**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN TỐT NGHIỆP**

**<Tên đề tài>**

**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** | |

# 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. 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 xét nghiệm hiện đại đang còn hạn chế, nhiều bước còn thực hiện thủ công hoặc phải lặp đi lặp lại công việc nhập liệu thông tin bệnh nhân.

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 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 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

[Lời cam kết ii](#_Toc74392958)

[Lời cảm ơn iii](#_Toc74392959)

[Tóm tắt iv](#_Toc74392960)

[Mục lục v](#_Toc74392961)

[Danh mục hình vẽ ix](#_Toc74392962)

[Danh mục bảng x](#_Toc74392963)

[Danh mục công thức xi](#_Toc74392964)

[Danh mục các từ viết tắt xii](#_Toc74392965)

[Danh mục thuật ngữ xiii](#_Toc74392966)

[Chương 1 Giới thiệu đề tài 1](#_Toc74392967)

[1.1 Đặt vấn đề 1](#_Toc74392968)

[1.2 Mục tiêu và phạm vi đề tài 1](#_Toc74392969)

[1.3 Định hướng giải pháp 1](#_Toc74392970)

[1.4 Bố cục đồ án 2](#_Toc74392971)

[Chương 2 Khảo sát và phân tích yêu cầu 4](#_Toc74392972)

[2.1 Khảo sát hiện trạng 4](#_Toc74392973)

[2.2 Tổng quan chức năng 4](#_Toc74392974)

[2.2.1 Biểu đồ use case tổng quan 4](#_Toc74392975)

[2.2.2 Biểu đồ use case phân rã XYZ 5](#_Toc74392976)

[2.2.3 Quy trình nghiệp vụ 5](#_Toc74392977)

[2.3 Đặc tả chức năng 5](#_Toc74392978)

[2.3.1 Đặc tả use case A 5](#_Toc74392979)

[2.3.2 Đặc tả use case B 5](#_Toc74392980)

[2.4 Yêu cầu phi chức năng 5](#_Toc74392981)

[Chương 3 Công nghệ sử dụng 6](#_Toc74392982)

[3.1 Frontend 6](#_Toc74392983)

[3.1.1 ReactJS 6](#_Toc74392984)

[3.1.2 Redux 7](#_Toc74392985)

[3.1.3 Bootstrap 7](#_Toc74392986)

[3.2 Backend 8](#_Toc74392987)

[3.2.1 NodeJS 8](#_Toc74392988)

[3.2.2 ExpressJS 8](#_Toc74392989)

[3.2.3 MongoDB 8](#_Toc74392990)

[3.3 Triển khai hệ thống 9](#_Toc74392991)

[3.3.1 Docker 9](#_Toc74392992)

[Chương 4 Phát triển và triển khai ứng dụng 10](#_Toc74392993)

[4.1 Thiết kế kiến trúc 10](#_Toc74392994)

[4.1.1 Lựa chọn kiến trúc phần mềm 10](#_Toc74392995)

[4.1.2 Thiết kế tổng quan 10](#_Toc74392996)

[4.1.3 Thiết kế chi tiết gói 11](#_Toc74392997)

[4.2 Thiết kế chi tiết 11](#_Toc74392998)

[4.2.1 Thiết kế giao diện 11](#_Toc74392999)

[4.2.2 Thiết kế lớp 12](#_Toc74393000)

[4.2.3 Thiết kế cơ sở dữ liệu 12](#_Toc74393001)

[4.3 Xây dựng ứng dụng 12](#_Toc74393002)

[4.3.1 Thư viện và công cụ sử dụng 12](#_Toc74393003)

[4.3.2 Kết quả đạt được 12](#_Toc74393004)

[4.3.3 Minh hoạ các chức năng chính 13](#_Toc74393005)

[4.4 Kiểm thử 13](#_Toc74393006)

[4.5 Triển khai 13](#_Toc74393007)

[Chương 5 Các giải pháp và đóng góp nổi bật 14](#_Toc74393008)

[Chương 6 Kết luận và hướng phát triển 15](#_Toc74393009)

[6.1 Kết luận 15](#_Toc74393010)

[6.2 Hướng phát triển 15](#_Toc74393011)

[Tài liệu tham khảo 16](#_Toc74393012)

[Chương 7 Works Cited 17](#_Toc74393013)

[Phụ lục A-1](#_Toc74393014)

[A Hướng dẫn viết đồ án tốt nghiệp A-1](#_Toc74393015)

[A.1 Quy định chung A-1](#_Toc74393016)

[A.2 Ngành học A-2](#_Toc74393017)

[A.3 Tạo đề mục A-2](#_Toc74393018)

[A.4 Bảng biểu A-3](#_Toc74393019)

[A.5 Kiểm trùng báo cáo A-4](#_Toc74393020)

[A.6 Hình vẽ A-4](#_Toc74393021)

[A.7 Tài liệu tham khảo A-4](#_Toc74393022)

[A.8 Công thức toán học A-5](#_Toc74393023)

[A.9 Tham chiếu chéo A-5](#_Toc74393024)

[A.10 Cập nhật mục lục và tham chiếu chéo A-6](#_Toc74393025)

[A.11 In quyển đồ án tốt nghiệp A-6](#_Toc74393026)

[B Đặc tả use case B-8](#_Toc74393027)

[B.1 Đặc tả use case “Thống kê tình hình mượn sách” B-8](#_Toc74393028)

[B.2 Đặc tả use case “Đăng ký làm thẻ mượn” B-8](#_Toc74393029)

[C Công nghệ sử dụng C-8](#_Toc74393030)

[C.1 Công nghệ bảo mật dữ liệu C-8](#_Toc74393031)

[C.2 Công nghệ blockchain C-8](#_Toc74393032)

[D Thiết kế gói D-8](#_Toc74393033)

[D.1 Thiết kế gói cho kiến trúc tổng quan D-8](#_Toc74393034)

[D.2 Thiết kế gói cho chức năng “Trả sách” D-9](#_Toc74393035)

[E Thiết kế lớp E-9](#_Toc74393036)

# Danh mục hình vẽ

[**Hình 1** Ví dụ biểu đồ phụ thuộc gói 7](#_Toc27562454)

[**Hình 2** Ví dụ thiết kế gói 8](#_Toc27562455)

[**Hình 3** Ví dụ hình vẽ A-4](#_Toc27562456)

[**Hình 4** Quy cách đóng quyển A-6](#_Toc27562457)

[**Hình 5** Quy cách ghi chữ phần gáy A-6](#_Toc27562458)

[**Hình 6** Hướng dẫn thiết lập in hai mặt A-7](#_Toc27562459)

# Danh mục bảng

[**Bảng 1** Danh sách thư viện và công cụ sử dụng 9](#_Toc27562460)

[**Bảng 2** Ví dụ sử dụng bảng A-2](#_Toc27562461)

# Danh mục công thức

[**Công thức 1** Khai triển Newton A-5](#_Toc27562462)

Lưu ý: Nếu ĐATN có ít hơn ba công thức toán học, sinh viên có thể xóa bỏ mục này.

# 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 |
|  |  |
|  |  |

# 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.

## 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. Cụ thể, 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 “…”. 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.

## 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 output 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.

## 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.

## Bố cục đồ án

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

# ­­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 hiện nay và phân tích chi tiết yêu cầu của hệ thống.

## Khảo sát hiện trạng

Thông thường, khảo sát chi tiết về hiện trạng và yêu cầu của phần mềm sẽ được lấy từ ba nguồn chính, đó là (i) người dùng/khách hàng, (ii) các hệ thống đã có, (iii) và các ứng dụng tương tự.

Sinh viên cần tiến hành phân tích, so sánh, đánh giá chi tiết ưu nhược điểm của các sản phẩm/nghiên cứu hiện có. Sinh viên có thể lập bảng so sánh nếu cần thiết. Kết hợp với khảo sát người dùng/khách hàng (nếu có), sinh viên nêu và mô tả sơ lược các tính năng phần mềm quan trọng cần phát triển.

## Tổng quan chức năng

Phần 2.2 này có nhiệm vụ tóm tắt các chức năng của phần mềm. Trong phần này, sinh viên lưu ý chỉ mô tả chức năng mức cao (tổng quan) mà không đặc tả chi tiết cho từng chức năng. Đặc tả chi tiết được trình bày trong phần 2.3.

### Biểu đồ use case tổng quan

Sinh viên vẽ biểu đồ use case tổng quan và giải thích các tác nhân tham gia là gì, nêu vai trò của từng tác nhân, và mô tả ngắn gọn các use case chính.

### Biểu đồ use case phân rã XYZ

Với mỗi use case mức cao trong biểu đồ use case tổng quan, sinh viên tạo một mục riêng như mục 2.2.2 và tiến hành phân rã use case đó. Lưu ý tên use case cần phân rã trong biểu đồ use case tổng quan phải khớp với tên đề mục.

Trong mỗi mục như vậy, sinh viên vẽ và giải thích ngắn gọn các use case phân rã.

### Quy trình nghiệp vụ

Nếu sản phẩm/hệ thống cần xây dựng có quy trình nghiệp vụ quan trọng/đáng chú ý, sinh viên cần mô tả và vẽ biểu đồ hoạt động minh họa quy trình nghiệp vụ đó. Sinh viên lưu ý đây không phải là luồng sự kiện của *từng use case*, mà là luồng hoạt động kết hợp nhiều use case để thực hiện một nghiệp vụ nào đó.

Ví dụ, một hệ thống quản lý thư viện có quy trình nghiệp vụ mượn trả với mô tả sơ bộ như sau: Sinh viên làm thẻ mượn, sau đó sinh viên đăng ký mượn sách, thủ thư cho mượn, và cuối cùng sinh viên trả lại sách cho thư viện. Một hệ thống có thể có một vài quy trình nghiệp vụ quan trọng như vậy.

## Đặc tả chức năng

Sinh viên lựa chọn từ 4 đến 7 use case quan trọng nhất của đồ án để đặc tả chi tiết. Mỗi đặc tả bao gồm ít nhất các thông tin sau: (i) Tên use case, (ii) Luồng sự kiện (chính và phát sinh), (iii) Tiền điều kiện, và (iv) Hậu điều kiện. Sinh viên chỉ vẽ bổ sung biểu đồ hoạt động khi đặc tả use case phức tạp.

### Đặc tả use case A

### Đặc tả use case B

## Yêu cầu phi chức năng

Trong phần này, sinh viên đưa ra các yêu cầu khác nếu có, bao gồm các yêu cầu phi chức năng như hiệu năng, độ tin cậy, tính dễ dùng, tính dễ bảo trì, hoặc các yêu cầu về mặt kỹ thuật như về CSDL, công nghệ sử dụng, v.v.

# 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. 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.

## Frontend

### ReactJS

ReactJS [3] là một thư viện Javascript, 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.

### 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.

### 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).

## Backend

### 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.

### 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.

### 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.

Sử dụng nhiều bộ nhớ: do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên sẽ bị dữ 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.

## Triển khai hệ thống

### 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.

# Phát triển và triển khai ứng dụng

## Thiết kế kiến trúc

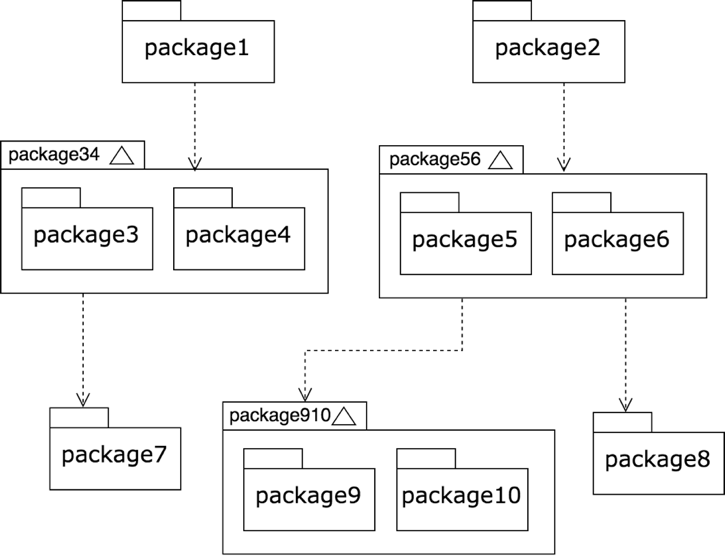
### Lựa chọn kiến trúc phần mềm

Mục này có độ dài từ một đến ba trang. Sinh viên cần lựa chọn kiến trúc phần mềm cho ứng dụng của mình như: kiến trúc ba lớp MVC, MVP, SOA, Microservice, v.v. rồi giải thích sơ bộ về kiến trúc đó (không giải thích chi tiết/dài dòng).

Sử dụng kiến trúc phần mềm đã chọn ở trên, sinh viên mô tả kiến trúc cụ thể cho ứng dụng của mình. Gợi ý: sinh viên áp dụng lý thuyết chung vào hệ thống/sản phẩm của mình như thế nào, có thay đổi, bổ sung hoặc cải tiến gì không. Ví dụ, thành phần M trong kiến trúc lý thuyết MVC sẽ là *những thành phần cụ thể nào* (ví dụ: là interface I + class C1 + class C2, v.v.) trong kiến trúc phần mềm của sinh viên.

### Thiết kế tổng quan

Sinh viên vẽ biểu đồ gói UML (UML package diagram), nêu rõ sự phụ thuộc giữa các gói (package). SV cần vẽ các gói sao cho chúng được phân theo các tầng rõ ràng, không được sắp đặt package lộn xộn trong hình vẽ. Sinh viên chú ý các quy tắc thiết kế (Các gói không phụ thuộc lẫn nhau, gói tầng dưới không phụ thuộc gói tầng trên, không phụ thuộc bỏ qua tầng, v.v.) và cần giải thích sơ lược về mục đích/nhiệm vụ của từng package. SV tham khảo ví dụ minh họa trong Hình 1.



**Hình 1** Ví dụ biểu đồ phụ thuộc gói

### Thiết kế chi tiết gói

Sinh viên thiết kế và lần lượt vẽ biểu đồ thiết kế cho từng package, hoặc một nhóm các package liên quan để giải quyết một vấn đề gì đó. Khi vẽ thiết kế gói, sinh viên chỉ cần đưa tên lớp, không cần chỉ ra các thành viên phương thức và thuộc tính. SV tham khảo ví dụ minh họa trong Hình 2.

Sinh viên cần vẽ rõ ràng quan hệ giữa các lớp trong biểu đồ. Các quan hệ bao gồm: phụ thuộc (dependency), kết hợp (association), kết tập (aggregation), hợp thành (composition), kế thừa (inheritance), và thực thi (implementation). Các quan hệ này đều đã được minh họa trong Hình 2.

Sau khi vẽ hình minh họa, sinh viên cần giải thích ngắn gọn về thiết kế của mình.



**Hình 2** Ví dụ thiết kế gói

## Thiết kế chi tiết

### Thiết kế giao diện

Phần này có độ dài từ hai đến ba trang. Sinh viên đặc tả thông tin về màn hình mà ứng dụng của mình hướng tới, bao gồm độ phân giải màn hình, kích thước màn hình, số lượng màu sắc hỗ trợ, v.v. Tiếp đến, sinh viên đưa ra các thống nhất/chuẩn hóa của mình khi thiết kế giao diện như thiết kế nút, điều khiển, vị trí hiển thị thông điệp phản hồi, phối màu, v.v. Sau cùng sinh viên đưa ra một số hình ảnh minh họa thiết kế giao diện cho các chức năng quan trọng nhất. Lưu ý, sinh viên không nhầm lẫn giao diện thiết kế với giao diện của sản phẩm sau cùng.

### Thiết kế lớp

Phần này có độ dài từ ba đến bốn trang. Sinh viên trình bày thiết kế chi tiết các thuộc tính và phương thức cho một số lớp chủ đạo/quan trọng nhất của ứng dụng (từ 2-4 lớp). Thiết kế chi tiết cho các lớp khác, nếu muốn trình bày, sinh viên đưa vào phần phụ lục.

Để minh họa thiết kế lớp, sinh viên thiết kế luồng truyền thông điệp giữa các đối tượng tham gia cho 2 đến 3 use case quan trọng nào đó bằng biểu đồ trình tự (hoặc biểu đồ giao tiếp).

### Thiết kế cơ sở dữ liệu

Phần này có độ dài từ hai đến bốn trang. Sinh viên thiết kế, vẽ và giải thích biểu đồ thực thể liên kết (E-R diagram). Từ đó, sinh viên thiết kế cơ sở dữ liệu tùy theo hệ quản trị cơ sở dữ liệu mà mình sử dụng (SQL, NoSQL, Firebase, v.v.)

## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

Sinh viên liệt kê các công cụ, ngôn ngữ lập trình, API, thư viện, IDE, công cụ kiểm thử, v.v. mà mình sử dụng để phát triển ứng dụng. Mỗi công cụ phải được chỉ rõ phiên bản sử dụng. SV nên kẻ bảng mô tả tương tự như Bảng 1. Nếu có nhiều nội dung trình bày, sinh viên cần xoay ngang bảng.

**Bảng 1** Danh sách thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Mục đích** | **Công cụ** | **Địa chỉ URL** |
| IDE lập trình | Eclipse Oxygen 64 bit | http://www.eclipse.org/ |
| v.v. | v.v. | v.v. |

### Kết quả đạt được

Sinh viên trước tiên mô tả kết quả đạt được của mình là gì, ví dụ như các sản phẩm được đóng gói là gì, bao gồm những thành phần nào, ý nghĩa, vai trò?

Sinh viên cần thống kê các thông tin về ứng dụng của mình như: số dòng code, số lớp, số gói, dung lượng toàn bộ mã nguồn, dung lượng của từng sản phẩm đóng gói, v.v. Tương tự như phần liệt kê về công cụ sử dụng, sinh viên cũng nên dùng bảng để mô tả phần thông tin thống kê này.

### Minh hoạ các chức năng chính

Sinh viên lựa chọn và đưa ra màn hình cho các chức năng chính, quan trọng, và thú vị nhất. Mỗi giao diện cần phải có lời giải thích ngắn gọn. Khi giải thích, sinh viên có thể kết hợp với các chú thích ở trong hình ảnh giao diện.

## Kiểm thử

Phần này có độ dài từ hai đến ba trang. Sinh viên thiết kế các trường hợp kiểm thử cho hai đến ba chức năng quan trọng nhất. Sinh viên cần chỉ rõ các kỹ thuật kiểm thử đã sử dụng. Chi tiết các trường hợp kiểm thử khác, nếu muốn trình bày, sinh viên đưa vào phần phụ lục.

Sinh viên sau cùng tổng kết về số lượng các trường hợp kiểm thử và kết quả kiểm thử. Sinh viên cần phân tích lý do nếu kết quả kiểm thử không đạt.

## Triển khai

Sinh viên trình bày mô hình và/hoặc cách thức triển khai thử nghiệm/thực tế. Ứng dụng của sinh viên được triển khai trên server/thiết bị gì, cấu hình như thế nào. Kết quả triển khai thử nghiệm nếu có (số lượng người dùng, số lượng truy cập, thời gian phản hồi, phản hồi người dùng, khả năng chịu tải, các thống kê, v.v.)

# Các giải pháp và đóng góp nổi bật

Chương này có độ dài tối thiểu 5 trang, tối đa không giới hạn. Sinh viên cần trình bày tất cả những nội dung đóng góp mà mình thấy tâm đắc nhất trong suốt quá trình làm ĐATN. Đó có thể là một loạt các vấn đề khó khăn mà sinh viên đã từng bước giải quyết được, là giải thuật cho một bài toán cụ thể, là giải pháp tổng quát cho một lớp bài toán, hoặc là mô hình/kiến trúc hữu hiệu nào đó được sinh viên thiết kế.

Chương này là **cơ sở quan trọng** để các thầy cô đánh giá sinh viên. Vì vậy, sinh viên cần phát huy tính sáng tạo, khả năng phân tích, phản biện, lập luận, tổng quát hóa vấn đề và tập trung viết cho thật tốt.

Mỗi giải pháp hoặc đóng góp của sinh viên cần được trình bày trong một mục độc lập bao gồm ba mục con: (i) dẫn dắt/giới thiệu về bài toán/vấn đề, (ii) giải pháp, và (iii) kết quả đạt được (nếu có).

Sinh viên lưu ý **không trình bày lặp lại nội dung**. Những nội dung đã trình bày chi tiết trong các chương trước không được trình bày lại trong chương này. Vì vậy, với nội dung hay, mang tính đóng góp/giải pháp, sinh viên chỉ nên tóm lược/mô tả sơ bộ trong các chương trước, đồng thời tạo tham chiếu chéo tới đề mục tương ứng trong Chương 5 này. Chi tiết thông tin về đóng góp/giải pháp được trình bày trong mục đó.

Ví dụ, trong Chương 4, sinh viên có thiết kế được kiến trúc đáng lưu ý gì đó, là sự kết hợp của các kiến trúc MVC, MVP, SOA, v.v. Khi đó, sinh viên sẽ chỉ mô tả ngắn gọn kiến trúc đó ở Chương 4, rồi thêm các câu có dạng: “Chi tiết về kiến trúc này sẽ được trình bày trong phần 5.1”.

# Kết luận và hướng phát triển

## Kết luận

Sinh viên so sánh kết quả nghiên cứu hoặc sản phẩm của mình với các nghiên cứu hoặc sản phẩm tương tự.

Sinh viên phân tích trong suốt quá trình thực hiện ĐATN, mình đã làm được gì, chưa làm được gì, các đóng góp nổi bật là gì, và tổng hợp những bài học kinh nghiệm rút ra nếu có.

## Hướng phát triển

Hệ thống liên kết trung gian 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] | "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,” [Trực tuyến]. Available: https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/. [Đã truy cập 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