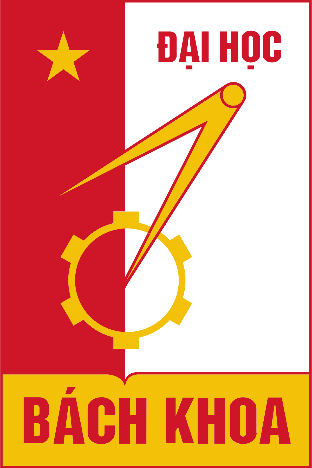
ĐẠI HỌC BÁCH KHOA HÀ NỘI

**KHOA CÔNG NGHỆ THÔNG TIN**



**KHÓA LUẬN TỐT NGHIỆP**

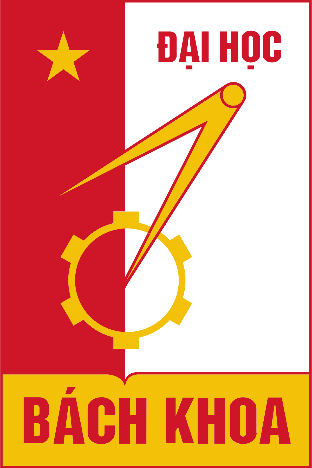
**ĐỀ TÀI:**

**Thiết Kế Website Quản Lý Đăng Ký Lịch Khám**

**HÀ NỘI - 2023**

ĐẠI HỌC BÁCH KHOA HÀ NỘI

**KHOA CÔNG NGHỆ THÔNG TIN**



**XÂY DỰNG HỆ THỐNG QUẢN LÝ ĐĂNG KÝ ĐẶT LỊCH KHÁM BỆNH**

**KHÓA LUẬN TỐT NGHIỆP**

**Chuyên ngành: CÔNG NGHỆ THÔNG TIN**

**Khóa học: 2018-2022**

**Người hướng dẫn: Nguyễn Thị Liệu**

**HÀ NỘI - 2023**

# LỜI CẢM ƠN

Trong quá trình học tập và rèn luyện tại Khoa Công nghệ thông tin – Trường Đại Học Bách Khoa Hà Nội và trong thời gian thực hiện thực tập tốt nghiệp đề tài “Xây dựng hệ thống quản lý đăng ký lịch khám bệnh”, em xin gửi lời cảm ơn về sự giúp đỡ của các thầy cô, giảng viên, cán bộ trong Khoa Công nghệ thông tin đã giúp em có được kiến thức và hoàn thành đề tài thực tập

Đặc biệt, em xin gửi lời cảm ơn tới giảng viên – Dương Thúy Hường – cô đã trực tiếp hướng dẫn và chỉ bảo cho em, giúp em rất nhiều trong quá trình tìm hiểu và hoàn thành đề tài.

Cuối cùng, với sự cố gắng của bản thân nhưng vốn kiến thức và thời gian còn hạn chế, báo cáo thực tập của em không thể tránh khỏi thiếu sót. Vì vậy em rất mong nhận được sự chỉ bảo của các thầy cô để em có thể nâng cao kiến thức của bản thân, và hơn nữa là phục vụ cho công việc sau này.

# TÓM TẮT NỘI DUNG ĐỒ ÁN

Hiện nay, các công nghệ tiên tiến phát triển ngày càng mạnh mẽ và được ứng dụng ngày càng nhiều vào các lĩnh vực kinh tế, sản xuất cũng như đời sống thường nhật của con người. Một điểm tiêu biểu trong việc phát triển các công nghệ đó phải kể đến việc ứng dụng công nghệ thông tin vào ngành y tế. Nhờ đó, giúp cho việc quản lý thời gian và đặt lịch khám bệnh đã không cần phải làm thủ công như ngày trước. Mọi người có thể sử dụng website để đặt lịch trực tuyến và đến khám theo giờ đã đặt

Từ điều trên, em đưa ra giải pháp tạo ra một trang web giúp mọi người có thể đặt lịch khám bệnh trực tuyến mà không cần phải đến bệnh viện, giúp mọi người có thể tiết kiệm được thời gian đi lại khi khám bệnh

Trang web này sẽ hỗ trợ bệnh nhân đặt lịch khám với bác sĩ một các nhanh chóng, người dùng chỉ cần chọn bác sĩ và thời gian muốn khám. Từ đó, giúp bệnh nhân tiết kiệm chi phí về thời gian đi lại và tiền bạc. Bên cạnh đó, trang web còn hỗ trợ bác sĩ quản lý khám của mình. Cuối cùng, trang web cho phép quản trị viên quản lý người dùng, cơ sở y tế và đội ngũ bác sĩ cũng như lịch đặt và chuyên khoa một các chính xác nhất.

**Mục Lục**

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 7](#_Toc152011970)

[1.1 Đặt vấn đề 7](#_Toc152011971)

[1.2 Mục tiêu và phạm vi đề tài 7](#_Toc152011972)

[1.3 Định hướng và giải pháp 8](#_Toc152011973)

[CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU 10](#_Toc152011974)

[2.1 Khảo sát hiện trạng 10](#_Toc152011975)

[2.1.1 Khảo sát các sản phẩm đặt lịch khám được áp dụng tại Việt Nam và nước ngoài 10](#_Toc152011976)

[2.1.1 Khảo sát yêu cầu của người dùng 10](#_Toc152011977)

[2.2 Tổng quan về chức năng 11](#_Toc152011978)

[2.2.1 Biểu đồ usecase tổng quát 13](#_Toc152011979)

[2.2.2 Biểu đồ usecase quản lý lịch khám của bác sỹ 14](#_Toc152011980)

[2.2.3 Biểu đồ usecase quản lý lịch khám của quản trị viên 15](#_Toc152011981)

[2.2.4 Biểu đồ usecase đặt lịch 16](#_Toc152011982)

[2.2.5 Quy trình đặt lịch khám 17](#_Toc152011983)

[2.2.6 Quy trình hủy lịch khám 18](#_Toc152011984)

[2.3 Đặc tả chức năng 19](#_Toc152011985)

[2.3.1 Đặc tả usecase đăng nhập 19](#_Toc152011986)

[2.3.2 Đặc tả usecase thêm mới bác sĩ 20](#_Toc152011987)

[2.3.3 Đặc tả usecase đăng ký tài khoản 21](#_Toc152011988)

[2.3.4 Đặc tả usecase đặt lịch khám 22](#_Toc152011989)

[2.3.5 Đặc tả usecase hủy lịch khám 23](#_Toc152011990)

[2.4 Yêu cầu phi chức năng 24](#_Toc152011991)

[2.4.1. Yêu cầu về bảo mật 24](#_Toc152011992)

[2.4.2. Yêu cầu về tính sử dụng 24](#_Toc152011993)

[2.4.3. Yêu cầu về tính tính thuận tiện 24](#_Toc152011994)

[2.4.4. Yêu cầu về bảo trì hệ thống 25](#_Toc152011995)

[CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG 26](#_Toc152011996)

[3.1 Hệ quản trị cơ sở dữ liệu mysql 26](#_Toc152011997)

[3.2 Ngôn ngữ lập trình java 27](#_Toc152011998)

[3.3 Spring boot framework 28](#_Toc152011999)

[3.4 Hibernate 29](#_Toc152012000)

[3.5 HTML 29](#_Toc152012001)

[3.6 CSS 30](#_Toc152012002)

[3.6 Javascript 30](#_Toc152012003)

[3.7 Công cụ sử dụng 31](#_Toc152012004)

[3.7.1. Intellij IDE 31](#_Toc152012005)

[3.7.2 Laragon 31](#_Toc152012006)

[3.7.3 Postman 32](#_Toc152012007)

[CHƯƠNG 4. THỰC NGHIỆM VÀ ĐÁNH GIÁ 33](#_Toc152012008)

[4.1 Thiết kế kiến trúc rest api 33](#_Toc152012009)

[4.1.1. Khái niệm về rest api 33](#_Toc152012010)

[4.1.2. Cách hoạt động của REST 35](#_Toc152012011)

[4.1.3 Ưu điểm của REST 35](#_Toc152012012)

[4.1.4 Nhược điểm của REST 36](#_Toc152012013)

[4.2 Thiết kế chi tiết 37](#_Toc152012014)

[4.2.1 Sơ đồ lớp 37](#_Toc152012015)

[4.2.2 Thiết kế cơ sở dữ liệu 39](#_Toc152012016)

[4.3 Thiết kế giao diện 45](#_Toc152012017)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 53](#_Toc152012018)

[5.1 kết luận 53](#_Toc152012019)

[5.2 Hướng phát triển 54](#_Toc152012020)

[CHƯƠNG 6. TÀI LIỆU THAM KHẢO 55](#_Toc152012021)

**DANH MỤC HÌNH VẼ**

[Hình 1 Sơ đồ usecase tổng quát 14](#_Toc152012159)

[Hình 2 Biểu đồ usecase quản lý lịch khám của bác sỹ 15](#_Toc152012160)

[Hình 3Biểu đồ usecase quản lý lịch khám của quản trị viên 16](#_Toc152012161)

[Hình 4 Biểu đồ usecase đặt lịch 17](#_Toc152012162)

[Hình 5 Quy trình đặt lịch khám 18](#_Toc152012163)

[Hình 6 Quy trình hủy lịch khám 19](#_Toc152012164)

[Hình 7 Mô hình rest api 35](#_Toc152012165)

[Hình 8 Thiết kế sơ đồ lớp 38](#_Toc152012166)

[Hình 9 Thiết kế cơ sở dữ liệu 40](#_Toc152012167)

[Hình 10 Giao diện trang chủ 46](#_Toc152012168)

[Hình 11 Giao diện đặt lịch khám 47](#_Toc152012169)

[Hình 12 Giao diện lịch sử đặt lịch 47](#_Toc152012170)

[Hình 13 Giao diện đăng nhâp 48](#_Toc152012171)

[Hình 14 Giao diện đăng ký 48](#_Toc152012172)

[Hình 15 Giao diện quản lý lịch đặt của bác sĩ 49](#_Toc152012173)

[Hình 16 Giao diện tạo phản hồi khám cho bác sĩ 49](#_Toc152012174)

[Hình 17 Giao diện cập nhật thông tin cá nhân của bác sĩ 50](#_Toc152012175)

[Hình 18 Giao diện quản lý danh sách bác sĩ 50](#_Toc152012176)

[Hình 19 Giao diện quản lý danh sách tài khoản 51](#_Toc152012177)

[Hình 20 Giao diện quản lý chuyên khoa 51](#_Toc152012178)

[Hình 21 Giao diện quản lý lịch đặt của admin 52](#_Toc152012179)

[Hình 22 Giao diện xác nhận thanh toán lịch đặt 52](#_Toc152012180)

[Hình 23 Giao diện quản lý bài viết 53](#_Toc152012181)

[Hình 24 Giao diện quản lý dịch vụ y tế 53](#_Toc152012182)

**DANH MỤC BẢNG BIỂU**

[Bảng 1 Đặc tả use case "Đăng nhập" 21](#_Toc152012252)

[Bảng 2 Đặc tả usecase thêm mới bác sĩ 22](#_Toc152012253)

[Bảng 3 Đặc tả usecase đăng ký tài khoản 23](#_Toc152012254)

[Bảng 4 Đặc tả usecase đặt lịch khám 24](#_Toc152012255)

[Bảng 5Đặc tả usecase hủy lịch khám 25](#_Toc152012256)

[Bảng 6 bảng authority 41](#_Toc152012257)

[Bảng 7 Bảng blog 42](#_Toc152012258)

[Bảng 8 Bảng detail\_invoice 42](#_Toc152012259)

[Bảng 9 Bảng doctor 43](#_Toc152012260)

[Bảng 10 Bảng image\_schedule 43](#_Toc152012261)

[Bảng 11 Bảng invoice 44](#_Toc152012262)

[Bảng 12 medical\_process 44](#_Toc152012263)

[Bảng 13 Bảng medicine 44](#_Toc152012264)

[Bảng 14 Bảng schedule\_as 45](#_Toc152012265)

[Bảng 15 Bảng specialist 46](#_Toc152012266)

[Bảng 16 Bảng user 46](#_Toc152012267)

[Bảng 17 Bảng user\_authority 47](#_Toc152012268)

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

## **1.1 Đặt vấn đề**

Trong lĩnh vực khám và chăm sóc sức khỏe, việc đặt lịch khám trực tuyến đang trở thành một xu hướng quan trọng và ngày càng phổ biến. Tuy nhiên việc đặt lịch khám trực tuyến, vẫn tồn tại nhiều rủi ro và hạn chế và hiệu quả của hệ thống đặt lịch khám trực tuyến.

Các vấn đề có thể gặp phải trong việc đặt lịch trực tuyến đó là:

* Khó khăn trong việc chọn bác sĩ phù hợp hay các cơ sở y tế
* Quy trình đặt lịch sẽ phức tạp đối với những người không am hiểu về công nghệ hay ít sử dụng internet
* Khó khăn trong việc quản lý việc hủy lịch hẹn của bệnh nhân hoặc bệnh nhân không đến khám gây lãng phí thời gian
* Các thông tin cá nhân sẽ được bảo mật, đây là yếu tố cực kỳ quan trọng trong việc đặt lịch khám trực tuyến

Quy trình đặt lịch khám trực tuyến giúp tiết kiệm thời gian và công sức cho người dùng, nâng cao chất lượng chăm sóc sức khỏe và sự hài lòng của người dùng. Giao diện đơn giản và dễ sử dụng, ít bước thực hiện và tính linh hoạt trong việc lựa chọn thời gian khám sẽ giảm bớt sự phiền toái và giúp người dùng nhanh chóng hoàn thành quá trình đặt lịch một cách thuận tiện và hiệu quả.

* 1. **Mục tiêu và phạm vi đề tài**

Hiện tại trên internet cung cấp rất nhều website, phần mềm giúp người dùng đặt lịch hẹn khám bệnh tại Việt Nam như Vinmec.com, CarePlus.com, benhvienthucuc.vn.... Tuy nhiên hầu hết các web này đều không đáp ứng được đầy đủ các nhu cầu của người dùng. Vấn đề chung của các web trên đó là giao

diện phức tạp, khó sử dụng, nhiều thông tin thừa không cần thiết.

Dựa trên nhu cầu của người dùng là đặt lịch khám, vì vậy em xin đưa ra một giải pháp đó là xây dựng một hệ thống website đặt lịch khám bệnh cho bệnh nhân, tập chung vào chức năng đặt lịch, quản lý lịch hẹn, dễ dàng thao tác xem và hủy lịch hẹn, phù hợp với đại đa số người dùng hiện nay. Website cũng cung cấp chức năng cho quản trị viên và bác sĩ. Quản trị viên có thể quản lý toàn bộ thông tin của hệ thống như quản lý bác sĩ, quản lý tài khoản, dịch vụ và lịch hẹn,… Bác sĩ có thể quản lý lịch hẹn của bệnh nhân với bác sĩ đó để bác sĩ có thể lên lịch khám cho bệnh nhân hoặc trả kết quả khám cho bệnh nhân.

* 1. **Định hướng và giải pháp**

Với các mục tiêu đã đề ra, em xin đề xuất xây dựng hệ thống đặt lịch khám trực tuyến dựa trên mô hình restful Api. Api sẽ cung cấp cho các dối tượng sử dụng là:

* Quản trị viên hệ thống
* Bác sĩ
* Bệnh nhân

Hệ thống sẽ được xây dựng trên nền tảng web app để tạo sự thuận tiện cho người sử dụng. Người dùng chỉ cần một thiết bị có kết nối internet như điện thoại hay máy tính đều có thể truy cập được hệ thống. Với định hướng là nền tảng web app, em dự định sẽ sử dụng các công nghệ sau để phát triển:

**Html, javascript**: đây là nền tảng cốt lỗi của lập trình front end, giúp xây dựng nên giao diện của người dùng và hiển thị dữ liệu

**Spring boot**: đây là một framework rất mạnh mẽ của java để xây dựng backend cho web hoặc app, spring hỗ trợ rất tốt cho việc lập trình Api cũng như rất nhiều các công nghệ như ORM, Spring security,… Spring boot là một framework giúp chúng ta phát triển ứng dụng web cũng như chạy ứng dụng một cách nhanh chóng và dễ ràng mở rộng.

**Mysql:** đây là một trong những hệ quản trị cơ sở dữ liệu phổ biến nhất cho các ứng dụng web. Là một hệ quản trị cơ sở dữ liệu miễn phí và mạnh mẽ, nó cho phép lưu trữ dữ liệu rạng quan hệ, điều này giúp dữ liệu trở nên chặt chẽ hơn so với những hệ quản trị cơ sở dữ liệu nosql

**CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU**

* 1. **Khảo sát hiện trạng**

**2.1.1 Khảo sát các sản phẩm đặt lịch khám được áp dụng tại Việt Nam và nước ngoài**

Trong thời đại công nghệ phát triển hiện đại ngày nay, việc đặt lịch khám bệnh trực tuyến đã trở nên phổ biến, hầu hết trên khắp các quốc gia đều có các hệ thống đặt lịch khám bệnh, và Việt Nam cũng giống như vậy.

Tại Việt Nam, chúng ta có thể tìm thấy các website đặt lịch khám bệnh chỉ với một lần click chuột như Vinmec, bookingcare, medpro, benhvienthucuc,…

Với các trang web nước ngoài, chúng ta có thể tìm thấy các website nổi tiếng như Zoodoc, Practo, Doctolib,…

Các website trên hầu hết đều có chung các chức năng như:

– Hiển thị thông tin các chuyên khoa, cơ sở y tế, bác sĩ

– Hiển thị thông tin lịch khám bệnh của từng bác sĩ

– Đặt lịch khám bệnh cho bệnh nhân

– Quản lý thông tin người dùng (bác sĩ, bệnh nhân)

– Quản lý thông tin chuyên khoa

– Quản lý lịch khám bệnh

**2.1.1 Khảo sát yêu cầu của người dùng**

Dựa trên việc khảo sát người dùng thực tế từ nhóm đối tượng đã từng sử dụng hệ thống đặt lịch khám và chưa từng sử dụng hệ thống đặt lịch khám. Kết quả thu được là hầu hết mọi người đều tập chung quan tâm đến chức năng đặt

Lịch, giá khám dịch vụ,chuyên môn của bác sĩ, cụ thể là:

* Xem thông tin chi tiết về bác sĩ
* Đặt lịch khám bệnh
* Quản lý lịch khám của bản thân
* Hủy lịch khám bệnh nếu muốn
* Xem phản hồi kết quả khám từ bác sĩ

## **2.2 Tổng quan về chức năng**

Hệ thống đặt lịch khám trực tuyến bao gồm 2 phần chính: Trang web đặt lịch khám dành cho người bệnh và Hệ thống quản lý dành cho quản trị viên và bác sĩ.

* Với đối tượng bệnh nhân
* Đăng ký tài khoản, cho phép người dùng tạo tài khoản và gửi xác thực tài khoản thông qua gmail
* Quản lý thông tin tài khoản của bản thân, cho phép người dùng sửa đổi thông tin tài khoản
* Quản lý lịch sử lịch hẹn của bản thân, cho phép người dùng xem lại lịch sử đăng ký khám, xem chi tiết từng lần khám
* Xem chi tiết phản hồi của bác sĩ, người dùng có thể xem phản hồi của bác sĩ trong từng lần khám
* Xem chi tiết hóa đơn khám bệnh và chi phí khám: Sau khi khám, quản trị viên sẽ tạo hóa đơn khám, người dùng sẽ thanh toán và có thể xem lại hóa đơn thanh toán từ lịch sử khám
* Tìm kiếm thông tin bác sĩ, xem chi tiết thông tin của từng bác sĩ
* Tìm kiếm bài viết, xem chi tiết nội dung bài viết
* Đặt lịch hẹn, cho phép người dùng chọn ngày, giờ khám và bác sĩ khám, đưa ra thông báo đặt lịch khám cho bệnh nhân
* Với đối tượng quản trị viên:
* Quản lý thông tin chuyên khoa, cho phép thêm sửa xóa thông tin các chuyên khoa
* Quản lý thông tin bác sĩ, cho phếp thêm sửa xóa thông tin bác sĩ
* Quản lý thông tin dịch vụ của bệnh viện, cho phép thêm sửa xóa thông tin, giá tiền các dịch vụ
* Quản lý thông tin lịch hẹn của bệnh nhân, cho phép quản trị viên đồng ý hoặc hủy lịch hẹn, xác thực thanh toán, tạo hóa đơn cho bệnh nhân
* Quản lý thông tin tài khoản của bệnh nhân và bác sĩ, cho phép quản trị viên tìm kiếm thông tin tài khoản hoặc khóa tài khoản.
* Với đối tượng bác sĩ:
* Quản lý lịch hẹn của bệnh nhân, cho phép bác sĩ có thể xác nhận hoặc hủy lịch hẹn, gửi thông báo đến bệnh nhân qua email, và gửi phản hồi khám về cho bệnh nhân

### **2.2.1 Biểu đồ usecase tổng quát**



Hình 1 Sơ đồ usecase tổng quát

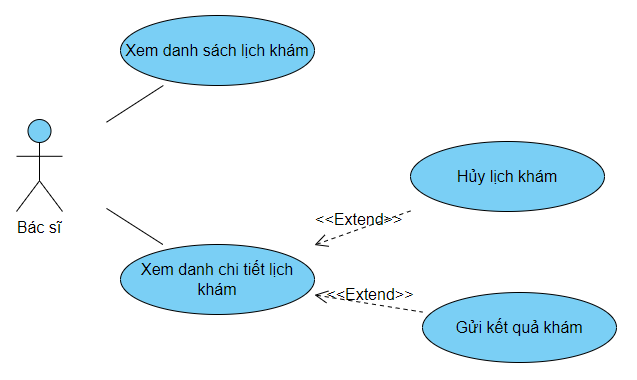
Hệ thống đặt lịch khám cơ bản gồm ba tác nhân chính là:

- Bệnh nhân, là một người dùng, tương tác trực tiếp với các tính năng trên trang web đặt lịch khám như đặt lịch, xem thông tin bác sĩ, xem lịch sử khám bệnh của bản thân

- Bác sĩ, người phụ trách quản lý lịch khám của bản thân và gửi phản hồi kết quả khám cho bệnh nhân sau khi khám xong

- Quản trị viên, đóng vai trò quản lý toàn bộ hệ thống như: quản lý người dùng, bác sĩ, cơ sở y tế, chuyên khoa, lịch khám, thanh toán,…

### **2.2.2 Biểu đồ usecase quản lý lịch khám của bác sỹ**

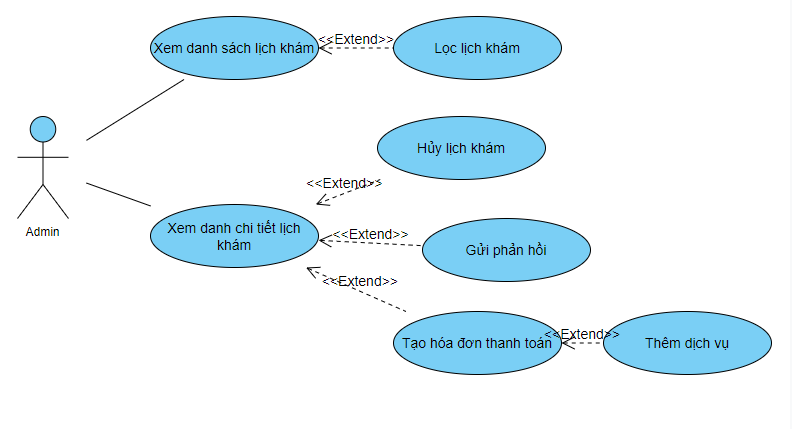


Hình 2 Biểu đồ usecase quản lý lịch khám của bác sỹ

Các chức năng thuộc use case “quản lý lịch khám của bác sỹ” dành cho tác nhân bác sĩ bao gồm các chức năng:

* Xem danh sách lịch khám
* Xem chi tiết lịch khám
* Hủy lịch khám
* Gửi phản hồi kết quả khám cho bệnh nhân

### **2.2.3 Biểu đồ usecase quản lý lịch khám của quản trị viên**

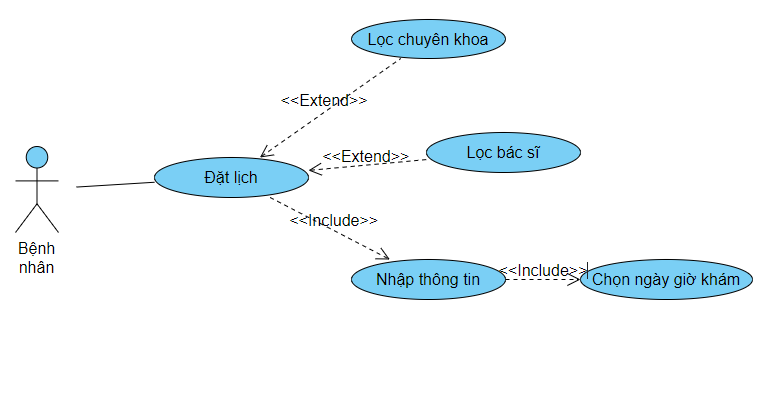


Hình 3Biểu đồ usecase quản lý lịch khám của quản trị viên

Các chức năng thuộc use case “quản lý lịch khám của quản trị viên” dành cho tác nhân quản trị viên bao gồm các chức năng:

* Xem danh sách lịch khám bệnh
* Lọc danh sách lịch khám theo ngày
* Xem chi tiết lịch khám
* Gửi phàn hồi khám
* Hủy lịch khám
* Tạo hóa đơn khám bệnh sau khi khám xong
* Thêm dịch vụ khám vào hóa đơn

### **2.2.4 Biểu đồ usecase đặt lịch**

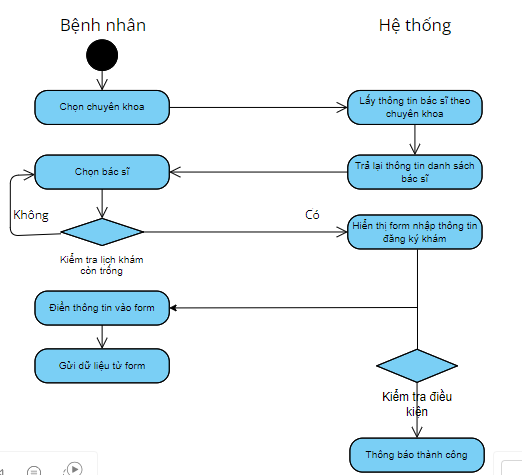


Hình 4 Biểu đồ usecase đặt lịch

Các chức năng thuộc use case “đặt lịch của bệnh nhân” dành cho tác nhân bệnh nhân viên bao gồm các chức năng:

* Lọc chuyên khoa
* Chọn bác sỹ muốn khám
* Nhập thông tin khám
* Chọn ngày giờ khám
* Xác nhận lịch khám

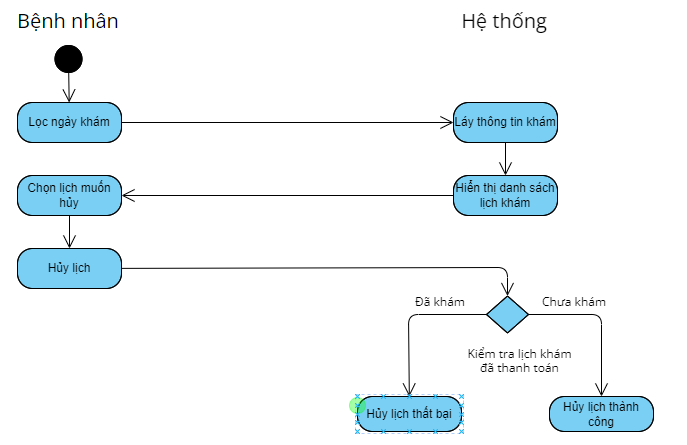
### **2.2.5 Quy trình đặt lịch khám**



Hình 5 Quy trình đặt lịch khám

Trang web có quy trình đặt lịch khám như sau: Bệnh nhân chọn chuyên khoa muốn khám, sau đó bệnh nhân chọn bác sĩ muốn khám, sau đó chọn lịch khám còn trống của bác sĩ. Sau khi chọn lịch khám, hệ thống sẽ hiển thị form đăng ký khám bệnh. Bệnh nhân nhập thông tin vào form đăng ký lịch khám và gửi yêu cầu khám, còn lại quản trị viên sẽ liên lạc lại để xác nhận

### **2.2.6 Quy trình hủy lịch khám**

****

Hình 6 Quy trình hủy lịch khám

Trang web người dùng có quy trình hủy lịch khám như sau: Bệnh nhân chọn khoảng ngày có lịch khám muốn hủy. Hệ thống hiển thị danh sách các lịch khám của bệnh nhân đó. Bệnh nhân chọn đến lịch muốn hủy và nhấn hủy. Hệ thống sẽ kiểm tra xem lịch khám này đã được khám hoặc thanh toán hay chưa, nếu lịch chưa được khám thì cho phép hủy lịch khám, nếu lịch đã khám sẽ thông báo lỗi cho bệnh nhân.

## **2.3 Đặc tả chức năng**

### **2.3.1 Đặc tả usecase đăng nhập**

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Usecase | UC001 | | |
| Tên Usecase | Đăng nhập | | |
| Mục đích | Cho phép người dùng đăng nhập vào hệ thông | | |
| Tác nhân | - Quản trị viên  - Bác sĩ  - Bệnh nhân | | |
| Tiền điều kiện | Đã có tài khoản trong hệ thống | | |
| Luồng sự kiện chính | STT | Tác nhân | Hành động |
| 1 | - Quản trị viên  - Bác sĩ  - Bệnh nhân | Nhập email và mật khẩu. Nhấn đăng nhập |
| 2 | Hệ thống | Kiểm tra các trường bắt buộc |
| 3 | Hệ thống | Kiểm tra tài khoản (email + mật khẩu) trong hệ thống |
| 4 | Hệ thống | Trả về giao diện màn hình chính |
| Luồng sự kiện thay thế | STT | Tác nhân | Hành động |
| 1 | Hệ thống | Thông báo lỗi: Chưa nhập thông tin vào 2 trường bắt buộc |
| 2 | Hệ thống | Thông báo lỗi:Đăng nhập thất bại |

Bảng 1 Đặc tả use case "Đăng nhập"

### **2.3.2 Đặc tả usecase thêm mới bác sĩ**

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Usecase | UC002 | | |
| Tên Usecase | Thêm mới bác sĩ | | |
| Mục đích | Cho phép quản trị viên thêm mới bác sĩ | | |
| Tác nhân | Quản trị viên | | |
| Tiền điều kiện | Đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Tác nhân | Hành động |
| 1 | Quản trị viên | Chọn thêm mới bác sĩ |
| 2 | Hệ thống | Hiển thị giao diện thêm mới bác sĩ |
| 3 | Quản trị viên | Nhập thông tin bác sĩ |
| 4 | Hệ thống | Kiểm tra trường bắt buộc, kiểm tra thông tin tài khoản |
| 5 | Hệ thống | Thông báo thành công |
| Luồng sự kiện thay thế | STT | Tác nhân | Hành động |
| 1 | Hệ thống | Thông báo lỗi: Chưa nhập thông tin vào các trường bắt buộc |
| 2 | Hệ thống | Thông báo lỗi:Tài khoản đã tổn tại |

Bảng 2 Đặc tả usecase thêm mới bác sĩ

### **2.3.3 Đặc tả usecase đăng ký tài khoản**

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Usecase | UC003 | | |
| Tên Usecase | Đăng ký tài khoản | | |
| Mục đích | Cho phép bệnh nhân đăng ký tài khoản mới | | |
| Tác nhân | Bệnh nhân | | |
| Tiền điều kiện |  | | |
| Luồng sự kiện chính | STT | Tác nhân | Hành động |
| 1 | Bệnh nhân | Chọn đăng ký tài khoản |
| 2 | Hệ thống | Hiển thị giao diện đăng ký |
| 3 | Bệnh nhân | Nhập thông tài khoản |
| 4 | Hệ thống | Kiểm tra trường bắt buộc  Kiểm tra email đã tồn tại chưa |
| 5 | Hệ thống | Thông báo thành công |
| Luồng sự kiện thay thế | STT | Tác nhân | Hành động |
| 1 | Hệ thống | Thông báo lỗi: Chưa nhập thông tin vào các trường bắt buộc |
| 2 | Hệ thống | Thông báo lỗi:Tài khoản đã tổn tại |

Bảng 3 Đặc tả usecase đăng ký tài khoản

### **2.3.4 Đặc tả usecase đặt lịch khám**

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Usecase | UC004 | | |
| Tên Usecase | Đặt lịch khám | | |
| Mục đích | Cho phép bệnh nhân đăng ký lịch khám | | |
| Tác nhân | Bệnh nhân | | |
| Tiền điều kiện | Đã đăng nhập tài khoản bệnh nhân | | |
| Luồng sự kiện chính | STT | Tác nhân | Hành động |
| 1 | Bệnh nhân | Chọn ngày khám, Chọn bác sĩ |
| 2 | Hệ thống | Hiển thị thời gian khám còn trống của bác sĩ |
| 3 | Bệnh nhân | Nhập thông tin, chọn giờ khám |
| 4 | Hệ thống | Kiểm tra trường bắt buộc  Kiểm tra giờ khám |
| 5 | Hệ thống | Thông báo thành công |
| Luồng sự kiện thay thế | STT | Tác nhân | Hành động |
| 1 | Hệ thống | Thông báo lỗi: Chưa nhập thông tin vào các trường bắt buộc |
| 2 | Hệ thống | Thông báo lỗi: Đã có người đặt lịch vào thời gian này |

Bảng 4 Đặc tả usecase đặt lịch khám

### **2.3.5 Đặc tả usecase hủy lịch khám**

|  |  |  |  |
| --- | --- | --- | --- |
| Mã Usecase | UC005 | | |
| Tên Usecase | Hủy lịch khám | | |
| Mục đích | Cho phép bệnh nhân hủy đăng ký lịch khám | | |
| Tác nhân | Bệnh nhân | | |
| Tiền điều kiện | Đã đăng nhập tài khoản bệnh nhân | | |
| Luồng sự kiện chính | STT | Tác nhân | Hành động |
| 1 | Bệnh nhân | Chọn lịch muốn hủy, nhấn hủy lịch |
| 2 | Hệ thống | Kiểm tra lịch đăng ký đã được khám hay chưa |
| 5 | Hệ thống | Thông báo thành công |
| Luồng sự kiện thay thế | STT | Tác nhân | Hành động |
| 1 | Hệ thống | Thông báo lỗi: Lịch khám đã hoàn thành, không thể hủy |

Bảng 5Đặc tả usecase hủy lịch khám

## **2.4 Yêu cầu phi chức năng**

### **2.4.1. Yêu cầu về bảo mật**

- Các form nhập phải được validate một cách hợp lý, tránh việc lưu sai, lưu thiếu dữ liệu sẽ bị lỗi sau này.

- Bệnh nhân chưa đăng nhập sẽ không được xem các thông tin về tài khoản, lịch sử khám, đặt lịch,…

- Thông tin tài khoản sẽ phải được mã hóa, bảo mật, mật khẩu đăng nhập cũng được mã hóa một chiều, ngay cả admin cũng không thể biết mật khẩu của tài khoản người dùng.

- Chức năng đăng ký hay quên mật khẩu đều phải xác thực thông qua mã code được gửi qua email.

- Trang quản trị thì chỉ được phép quản trị viên mới được truy cập thông qua quyền của tài khoản.

### **2.4.2. Yêu cầu về tính sử dụng**

- Giao diện đơn giản, dễ dàng sử dụng

- Chức năng chịu tải tốt khi có nhiều user cùng truy cập hệ thống

- Hệ thống phản hồi nhanh khi người dùng gửi yêu cầu

- Tương thích với nhiều loại trình duyệt, nhiều hệ điều hành

- Bảo mật thông tin tài khoản người dùng tốt

- Dữ liệu quan trọng phải được mã hóa, chống lấy cắp thông tin

### **2.4.3. Yêu cầu về tính tính thuận tiện**

- Hệ thống đưa đến cho người dùng trải nghiệm sử sụng sản phẩm một cách dễ dàng với giao diện thân thiện và các chức năng được đơn giản hóa

### **2.4.4. Yêu cầu về bảo trì hệ thống**

- Hệ thống phải dễ bảo trì các thành phần mà không ảnh hưởng đến hoạt động của toàn hệ thống. Các chức năng, thành phần phải được phân chia riêng biệt ví dụ backend và frontend được chia thành các folder riêng biệt, giao diện được chia thành các component. Độc lập với nhau để có thể dễ dàng bảo trì chức năng, thành phần này mà không ảnh hưởng đến toàn bộ hệ thống..

**Tổng kết chương**

Trong chương 2, tôi đã đưa ra sơ bộ khảo sát về các sản phẩm đặt lịch khám trực tuyến đang thịnh hành tại Việt Nam cũng như các nước trên thế giới và các tính năng cơ bản của chúng. Thông qua các khảo sát, ta có thể thấy rằng một hệ thống đặt lịch khám trực tuyến thành công cần phải đảm bảo được các yếu tố về đầu ra các thành phần và dịch vụ để đẩy nhanh quá trình xây dựng hệ thống với khả năng sử dụng linh hoạt với nhu cầu của khách hàng. Đây là tiền đề giúp tôi lựa chọn hướng tiếp cận các công nghệ thích hợp để xây dựng các dịch vụ trong hệ thống. Trong chương 3, các nội dung này sẽ được triển khai một cách chi tiết.

**CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG**

## **3.1 Hệ quản trị cơ sở dữ liệu mysql**

MySQL là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất thế giới và được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng. Vì MySQL là hệ quản trị cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, có tính khả chuyển, hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh. Với tốc độ và tính bảo mật cao, MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên internet. Người dùng có thể tải về MySQL miễn phí từ trang chủ. MySQL có nhiều phiên bản cho các hệ điều hành khác nhau: phiên bản Win32 cho các hệ điều hành dòng Windows, Linux, Mac OS X, Unix, FreeBSD, NetBSD, Novell NetWare, SGI Irix, Solaris, SunOS,..

Một số đặc trưng của MySQL:

• Dễ sử dụng: MySQL là cơ sở dữ liệu tốc độ cao, ổn định, dễ sử dụng và hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.

• Độ bảo mật cao: MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên Internet khi sở hữu nhiều nhiều tính năng bảo mật thậm chí là ở cấp cao.

• Đa tính năng: MySQL hỗ trợ rất nhiều chức năng SQL được mong chờ từ một hệ quản trị cơ sở dữ liệu quan hệ cả trực tiếp lẫn gián tiếp.

• Khả năng mở rộng và mạnh mẽ: MySQL có thể xử lý rất nhiều dữ liệu và hơn thế nữa nó có thể được mở rộng nếu cần thiết.

• Nhanh chóng: Việc đưa ra một số tiêu chuẩn cho phép MySQL để làm việc rất hiệu quả và tiết kiệm chi phí, do đó nó làm tăng tốc độ thực thi.

## **3.2 Ngôn ngữ lập trình java**

- Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động. Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak.

- Không như một vài ngôn ngữ khác chỉ chạy được ở một vài môi trường nhất định thì một chương trình viết bằng ngôn ngữ Java có thể chạy tốt ở nhiều môi trường khác nhau. Gọi là khả năng “cross-platform”. Khả năng độc lập phần cứng và hệ điều hành được thể hiện ở 2 cấp độ là cấp độ mã nguồn và cấp độ nhị phân.

- Java là một ngôn ngữ lập trình mạnh mẽ, cơ động, có thể lập trình được cả website, app, AI hay lập trình nhúng,…

Java có rất nhiều ưu điểm vượt trội như:

* Là một ngôn ngữ lập trình yêu cầu chặt chẽ về kiểu dữ liệu.
* Java kiểm soát chặt chẽ việc truy nhập đến mảng, chuỗi. Không cho phép sử dụng các kỹ thuật tràn. Do đó các truy nhập sẽ không vượt quá kích thước của mảng hoặc chuỗi.
* Quá trình cấp phát và giải phóng bộ nhớ được thực hiện tự động
* Cơ chế xử lý lỗi giúp việc xử lý và phục hồi lỗi dễ dàng hơn

Ngoài ra, java cũng có một vài điểm hạn chế so với các ngôn ngữ khác:

Cũng như các ngôn ngữ thông dịch khác, quá trình chạy các đoạn mã Java là chậm hơn các ngôn ngữ biên dịch khác (tuy nhiên vẫn ở trong một mức chấp nhận được)

## **3.3 Spring boot framework**

Spring là một Framework phát triển các ứng dụng Java được sử dụng bởi hàng triệu lập trình viên. Nó giúp tạo các ứng dụng có hiệu năng cao, dễ kiểm thử, sử dụng lại code

Spring là một mã nguồn mở, được phát triển, chia sẻ và có cộng đồng người dùng rất lơn. Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là: Dependency Injection và Aspect Oriented Programming. Những tính năng core (cốt lõi) của Spring có thể được sử dụng để phát triển Java Desktop, ứng dụng mobile, Java Web. Mục tiêu chính của Spring là giúp phát triển các ứng dụng J2EE một cách dễ dàng hơn dựa trên mô hình sử dụng POJO (Plain Old Java Object)

Spring được chia làm nhiều module khác nhau, tùy theo mục đích phát triển ứng dụng mà ta dùng 1 trong các module đó.

Các lợi ích của spring framework:

* Spring cho phép lập trình viên sử dụng POJOs. Việc sử dụng POJOs giúp chúng ta không phải làm việc với EJB, ứng dụng, các luồng chạy, cấu hình… đơn giản hơn rất nhiều.
* Spring được tổ chức theo kiểu mô đun. Số lượng các gói và các lớp khá nhiều, nhưng chúng ta chỉ cần quan tâm đến những gì chúng ta cần và không cần quan tâm đến phần còn lại.
* Spring hỗ trợ sử dụng khá nhiều công nghệ như ORM Framework, các logging framework, JEE, các thư viện tạo lịch trình (Quartz và JDK timer)…
* Module Web của Spring được thiết kế theo mô hình MVC nên nó cung cấp đầy đủ các tính năng giúp thay thế các web framework khác như Struts.
* Spring cung cấp một dự án là Spring Security Cung cấp các cơ chế xác thực (authentication) và phân quyền (authorization) cho project của chúng ta. Nó giúp chúng ta phân quyền cho người dùng và xác thực người dùng truy cập một cách bảo mật tuyệt vời
* Spring Boot: là một framework giúp chúng ta phát triển ứng dụng web cũng như chạy ứng dụng một cách nhanh chóng.

## **3.4 Hibernate**

Hibernate là một thư viện ORM (Object Relational Mapping) mã nguồn mở giúp lập trình viên viết ứng dụng Java có thể map các objects (pojo) với hệ quản trị cơ sở dữ liệu quan hệ, và hỗ trợ thực hiện các khái niệm lập trình hướng đối tượng với cớ dữ liệu quan hệ.

Chúng ta sẽ không phải làm việc trực tiếp với database như khi dùng jdbc mà chúng ta sẽ làm việc với database thông qua class đại diện cho các table trong database. Trước khi có hibernate, chúng ta phải sử dụng jdbc để viết câu lệnh truy vấn rất dài dòng và hiệu suất kém vì phải đóng mở connection liên tục. Khi hibernate được sử dụng, các lập trình viên sẽ không phải viết các câu truy vấn dài dòng nữa và không cần phải đóng mở connection nhiều lần

## **3.5 HTML**

HTML là từ viết tắt của Hypertext Markup Language ( Hay còn gọi là “Ngôn ngữ đánh dấu siêu văn bản”), là ngôn ngữ được sử dụng rộng rãi nhất để viết các trang Web.

HTML là ngôn ngữ đánh dấu bằng thẻ (Markup Language). Nghĩa là, sử dụng HTML để đánh dấu một tài liệu text bằng các thẻ (tag) để nói cho trình duyệt Web cách để cấu trúc nó để hiển thị ra màn hình.

Có thể tạm hiển, HTML giúp chúng ta tạo ra “khung xương” của một website.

Ưu điểm của HTML:

- Có nhiều tài nguyên hỗ trợ với cộng đồng người dùng vô cùng lớn

- Có thể hoạt động mượt mà trên hầu hết mọi trình duyệt hiện nay

- Các markup sử dụng trong HTML thường ngắn gọn, có độ đồng nhất cao

- Sử dụng mã nguồn mở, hoàn toàn miễn phí

- HTML là chuẩn web được vận hành bởi W3C

- Dễ dàng để tích hợp với các loại ngôn ngữ backend (ví dụ như: PHP, Node.js, Java,…)

## **3.6 CSS**

CSS là từ viết tắt của Cascading Style Sheet, tức là ngôn ngữ dùng để định dạng các phần tử do ngôn ngữ đánh dấu (như HTML) tạo ra. Hay nói cách khác, HTML định dạng các phần tử có trên website như tiêu đề, bảng biểu, tạo đoạn văn bản,… còn CSS giúp cho các phần tử của HTML trở nên phong phú, nổi bật hơn bằng cách trang trí, đổi màu chữ, thêm màu sắc trang hay thay đổi cấu trúc trang,…

Có thể hiểu, html là thứ giúp chúng ta tạo nên “khung xương” của website thì css là thứ giúp chúng ta tạo nên màu sắc, trang điểm cho website

## **3.6 Javascript**

JavaScript là ngôn ngữ lập trình website phổ biến hiện nay, nó được tích hợp và nhúng vào HTML giúp website trở nên sống động hơn. JavaScript đóng vai trò như là một phần của trang web, thực thi cho phép Client-side script từ phía người dùng cũng như phía máy chủ (Nodejs) tạo ra các trang web động

JavaScript là một ngôn ngữ lập trình thông dịch với khả năng hướng đến đối tượng. Là một trong 3 ngôn ngữ chính trong lập trình web và có mối liên hệ lẫn nhau để xây dựng một website sống động, chuyên nghiệp

Nhiệm vụ của Javascript là xử lý những đối tượng HTML trên trình duyệt. Nó có thể can thiệp với các hành động như thêm / xóa / sửa các thuộc tính CSS và các thẻ HTML một cách dễ dàng. Hay nói cách khác, Javascript là một ngôn ngữ lập trình trên trình duyệt ở phía client.

Javascript có thể lấy dữ liệu từ server thông qua các api trả ra dưới dạng chuỗi json và trả ra giao diện người dùng hoặc có thể lấy dữ liệu từ form nhập vào trên giao diện và gọi đến api để lưu dữ liệu vào server.

## **3.7 Công cụ sử dụng**

### **3.7.1. Intellij IDE**

Intellij IDEA là một trình IDE dùng để lập trình Java (nó cũng được sử dụng để lập trình một số ngôn ngữ khác như Node.js, python…)

Nhìn chung Intellij IDEA khá giống với Eclipse vì nó chủ yếu dùng cho Java nhưng vẫn có thể hỗ trợ các ngôn ngữ khác và có rất nhiều các plugin hỗ trợ.

Intellij IDEA có 2 bản là bản miễn phí (community) và bản trả phí (ultimate). Bản trả phí thì hỗ trợ thêm JavaScript, TypeScript, các plugin GWT, Vaadin… check các đoạn code trùng lặp…

### **3.7.2 Laragon**

Là một chương trình cung cấp môi trường WAMP cực kỳ mạnh mẽ và hoàn toàn miễn phí, với phần mềm này, bạn có thể tự tạo một web server ngay trên máy tính cá nhân mà không cần phải đăng ký một web server.

Là chương trình giúp tạo web server được dùng trên nhiều hệ điều hành như Windows, Linux, MacOS, Cross- platform và Soliaris

Laragom cung cấp phiên bản mysql trên web, giúp giao diện dễ sử dụng hơn so với mysql workbench

### **3.7.3 Postman**

Postman là một công cụ cho phép chúng ta thao tác với API, phổ biến nhất là REST. Postman hiện là một trong những công cụ phổ biến nhất được sử dụng trong thử nghiệm các API. Với Postman, ta có thể gọi Rest API mà không cần viết dòng code nào.

Postman hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, PATCH, DELETE, …). Bên cạnh đó, nó còn cho phép lưu lại lịch sử các lần request, rất tiện cho việc sử dụng lại khi cần.

Postman không chỉ cho phép thực hiện các thao tác với rest api mà còn có thể thao tác với websocket, đây là chức năng mới trong bản beta của postman.

**CHƯƠNG 4. THỰC NGHIỆM VÀ ĐÁNH GIÁ**

## **4.1 Thiết kế kiến trúc rest api**

### **4.1.1. Khái niệm về rest api**

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho ứng dựng web để quản lí resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile, webserice...) khác nhau có thể giao tiếp với nhau

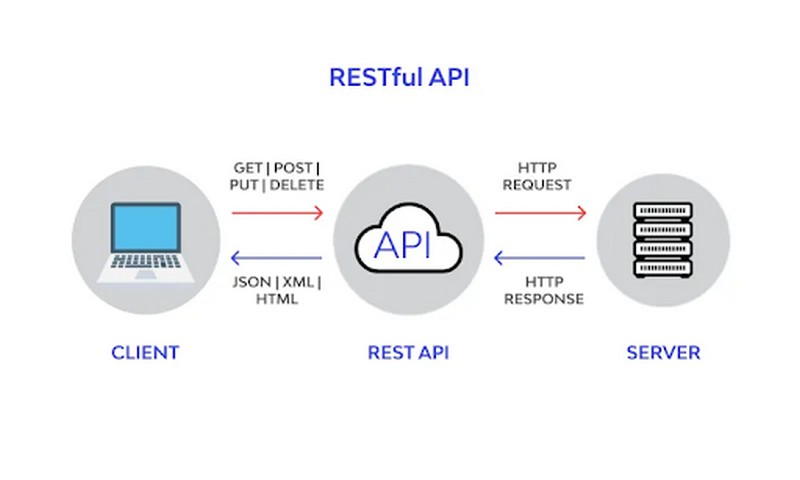
API là viết tắt của Application Programming Interface (Giao diện lập trình ứng dụng) là một tập hợp các định nghĩa, giao thức, và công cụ chương trình con để xây dựng để xây dựng phần mềm và ứng dụng. Hiểu đơn giản API là một trung gian phần mềm cho phép hai ứng dụng giao tiếp với nhau. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML. Với API, bạn có thể tiếp cận, truy xuất dữ liệu từ máy chủ thể hiện chúng trên ứng dụng phần mềm hoặc website của mình một cách dễ dàng hơn. Tính tới nay, API đã phát triển với nhiều loại ứng dụng và phần mềm khác nhau. Thế hệ mới nhất của web/ app API có thể ứng dụng được ở mọi hệ thống từ cơ sở dữ liệu, hệ điều hành, hệ thống nền web, thư viện hay thậm chí là phần cứng máy tính.

REST là một kiểu kiến trúc được sử dụng trong việc giao tiếp giữa các máy tính (máy tính cá nhân và máy chủ của trang web) trong việc quản lý các tài nguyên trên internet.

REST được sử dụng rất nhiều trong việc phát triển các ứng dụng Web Services sử dụng giao thức HTTP trong giao tiếp thông qua mạng internet. Các ứng dụng sử dụng kiến trúc REST này thì sẽ được gọi là ứng dụng phát triển theo kiểu RESTful. Một RESTful API đơn giản là một API tuân thủ các nguyên tắc và ràng buộc của REST.

REST bao gồm 3 bộ phận cơ bản như: bộ máy chủ ngoài chứa dữ liệu (external server), máy chủ API và máy chủ khách (client). Trong đó, máy khách có thể là bất cứ thứ gì, ứng dụng trên nền web, thư viện hoặc thậm chí là các phần mềm khác nhau của một phần mềm máy tính.

REST cho phép các máy khách truy cập máy chủ API và thực hiện các lệnh lấy về, chỉnh sửa hay xóa dữ liệu từ external server. Các lập trình viên có thể thoải mái truy xuất, chỉnh sửa dữ liệu từ máy chủ mà không cần biết hệ thống hoạt động như thế nào.



Hình 7 Mô hình rest api

### **4.1.2. Cách hoạt động của REST**

Khi làm việc với server sẽ gồm 4 hoạt động thiết yếu là:

* Lấy dữ liệu ở một định dạng nào đó (JSON)
* Tạo mới dữ liệu
* Cập nhật dữ liệu
* Xóa dữ liệu

REST hoạt động chủ yếu dựa vào giao thức HTTP. Mỗi trong 4 hoạt động cơ bản trên sẽ sử dụng những phương thức HTTP riêng (HTTP method):

* POST (CREATE) : Tạo mới một Resource.
* GET (READ) : Trả về một Resource hoặc một danh sách Resource.
* PUT (UPDATE) : Cập nhật, thay thế thông tin cho Resource.
* DELETE (DELETE) : Xoá một Resource.

REST là một kiến trúc thống nhất giúp thiết kế các website để có thể dễ dàng quản lý các tài nguyên. Nó không phải là một quy luật buộc bạn phải tuân theo mà đơn giản là một kiến trúc được đề xuất ra và kiến trúc này hiện đang được sử dụng rất phổ biến vì tính đơn giản, dễ hiểu và rất ưu việt của nó. Với các ứng dụng web được thiết kế sử dụng RESTful, ban có thể dễ dàng biết được URL và HTTP method để quản lý một resource.

### **4.1.3 Ưu điểm của REST**

REST cũng có ưu điểm khi sử dụng giao thức stateless (không trạng thái). Hệ thống này không sử dụng session, cookie, không cần biết những thông tin đó trong mỗi lần request đến máy chủ ngoài. Điều này giúp REST giảm tải cho máy chủ ngoài, nâng cao hiệu suất làm việc.

Tính khả biến: với các hệ thống cần thay đổi các tài nguyên liên tục, sử dụng REST với việc tạo request đơn giản sẽ giúp mọi chuyện trở nên đơn giản hơn.

Tính mở rộng: các hệ thống REST có khả năng mở rộng rất cao nhờ sự tách biệt giữa các thành phần và các quy ước giao tiếp được quy định sẵn.

Tính linh hoạt: việc chuẩn hoá interface giúp hệ thống trở nên linh hoạt hơn, có thể sử dụng cho cho nhiều nền tảng khác nhau, mobile, web,...

Trong sáng: trong giao tiếp giữa các thành phần, các request trở nên rất rõ ràng, dễ hiểu.

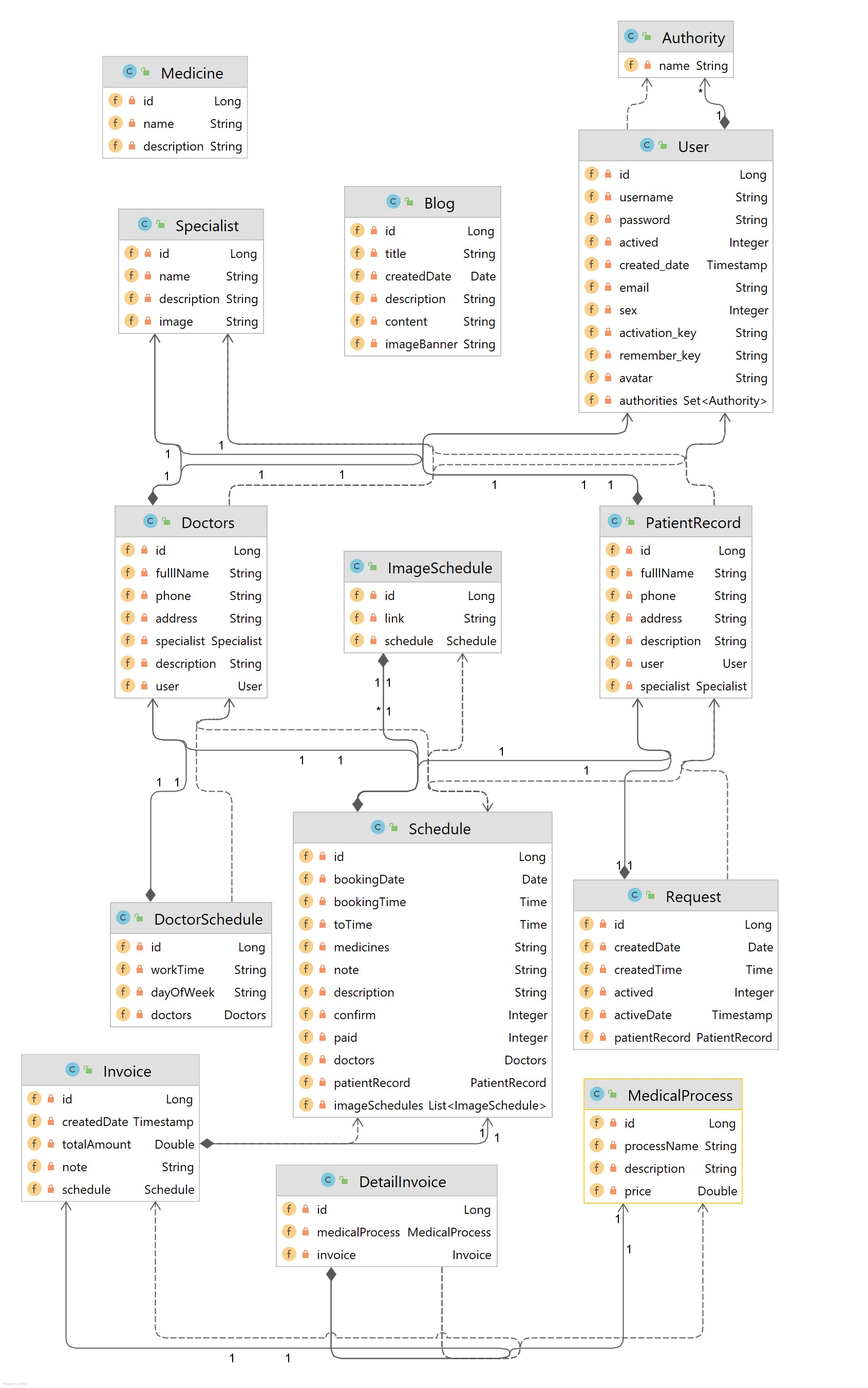
### **4.1.4 Nhược điểm của REST**

REST Chỉ hoạt động trên các giao thức HTTP

Ưu điểm của việc sử dụng các cấu trúc HTTP lại cũng chính là một hạn chế. Bởi các hạn chế của HTTP cũng trở thành các hạn chế của kiến trúc REST. Ví dụ: HTTP không lưu trữ thông tin về trạng thái giữa các chu kỳ khi phản hồi yêu cầu, và như vậy các ứng dụng dựa trên REST rơi vào tình trạng không rõ trạng thái và mọi tác vụ quản lý trạng thái phải được máy khách thực hiện.

## **4.2 Thiết kế chi tiết**

### **4.2.1 Sơ đồ lớp**

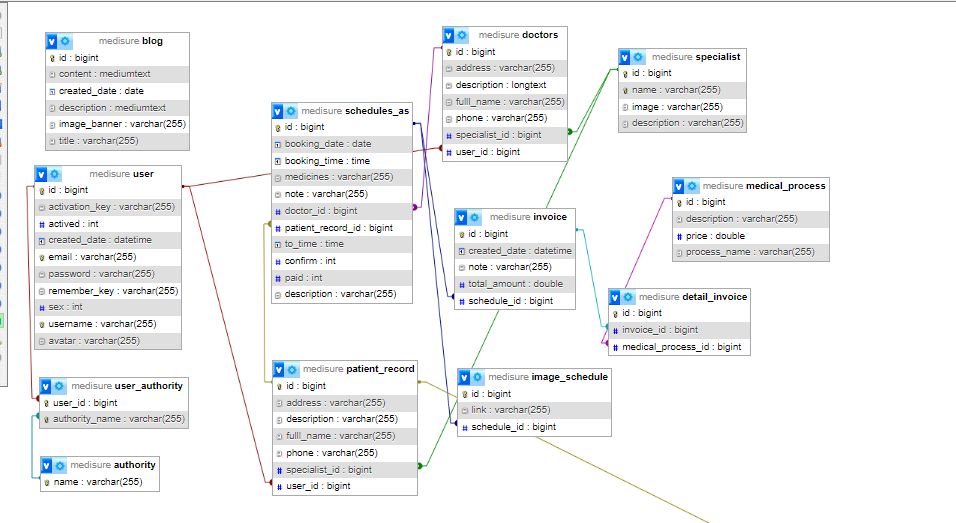


Hình 8 Thiết kế sơ đồ lớp

Hình 8 của hệ thống chịu trách nhiệm định nghĩa các lớp thực thể, mô hình hoá dữ liệu giao tiếp giữa client và server. Các lớp được minh họa trong sơ đồ là các lớp được thiết kế để phù hợp với nhu cầu chung nhất. Trong đó:

* Authority: Chứa các quyền của hệ thống
* Blog: Chứa thông tin các bài viết
* DetailInvoice: chứa chi tiết đơn khám
* Invoice: Đơn khám
* Doctor: Chứa thông tin bác sĩ
* ImageSchedule: Chứa ảnh chụp kết quả khám
* MedicalProcess: Chứa thông tin dịch vụ
* Medicine: Chứa thông tin thuốc
* Schedule: Chứa thông tin đặt lịch của bệnh nhân
* Specialist: Chứa thông tin các chuyên khoa
* User: Chứa thông tin tài khoản

### **4.2.2 Thiết kế cơ sở dữ liệu**



Hình 9 Thiết kế cơ sở dữ liệu

Chi tiết dữ liệu trong bảng

**Bảng authority**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| name | Varchar(255) | Tên quyền |

Bảng 6 bảng authority

**Bảng blog**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| content | mediumtext | Nội dung bài viết |
| created\_date | date | Ngày tạo bài viết |
| description | mediumtext | Mô tả bài viết |
| image\_banner | varchar(255) | ảnh nền bài viết |
| title | varchar(255) | Tiêu đề bài viết |

Bảng 7 Bảng blog

**Bảng detail\_invoice**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| invoice\_id | bigint | Liên kết bảng hóa đơn |
| medical\_process\_id | bigint | Liên kết bảng dịch vụ y tế |

Bảng 8 Bảng detail\_invoice

**Bảng doctor**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| address | varchar(255) | Địa chỉ |
| description | longtext | Mô tả thông tin bác sĩ |
| fulll\_name | varchar(255) | Họ tên bác sĩ |
| phone | varchar(255) | Số điện thoại |
| specialist\_id | bigint | Chuyên khoa |
| user\_id | bigint | Liên kết bảng user |

Bảng 9 Bảng doctor

**Bảng image\_schedule**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| link | varchar(255) | Link file ảnh |
| schedule\_id | bigint | Liên kết bảng schedule |

Bảng 10 Bảng image\_schedule

**Bảng invoice**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| created\_date | datetime | Ngày tạo |
| note | varchar(255) | Ghi chú |
| total\_amount | double | Tổng tiền |
| schedule\_id | bigint | Liên kết bảng schedule |

Bảng 11 Bảng invoice

**Bảng medical\_process**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| description | varchar(255) | Mô tả dịch vụ |
| price | double | Giá tiền |
| process\_name | varchar(255) | Tên dịch vụ |

Bảng 12 medical\_process

**Bảng medicine**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| description | varchar(255) | Mô tả thuốc |
| name | varchar(255) | Tên thuốc |

Bảng 13 Bảng medicine

**Bảng schedule\_as**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| booking\_date | date | Ngày khám |
| booking\_time | time | Giờ khám |
| medicines | varchar(255) | Đơn thuốc |
| note | varchar(255) | Ghi chú của bác sĩ |
| doctor\_id | bigint | Bác sĩ khám |
| patient\_record\_id | bigint | Hồ sơ bệnh án |
| to\_time | time | Giờ kết thúc khám |
| confirm | int | Xác nhận khám của bác sĩ |
| paid | int | Trạng thái thanh toán |
| description | varchar(255) | Mô tả khám |

Bảng 14 Bảng schedule\_as

**Bảng specialist**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| name | varchar(255) | Tên chuyên khoa |
| description | varchar(255) | Mô tả |
| image | varchar(255) | ảnh chuyên khoa |

Bảng 15 Bảng specialist

**Bảng user**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| id | bigint | Khóa chính |
| activation\_key | int | Khóa kích hoạt |
| actived | time | Trạng thái tài khoản |
| created\_date | datetime | Ngày tạo |
| email | varchar(255) | Email |
| password | varchar(255) | Mật khẩu |
| remember\_key | varchar(255) | Khóa xác nhận quên mật khẩu |
| sex | int | Giới tính |
| username | varchar(255) | Tên đăng nhập |
| avatar | varchar(255) | Ảnh đại diện |

Bảng 16 Bảng user\

**Bảng user\_authority**

|  |  |  |
| --- | --- | --- |
| Tên cột | Kiểu dữ liệu | Ý nghĩa |
| user\_id | bigint | Liên kết bảng user |
| authority\_name | varchar(255) | Liên kết bảng authority |

Bảng 17 Bảng user\_authority

**CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **5.1 kết luận**

Sau quá trình tìm hiểu và phân tích, em đã xây dựng một Hệ thống đặt lịch khám trực tuyến cho đồ án tốt nghiệp của mình. Sử dụng những kỹ năng lập trình và kiến thức đã học được, như ngôn ngữ JavaScript, Html, Java Spring boot và cơ sở dữ liệu MySQL, tôi đã tạo ra một hệ thống hoàn chỉnh.

Hệ thống này có thể mang lại nhiều lợi ích cho người dùng, cho phép họ dễ dàng xem thông tin về các chuyên khoa, bác sĩ. Đặc biệt, người dùng có thể đặt lịch khám trực tuyến một cách trực quan và nhanh chóng, giúp tiết kiệm thời gian và công sức. Bên cạnh đó, hệ thống cung cấp tính năng cho phép bác hủy lịch khám khi cần thiết và gửi kết quả phản hồi cho bệnh nhân sau khi khám xong. Giao diện của hệ thống được thiết kế đẹp mắt, đơn giản, thân thiện và dễ sử dụng đối với người dùng.

Việc áp dụng các kỹ năng lập trình và công nghệ hiện đại như Java spring boot và MySQL đã giúp tôi tạo ra một hệ thống đáng tin cậy và hiệu quả. Tôi đã sử dụng khả năng xử lý dữ liệu và logic của spring framework để xây dựng các API hỗ trợ việc đặt lịch khám và quản lý lịch khám. Cùng với đó, sử dụng MySQL, tôi đã lưu trữ và truy vấn dữ liệu một cách hiệu quả để đảm bảo hệ thống hoạt động mượt mà và đáng tin cậy.

Ngoài ra, việc thực hiện đồ án đã giúp tôi áp dụng và rèn luyện kỹ năng lập trình, quản lý dự án và kiểm thử. Tôi đã học cách xây dựng một dự án phức tạp từ đầu đến cuối, từ việc phân tích yêu cầu, thiết kế hệ thống, triển khai và kiểm thử. Điều này đã giúp tôi phát triển khả năng làm việc độc lập, quản lý thời gian và xử lý các vấn đề phát sinh trong quá trình phát triển.

Tuy nhiên, như mọi dự án phát triển, hệ thống của tôi cũng còn một số thiếu sót và khả năng cải thiện. Trong tương lai, tôi sẽ tiếp tục nâng cấp và phát triển hệ thống để đáp ứng tốt hơn các yêu cầu của người dùng. Tôi sẽ tiếp tục học hỏi và ápdụng những kỹ thuật mới và tiến bộ để cải thiện hiệu suất và tính năng của hệ thống. Tôi sẽ tìm cách tối ưu hóa mã nguồn và cải thiện giao diện người dùng để mang lại trải nghiệm tốt nhất cho người dùng.

Tôi tự tin rằng kiến thức và kỹ năng mà tôi đã học được từ nó sẽ tiếp tục đóng góp cho sự phát triển của tôi trong lĩnh vực phát triển phần mềm.

## **5.2 Hướng phát triển**

Đây là môt đề tài không phải mới nhưng có giá trị thực tiễn cao. Tuy nhiên, do giới hạn về mặt thời gian, hệ thống vẫn còn một số điểm hạn chế và thiếu sót về mặt chức năng. Để đề tài này thêm phóng phú và mang tính thực tế cao hơn, có khả năng ứng dụng cao hơn thì đề tài cần bổ sung thêm nhiều tính năng hơn nữa.

Hướng phát triển thêm các chức năng khác:

Về chức năng thông báo cho người bệnh khi bác sĩ hủy lịch khám hay phản hồi kết quả khám sẽ có thêm thông báo realtime qua Zalo hoặc trực tiếp qua firebase về điện thoại, giúp người dùng có thể nhận được phản hồi theo thời gian thực

Thêm chức năng về chatbot, giúp bệnh nhân có thể nhắn tin trực tiếp đến cơ sở khám để có thể được tư vấn nhanh chóng về bệnh cũng như dịch vụ tại bệnh viện

Tích hợp thanh toán online qua momo hoặc cổng thanh toán bất kì để bệnh nhân có thể thanh toán tiền khám online

**CHƯƠNG 6. TÀI LIỆU THAM KHẢO**

1. Boyarsky & Scott Selikoff. 2005. oracle certified associate java se 8 programmer, Jeanne

2. Phillip Webb, Dave Syer, Josh Long, Stéphane Nicoll, Rob Winch, Andy Wilkinson, Marcel Overdijk, Christian Dupuis, Sébastien Deleuze, Michael Simons, Vedran Pavić, Jay Bryant, Madhura Bhave. 2012-2019. Spring Boot Reference Documentation

3. Venkatasubramaniam Iyer, Elizabeth Hanes Perry, Brian Wright, Thomas Pfaeffle. 2010. Oracle Database JDBC Developer’s Guide and Reference 10g Release 2 (10.2)

4. Donald Bell. 2010. UML basics. Part II: The activity diagram. The activity diagram's purpose. by Donald Bell IBM Global Services

5. Oliver Gierke, Thomas Darimont, Christoph Strobl, Mark Paluch. 2017. Spring Data JPA - Reference Documentation

Các trang web tham khảo

<https://www.baeldung.com/spring-boot>

<https://www.baeldung.com/java-config-spring-security>

<https://www.baeldung.com/spring-security-oauth-jwt>

<https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch>

<https://www.baeldung.com/thymeleaf-in-spring-mvc>