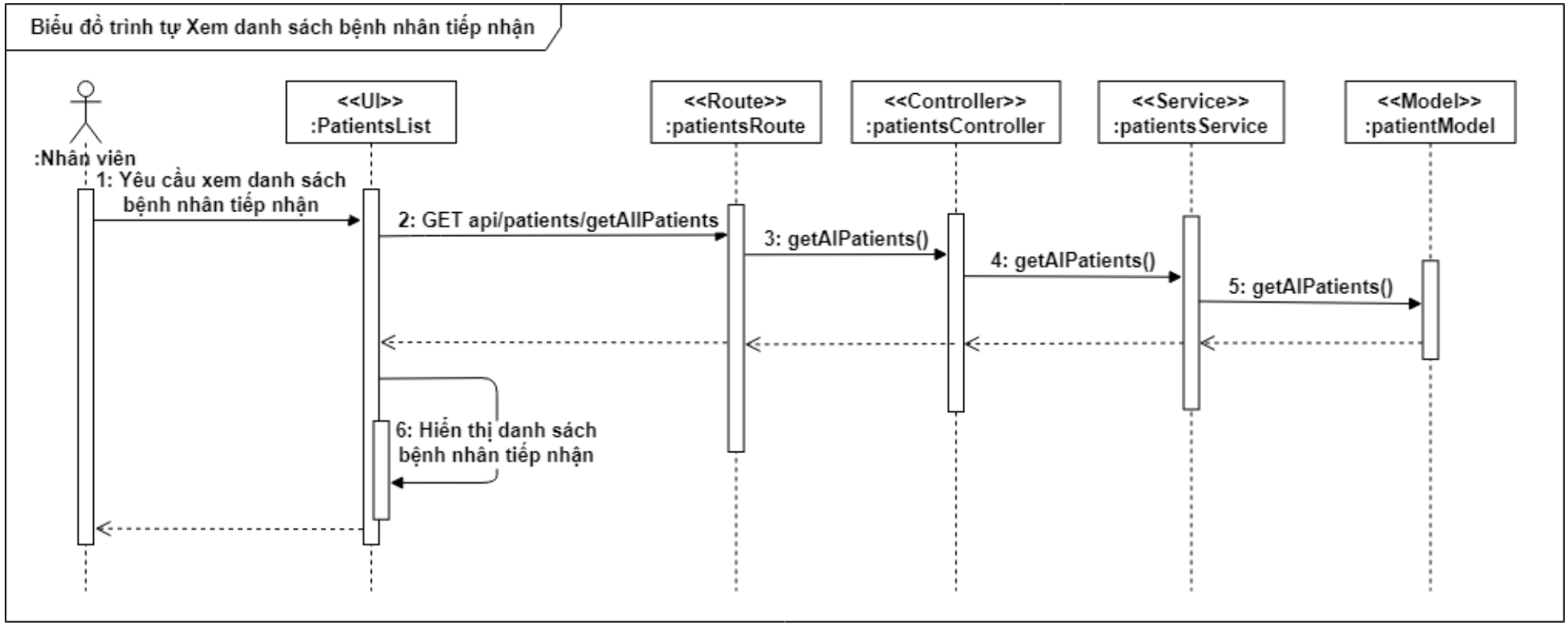
Trong phần này, em sẽ trình bày biểu đồ trình tự mô tả các chức năng của hệ thống. Tuy nhiên, do khuôn khổ báo cáo đồ án có hạn, nên em sẽ trình bày biểu đồ trình tự của hai chức năng chính là Chẩn đoán bệnh nhân và Xem danh sách bệnh nhân tiếp nhận.

4.3.1.1 Luồng hoạt động Chẩn đoán bệnh nhân

Hình 13 mô tả trình tự xử lý của chức năng Chẩn đoán bệnh nhân. Đầu tiên, tác nhân là Bác sĩ thao tác trên giao diện và nhập các trường dữ liệu cần thiết của phiếu chẩn đoán. Trong quá trình thao tác, các thành phần UI sẽ tự động kiểm tra dữ liệu và hiển thị thông báo lỗi tương ứng nếu có. Sau khi người dùng gửi yêu cầu, các thông tin đã nhập sẽ được gửi về phía server thông qua giao thức HTTP. Các Router sẽ định tuyến và đến Controller tương ứng, sau đó, Controller sẽ gọi đến Service tương ứng và tương tác với Model để cập nhật dữ liệu nhận được. Sau khi server xử lý xong, response được gửi trả về tương ứng phía giao diện, giao diện hiển thị danh sách bệnh nhân với thông tin chẩn đoán đã được cập nhật.



Hình 13 Biểu đồ trình tự Chẩn đoán bệnh nhân



Hình 14 Biểu đồ trình tự Xem danh sách tiếp nhận bệnh nhân

4.3.1.2 Luồng hoạt động Xem danh sách bệnh nhân tiếp nhận

Hình 14 mô tả trình tự xử lý của chức năng Xem danh sách bệnh nhân tiếp nhận. Đầu tiên, Nhân viên thao tác trên giao diện với yêu cầu xem danh sách bệnh nhân tiếp nhận. Sau khi người dùng gửi yêu cầu, phương thức HTTP được gọi để kết nối đến yêu cầu đến phía server. Các Router sẽ định tuyến và đến Controller tương ứng, sau đó, Controller sẽ gọi đến Service tương ứng và tương tác với Model để cập nhật dữ liệu nhận được. Sau khi server xử lý xong, response được gửi trả về tương ứng phía giao diện để giao diện hiển thị danh sách bệnh nhân tiếp nhận.

4.3.2 Thiết kế API

Các thành phần chính trong hệ thống tương tác với nhau thông qua API, vì vậy, trong mục này, em sẽ mô tả các API được sử dụng trong hệ thống.

Bảng 9 cung cấp danh sách 36 API chính được sử dụng trong hệ thống, được phân loại theo 4 trường là: Nhóm API, Mục đích, Phương thức, Địa chỉ. Nhóm API nêu mục đích chung của một nhóm các API hướng đến một nhóm người dùng với một vai trò cụ thể hoặc một nhóm các kết quả, bao gồm cả baseURL của nhóm; Mục đích nêu rõ mục đích sử dụng của một API; Phương thức mô tả giao thức HTTP với các phương thức GET, POST, PUT, DELETE dùng để giao tiếp và thao tác với dữ liệu; Địa chỉ nêu rõ địa chỉ cụ thể của từng API.

Bảng 9 Danh sách các API của hệ thống

STT	Nhóm API	Mục đích	Phương thức	Địa chỉ
1	Quản lý người dùng: /api/users	Xác thực người dùng	GET	/auth
2		Tạo mới tài khoản người dùng	POST	/register
3		Đăng nhập	POST	/login
4		Đăng xuất	GET	/logout
5		Cập nhật thông người dùng	PUT	/updateInformation

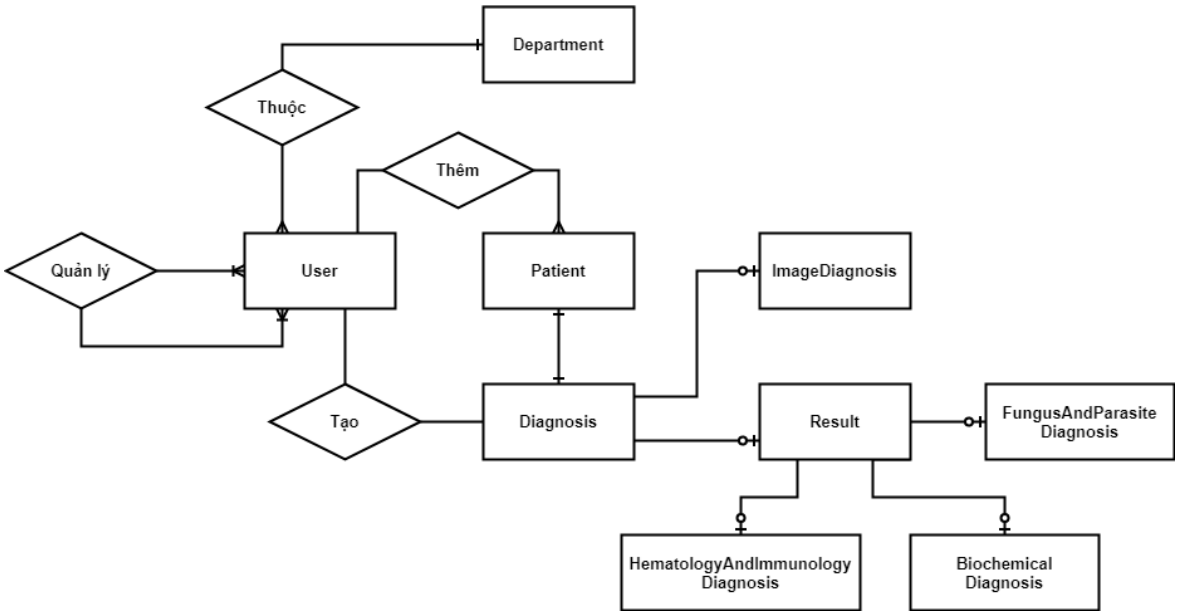
STT	Nhóm API	Mục đích	Phương thức	Địa chỉ
6		Lấy toàn bộ danh sách người dùng	GET	/getAllUsers
7		Xóa người dùng khỏi hệ thống	DELETE	/deleteUser
8	Quản lý bệnh nhân: /api/patients	Thêm mới thông tin bệnh nhân	POST	/uploadInfo
9		Cập nhật thông tin bệnh nhân	PUT	/updateInfo
10		Lấy thông tin bệnh nhân bằng mã bệnh nhân	GET	/getPatientById
11		Lấy danh sách bệnh nhân	GET	/getAllPatients
12		Lấy danh sách bệnh nhân theo phòng khám	GET	/getPatientsBy-Department
13	Quản lý chẩn đoán: /api/diagnosis/	Cập nhật chẩn đoán hình ảnh	PUT	/updateImaging-Diagnosis
14		Cập nhật chẩn đoán đầy đủ	PUT	/updateDiagnosis
15		Lấy chẩn đoán của bệnh nhân dựa trên mã bệnh nhân	GET	/getDiagnosisById
16		Lưu chẩn đoán của bệnh nhân	POST	/putDiagnosis
17		Cập nhật chẩn kết quả xét nghiệm nấm, kí sinh trùng	PUT	/updateFungus-Diagnosis

STT	Nhóm API	Mục đích	Phương thức	Địa chỉ
18		Cập nhật chẩn kết quả xét nghiệm huyết học, miễn dịch	PUT	/updateHematology-Diagnosis
19		Cập nhật chẩn kết quả xét nghiệm tổng quát	PUT	/updateResult-Diagnosis
20		Cập nhật kết quả xét nghiệm sinh hóa máu	PUT	/updateBiochemical-Diagnosis
21		Lấy kết quả chẩn đoán dựa trên chẩn đoán của bác sĩ	GET	/getDiagnosis
22	Nhóm chẩn đoán hình ảnh: /api/diagnosis/imagingDiagnosis	Tải ảnh lên storage của hệ thống	POST	/uploadImage
23		Lưu ảnh tương ứng với thông tin bệnh nhân	POST	/saveImage
24		Lấy kết quả chẩn đoán hình ảnh dựa trên mã bệnh nhân	GET	/getImagingDiagnosis-ById
25	Nhóm phiếu xét nghiệm nấm – kí sinh trùng: /api/diagnosis/fungusAnd-ParasiteDiagnosis	Lưu phiếu xét nghiệm nấm – kí sinh trùng	POST	/saveFungusAnd-ParasiteForm
26		Lấy kết quả phiếu xét nghiệm dựa trên mã bệnh nhân	GET	/getFungusDiagnosis-ById
27		Cập nhật phiếu xét nghiệm nấm – kí sinh trùng	PUT	/updateFungus-DiagnosisById

STT	Nhóm API	Mục đích	Phương thức	Địa chỉ
28	Nhóm phiếu xét nghiệm huyết học – miễn dịch: /api/diagnosis/hematologyDiagnosis	Lưu phiếu xét nghiệm huyết học – miễn dịch	POST	/saveHematologyForm
29		Lấy kết quả phiếu xét nghiệm dựa trên mã bệnh nhân	GET	/getHematology-DiagnosisById
30		Cập nhật phiếu xét nghiệm huyết học – miễn dịch	PUT	/updateHematology-DiagnosisById
31	Nhóm phiếu xét nghiệm sinh hóa máu: /api/diagnosis/biochemicalDiagnosis	Lưu phiếu xét nghiệm sinh hóa máu	POST	/saveBiochemicalForm
32		Lấy kết quả phiếu xét nghiệm dựa trên mã bệnh nhân	GET	/getBiochemical-DiagnosisById
33		Cập nhật phiếu xét nghiệm sinh hóa máu	PUT	/updateBiochemical-DiagnosisById
34	Nhóm phiếu xét nghiệm tổng quan: /api/diagnosis/result	Lưu phiếu xét nghiệm tổng quan	POST	/saveResultForm
35		Lấy kết quả phiếu xét nghiệm dựa trên mã bệnh nhân	GET	/getResultDiagnosis-ById
36		Cập nhật phiếu xét nghiệm tổng quan	PUT	/updateResultDiagnosisById

4.4 Thiết kế cơ sở dữ liệu

4.4.1 Biểu đồ thực thể liên kết

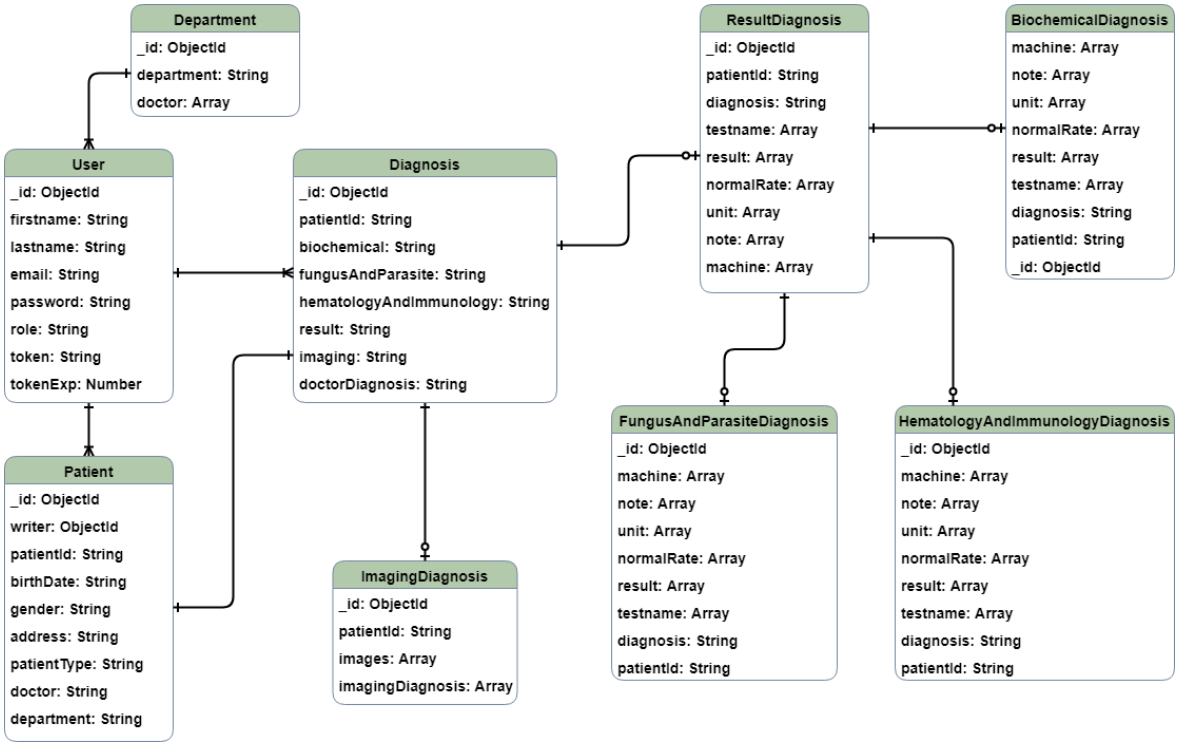


Hình 15 Biểu đồ thực thể liên kết

Hình 15 hiện biểu đồ thực thể liên kết của hệ thống. Các thực thể bao gồm User, Patient, Diagnosis, Department, ImageDiagnosis, Result, HematologyAndImmunologyDiagnosis, FungusAndParasiteDiagnosis và BiochemicalDiagnosis. Cụ thể, thực thể User chỉ người dùng hệ thống với các vai trò cụ thể như admin, nhân viên, bác sĩ, nhân viên chẩn đoán hình ảnh, nhân viên xét nghiệm. Admin có thể tạo mới User, nhân viên có thể thêm mới bệnh nhân, bác sĩ thuộc một phòng khám cố định và có thể chẩn đoán cho bệnh nhân. Dựa vào chẩn đoán đó bệnh nhân có thể phải thực hiện chẩn đoán hình ảnh hoặc xét nghiệm. Thực thể Result chỉ phiếu xét nghiệm tổng quan, có thể bao gồm ít nhất một trong ba loại phiếu xét nghiệm thành phần là phiếu xét nghiệm nấm – kí sinh trùng, phiếu xét nghiệm sinh hóa máu và phiếu xét nghiệm huyết học – miễn dịch.

4.4.2 Thiết kế tổng quan cơ sở dữ liệu

Dựa vào biểu đồ thực thể liên kết trình bày ở trên, ở phần này, em mô tả tổng quan thiết kế cơ sở dữ liệu ở **Hình 16**.



Hình 16 Thiết kế tổng quan cơ sở dữ liệu

4.4.3 Thiết kế chi tiết cơ sở dữ liệu

Trong phần này, em xin trình bày chi tiết về các trường dữ liệu của các collection có trong cơ sở dữ liệu. Do giới hạn độ dài đồ án tốt nghiệp, em sẽ trình bày chi tiết một số collection chính liên quan đến trực tiếp đến các chức năng chính của hệ thống, bao gồm: User, Patient. Department, Diagnosis, ImagingDiagnosis. Chi tiết về các collection được trình bày ở **Bảng 10** bên dưới.

Bảng 10 Thiết kế chi tiết cơ sở dữ liệu của hệ thống

Collection	Tên trường	Kiểu dữ liệu	Ý nghĩa
User	_id	ObjectId	ID được tự động generate bởi hệ thống
	firstname	String	Họ của người dùng
	lastname	String	Tên của người dùng

Collection	Tên trường	Kiểu dữ liệu	Ý nghĩa
	email	String	Email đăng nhập của người dùng
	password	String	Mật khẩu đăng nhập của người dùng, được mã hóa với bcrypt
	role	String	Vai trò của người dùng trong hệ thống
	token	String	Token được tạo ra lúc người dùng đăng nhập hệ thống
	tokenExp	Number	Thời gian tài khoản có thể hoạt động được trước khi bị đăng xuất khỏi hệ thống, tính bằng giây
Patient	_id	ObjectId	ID được tự động generate bởi hệ thống
	writer	ObjectId	Tham chiếu đến thông tin người dùng (nhân viên bệnh viện) nhập thông tin bệnh nhân
	patientId	String	Mã bệnh nhân
	name	String	Tên bệnh nhân
	birthdate	String	Ngày sinh
	gender	String	Giới tính
	address	String	Địa chỉ

Collection	Tên trường	Kiểu dữ liệu	Ý nghĩa
	patientType	String	Loại bệnh nhân: Khám trong giờ hoặc Khám ngoài giờ
	doctor	String	Bác sĩ điều trị
	department	String	Khoa phòng của bác sĩ điều trị
Department	_id	ObjectId	ID được tự động generate bởi hệ thống
	department	String	Phòng khám
	doctor	Array	Danh sách các bác sĩ trực thuộc phòng khám
Diagnosis	_id	ObjectId	ID được tự động generate bởi hệ thống
	patientId	String	Mã bệnh nhân
	biochemical	String	Trạng thái xét nghiệm sinh hóa máu
	fungusAndParasite	String	Trạng thái xét nghiệm nấm – ký sinh trùng
	hematologyAnd-Immunology	String	Trạng thái xét nghiệm huyết học – miễn dịch
	result	String	Trạng thái xét nghiệm tổng quan
	imaging	String	Trạng thái chụp chẩn đoán hình ảnh

Collection	Tên trường	Kiểu dữ liệu	Ý nghĩa
	doctorDiagnosis	String	Chẩn đoán của bác sĩ
ImagingDiagnosis	_id	ObjectId	ID được tự động generate bởi hệ thống
	patientId	String	Mã bệnh nhân
	images	Array	URL ảnh chụp chẩn đoán của bệnh nhân
	imagingDiagnosis	Array	Kết quả chẩn đoán ung thư hắc tố da từng ảnh của bệnh nhân

4.5 Xây dựng ứng dụng

4.5.1 Thư viện và công cụ sử dụng

Trong quá trình xây dựng và phát triển hệ thống, em có sử dụng một số thư viện và công cụ hỗ trợ. Các thư viện và công cụ hỗ trợ được liệt kê chi tiết ở **Bảng 11**.

Bảng 11 Danh sách thư viện và công cụ sử dụng

	Mục đích	**Công cụ**	**Địa chỉ URL**
Công cụ hỗ trợ và thư viện chung	IDE lập trình	Visual Studio Code	https://code.visualstudio.com
	Ngôn ngữ lập trình	Javascript	https://developer.mozilla.org/-enUS/docs/Web/JavaScript
	Triển khai hệ thống	Docker	https://www.docker.com
	Thư viện thao tác với HTTP Request	Axios	https://github.com/axios/axios
	Thao tác với	MongoDBCloud	https://cloud.mongodb.com/

	Mục đích	Công cụ	Địa chỉ URL
Công cụ hỗ trợ và thư viện phía Backend	CSDL MongoDB		
	Nền tảng xây dựng backend	NodeJS	https://nodejs.org
	Framework backend	Express	https://expressjs.com
	Thư viện hỗ trợ mô hình hóa thực thể khi thao tác với MongoDB	Moongoose	https://mongoosejs.com
Công cụ hỗ trợ và thư viện phía Frontend	Thư viện xây dựng Frontend	ReactJS	https://reactjs.org/
	Thư viện quản lý trạng thái ứng dụng	ReduxJS	https://redux.js.org/
	Framework frontend	Bootstrap	https://getbootstrap.com/

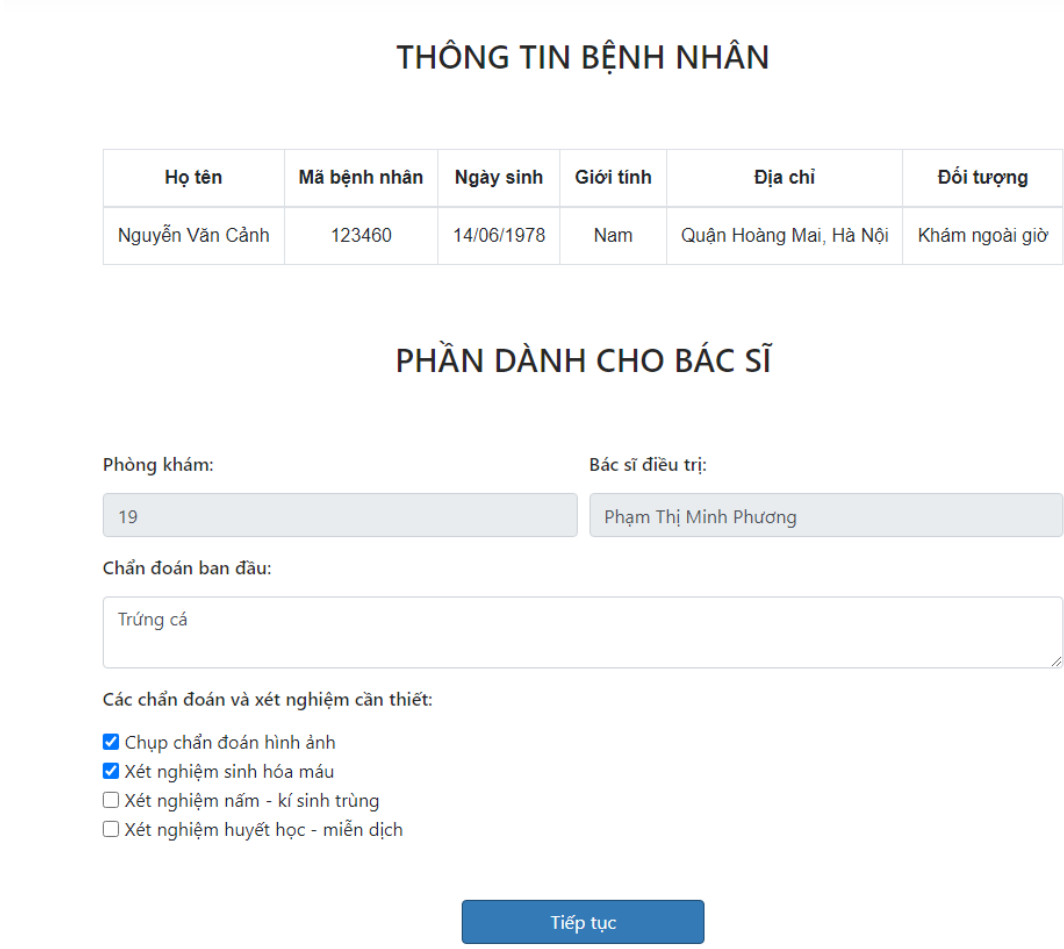
4.5.2 Kết quả đạt được

Qua quá trình xây dựng và triển khai hệ thống, em đã cố gắng xây dựng hoàn chỉnh các chức năng chính của hệ thống và hoàn thiện giao diện cơ bản ở mức dễ nhìn và trực quan. Đồng thời hệ thống đã được triển khai lên server, các thông số chi tiết về server được trình bày chi tiết ở mục **Triển khai hệ thống**.

4.5.3 Minh hoạ các chức năng chính

Sau khi trình bày về thiết kế kiến trúc tổng quan, thiết kế về cơ sở dữ liệu và công cụ sử dụng để xây dựng và phát triển hệ thống, trong phần này, em sẽ minh họa một số chức năng chính của hệ thống với thiết kế giao diện và giải thích đầy đủ. Cụ thể, các chức năng sẽ được minh họa bao gồm: Chẩn đoán bệnh nhân (1), Chẩn đoán hình ảnh (2), Xem kết quả báo cáo chẩn đoán hình ảnh (3), Điền phiếu xét nghiệm sinh hóa máu (4) và Xem kết quả phiếu xét nghiệm tổng quan (5).

4.5.3.1 Minh họa chức năng Chẩn đoán bệnh nhân



Hình 17 Minh họa giao diện chẩn đoán bệnh nhân

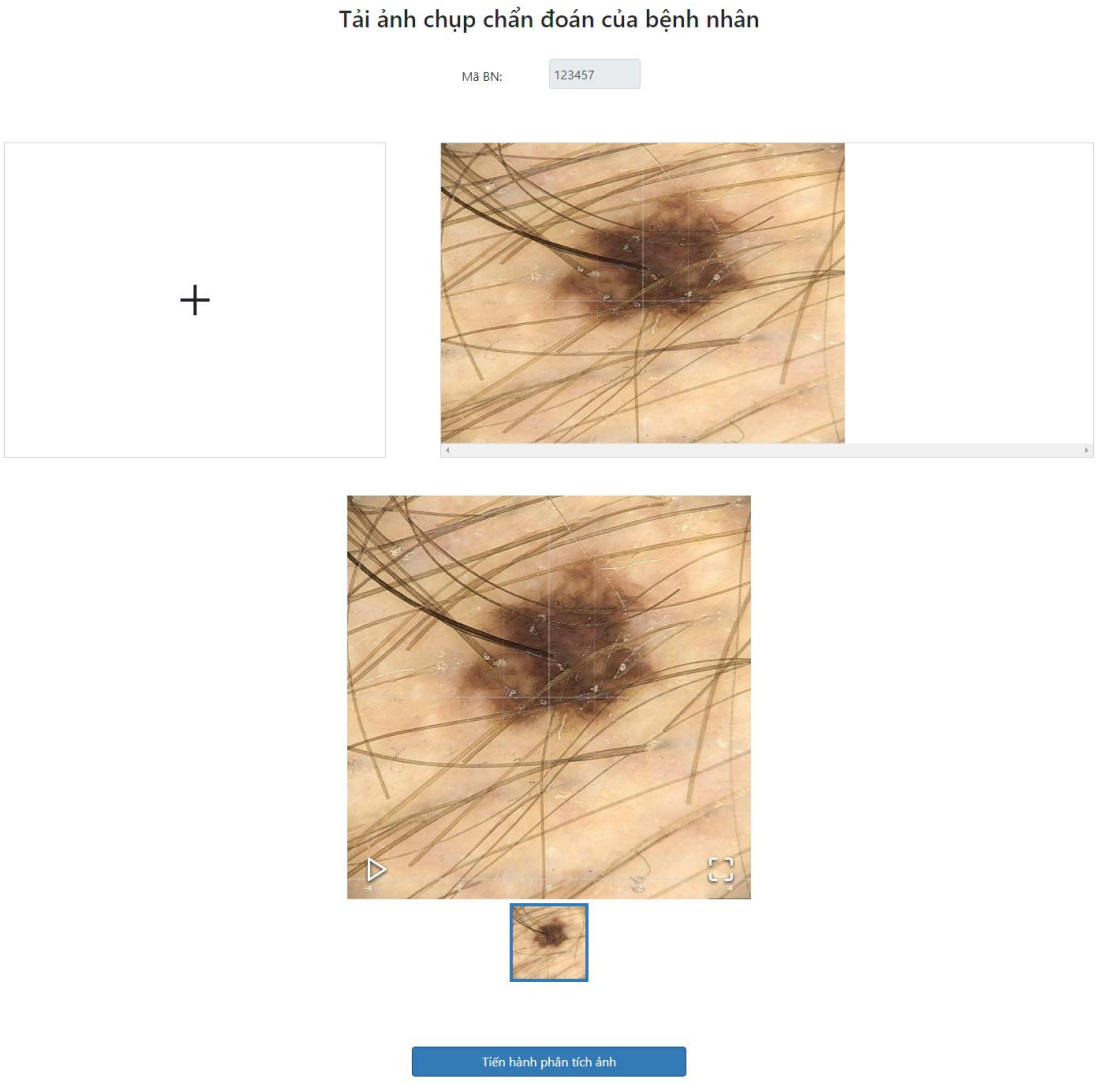
Dựa trên danh sách bệnh nhân tiếp nhận như ở **Hình 18**, bác sĩ tiến hành lựa chọn bệnh nhân cần chẩn đoán hình ảnh để tiến hành chẩn đoán bệnh cho bệnh nhân. Sau khi lựa chọn bệnh nhân cần chẩn đoán, bác sĩ sẽ được chuyển tiếp đến giao diện ở **Hình 17**. Giao diện cung cấp thông tin cơ bản của bệnh nhân cho bác sĩ, đồng thời, bác sĩ cần tiến hành điền đủ các trường thông tin cần thiết trong phiếu chẩn đoán. Sau khi hoàn thành, ấn nút Tiếp tục, hệ thống sẽ chuyển về màn hình giao diện Danh sách bệnh nhân tiếp nhận với các trường thông tin được cập nhật đầy đủ.

Hình 18 Minh họa giao diện danh sách bệnh nhân tiếp nhận

4.5.3.2 Minh họa chức năng Chẩn đoán hình ảnh

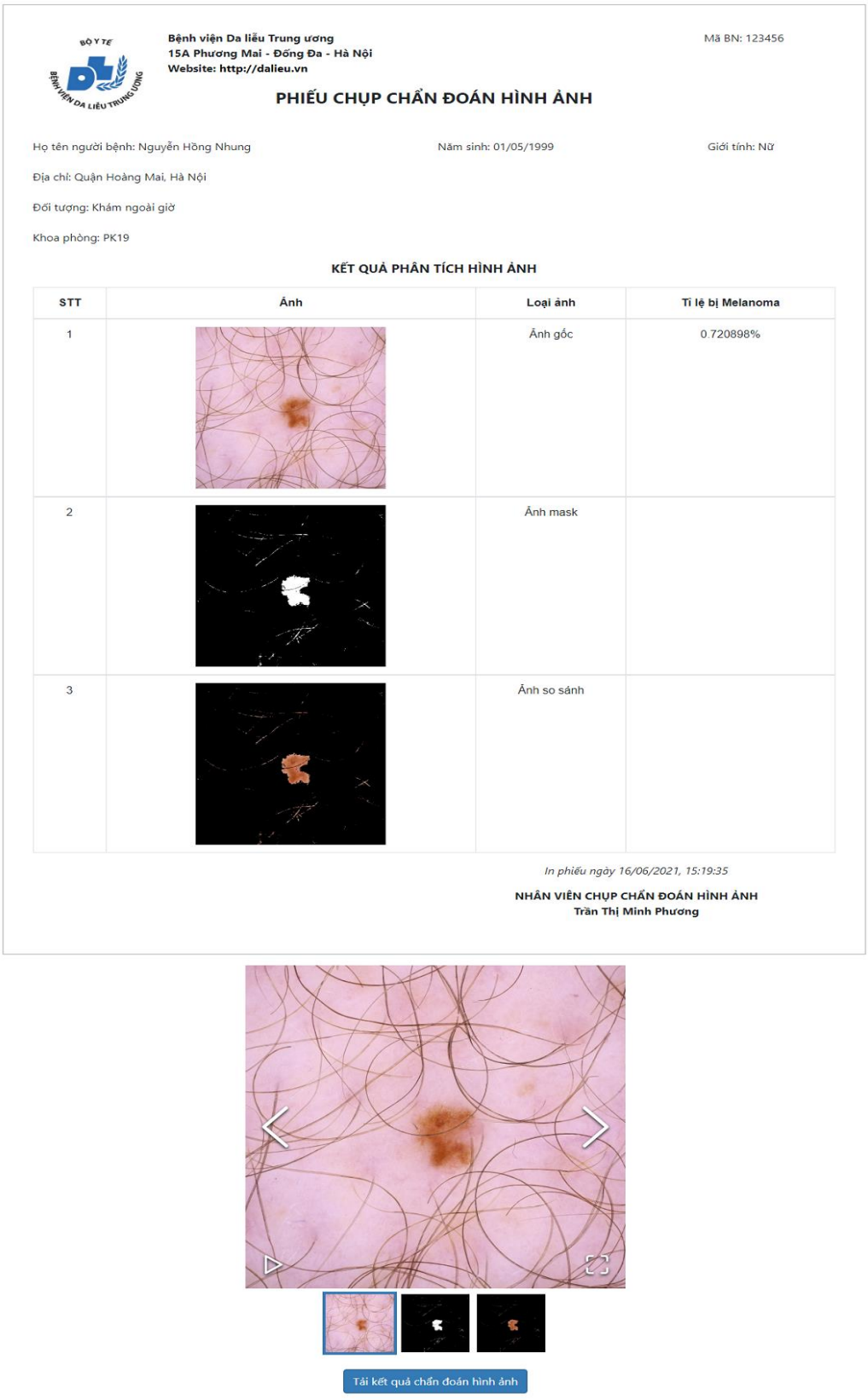
Hình 19 Minh họa giao diện Danh sách bệnh nhân tiếp nhận cần chẩn đoán hình ảnh

Hình 19 minh họa danh sách bệnh nhân tiếp nhận phía nhân viên chẩn đoán hình ảnh. Qua danh sách, nhân viên chẩn đoán hình ảnh biết mình cần chụp chẩn đoán cho bệnh nhân nào, khi ấn vào tên bệnh nhân tương ứng, giao diện **Hình 20** sẽ là giao diện để nhân viên chẩn đoán hình ảnh tải ảnh chụp chẩn đoán thu được từ máy chụp lên. Giao diện ảnh bên dưới cho phép phóng to hình ảnh để kiểm tra trước khi tiến hành phân tích ảnh.



Hình 20 Minh họa giao diện tải ảnh chẩn đoán hình ảnh của nhân viên chẩn đoán hình ảnh

4.5.3.3 Minh họa chức năng Xem kết quả báo cáo chẩn đoán hình ảnh



Bệnh viện Da liễu Trung ương
15A Phương Mai - Đống Đa - Hà Nội
Website: http://dalieu.vn

Mã BN: 123456

PHIẾU CHỤP CHẨN ĐOÁN HÌNH ẢNH

Họ tên người bệnh: Nguyễn Hồng Nhung | Năm sinh: 01/05/1999 | Giới tính: Nữ

Địa chỉ: Quận Hoàng Mai, Hà Nội

Đối tượng: Khám ngoài giờ

Khoa phòng: PK19

KẾT QUẢ PHÂN TÍCH HÌNH ẢNH

STT	Ảnh	Loại ảnh	Tỉ lệ bị Melanoma
1		Ảnh gốc	0.720898%
2		Ảnh mask	
3		Ảnh so sánh	

In phiếu ngày 16/06/2021, 15:19:35

NHÂN VIÊN CHỤP CHẨN ĐOÁN HÌNH ẢNH
Trần Thị Minh Phương

Hình 21 Giao diện minh họa Báo cáo chẩn đoán hình ảnh

Từ mục danh sách bệnh nhân, với những bệnh nhân có kết quả báo cáo chẩn đoán hình ảnh với trạng thái là "Đã xong" trong **Hình 18** và **Hình 19**, hay sau khi nhân viên chẩn đoán hình ảnh ấn nút "Tiến hành phân tích ảnh" thuộc giao diện **Hình 20** thì đều có thể xem được kết quả báo cáo chẩn đoán hình ảnh của bệnh nhân như giao diện minh họa ở **Hình 21**. Giao diện bao gồm các trường thông tin mẫu và thông tin bệnh nhân tuân theo quy định của Bộ Y tế, đồng thời có đính kèm ảnh và kết quả phân tích. Người xem báo cáo có thể ấn xem hình ảnh phóng to của các hình ảnh chẩn đoán bên dưới, hoặc có thể xuất báo cáo dưới dạng PDF bằng cách ấn vào nút "Tải kết quả chẩn đoán hình ảnh".

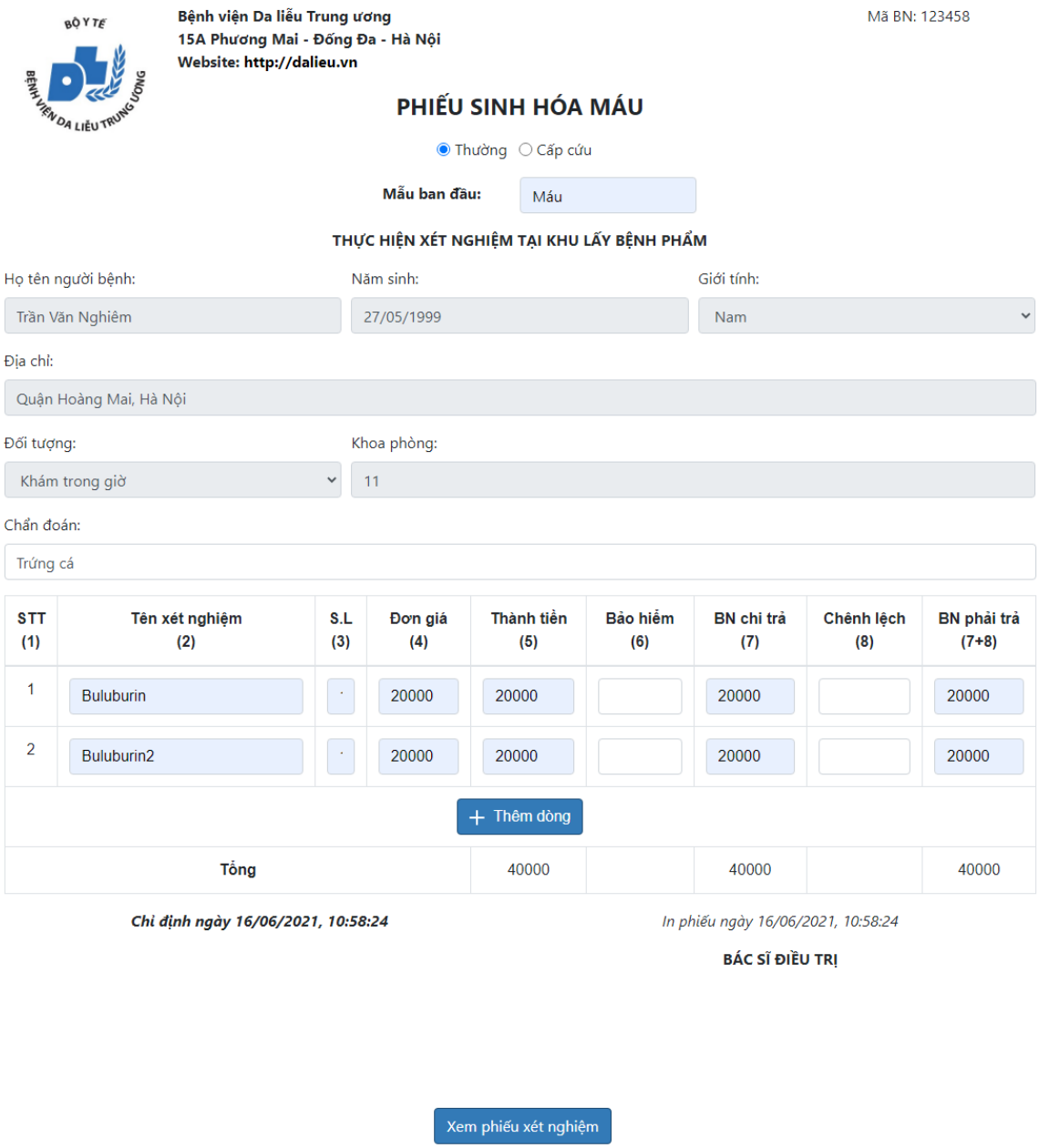
4.5.3.4 Minh họa chức năng Điền phiếu xét nghiệm sinh hóa máu

DANH SÁCH BỆNH NHÂN TIẾP NHẬN

STT	Họ tên	Mã bệnh nhân	Ngày sinh	Giới tính	Chẩn đoán của bác sĩ	Cần xét nghiệm sinh hóa máu	Cần xét nghiệm nấm - ký sinh trùng	Cần xét nghiệm huyết học - miễn dịch	Xét nghiệm tổng quát
1	Nguyễn Hồng Nhung	123456	01/05/1999	Nữ	Trứng cá	Đã xong	Đã xong	Đã xong	Đã xong
2	Phạm Ngọc Hưng	123457	27/04/1999	Nam	Chưa chẩn đoán	Không	Không	Không	Không
3	Trần Văn Nghiêm	123458	27/05/1999	Nam	Viêm da	Có	Có	Có	Có
4	Nguyễn Thị Lâm	123459	13/06/1966	Nữ	Trứng cá	Có	Có	Có	Không
5	Nguyễn Văn Cảnh	123460	14/06/1978	Nam	Trứng cá	Có	Không	Không	Không

Hình 22 Minh họa giao diện Danh sách bệnh nhân tiếp nhận cần xét nghiệm

Nhân viên xét nghiệm sẽ nhận được danh sách bệnh nhân cần xét nghiệm với thông tin được liệt kê chi tiết như ở **Hình 22**, từ màn hình này, nhân viên xét nghiệm lựa chọn điền phiếu xét nghiệm cho bệnh nhân bằng cách ấn vào chữ "Có". Sau khi lựa chọn được bệnh nhân để điền phiếu xét nghiệm, hệ thống hiển thị giao diện như **Hình 23**. Các thông tin bệnh nhân được tự động điền vào các mục cần thiết, các mục còn lại của phiếu xét nghiệm do nhân viên xét nghiệm điền. Sau khi điền xong các trường cần thiết của phiếu xét nghiệm, nhân viên xét nghiệm ấn nút "Xem phiếu xét nghiệm" để được chuyển đến màn hình xem kết quả phiếu xét nghiệm.



Hình 23 Minh họa giao diện Điền phiếu xét nghiệm Sinh hóa máu

4.5.3.5 Minh họa chức năng Xem phiếu xét nghiệm tổng quan

Từ mục danh sách bệnh nhân, với những bệnh nhân có kết quả phiếu xét nghiệm với trạng thái là "Đã xong" trong **Hình 18** và **Hình 22**, hay sau khi nhân viên xét nghiệm ấn nút "Xem phiếu xét nghiệm" thuộc giao diện **Hình 23** thì đều có thể xem được kết quả phiếu xét nghiệm tổng quan của bệnh nhân như giao diện minh họa ở **Hình 24**. Phiếu xét nghiệm tổng quan tham khảo theo phiếu xét nghiệm mẫu sau khi khảo sát tại Bệnh viện Da liễu Trung ương.

Hình 24 Minh họa giao diện Xem phiếu xét nghiệm tổng quan

4.6 Kiểm thử và triển khai

4.6.1 Kiểm thử hệ thống

Em đã sử dụng kĩ thuật kiểm thử hộp đen với 33 test case để kiểm thử cho hệ thống. Danh sách các test case kiểm thử được liệt kê chi tiết ở **Bảng 12**. Chi tiết nội dung kiểm thử của các test case em sẽ trình bày trong phần Phụ lục.

Bảng 12 Danh sách test case

STT	Nhóm chức năng/ chức năng chính	Tên test case
1	Quản lý người dùng	Kiểm tra hiển thị danh sách người dùng
2		Kiểm tra nhập thông tin người dùng
3		Kiểm tra thêm người dùng
4		Kiểm tra cập nhật thông tin người dùng
5		Kiểm tra xóa tài khoản người dùng
6	Quản lý danh sách bệnh nhân	Kiểm tra hiển thị danh sách bệnh nhân tiếp nhận
7		Kiểm tra nhập thông tin bệnh nhân
8		Kiểm tra thêm bệnh nhân
9		Kiểm tra cập nhật thông tin bệnh nhân
10		Kiểm tra hiển thị danh sách bệnh nhân sau khi cập nhật
11	Chẩn đoán bệnh nhân	Kiểm tra hiển thị danh sách bệnh nhân tiếp nhận
12		Kiểm tra nhập thông tin chẩn đoán bệnh nhân
13		Kiểm tra lưu thông tin chẩn đoán bệnh nhân
14		Kiểm tra hiển thị danh sách bệnh nhân sau khi đã chẩn đoán
15	Tìm kiếm thông tin bệnh nhân	Kiểm tra nhập thông tin tìm kiếm
16		Kiểm tra hiển thị nhập thông tin tìm kiếm

STT	Nhóm chức năng/ chức năng chính	Tên test case
17		Kiểm tra kết quả tìm kiếm
18		Kiểm tra hiển thị chi tiết kết quả tìm kiếm
19	Tiếp nhận bệnh nhân chẩn đoán hình ảnh	Kiểm tra hiển thị danh sách bệnh nhân chẩn đoán hình ảnh
20		Kiểm tra tải ảnh chụp chẩn đoán lên hệ thống
21		Kiểm tra lưu ảnh chụp chẩn đoán
22		Kiểm tra hiển thị kết quả chẩn đoán hình ảnh
23		Kiểm tra hiển thị danh sách bệnh nhân sau khi chẩn đoán hình ảnh
24		Kiểm tra kết quả xuất báo cáo sau khi chẩn đoán hình ảnh
25	Tiếp nhận bệnh nhân xét nghiệm	Kiểm tra hiển thị danh sách bệnh nhân cần xét nghiệm
26		Kiểm tra hiển thị phiếu xét nghiệm
27		Kiểm tra nhập thông tin các phiếu xét nghiệm
28		Kiểm tra lưu các phiếu xét nghiệm
29		Kiểm tra hiển thị danh sách bệnh nhân sau khi điền phiếu xét nghiệm
30		Kiểm tra kết quả xuất báo cáo sau điền phiếu xét nghiệm

STT	Nhóm chức năng/ chức năng chính	Tên test case
31	Đăng nhập	Kiểm tra hiển thị màn hình đăng nhập
32		Kiểm tra nhập thông tin đăng nhập
33		Kiểm tra kết quả đăng nhập

4.6.2 Triển khai hệ thống

Hệ thống được triển khai thử nghiệm tại https://103.159.51.183. Thông tin cụ thể về server dùng để cài đặt và triển khai hệ thống được mô tả chi tiết ở **Bảng 13**.

Bảng 13 Thông số cấu hình server triển khai hệ thống

Tên cấu hình	Thông số
Hệ điều hành	Ubuntu 16.04
CPU	Intel® Xeon® E5-26XX
RAM	1.5GB
SSD	15GB

4.7 Kết chương

Trong chương 4, em đã trình bày chi tiết về thiết kế kiến trúc của hệ thống cũng như cơ sở dữ liệu. Em cũng mô tả cách áp dụng các công nghệ giới thiệu ở chương 3 để xây dựng và triển khai hệ thống. Trong chương 5, em sẽ trình bày về các giải pháp nổi bật mà bản thân tâm đắc nhất trong quá trình thực hiện đồ án tốt nghiệp.

Chương 5 Các giải pháp nổi bật

Trong 4 chương trước, hệ thống em xây dựng và phát triển trong đồ án lần này đã được mô tả khá chi tiết và đầy đủ. Tuy nhiên, giải pháp về thiết kế kiến trúc hệ thống mà em cảm thấy phù hợp, tâm đắc và là nền tảng để em xây dựng hệ thống thì chưa được mô tả chi tiết. Vì vậy, trong chương 5 này, em sẽ làm rõ về kiến trúc Microservices qua các mục sau: (1) Đặt vấn đề, (2) Giải pháp và (3) Kết quả đạt được. Đồng thời em cũng nêu giới thiệu chi tiết về kiến trúc Frontend của hệ thống.

5.1 Kiến trúc Microservices

5.1.1 Đặt vấn đề

Sau khi phân tích vấn đề ở mục **1.1** và khảo sát hiện trạng ở mục **2.1**, em nhận thấy các phần mềm có mặt trên thị trường hiện nay còn khá nhiều điểm hạn chế, cụ thể:

Số lượng các phần mềm có hỗ trợ các tính năng quản lý chẩn đoán hình ảnh, xét nghiệm và thăm dò chức năng còn hạn chế. Nếu có thì thường nằm trong gói tính năng nâng cao với chi phí phải trả cho phần mềm là khá cao.

Giao diện các phần mềm thường sử dụng Winform nên thường đơn điệu về màu sắc, không trực quan và mất khá nhiều thời gian để làm quen.

Quan trọng nhất là các hệ thống phần mềm mà em khảo sát phần lớn đều áp dụng kiến trúc Monolithic (kiến trúc đơn khối). Kiến trúc đơn khối có khá nhiều ưu điểm như: đơn giản khi phát triển do công nghệ là thống nhất; dễ kiểm thử và triển khai do toàn bộ project được đóng gói thành một package; các bước xây dựng và phát triển ban đầu khá nhanh, yêu càu về cơ sở hạ tầng đơn giản. Tuy nhiên, khi project dần lớn hơn và trở nên phức tạp trong quá trình phát triển, việc tái cấu trúc và mở rộng hệ thống là vô cùng khó khăn do các thành phần trong hệ thống liên kết với nhau chặt chẽ, chỉ cần một thay đổi nhỏ ở thành phần này cũng làm ảnh hưởng đến toàn bộ thành phần khác. Việc phát hiện lỗi và sửa lỗi cũng mất rất nhiều thời gian do không xác định được lỗi nằm ở đâu. Đồng thời, việc áp dụng công nghệ mới vào hệ thống được thiết kế dựa trên kiến trúc đơn khối là một việc khó khăn và tốn thời gian vì lúc đó toàn bộ hệ thống cần thay đổi và cập nhật công nghệ. Do đó, các phần mềm ứng dụng kiến trúc Monolithic thường sử dụng các công nghệ đã cũ và lỗi thời.

Ngoài các điểm hạn chế phía trên, xét về thực tế, hệ thống được phát triển nhằm phục vụ nhiều nhóm người có chức năng và vai trò khác nhau. Do đó, hệ thống cần được phân chia thành nhiều thành phần nhỏ cung cấp các dịch vụ khác nhau với các chức năng cụ thể giải quyết nghiệp vụ chuyên môn riêng của từng nhóm người dùng. Việc phân chia và quản lý tài nguyên hệ thống sao cho phù hợp cũng là một bài toán cần cân nhắc giải quyết.

5.1.2 Giải pháp

Qua tìm hiểu, em nhận thấy kiến trúc Microservices là chìa khóa để giải quyết những vấn đề này. Đây là loại kiến trúc đang phổ biến trong những năm gần đây do ngoài cung cấp phần lớn các tính năng của kiến trúc Monolithic, kiến trúc này còn cung cấp nhiều tính năng cho phép thực hiện các thay đổi và mở rộng một cách linh hoạt. Cụ thể, kiến trúc Microservices sẽ chia hệ thống thành các thành phần dịch vụ nhỏ hơn, mỗi thành phần dịch vụ sẽ hoạt động độc lập, do đó cải thiện được các khuyết điểm của kiến trúc Monolithic.

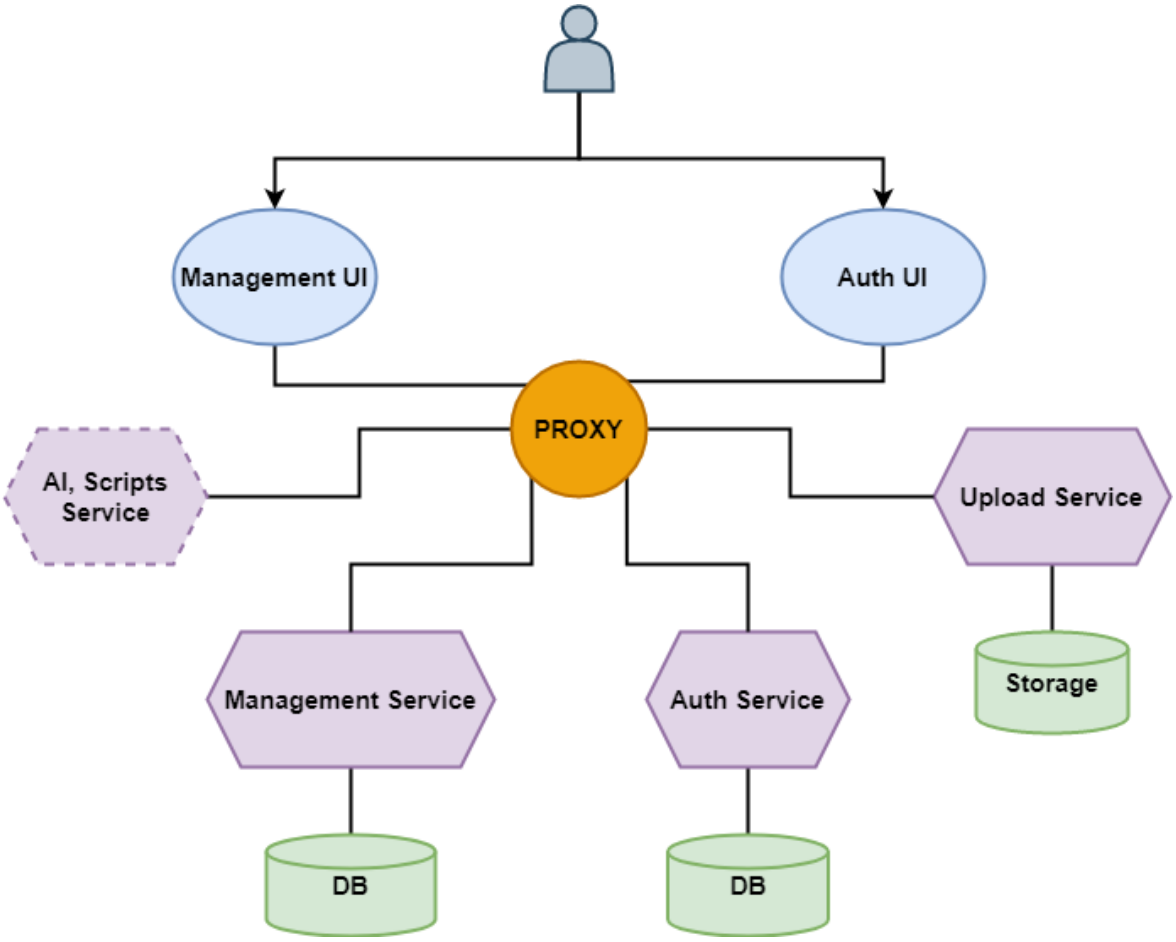
Cụ thể, dựa trên kiến trúc Microservices, em đã chia hệ thống thành các thành phần dịch vụ như sau: dịch vụ quản lý (Managament service), dịch vụ xác thực phân quyền người dùng (Auth service) và dịch vụ lưu trữ (Upload service), **Hình 25** mô tả tổng quan kiến trúc hệ thống dựa trên kiến trúc Microservices.

Nhóm dịch vụ quản lý đảm nhiệm thực hiện các chức năng quản lý (thêm mới, cập nhật, xóa) liên quan đến người dùng hệ thống, bệnh nhân, chẩn đoán, đồng thời cũng chịu trách nhiệm quản lý, vận hành hệ thống.

Dịch vụ xác thực phân quyền người dùng đảm nhiệm việc xác thực người dùng hệ thống. Cụ thể, khi người dùng sử dụng hệ thống, dịch vụ này sẽ xác thực người dùng, kiểm tra vai trò và quyền truy cập của người dùng, nếu hợp lệ sẽ được quyền sử dụng hệ thống dựa trên vai trò tương ứng sở hữu.

Việc quản lý và lưu trữ các tệp tin hệ thống như hình ảnh, tệp văn bản, mã nguồn sẽ do dịch vụ lưu trữ đảm nhiệm.

Ngoài các dịch vụ trên, hệ thống còn cần phát triển dịch vụ tương tác với các chương trình, mô hình chẩn đoán chạy song song bên ngoài. Dịch vụ này sẽ chịu trách nhiệm lấy các dữ liệu hệ thống làm đầu vào các mô hình chẩn đoán và chương trình thực thi, đợi chúng chạy xong và lấy các kết quả trả về lưu vào trong hệ thống.

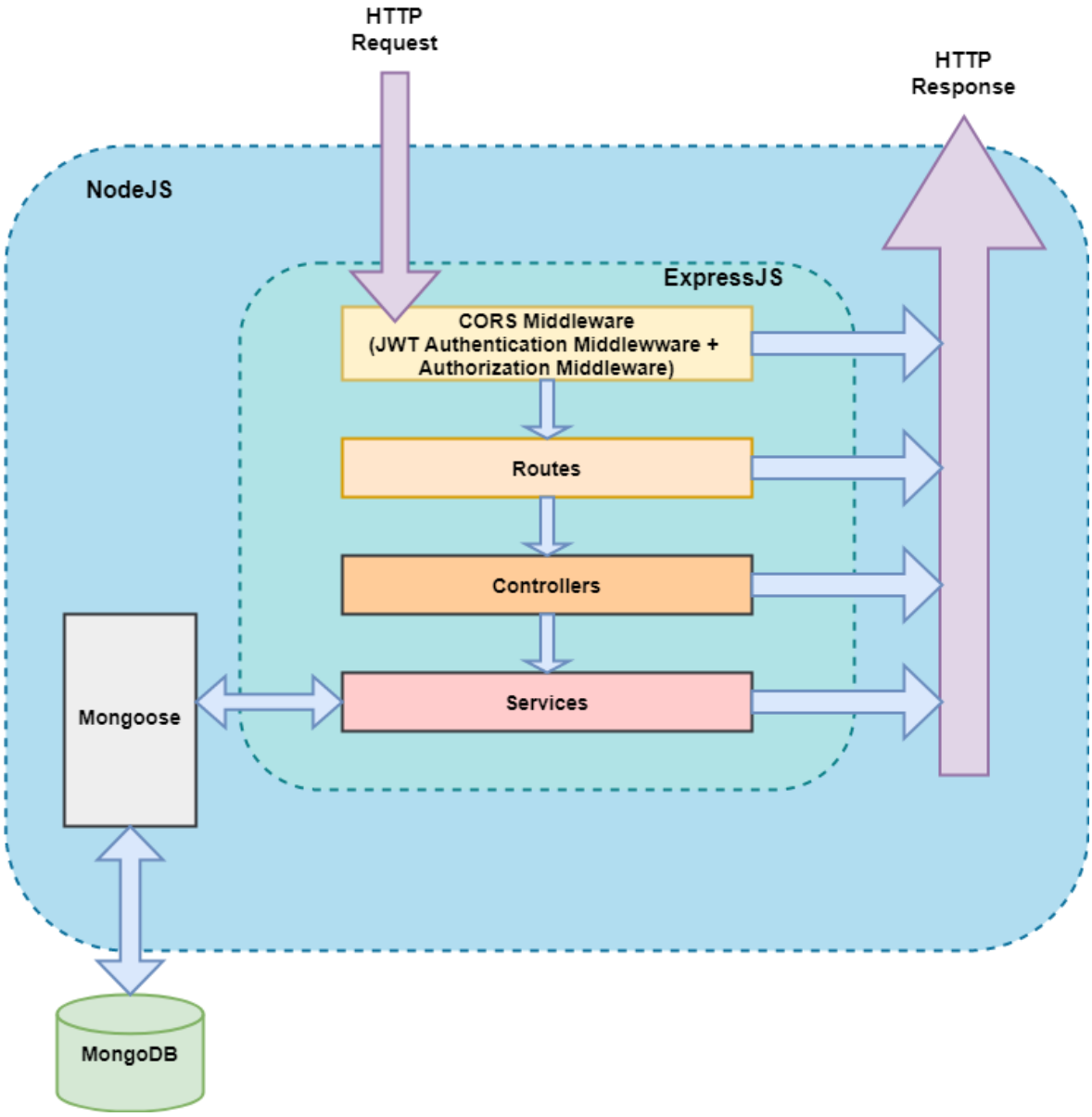


Hình 25 Tổng quan kiến trúc hệ thống dựa trên kiến trúc Microservices

5.1.2.1 Dịch vụ quản lý

Dịch vụ quản lý là dịch vụ cốt lõi của hệ thống mà em xây dựng. Dịch vụ quản lý đảm nhiệm chức năng quản lý, vận hành và lưu trữ những thông tin cốt lõi của hệ thống, đồng thời cũng là dịch vụ dành cho những nhóm người dùng có vai trò quản lý hệ thống. **Hình 8** mô tả kiến trúc tổng quan của dịch vụ với 4 thành phần chính là Routes, Controllers, Services và Models. Các yêu cầu sẽ được thành phần Routes tiếp nhận và điều hướng đến Controllers, Controllers nhận và xử lý yêu càu, sau đó lại chuyển tiếp xuống cho Services. Services sẽ thực hiện truy vấn cơ sở dữ liệu thông qua các Models.

5.1.2.2 Dịch vụ xác thực phân quyền người dùng



Hình 26 Kiến trúc dịch vụ xác thực phân quyền người dùng dựa trên thư viện JWT

Dịch vụ xác thực phân quyền người dùng có kiến trúc được mô tả ở **Hình 26**. Xác thực và phân quyền người dùng trong hệ thống là phần cơ bản mà bất kỳ hệ thống nào hiện nay cũng cần phải có. Authentication (xác thực) là quá trình hệ thống kiểm tra, xác định định danh của người dùng khi truy cập vào hệ thống. Về mặt kỹ thuật, Authorization (phân quyền) là quá trình xảy ra sau quá trình Authentication kết thúc, nghĩa là quá trình này xác định xem người dùng vừa xác thực có vai trò gì trong hệ thống. Quá trình xác thực và phân quyền của dịch vụ xác thực và phân quyền được hỗ trợ bởi thư viện JWT. JWT cung cấp token-based authentication. Cụ thể, trong xác thực người dùng, khi người dùng đăng nhập thành công,

server sẽ trả về một chuỗi token cho browser và chuỗi token này thường lưu trong LocalStorage hoặc Cookies của browser, lợi thế hơn việc phải lưu cả một Session vào Cookies như cách truyền thống. Bất cứ khi nào người dùng muốn truy cập vào những route được bảo vệ hoặc những route chỉ có những vai trò nhất định được truy cập, thì browser sẽ đính kèm token này trong Header Authorization của request rồi mới gửi đi. Single Sign On cũng là một chức năng có sử dụng JWT một cách rộng rãi, bởi vì chuỗi JWT có kích thước đủ nhỏ để đính kèm trong request và sử dụng ở nhiều hệ thống thuộc các domain khác nhau. Đây là một ứng dụng của Stateless authentication. Trạng thái của người dùng sẽ không được lưu trong bộ nhớ của server mà được đóng gói vào JWT. Như trong **Hình 26**, luồng hoạt động của dịch vụ được mô tả như sau:

Thông qua Express, các HTTP request hợp lệ và đúng với route đã thiết kế sẽ được kiểm tra bởi CORS Middleware, với Security layer bao gồm hai thành phần: JWT Authentication Middleware - có nhiệm vụ xác minh người dùng và chuỗi token; Authorization Middleware - Kiểm tra vai trò của người dùng với các thông tin được lưu trong cơ sở dữ liệu.

Nếu gặp bất kỳ lỗi nào trong toàn bộ quá trình trên sẽ lập tức phản hồi lại cho client dưới dạng HTTP response (error code).

5.1.2.3 Dịch vụ lưu trữ

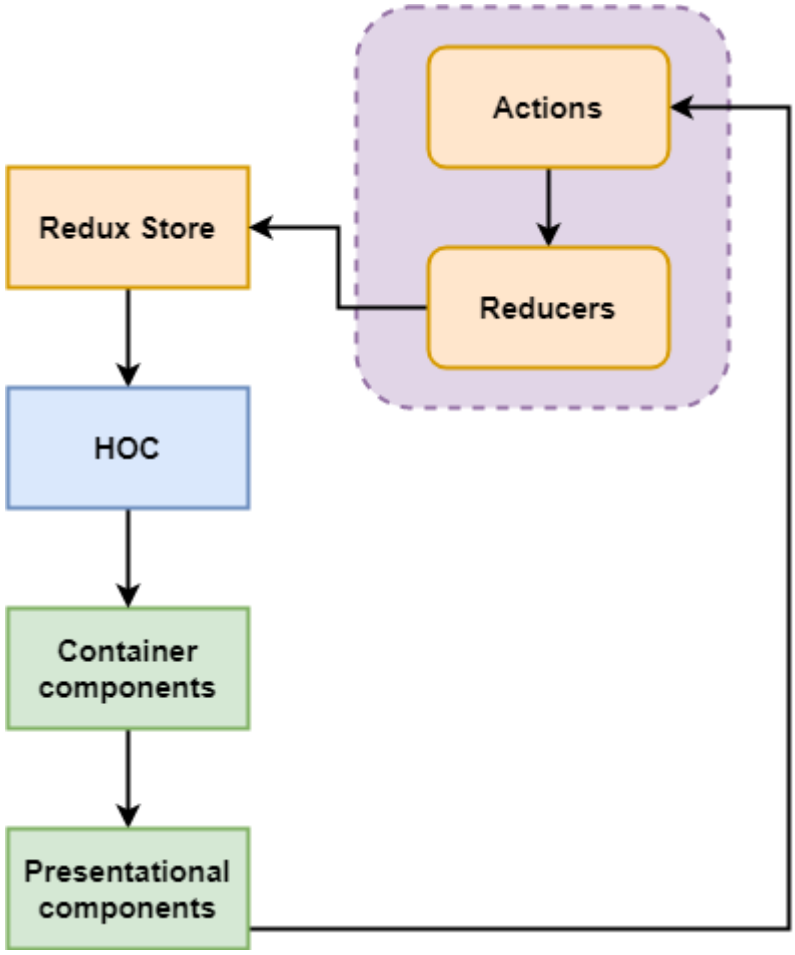
Để xây dựng hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm, việc lưu trữ các ảnh chụp chẩn đoán, ảnh sau khi phân tích dựa trên mô hình chẩn đoán ung thư hắc tố da hay các tệp tin lưu trữ kết quả chẩn đoán, kết quả xét nghiệm là vô cùng cần thiết. Dịch vụ lưu trữ được tách ra làm một dịch vụ riêng vì dịch vụ lưu trữ yêu cầu một kho lưu trữ lớn mà không thao tác nhiều với server và cơ sở dữ liệu. URL về nơi lưu trữ các tệp tin và hình ảnh được lưu trữ trên cơ sở dữ liệu.

5.1.3 Kết quả đạt được

Hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm là một hệ thống được xây dựng hướng đến tiêu chí dễ bảo trì, nâng cấp và mở rộng. Việc triển khai kiến trúc Microservices hoàn toàn đáp ứng được các tiêu chí của hệ thống do mỗi thành phần dịch vụ của hệ thống hoạt động độc lập. Mã nguồn khi xây dựng và triển khai mỗi dịch vụ cũng tập trung vào giải quyết một hoặc một số chức năng nghiệp vụ chính nên dễ đọc, dễ gỡ lỗi và kiểm thử.

5.2 Kiến trúc Frontend

Giao diện của hệ thống được xây dựng dựa trên sự kết hợp của React và Redux. Kiến trúc tổng quan về giao diện của hệ thống được mô tả ở **Hình 27**.



Hình 27 Kiến trúc tổng quan Frontend

5.2.1 Các thành phần giao diện trong ReactJS

5.2.1.1 Đặt vấn đề

Bản chất khi ta làm việc với React là ta đang làm việc với các component. React đã quá nổi tiếng và thành công trong việc chia nhỏ giao diện thành các thành phần view nhỏ hơn là các component. Nhưng giao diện không chỉ có mỗi chức năng hiển thị, giao diện cũng cần phải xử lý các logic nghiệp vụ và giao tiếp với cơ sở dữ liệu để lấy về dữ liệu, đồng thời giao diện cũng phải xử lý dữ liệu để cho ra được dữ liệu cần thiết cho việc hiển thị. Việc thiết kế các component vừa xử lý cả logic lẫn hiển thị dữ liệu là hoàn toàn không phù hợp với tiêu chí hướng đến một hệ thống có khả năng mở rộng, dễ bảo trì và nâng cấp vì khi đó mã nguồn của các component này sẽ rắc rối, phức tạp, chồng chéo, khó đọc và khó gỡ lỗi, khó có thể tối ưu mã nguồn để nâng cao hiệu suất cho hệ thống.

5.2.1.2 Giải pháp

Hai khái niệm chính là giải pháp khi xây dựng giao diện với ReactJS là Container components và Presentational components, hay còn gọi tắt là Container và component. Cụ thể:

Container components là các component đảm nhận chức năng kết nối với phía backend, xử lý các logic nghiệp vụ và cung cấp dữ liệu cho các component nhỏ hơn. Các component này thường là các stateful components, hay còn được gọi là các component cha.

Presentational components là các component con nhận dữ liệu từ các component cha và chịu trách nhiệm hiển thị các dữ liệu đó lên trên giao diện. Xử lý logic nếu có ở các component này thường là các logic chỉ liên quan đến giao diện chứ không phải là các logic liên quan đến xử lý dữ liệu. Chúng thường được gọi là các stateless components.

5.2.1.3 Kết quả đạt được

Việc phân chia các thành phần giao diện thành các component với chức năng riêng biệt khiến hệ thống tận dụng được tối đa hiệu quả của thư viện ReactJS mang lại. Mã nguồn không chỉ trong sáng, dễ hiểu mà còn dễ bảo trì, gỡ lỗi và mở rộng, phù hợp với tiêu chí xây dựng một hệ thống có tính mở rộng cao.

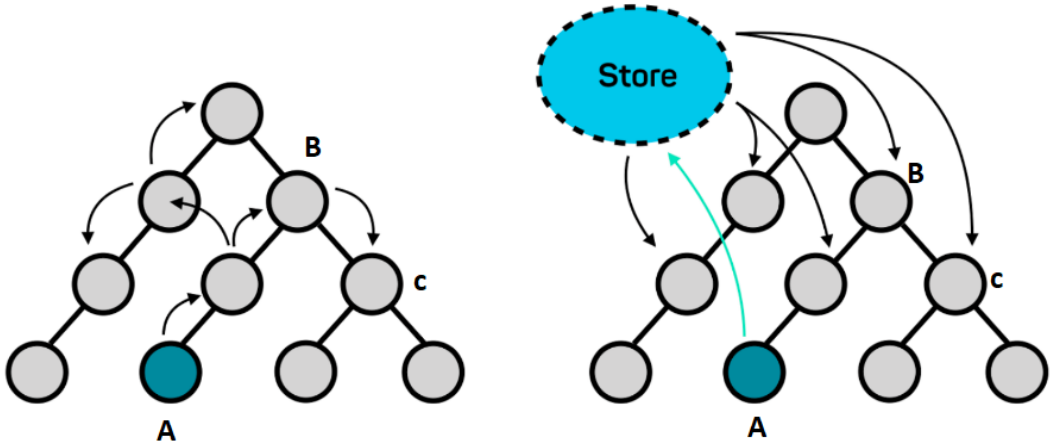
5.2.2 Redux

5.2.2.1 Đặt vấn đề

Các component thuộc React có hai thuộc tính cơ bản là Props và State. State được định nghĩa là một object có thể được sử dụng để chứa dữ liệu hoặc thông tin về components. Trong một React Component, state chỉ tồn tại trong phạm vi của components chứa nó, mỗi khi state thay đổi thì components đó sẽ được render lại. Lý tưởng nhất khi xây dựng một ứng dụng sử dụng React phía frontend là state của mỗi component không cần phải truyền đến một component cách xa nó trong cây component, nghĩa là một component không cần phải chia sẻ state qua nhiều component mới đến được component cần sử dụng state đó. Việc quản lý và chia sẻ state càng trở nên phức tạp khi quy mô hệ thống mở rộng, số lượng mã nguồn và logic xử lý các state cũng tăng lên, khiến việc đọc hiểu và bảo trì, nâng cấp hệ thống khó khăn.

5.2.2.2 Giải pháp

Redux là một giải pháp để giải quyết vấn đề chia sẻ state trong ReactJS. Cụ thể, Redux là một công cụ quản lý trạng thái ứng dụng. Sự khác biệt giữa việc không dùng Redux và dùng Redux để quản lý trạng thái được mô tả bằng **Hình 28**.



Hình 28 Nguyên lý hoạt động của Redux

Giả sử khi cần truyền dữ liệu từ component A đến component C, ta cần chia sẻ và cập nhật state qua rất nhiều component trung gian như component B, nhưng đó là cách làm truyền thống. Với Redux, việc chia sẻ state sẽ dễ dàng và thuận tiện hơn. Các state cần được chia sẻ sẽ được lưu vào một store trung gian, component nào cần đến state đó có thể kết nối đến store và lấy về state. Như ví dụ trên, khi muốn chia sẻ dữ liệu từ component A đến component C, chỉ cần lưu state của component A vào store, store sẽ cấp phát dữ liệu đó cho component C.

Ba khái niệm quan trọng trong Redux là Store, Actions và Reducers:

Actions được hiểu đơn giản là các event, với vai trò là kênh gửi dữ liệu duy nhất từ ứng dụng đến store, được gọi thông qua phương thức dispatch() của store. Bất cứ khi nào trạng thái của ứng dụng hay là render của view thay đổi thì việc đầu tiên là một action sẽ được tạo ra.

Reducers là các hàm thuần túy lấy trạng thái hiện tại của ứng dụng, thực hiện một action và trả về trạng thái mới. Reducers tạo ra các bản sao trạng thái và thao tác trên các bản sao này để trả về một trạng thái mới cho store dựa trên hành động đã thực hiện mà không thao tác trực tiếp lên trạng thái ứng dụng. Store sẽ sử dụng trạng thái mới đó như trạng thái chính thức mới.

Store được coi là phần quan trọng nhất trong Redux, đảm nhận nhiệm vụ quản lý và lưu trữ trạng thái của toàn bộ ứng dụng.

Kết hợp với các khái niệm Container components (containers) và Presentational components (components) nêu ở mục **5.2.1.2**, ta có thể thấy được luồng làm việc cơ bản phía Frontend của hệ thống. Người dùng sẽ thao tác với các thành phần giao diện là các components và sinh ra các actions để thay đổi trạng thái hoặc render view của các components này. Các actions được thực thi thông qua việc gọi đến phương thức dispatch(), các actions sau đó sẽ được gửi đến reducers. Reducers thực hiện hành động dựa vào action được gửi đến, đồng thời lưu vào store trạng thái mới của component. Store sẽ trả về trạng thái mới đó cho containers xử lý. Containers tiếp nhận state và xử lý nếu cần thiết, sau đó gửi xuống các components để tiến hành cập nhật trạng thái và render giao diện.

5.2.2.3 Kết quả đạt được

Khi sử dụng React và Redux, mã nguồn của hệ thống trong sáng, tối ưu hơn, không cần thiết phải viết những đoạn code lặp đi lặp lại khi cập nhật hay truyền trạng thái từ component này đến các component khác trong component tree. Điều này có nghĩa hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm có thể đáp ứng được tiêu chí dễ mở rộng, nâng cấp và bảo trì trong tương lai.

5.2.3 HOC

5.2.3.1 Đặt vấn đề

Component là đơn vị cơ bản nhất khi làm việc với React. Tuy nhiên một số component có cách thức hoạt động không thực sự phù hợp với định nghĩa các component truyền thống. Cụ thể, có một số component không giống nhau, có nghĩa là chúng gọi đến các phương thức và trả về các kết quả khác nhau, nhưng cách thức và trình tự thực thi thì lại giống nhau và cùng tương tác với một nguồn dữ liệu hay một sự kiện nào đó. Trong một hệ thống lớn, những pattern như vậy lặp đi lặp lại rất nhiều lần, nếu chúng ta liên tục viết đi viết lại mã nguồn giống nhau thì tổng thể mã nguồn của chúng ta không thực sự được coi là tái sử dụng như tiêu chí của React.

5.2.3.2 Giải pháp

HOC (higher-order component) là một kỹ thuật nâng cao trong React với khả năng tái sử dụng logic của component. Cụ thể, chúng nhận đầu vào là một component và trả về một component, HOC giúp tránh lặp lại code và sử dụng một logic chung cho nhiều component khác nhau mà không làm thay đổi component ban đầu. HOC sẽ bọc các component gốc trong một container, truyền thêm dữ liệu và props cho component gốc đó. HOC không can thiệp vào quá trình xử lý dữ liệu của các component gốc. Các component gốc cũng không quan

tâm dữ liệu và props đến từ đâu, chúng chỉ việc gọi HOC và dùng các dữ liệu, props đó để xử lý.

5.2.3.3 Kết quả đạt được

Trong hệ thống, em lựa chọn sử dụng HOC cho phần logic xác thực người dùng. Bất kỳ người dùng nào với vai trò gì, được gắn với các component và route nào, chỉ cần gọi HOC để xác thực trước khi sử dụng hệ thống, không cần viết đi viết lại các mã nguồn xử lý logic việc xác thực người dùng cho từng vai trò.

Chương 6 Kết luận và hướng phát triển

6.1 Kết luận

Qua 5 chương vừa trình bày ở trên, em đã mô tả xong về đồ án của mình. Hệ thống được thiết kế theo kiến trúc Microservices, đồng thời có khả năng đáp ứng yêu cầu mở rộng hệ thống sau này.

Để phát triển hệ thống, em đã dành thời gian đi khảo sát thực tế tại bệnh viện, nghiên cứu và tìm hiểu về quy trình làm việc của bệnh viện cũng như thông tin về loại máy chụp chẩn đoán hình ảnh đang được dùng tại bệnh viện. Đồng thời, em cũng đã tập trung tìm hiểu về một số phần mềm hỗ trợ các chức năng tương đương trên thị trường, nhưng thông tin thu về còn hạn chế do diễn biến phức tạp của dịch bệnh nên không thể đi khảo sát trực tiếp xem có bệnh viện nào đang triển khai hay không. Hệ thống của em được xây dựng và phát triển với đầy đủ các chức năng cơ bản mà em đã nêu ra và phân tích ở chương 2 và chương 4. Trong tương lai, nó sẽ còn kết hợp với mô hình chẩn đoán ung thư hắc tố da của bạn Nguyễn Trí Hùng dưới sự hướng dẫn của TS. Nguyễn Hồng Quang. Tuy nhiên, hiện tại hệ thống còn nhiều điểm cần cải thiện về mặt giao diện, hiệu năng truy vấn dữ liệu, cân bằng tải nếu có nhiều người dùng cùng hoạt động trong hệ thống. Đồng thời, chưa có nhiều mẫu báo cáo xét nghiệm được thiết kế có sẵn trên hệ thống do nguồn thông tin hình ảnh khảo sát được tại bệnh viện còn hạn chế. Hệ thống hiện tại cũng chỉ thực hiện demo phân tích file output trả về từ mô hình chẩn đoán là file csv và file hình ảnh.

Trong quá trình phát triển hệ thống, em đã gặp nhiều vấn đề cả về kiến thức chuyên môn và kĩ năng mềm nhưng đã kịp thời giải quyết được chúng và rút được nhiều bài học kinh nghiệm quý giá. Thứ nhất, đây là một hệ thống đề cao tính thực tế nên cần khảo sát nhiều, nhưng trong quá trình em thực hiện đồ án, tình hình dịch bệnh lại diễn biến phức tạp, đồng thời các tính năng của hệ thống có thể đã được phát triển bởi một số phần mềm nhưng nằm trong gói phần mềm nâng cao, rất khó để khảo sát và dùng thử. Thứ hai, cần phải lưu ý đặc biệt về UI/UX của hệ thống do người dùng không chuyên và giao diện cần hết sức trực quan, dễ dùng để thời gian tiếp cận và triển khai phần mềm được hiệu quả nhất. Thứ ba, hệ thống được em xây dựng và liên tục thay đổi trong quá trình phát triển để đáp ứng những yêu cầu mở rộng của hệ thống. Do vậy, lượng kiến thức chuyên môn trong quá trình phát triển hệ thống mà em cần đọc và tìm hiểu thêm là khá nhiều nhưng do kỹ năng quản lý thời

gian của em còn chưa tốt nên trong giai đoạn đầu của đồ án, em tốn khá nhiều thời gian để định hướng cũng như lựa chọn thiết kế cho hệ thống cũng như tìm hiểu các công nghệ cần và nên sử dụng. Thứ tư, do trình độ bản thân còn hạn chế và nhiều thiếu sót, nên nhiều đoạn mã nguồn trong hệ thống có thể vẫn chưa được tối ưu và dễ đọc.

Nhìn chung, em đã học hỏi được rất nhiều kiến thức, kĩ năng và kinh nghiệm sau khi thực hiện đồ án lần này. Đó là kỹ năng phân tích, thiết kế hệ thống, kỹ năng chọn lọc thông tin, kỹ năng quản lý thời gian và kỹ năng viết báo cáo. Chính những kiến thức và kinh nghiệm đó sẽ là công cụ giúp em hoàn thiện đồ án này hơn nữa trong tương lai gần, cũng như hoàn thiện kỹ năng của bản thân để có thể áp dụng vào công việc sau khi tốt nghiệp.

6.2 Hướng phát triển

Hệ thống hỗ trợ chẩn đoán hình ảnh và xét nghiệm em xây dựng trong khuôn khổ đồ án lần này vẫn đang được phát triển và cải tiến trong tương lai. Em dự định sẽ chú trọng hơn vào phần cải thiện giao diện người dùng, hướng đến hoàn thiện thiện tiêu chí đẹp mắt, thân thiện với người dùng và đặc biệt là phải trực quan do hướng đến đối tượng người dùng là các nhân viên và y bác sĩ trong bệnh viện. Không chỉ vậy, em còn hướng đến việc tối ưu code sao cho hiệu suất của hệ thống được cải thiện và nâng cao nhưng vẫn phải giữ được sự ổn định và khả năng bảo trì, mở rộng.

Trên đây là những kết luận và hướng phát triển của đồ án. Em mong nhận được những nhận xét, góp ý của thầy cô, của người dùng để hệ thống ngày càng phát triển và hoàn thiện hơn.

Tài liệu tham khảo

[1] "Cổng thông tin điện tử Bộ Y tế," [Online]. Available: https://moh.gov.vn/tin-noi-bat/-/asset_publisher/3Yst7YhbkA5j/content/hoi-nghi-hoi-ong-tu-van-ung-dung-phat-trien-cong-nghe-thong-tin-trong-y-te. [Accessed 12 6 2021].

[2] "Cục quản lý khám chữa bệnh Bộ Y tế," [Online]. Available: http://kcb.vn/vanban/mau-ho-so-benh-an-dung-trong-benh-vien. [Accessed 12 6 2021].

[3] "ReactJS," [Online]. Available: https://reactjs.org. [Accessed 12 6 2021].

[4] “State of JS,” [Online]. Available: https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/. [Accessed 12 6 2021].

[5] "Redux," [Online]. Available: https://redux.js.org. [Accessed 12 6 2021].

[6] "Bootstrap," [Online]. Available: https://getbootstrap.com/. [Accessed 12 6 2021].

[7] "NodeJS," [Online]. Available: https://nodejs.org. [Accessed 12 6 2021].

[8] "Express," [Online]. Available: https://expressjs.com. [Accessed 12 6 2021].

[9] "MongoDB," [Online]. Available: https://www.mongodb.com. [Accessed 12 6 2021].

[10] "Docker," [Online]. Available: https://www.docker.com/. [Accessed 12 6 2021].

Phụ lục

A Kịch bản kiểm thử

A.1 Kiểm thử chức năng Đăng nhập hệ thống

STT	Mô tả	Quy trình kiểm thử	Kết quả mong muốn	Kết quả thực tế	Trạng thái
1	Xác thực các trường thông tin email và password	Bỏ trống cả 2 trường email và password Nhấn nút đăng nhập	Hiển thị lỗi: Yêu cầu nhập địa chỉ email, Yêu cầu nhập password	Hiển thị lỗi: Yêu cầu nhập địa chỉ email, Yêu cầu nhập password	Đạt
2	Xác thực trường thông tin email	Nhập email sai quy định, trường password nhập đúng Nhấn nút đăng nhập	Hiển thị lỗi: Email không hợp lệ	Hiển thị lỗi: Email không hợp lệ	Đạt
3	Xác thực tài khoản	Nhập sai thông tin trường email hoặc password Nhấn nút đăng nhập	Hiển thị lỗi: Kiểm tra lại email hoặc password của bạn	Hiển thị lỗi: Kiểm tra lại email hoặc password của bạn	Đạt
4	Đăng nhập thành công	Nhập đủ, đúng thông tin trường email và password	Người dùng đăng nhập thành công	Người dùng đăng nhập thành công	Đạt

STT	Mô tả	Quy trình kiểm thử	Kết quả mong muốn	Kết quả thực tế	Trạng thái
		Nhấn nút đăng nhập			

A.2 Kiểm thử chức năng Nhập thông tin bệnh nhân

STT	Mô tả	Quy trình kiểm thử	Kết quả mong muốn	Kết quả thực tế	Trạng thái
1	Xác thực các trường thông tin của bệnh nhân	Bỏ trống cả tất cả các trường thông tin Nhấn nút Thêm thông tin bệnh nhân	Hiển thị lỗi: Vui lòng điền vào các trường thông tin cần thiết	Hiển thị lỗi: Vui lòng điền vào các trường thông tin cần thiết	Đạt
2	Xác thực các trường thông tin của bệnh nhân	Không nhập đủ các trường thông tin Nhấn nút Thêm thông tin bệnh nhân	Hiển thị lỗi: Vui lòng điền vào các trường thông tin còn thiếu	Hiển thị lỗi: Vui lòng điền vào các trường thông tin còn thiếu	Đạt
3	Thêm thông tin bệnh nhân thành công	Nhập đủ các trường thông tin được yêu cầu Nhấn nút Thêm thông tin bệnh nhân	Thêm thông tin bệnh nhân thành công	Thêm thông tin bệnh nhân thành công	Đạt