|  |
| --- |
| **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **---------------------------------------** |
| ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC |
| *Ngành Công Nghệ Thông Tin* |
| **XÂY DỰNG WEBSITE ĐẶT LỊCH KHÁM BỆNH BOOKING CARE SỬ DỤNG REACTJS, NODEJS, MYSQL VÀ REDUX**  **CBHD: ThS. Đoàn Văn Trung** |
| **Sinh viên: Lưu Thế Minh** |
| **Mã số sinh viên: 2018603624** |
| Hà Nội - 2022 |

**MỤC LỤC**

[LỜI CẢM ƠN 5](#_Toc103977770)

[MỞ ĐẦU 6](#_Toc103977771)

[1. Lý do chọn đề tài 6](#_Toc103977772)

[2. Đối tượng, mục tiêu và phạm vi nghiên cứu 7](#_Toc103977773)

[3. Ý nghĩa khoa học và thực tiễn của đề tài 8](#_Toc103977774)

[Chương 1. TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU 10](#_Toc103977775)

[1.1 Giới thiệu chung 10](#_Toc103977776)

[1.2 Cơ sở lý thuyết 10](#_Toc103977777)

[**1.2.1. Tổng quan về đặt lịch khám bệnh trực tuyến 10**](#_Toc103977778)

[1.2.1.1 Khái niệm về đặt lịch khám bệnh trực tuyến 10](#_Toc103977779)

[1.2.1.2 Đặc điểm của đặt lịch khám bệnh trực tuyến 11](#_Toc103977780)

[1.2.1.3 Lợi ích của thương mại điện tử đối với doanh nghiệp: 11](#_Toc103977781)

[**1.2.2** **Giới thiệu về các công cụ lập trình** 13](#_Toc103977782)

[1.2.2.1 Công cụ Visual Studio Code Giới thiệu Visual Studio Code: 13](#_Toc103977783)

[1.2.2.2 Công cụ XAMPP 14](#_Toc103977784)

[1.2.2.3 Tổng quan về mô hình MVC 15](#_Toc103977785)

[1.2.2.4 Thiết kế giao diện website với thư viện Reactjs và Redux 17](#_Toc103977786)

[1.2.2.5 Tổng Quan Về API RESTful 20](#_Toc103977787)

[1.2.2.6 Lập trình web với Nodejs, Express 21](#_Toc103977788)

[Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 24](#_Toc103977789)

[2.1 Khảo sát bài toán 24](#_Toc103977790)

[2.2 Giới thiệu về hệ thống 24](#_Toc103977791)

[2.3 Phân tích yêu cầu 24](#_Toc103977792)

[2.3.1 Về hệ thống 24](#_Toc103977793)

[2.3.2 Về người sử dụng 24](#_Toc103977794)

[2.3.3 Yêu cầu về chức năng 25](#_Toc103977795)

[2.4 Thiết kế hệ thống 25](#_Toc103977796)

[2.4.1 Mô tả use case Đăng nhập 28](#_Toc103977797)

[2.4.2 Mô tả use case Quản lý tài khoản 29](#_Toc103977798)

[2.4.3 Mô tả use case Thêm thông tin bác sĩ 31](#_Toc103977799)

[2.4.4 Mô tả usecase Quản lý chuyên khoa 32](#_Toc103977800)

[2.4.5 Mô tả usecase Quản lý kế hoạch khám bệnh 34](#_Toc103977801)

[2.4.6 Mô tả usecase Quản lý bệnh nhân 35](#_Toc103977802)

[2.4.7 Mô tả usecase Xem chi tiết phòng khám 37](#_Toc103977803)

[2.4.8 Mô tả usecase Xem chi tiết chuyên khoa 38](#_Toc103977804)

[2.4.9 Mô tả usecase Nhận email xác nhận lịch khám 40](#_Toc103977805)

[2.4.10 Mô tả usecase Đặt lịch khám bệnh 41](#_Toc103977806)

[2.4.11 Mô tả usecase Tìm kiếm bác sĩ 42](#_Toc103977807)

[2.4.12 Mô tả usecase Xem thông tin bác sĩ 44](#_Toc103977808)

[2.5 Thiết kế giao diện 46](#_Toc103977809)

[2.5.1 Màn hình Đăng nhập 46](#_Toc103977810)

[2.5.2 Màn hình thông tin bác sĩ 47](#_Toc103977811)

[2.5.3 Màn hình quản lý người dùng 48](#_Toc103977812)

[2.5.4 Màn hình Tạo thêm thông tin bác sĩ 49](#_Toc103977813)

[2.5.5 Màn hình Quản Lý Chuyên Khoa 49](#_Toc103977814)

[2.5.6 Màn hình Xem chi tiết sản phẩm 50](#_Toc103977815)

[2.5.7 Màn hình Quản lý bệnh nhân khám bệnh 50](#_Toc103977816)

[2.5.8 Màn hình Tìm Kiếm Bác Sĩ 51](#_Toc103977817)

[2.5.9 Màn hình Chi tiết chuyên khoa 52](#_Toc103977818)

[2.6 Thiết kế cơ sở dữ liệu 53](#_Toc103977819)

[2.6.1 Bảng All Code 54](#_Toc103977820)

[2.6.2 Bảng Bookings 54](#_Toc103977821)

[2.6.3 Bảng Clinics 55](#_Toc103977822)

[2.6.4 Bảng Doctor Infor 55](#_Toc103977823)

[2.6.5 Bảng Markdown 56](#_Toc103977824)

[2.6.6 Bảng Schedules 57](#_Toc103977825)

[2.6.7 Bảng Specialty 57](#_Toc103977826)

[2.6.8 Bảng Users 58](#_Toc103977827)

[Chương 3. KẾT QUẢ 59](#_Toc103977828)

[3.1 Kết quả thu được 59](#_Toc103977829)

[1. Kết quả đạt được 61](#_Toc103977830)

[2. Hạn chế của đề tài 61](#_Toc103977831)

[3. Hướng phát triển đề tài 61](#_Toc103977832)

[TÀI LIỆU THAM KHẢO 63](#_Toc103977833)

**DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **Stt** | **Viết tắt** | **Dịch nghĩa** |
| 1 | MySQL | My Structured Query Language |
| 2 | MVC | Model-View-Controller |
| 3 | RDBMS | Relational Database Management  System |
| 4 | ORDBMS | Object – relational database management system |
| 5 | URL | Uniform Resource Locator |
| 6 | CSDL | Cơ sở dữ liệu |

**DANH MỤC CÁC BẢNG BIỂU**

[Bảng 2. 1.Mô tả usecase Đăng nhập 25](#_heading=h.vx1227)

[Bảng 2. 2. Mô tả use case Quản lý tài khoản 27](#_heading=h.1v1yuxt)

[Bảng 2. 3. Mô tả use case Thêm thông tin bác sĩ 28](#_heading=h.2u6wntf)

[Bảng 2. 4. Mô tả use case Quản lý chuyên khoa 30](#_heading=h.3tbugp1)

[Bảng 2. 5. Mô tả use case Thêm kế hoạch khám bệnh 32](#_heading=h.nmf14n)

[Bảng 2. 6. Mô tả use case Quản lý bệnh nhân 34](#_heading=h.46r0co2)

[Bảng 2. 7. Mô tả use case Xem chi tiết phòng khám 35](#_heading=h.111kx3o)

[Bảng 2. 8. Mô tả use case Xem chi tiết chuyên khoa 37](#_heading=h.206ipza)

[Bảng 2. 9. Mô tả ưu case Nhận email xác nhận lịch khám 38](#_heading=h.1egqt2p)

[Bảng 2. 10. Mô tả use case Đặt lịch khám bệnh 40](#_heading=h.2dlolyb)

[Bảng 2. 11. Mô tả use case Xem thông tin bác sĩ 43](#_heading=h.1rvwp1q)

**DANH MỤC HÌNH VẼ, ĐỒ THỊ**

[Hình 2. 1.Biểu đồ lớp của hệ thống 24](#_heading=h.1hmsyys)

[Hình 2. 2.Biểu đồ của hệ thống 25](#_heading=h.41mghml)

[Hình 2. 2..Biểu đồ trình tự đăng nhập](about:blank) 27

[Hình 2. 4.Biểu đồ trình tự quản lý tài khoản](about:blank) 29

[Hình 2. 5. Biểu đồ trình tự hóa thêm thông tin bác sĩ](about:blank) 31

[Hình 2.6. Biểu đồ trình tự hóa quản lý chuyên khoa](about:blank) 33

[Hình 2 .7. Thêm kế họạch khám](about:blank) 35

[Hình 2. 9. Biểu đồ trình tự hóa xem chi tiết phòng khám](about:blank) 39

[Hình 2. 9. Biểu đồ tuần tự quy trình xem chi tiết chuyên khoa phòng khám 41](#_heading=h.4k668n3)

[Hình 2. 11. Biều đồ tuần tự quy trình xác nhận khám](about:blank) 42

[Hình 2. 13. Biểu đồ tuần tự tìm kiếm bác sĩ](about:blank) 46

[Hình 2. 15. Thiết kế màn hình đăng nhập 49](#_heading=h.1664s55)

[Hình 2. 16. Màn hình xem thông tin bác sĩ 50](#_heading=h.25b2l0r)

[Hình 2. 17. Thiết kế màn hình quản lý người dùng 51](#_heading=h.34g0dwd)

[Hình 2. 18. Thiết kế màn hình thêm thông tin bác sĩ 52](#_heading=h.43ky6rz)

[Hình 2. 19. Thiết kế màn hình quản lý chuyên khoa 53](#_heading=h.xvir7l)

[Hình 2. 20. Thiết kế màn hình xem chi tiết sản phẩm 54](#_heading=h.1x0gk37)

[Hình 2. 21. Thiết kế màn hình quản lý bệnh nhân khám bệnh 54](#_heading=h.2w5ecyt)

[Hình 2. 22. Thiết kế màn hình tìm kiếm bác sĩ 55](#_heading=h.3vac5uf)

[Hình 2. 23. Thiết kế màn hình xem chi tiết thông tin chuyên khoa 56](#_heading=h.pkwqa1)

[Hình 2. 24. Cơ sở dữ liệu 57](#_heading=h.1opuj5n)

[Hình 2. 25. Bảng All Code](about:blank) 58

[Hình 2. 26. Bảng Booking 58](#_heading=h.3mzq4wv)

[Hình 2. 27. Bảng Clinic 59](#_heading=h.319y80a)

[Hình 2. 29. Bảng Markdown 61](#_heading=h.4du1wux)

[Hình 2. 30. Bảng Schedule 61](#_heading=h.3s49zyc)

[Hình 2. 31. Bảng Specialty 62](#_heading=h.36ei31r)

[Hình 2.32. Bảng User](about:blank) 63

[Hình 2. 33. Giao diện trang chủ người dùng 64](#_heading=h.zu0gcz)

[Hình 2.34. Giao diện trang chủ admin 65](#_heading=h.1yyy98l)

# LỜI CẢM ƠN

Để hoàn thành báo cáo cho đề tài đồ án tốt nghiệp này, trước hết em xin được gửi lời cảm ơn tới trường Đại học Công nghiệp Hà Nội, khoa Công nghệ thông tin đã tạo điều kiện cho em được học tập và nghiên cứu để hoàn thành đồ án tốt nghiệp của mình. Em xin gửi lời cảm ơn chân thành đến các thầy cô trong khoa Công nghệ thông tin đã giảng dạy bằng tất cả tâm huyết, sự nhiệt tình, truyền đạt cho chúng em những tri thức, kinh nghiệm quý báu của mình để từ đó chúng em có một hành trang tốt nhất bước vào con đường lập nghiệp của riêng mình. Đặc biệt em xin gửi lời cảm ơn sâu sắc đến thầy giáo **ThS. Đoàn Văn Trung**, thầy đã luôn tận tình chỉ dạy, định hướng và giúp đỡ em trong trong suốt quá trình nghiên cứu và thực hiện đồ án tốt nghiệp.

Em đã cố gắng hoàn thiện báo cáo đồ án tốt nghiệp một cách tốt nhất, tuy nhiên không thể tránh được những thiếu sót. Em rất mong nhận được sự góp ý của các thầy cô để đồ án tốt nghiệp này của em được hoàn thiện hơn nữa. Lời cuối cùng em xin chúc thầy cô luôn dồi dào sức khỏe và thành công trong cuộc sống.

Em xin chân thành cảm ơn !

# MỞ ĐẦU

## Lý do chọn đề tài

Thiết kế website phòng khám, bệnh viện hiện là tiêu chí hàng đầu cho nhu cầu truyền thông dịch vụ của nhiều trung tâm chăm sóc và điều trị sức khỏe cả công và tư.

Một trong số những ngành dịch vụ quan trọng nhất hiện nay chính là dịch vụ chăm sóc sức khỏe. Trong chăm sóc sức khỏe lại được chia ra thành nhiều chuyên khoa như khoa nội, khoa ngoại, khoa mắt, thần kinh, khoa răng miệng,… Trước đây, khi những dịch vụ này chưa được chú ý phát triển ở nước ta, hầu như ngành dịch vụ này không cần được truyền thông nhắc đến vẫn thu hút “khách hàng”. Người dân khi có nhu cầu sẽ trực tiếp đến các trung tâm chăm sóc sức khỏe, bệnh viện công để khám và chữa bệnh. Nhưng hiện nay, vì nhiều nguyên nhân mà mọi người có nhu cầu chăm sóc sức khỏe nhiều hơn, không cần phải mắc bệnh rồi mới đến kiểm tra và chữa trị. Nhu cầu tăng đột ngột dẫn đến sự ra đời của rất nhiều phòng khám, bệnh viện tư nhân. Khi ấy, ngoài chất lượng dịch vụ luôn được đảm bảo, phòng khám nào có danh tiếng hơn, phòng khám ấy chắc chắn được nhiều bệnh nhân trao gửi niềm tin. Hiện nay, công cụ truyền thông quan trọng nhất đối với phòng khám chính là website giới thiệu chuyên nghiệp và chất lượng.

Bây giờ là thời đại công nghệ, đã có khoảng ít nhất 70% dân số sử dụng điện thoại thông minh. Trong đó khoảng 80% những người đó rất thông thạo việc tìm kiếm trên mạng. Việc tìm kiếm thông tin về phòng khám bây giờ có thể hoàn tất trong vòng vài giây. Như vậy, để phòng khám, bệnh viện tiếp cận với khách hàng của mình chỉ có cách duy nhất là thiết kế và phát triển website chuyên nghiệp.

Sự gắn kết chặt chẽ giữa phòng khám với người dân được thể hiện hoàn toàn rõ ràng trên website của mình thông qua bình luận, những câu chuyện chia sẻ, những câu hỏi và tư vấn từ bác sĩ chuyên môn.

Một trong những khó khăn trong ngành chăm sóc sức khỏe hiện nay là việc người dân có quá ít kiến thức về bệnh tật cũng như cách tự chăm sóc và bảo vệ sức khỏe của mình. Trước hiện trạng nhiều loại bệnh tật xuất hiện và gây nguy hiểm đến tính mạng hiện nay, nhiều người dần hình thành cảm giác lo lắng và mong muốn được bác sĩ giúp đỡ, tư vấn. Khi nhu cầu quá lớn thì quỹ thời gian của bác sĩ không thể nào đáp ứng được cho từng người cụ thể. Do đó, website chính là nơi để bác sĩ giải quyết vấn đề này. Giải đáp thắc mắc một lần cho tất cả mọi người. Hiệu năng của công việc cũng nhờ đó mà tăng lên gấp bội

## Đối tượng, mục tiêu và phạm vi nghiên cứu

* 1. Mục tiêu nghiên cứu
     + Xác định và làm rõ các vấn đề liên quan đến đề tài.
     + Khảo sát, phân tích thiết kế hệ thống cho website.
     + Xây dựng website bán thiết bị đo lường cho công ty BookingCare đáp ứng đầy đủ các yêu cầu được đặt ra.
     + Xây dựng Website đáp ứng nhu cầu đặt lịch qua mạng, tra cứu thông tin bệnh tình, tư vấn trực tuyến các vấn đề về sức khỏe, dễ dàng thao tác với nhiều đối tượng, tiết kiệm thời gian.
     + Xây dựng được chức năng tạo lịch khám bệnh giúp bác sĩ chủ động được thời gian làm việc
  2. Đối tượng nghiên cứu
     + Quy trình đặt lịch khám bệnh
     + Quy trình tạo lịch khám bệnh
     + Cách tổ chức và quản lý dữ liệu về thông tin bệnh nhân và bác sĩ

## Ý nghĩa khoa học và thực tiễn của đề tài

* Ý nghĩa khoa học, thực tiễn

Trong lĩnh vực y tế, một trong những khó khăn trong ngành chăm sóc sức khỏe hiện nay là việc người dân có quá ít kiến thức về bệnh tật cũng như cách tự chăm sóc và bảo vệ sức khỏe của mình. Trước hiện trạng nhiều loại bệnh tật xuất hiện và gây nguy hiểm đến tính mạng hiện nay, nhiều người dần hình thành cảm giác lo lắng và mong muốn được bác sĩ giúp đỡ, tư vấn. Khi nhu cầu quá lớn thì quỹ thời gian của bác sĩ không thể nào đáp ứng được cho từng người cụ thể. Do đó, website chính là nơi để bác sĩ giải quyết vấn đề này. Giải đáp thắc mắc một lần cho tất cả mọi người. Hiệu năng của công việc cũng nhờ đó mà tăng lên gấp bội.

* + Tìm hiểu về hoạt động đặt lịch khám bệnh
  + Khảo sát về hoạt động quy trình và cách vận hành việc đặt lịch khám bệnh. Trao đổi nắm bắt các thông tin để tìm giải pháp sao cho tối ưu.
  + Phân tích chức năng chính đảm bảo, yêu cầu cần thiết của hệ thống.
  + Lựa chọn công nghệ phù hợp xử lý được các yêu cầu của khách hàng, các công cụ lập trình để xây dựng website.Tiến hành triển khai và xây dựng website đặt lịch khám bệnh BookingCare
  + Tìm hiểu thêm nguồn thông tin trên Internet
* Kết quả mong muốn
  + Xác định và làm rõ các vấn đề liên quan đến đề tài.
  + Khảo sát, phân tích thiết kế hệ thống cho website.
  + Xây dựng website đặt lịch khám bệnh đáp ứng đầy đủ các yêu cầu được đặt ra.
  + Về phía khách hàng, đảm bảo được quá trình xem, đặt lịch tra cứu thông tin một cách chính xác, nhanh gọn
  + Về phía bác sĩ giúp quản lý, lên lịch trình khám bệnh một cách chủ động trong các tình huống

# Chương 1. TỔNG QUAN VỀ NỘI DUNG NGHIÊN CỨU

## Giới thiệu chung

Hiện nay, việc hiện đại hóa nhanh chóng các việc tìm kiếm thông tin về phòng khám bây giờ có thể hoàn thành trong vài phút. Điều này giúp cho việc đặt lịch giữa các bệnh nhân và bác sĩ có thể nhanh chóng hơn hiệu quả hơn, bên cạnh đó giúp cho phong khám, bác sĩ tốt sẽ được nhiều người biết đến hơn

Để xây dựng lên, em dự định sẽ sử dụng mô hình MVC, sử dụng thư viện Reactjs bên phía Front-End và Nodejs cho bên Back-End

* + - Những vấn đề còn tồn tại
      * Giải quyết vấn đề lòng tin, sự tin tưởng của khách hàng khi đặt lịch online.
      * Quảng bá sản phẩm rộng rãi hơn.
    - Định hướng phát triển
      * Xây dựng một website đặt lịch khám bệnh
      * Giúp khách hàng và bac sĩ có thể dễ dàng hơn trong việc lên lịch khám bệnh
      * Quản lý lịch khám bệnh dễ dàng hơn đối với bác sĩ

## Cơ sở lý thuyết

### **1.2.1. Tổng quan về đặt lịch khám bệnh trực tuyến**

#### 1.2.1.1 Khái niệm về đặt lịch khám bệnh trực tuyến

* Dịch vụ đặt lịch khám tại bệnh viện (hay còn gọi là đặt lịch khám trước, khám theo yêu cầu) giúp khách hàng chủ động thời gian đi thăm khám, chọn lựa đươc chuyên gia , giáo sư hay các bác sĩ chuyên khoa giàu kinh nghiệm để đảm bảo chất lượng khám chữa bệnh diễn ra đạt kết quả tốt nhất.

#### 1.2.1.2 Đặc điểm của đặt lịch khám bệnh trực tuyến

* Đặt lịch khám bệnh trực tuyến là việc ứng dụng công nghệ thông tin vào trong mọi hoạt động đặt lịch khám bệnh, chính vì lẽ đó mà sự phát triển của công nghệ thông tin sẽ thúc đẩy dịch vụ khám bệnh phát triển nhanh chóng, ngược lại, sự phát triển của việc đặt lịch khám bệnh trực tuyến giúp quảng bá hình ảnh bệnh viện cũng như thông tin về các bác sĩ.
* Về hình thức: Đặt lịch khám bệnh là hoàn toàn qua mạng. Trong hoạt động đặt lịch khám bệnh truyền thống, bệnh nhân phải gặp trực tiếp bác sĩ ở bệnh viện để đặt lịch khám bệnh
* Phạm vi hoạt động: Trên khắp toàn cầu hay thị trường trong nước. Điều này thể hiện ở chỗ mọi người ở tất cả các quốc gia trên khắp toàn cầu không phải di chuyển tới bất kì địa điểm nào mà vẫn có thể tham gia vào cùng một giao dịch bằng cách truy cập vào các website đặt lịch hoặc vào các trang mạng xã hội.
* Thời gian không giới hạn: Các bên tham gia vào hoạt động đều có thể tiến hành các giao dịch suốt 24 giờ/ 7 ngày trong vòng 365 ngày liên tục ở bất cứ nơi nào có mạng viễn thông và có các phương tiện điện tử kết nối với các mạng này, đây là các phương tiện có khả năng tự động hóa cao giúp đẩy nhanh quá trình giao dịch.
* Trong thương mại điện tử các bên không phải gặp gỡ nhau trực tiếp mà vẫn có thể tiến hành hoạt động đặt lịch. Để làm được điều này các bên phải truy cập vào hệ thống thông tin của nhau hay hệ thống thông tin của các giải pháp tìm kiếm thông qua mạng internet để tìm hiểu thông tin về nhau từ đó tiến hành trao đổi mua bán.

#### 1.2.1.3 Lợi ích của thương mại điện tử đối với doanh nghiệp:

* Mở rộng quy mô thị trường: Thị trường trong dịch vụ khám bệnh là thị trường toàn cầu không biên giới. Nhờ kết nối internet mà các tổ chức có thể tiếp cận tới mọi thị trường lớn nhỏ khác nhau trên toàn cầu một cách nhanh chóng. Đặt lịch khám bệnh thực sự có ý nghĩa và hiệu quả đối với các doanh nghiệp vừa và nhỏ trong giai đoạn đầu thành lập khi nguồn vốn còn hạn chế.
* Tiết kiệm chi phí: Bao gồm chi phí marketing, sản xuất, phân phối và các chi phí hành chính giấy tờ.
* Tăng lợi nhuận: Đặt lịch khám bệnh giúp cho các bệnh viện nâng cao hiệu quả khám bệnh, quản lý tốt hơn quá trình đặt lịch, điều phối và quan hệ bệnh nhân nên giúp cho bệnh viện giảm thiểu được chi phí tăng, lợi nhuận.
* Kết nối với khách hàng liên tục: 24h/24h trong cả 7 ngày nên có thể phản ứng nhanh trước những thay đổi về nhu cầu của khách hàng.
* Tiếp cận nhanh chóng và hiệu quả với thị trường khách hàng mục tiêu, do đó đáp ứng tốt nhất mọi nhu cầu của khách hàng.
* Củng cố quan hệ khách hàng: Việc loại bớt trung gian giúp cho tổ chức có thể tiếp cận gần hơn với khách hàng, tạo được lòng trung thành.
* Thông tin cập nhật: Thông tin về bác sĩ, chuyên khoa và lịch khám được cập nhật từng phút trên các website đặt lịch trực tuyến.
* Đối với người có nhu cầu đặt lịch khám:
* Nhiều sự chọn lựa: Khách hàng có nhiều sự lựa chọn hơn do đó họ có thể chọn cho mình một sản phẩm tốt nhất với giá cả phải chăng từ thị trường trong và ngoài nước, cũng như có thể đặt lịch khám bệnh từ nơi mình muốn
* Thông tin phong phú, cập nhật: Các bệnh nhân và khách hàng có thể tìm kiếm thông tin về một loại hàng cực kỳ nhanh chóng.
* Đặt lịch mọi lúc, mọi nơi cho phép người khám bệnh có thể đặt lịch khám bệnh liên tục trong ngày.

### **Giới thiệu về các công cụ lập trình**

#### 1.2.2.1 Công cụ Visual Studio Code Giới thiệu Visual Studio Code:



* Visual Studio Code là một môi trường phát triển tích hợp từ Microsoft. Là sự kết hợp hoàn hảo giữu IDE và Code Editor. Nó được sử dụng để phát triển chương trình máy tính cho Microsoft Windows, cũng như các trang web, các ứng dụng web và các dịch vụ web. Visual Studio Code sử dụng nền tảng phát triển phần mềm của Microsoft như Windows API, Windows Forms, …
* Visual Studio bao gồm:
* Một trình soạn thảo mã hỗ trợ IntelliSense cũng như cải tiến mã nguồn.
* Trình gỡ lỗi tích hợp hoạt động cả về trình gỡ lỗi mức độ mã nguồn và gỡ lỗi mức độ máy.
* Công cụ tích hợp khác bao gồm một mẫu thiết kế các hình thức xây dựng giao diện ứng dụng, thiết kế web, thiết kế lớp và thiết kế giản đồ cơ sở dữ liệu.
* Nó chấp nhận các plug-in nâng cao các chức năng ở hầu hết các cấp bao gồm thêm hỗ trợ cho các hệ thống quản lý phiên bản (như Subversion).
* Visual Studio Code hỗ trợ nhiều ngôn ngữ lập trình khác nhau và cho phép trình biên tập mã và gỡ lỗi để hỗ trợ (mức độ khác nhau) hầu hết mọi ngôn ngữ lập trình. Nó hỗ trợ mạnh mẽ cho Javascript, Nodejs và TypeScript

#### 1.2.2.2 Công cụ XAMPP

Giới thiệu XAMPP:



* Phần mềm Xampp là một trong những phần mềm được nhiều lập trình viên sử dụng để thiết lập website theo ngôn ngữ PHP. XAMPP có công dụng thiết lập web server có cài đặt sẵn các công cụ như PHP, Apache, MySQL… Xampp sở hữu thiết kế giao diện thân thiện với người dùng, cho phép các lập trình viên có thể đóng mở hoặc reboot các tính năng của server mọi lúc. Ngoài ra, Xampp cũng được xây dựng theo source code mở.
* Đánh giá một cách tổng thể, **Xampp** được đánh giá là một trong những phần mềm trọng yếu trong việc xây dựng website dành cho các lập trình viên PHP. Xampp cài đặt các thành tố trọng yếu, hỗ trợ lẫn nhau bao gồm:
* Apache
* PHP (thiết lập nền tảng để các tập tin script \*.php hoạt động);
* MySql (hệ quản trị dữ liệu)
* Perl
* MySQL là một hệ thống quản lý cơ sở dữ liệu liên kết có ưu điểm nhanh chóng, dễ dùng cho các lập trình viên. MySQL được ứng dụng khá nhiều cho các website - thương mại điện tử hiện nay.
* Hệ thống được phát triển bởi công ty Thụy Điển này hiện được nhiều lập trình viên hàng đầu trên thế giới ưa chuộng bởi những ưu điểm sau.
* MySQL là một opensource, hoàn toàn miễn phí.
* Hiệu suất hoạt động của MySQL rất mạnh mẽ, kể cả khi khối lượng dữ liệu khổng lồ. Có thể kiểm tra toàn vẹn dữ liệu với SQL Trigger
* Cái tên nói lên tất cả, hệ thống này sử dụng ngôn ngữ dữ liệu SQL để cấu trúc hóa website của bạn.
* MySQL tương thích với hầu hết các hệ điều hành (Windows, Mac, Linux…) cùng các ngôn ngữ lập trình như PHP, C++, Java…
* MySQL rất tương thích với PHP, ngôn ngữ lập trình phổ biến của nhiều lập trình viên.
* MySQL có hệ thống thông tin khổng lồ, chứa đến 50 triệu hàng dữ liệu trong 1 board. Dữ liệu tối thiểu của board này là 4GB, còn tối đa là 8TB (tùy theo độ mạnh của hệ điều hành)
* MySQL cho phép điều chỉnh tùy ý để các lập trình viên có thể thao tác thay đổi cho website của mình.

#### 1.2.2.3 Tổng quan về mô hình MVC

* Khái niệm
* MVC là từ viết tắt bởi ba từ Model – View – Controller. Đây là mô hình thiết kế sử dụng trong kỹ thuật phần mềm. Mô hình được chia thành ba phần, tương ứng mỗi từ. Mỗi từ tương ứng với một hoạt động tách biệt trong một mô hình:
* Model (M) : Là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng. Bộ phận này là một cầu nối giữa hai thành phần bên dưới là View và Controller. Có khi chỉ đơn giản là một tệp XML bình thường.
* View (V) : Đây là phần giao diện dành cho người sử dụng. Nơi mà người dùng có thể lấy được thông tin dữ liệu của MVC thông qua các thao tác truy vấn như tìm kiếm hoặc sử dụng thông qua các website. View không có mối quan hệ trực tiếp với Controller, cũng không được lấy dữ liệu từ Controller mà chỉ hiển thị yêu cầu chuyển cho Controller.
* Controller (C) : Bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua view. Từ đó, Controller đưa ra dữ liệu phù hợp với người dùng. Bên cạnh đó, Controller còn có chức năng kết nối với model.
* Ưu và nhược điểm của mô hình MVC

**Ưu điểm:**

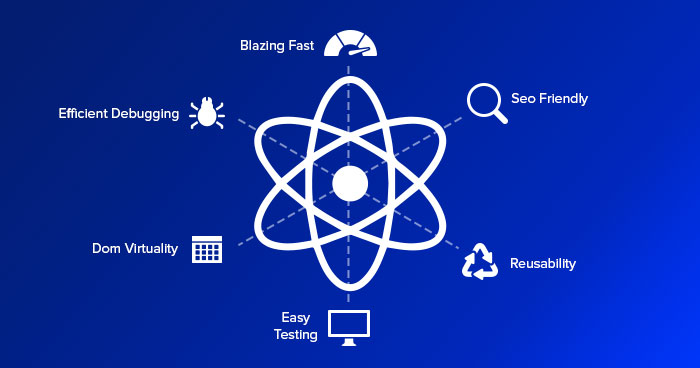
* Kiểm tra dễ dàng: Với MVC, bạn có thể dễ dàng kiểm tra, rà soát lỗi phần mềm trước khi tới tay người tiêu dùng, đảm bảo chất lượng và độ uy tín cao hơn.
* Chức năng control: Trên các nền website thì việc sử dụng các ngôn ngữ như CSS, HTML, Javascript có một vai trò vô cùng quan trọng. Việc sử dụng mô hình MVC sẽ giúp bạn có một bộ control ưu việt trên nền tảng các ngôn ngữ hiện đại với nhiều hình thức khác nhau.
* View và kích thước: View sẽ là nơi lưu trữ các dữ liệu. Càng nhiều yêu cầu được thực hiện thì kích thước tệp càng lớn. Khi đó, đường truyền mạng cũng giảm tốc độ tải trang. Việc sử dụng mô hình MVC sẽ giúp bạn tiết kiệm được diện tích băng thông một cách tối ưu.
* Tính kết hợp: Việc tích hợp ở mô hình MVC cho phép bạn thoải mái viết mã trên nền tảng website. Khi đó, server của bạn sẽ được giảm tải khá nhiều.
* Đơn giản: Đây là một mô hình với kết cấu tương đối đơn giản. Dù bạn không có quá nhiều chuyên môn cũng có thể sử dụng được.

**Nhược điểm:**

* MVC thường được sử dụng vào những dự án lớn. Do đó, với các dự án nhỏ, mô hình MVC có thể gây cồng kềnh, tốn thời gian trong quá trình phát triển cũng như thời gian trung chuyển dữ liệu.
* MVC đang là mô hình được ứng dụng rất nhiều trong lập trình. Hệ thống MVC phát triển tốt sẽ cho phép phát triển front – end, back – end cùng trên hệ thống mà không có sự can thiệp, chia sẻ, chỉnh sửa các tập tin trong khi một hoặc hai bên vẫn đang làm việc. Việc sử dụng mô hình tương đối đơn giản. Chỉ cần hiểu rõ quy trình vận hành, nắm được các chức năng của từng bộ phận thì việc triển khai mô hình MVC tương đối dễ dàng.

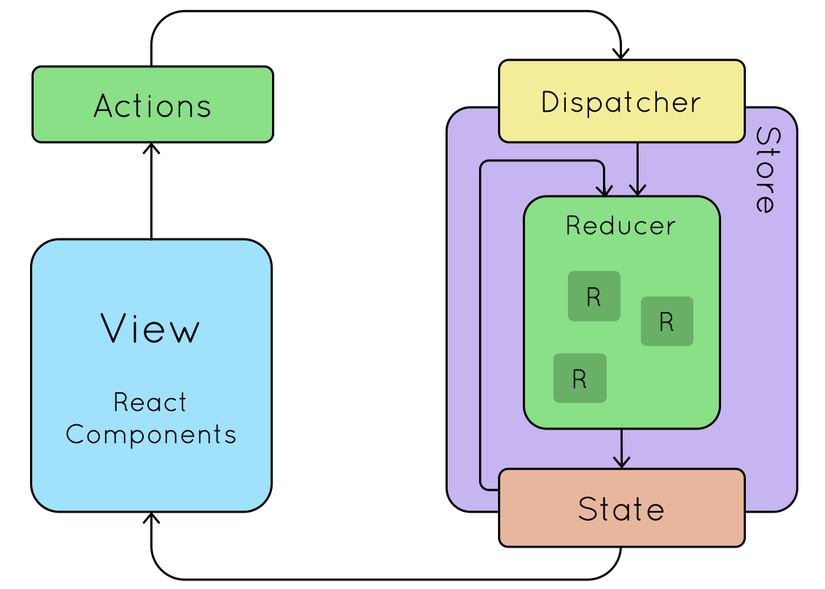
#### 1.2.2.4 Thiết kế giao diện website với thư viện Reactjs và Redux

* Reactjs là gì ?



* React là một thư viện JavaScript nhằm đơn giản hóa việc phát triển giao diện người dùng.
* Được phát triển bởi Facebook và bản release đầu tiên trên thế giới vào nằm 2013 và được sử dụng để viết ra Facebook, instagram và được sự ủng hộ sử dụng của rất nhiều công ty khác trên thế giới.
* Mục tiêu của react chính là đơn giản để phát triển. Tất cả trạng thái đều được tập trung tại một thời điểm, bằng cách chia giao diện người dùng thành tập hợp các thành phần (components)
* React được sử dụng để xây dựng single-page web applications.
* Tính năng ReactJs
* **JSX** – JSX là phần mở rộng cú pháp JavaScript. Không nhất thiết phải sử dụng JSX trong phát triển React, nhưng nó được khuyến khích.
* **Các thành phần** – React là tất cả về các thành phần. Bạn cần nghĩ mọi thứ như một thành phần. Điều này sẽ giúp bạn duy trì mã khi làm việc trên các dự án quy mô lớn hơn.
* **Luồng dữ liệu một chiều và Flux** – React triển khai luồng dữ liệu một chiều giúp bạn dễ dàng suy luận về ứng dụng của mình. Flux là một mẫu giúp giữ dữ liệu của bạn không theo hướng.
* **Giấy phép** – React được cấp phép theo Facebook Inc. Tài liệu được cấp phép theo CC BY 4.0.
* Ưu điểm của ReactJs
* Sử dụng DOM ảo là đối tượng Javascript. Điều này sẽ cải thiện hiệu suất phản ứng, vì DOM ảo Javascript nhanh hơn DOM thông thường.
* Có thể được sử dụng trên máy khách và máy chỉ cũng như với các khuôn khổ khác
* Các mẫu thành phần và dữ liệu cải thiện khả đọc, giúp duy trì các ứng dụng lớn hơn

**Redux là gì ?**

****

* Redux là một predictable state management tool cho các ứng dụng Javascript. Nó giúp bạn viết các ứng dụng hoạt động một cách nhất quán, chạy trong các môi trường khác nhau (client, server, and native) và dễ dàng để test. Redux ra đời lấy cảm hứng từ tư tưởng của ngôn ngữ Elm và kiến trúc Flux của Facebook. Do vậy Redux thường dùng kết hợp với React.
* Lợi ích Redux mang lại:
* Trạng thái có thể dự đoán được: Nếu cùng một state và action được chuyển cho reducer, thì kết quả tương tự luôn được tạo ra vì reducer là các pure function. Nhờ đó mà chúng ta có thể chuyển qua lại giữa các state trước đó và xem kết quả trong thời gian thực.
* Việc đồng bộ dễ dàng hơn: Với chỉ một "Source of Truth" (store), chúng ta sẽ gặp ít vấn đề trong việc sync state giữa các component với nhau hơn.
* Dễ dàng testing: vì các functions được sử dụng để thay đổi state là các pure functions.
* Khả năng maintain: Redux có bộ docs vô cùng chặt chẽ hướng dẫn chi tiết cách tổ chức code như thế nào. Do đó, nếu nắm vứng kiến thức về redux thì việc maintain code vô cùng dễ dàng

#### 1.2.2.5 Tổng Quan Về API RESTful



* + RESTful API là một tiêu chuẩn dùng trong việc thiết kế các thiết kế API cho các ứng dụng web để quản lý các *resource*. *RESTful* là một trong những kiểu thiết kế API được sử dụng phổ biến nhất ngày nay.
  + Trọng tâm của REST quy định cách sử dụng các HTTP method (như *GET*, *POST*, *PUT*, *DELETE*...) và cách định dạng các URL cho ứng dụng web để quản các resource
  + Với các ứng dụng web được thiết kế sử dụng RESTful, lập trình viên có thể dễ dàng biết được URL và HTTP method để quản lý một resource. Cũng cần lưu ý bản thân RESTful không quy định logic code ứng dụng và RESTful cũng không giới hạn bởi ngôn ngữ lập trình ứng dụng. Bất kỳ ngôn ngữ lập trình (hoặc framework) nào cũng có thể áp dụng RESTful trong việc thiết kế API cho ứng dụng web.

#### 1.2.2.6 Lập trình web với Nodejs, Express

**Tổng quan về Nodejs**

****

* Nodejs là một môi trường thực thi nguồn mở, đa nền tảng để chạy code JavaScript ở phía server.
* Có nghĩa là:
* Node.JS là môi trường thực thi (runtime environment) để chạy code JavaScript, chứ không phải là một ngôn ngữ lập trình hay framework.
* Đa nền tảng, tức là có thể chạy ở nhiều OS khác nhau như Linux, Windows, MacOS…
* Chuyên biệt cho việc lập trình server.
* Trước đây, JavaScript thường chỉ được dùng ở phía client, nó được các lập trình viên nhúng vào mã HTML sau đó được chạy bởi JavaScript engine tích hợp trong trình duyệt web. Giờ đây với sự xuất hiện của Node.js, code JavaScript đã có thể chạy ở phía server, tạo ra các nội dung web động trước khi gửi về trình duyệt. Và như vậy, JavaScript đã có mặt ở khắp mọi nơi, từ frontend cho tới backend. Các lập trình viên cũng mừng ra mặt, bởi vì chỉ với một ngôn ngữ duy nhất, họ đã trở thành một lập trình viên Fullstack.Việc xây dựng một hệ thống web cũng trở nên dễ dàng hơn, bởi vì team frontend và team backend giờ đây đã có thể “nói chuyện” với nhau qua cùng một ngôn ngữ.
* Node.js có kiến trúc hướng sự kiện và không đồng bộ. Lựa chọn kiến trúc phần mềm như này là để tối ưu băng thông và khả năng mở rộng của ứng dụng web, cũng như các ứng dụng web thời gian thực , nó sử dụng một tập các module để thực hiện các tác vụ khác nhau, các module này được thiết kế như là các API nhằm giảm thiểu độ phức tạp của ứng dụng. Sự mềm dẻo linh hoạt của Node.js còn ở chỗ nó có thể chạy trên gần như tất cả các nền tảng, cũng như có thể được viết bằng các ngôn ngữ lập trình khác nhau như CoffeeScript, Dart, TypeScript,… hay bất kỳ ngôn ngữ nào mà có thể dịch ra được thành mã JavaScript.
* Ứng dụng phổ biến nhất của Node.js là để viết nên các web server.Node.js đã mang lập trình hướng sự kiện tới với web server, đồng nghĩa với việc tạo ra được các web server siêu nhanh bằng JavaScript, các hàm trong Node.js là non-blocking, có nghĩa là các dòng lệnh được thực thi song song với nhau So với PHP, một ngôn ngữ đang rất phổ biến ở server-side, thì điểm khác biệt lớn nhất đó là các hàm trong PHP bị block (các dòng lệnh phải đợi cho tới khi các lệnh trước đó được hoàn thành)

**Tổng quan về Express**

****

* Express js là một Framework nhỏ, nhưng linh hoạt được xây dựng trên nền tảng của Nodejs. Nó cung cấp các tính năng mạnh mẽ để phát triển web hoặc mobile
* Về các package hỗ trợ: Expressjs có vô số các package hỗ trợ nên các bạn không phải lo lắng khi làm việc với Framework này.
* Về performance: Express cung cấp thêm về các tính năng (feature) để dev lập trình tốt hơn. Chứ không làm giảm tốc độ của NodeJS.
* Và hơn hết, các Framework nổi tiếng của NodeJS hiện nay đều sử dụng ExpressJS như một core function, chẳng hạn: SailsJS, MEAN,....

# Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Khảo sát bài toán

* Với mục tiêu phát triển thị trường dịch vụ y tế online, công ty cổ phần Booking Care đã thành lập và đưa ra thị trường các dịch vụ về y tế tới mọi người toàn quốc. Đối với một doanh nghiệp hoạt động như vậy thì việc quảng bá và giới thiệu các dịch vụ đến khách hàng đáp ứng nhu cầu y tế ngày càng cao của khách hàng sẽ là cần thiết.
* Hệ thống website đặt lịch khám bệnh của công ty ProStore giúp cho khách hàng có cách nhìn tổng quan về các dịch vụ mà công ty cung ứng ra thị trường, có thể đặt lịch khám bệnh một cách nhanh chóng mà không cần phải đến trực tiếp các bệnh viện . Hơn hết, nó cũng là nơi để quảng bá sản phẩm một cách hiệu quả của công ty đến với mọi khách hàng, giúp cho doanh nghiệp có cái nhìn trực quan về tình hình dịch vụ y tế hiện tại.

## Giới thiệu về hệ thống

* Để đẩy mạnh phát triển các dịch vụ y tế. Hệ thống website đặt lịch khám bệnh của công ty đã được xây dựng. Hệ thống đã sử dụng mô hình MVC kết hợp với thư viện ReactJS , Nodejs và Redux cùng với cơ sở dữ liệu MySQL Server, là một website đặt lịch khám bệnh. Đồng thời giúp cho khách hàng và bác sĩ có thể kiểm soát lịch khám bệnh của mình một cách dễ dàng hơn

## Phân tích yêu cầu

### 2.3.1 Về hệ thống

* Xây dựng hệ thống có giao diện đẹp, thân thiện, dễ sử dụng và có bố cục cũng như trang trí phù hợp với một website đặt kịch khám

### 2.3.2 Về người sử dụng

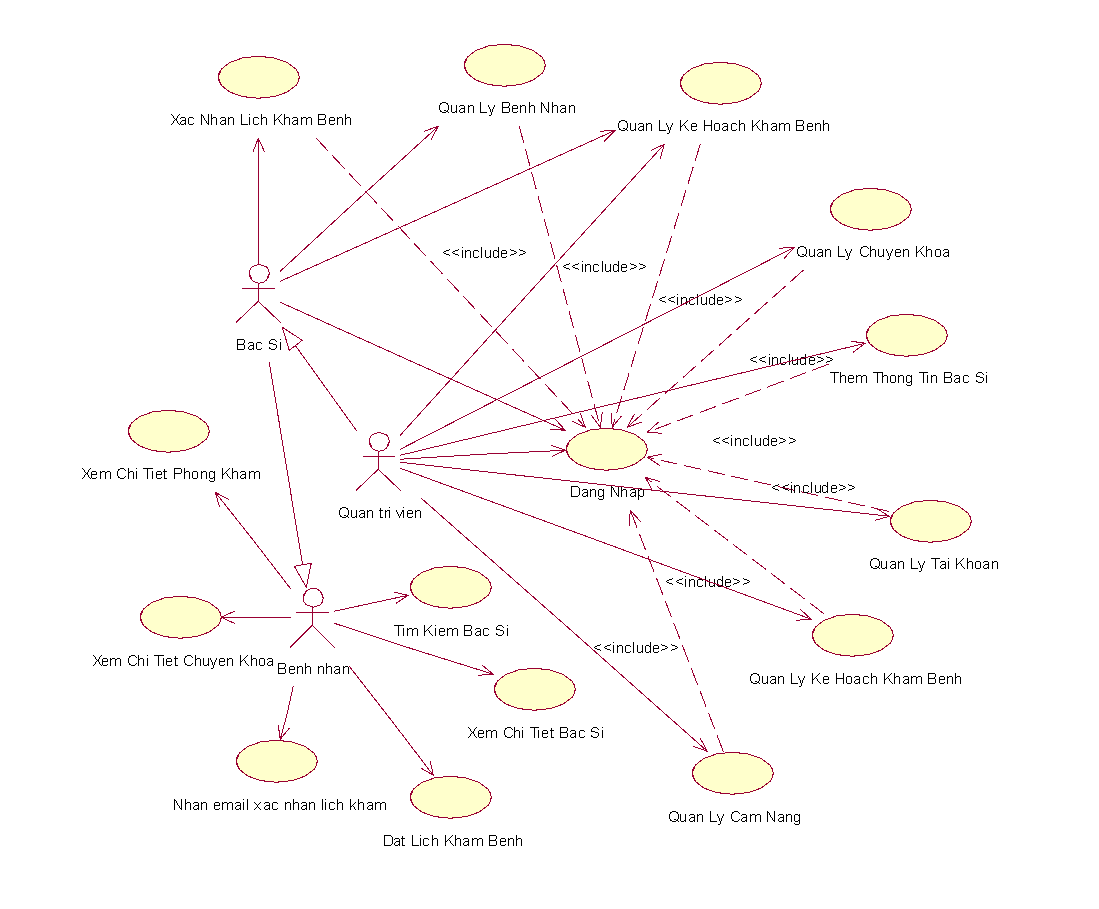
* Người sử dụng có thể nắm bắt được các thông tin cơ bản về bác sĩ, chuyên khoa và phòng khám. Ngoài việc có thể xem danh sách các bác sĩ, phong khám…. Người quản trị có thể quản lý kế hoạch khám bệnh, bệnh nhân đăng ki khám bệnh

### 2.3.3 Yêu cầu về chức năng

* Các chức năng chính của hệ thống:
  + Đăng nhập/ Đăng ký
  + Tìm kiếm bác sĩ
  + Đặt lịch khám bệnh
  + Xem danh sách chuyên khoa/ bệnh viện / bác sĩ
  + Quản lý lịch khám bệnh
  + Quản lý người dùng
  + Quản lý danh sách chuyên khoa
  + Lọc bệnh viện theo địa chỉ
  + Quản lý lịch đặt khám

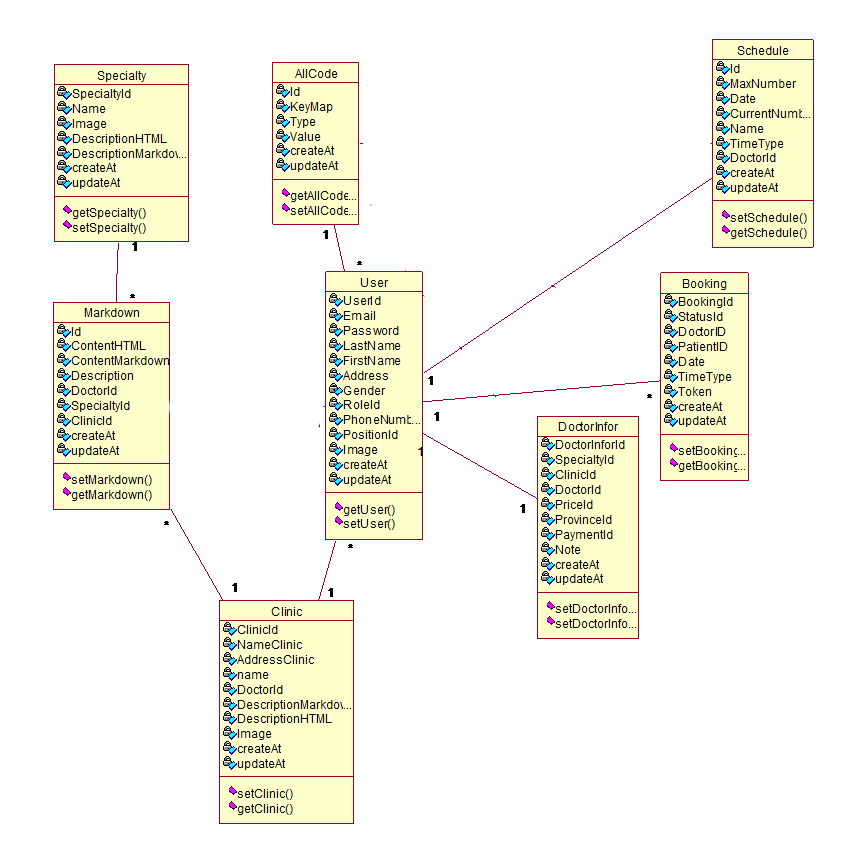
## Thiết kế hệ thống

* Biểu đồ Use case:



*Hình 2. 1.Biểu đồ lớp của hệ thống*

* Biểu đồ lớp của hệ thống



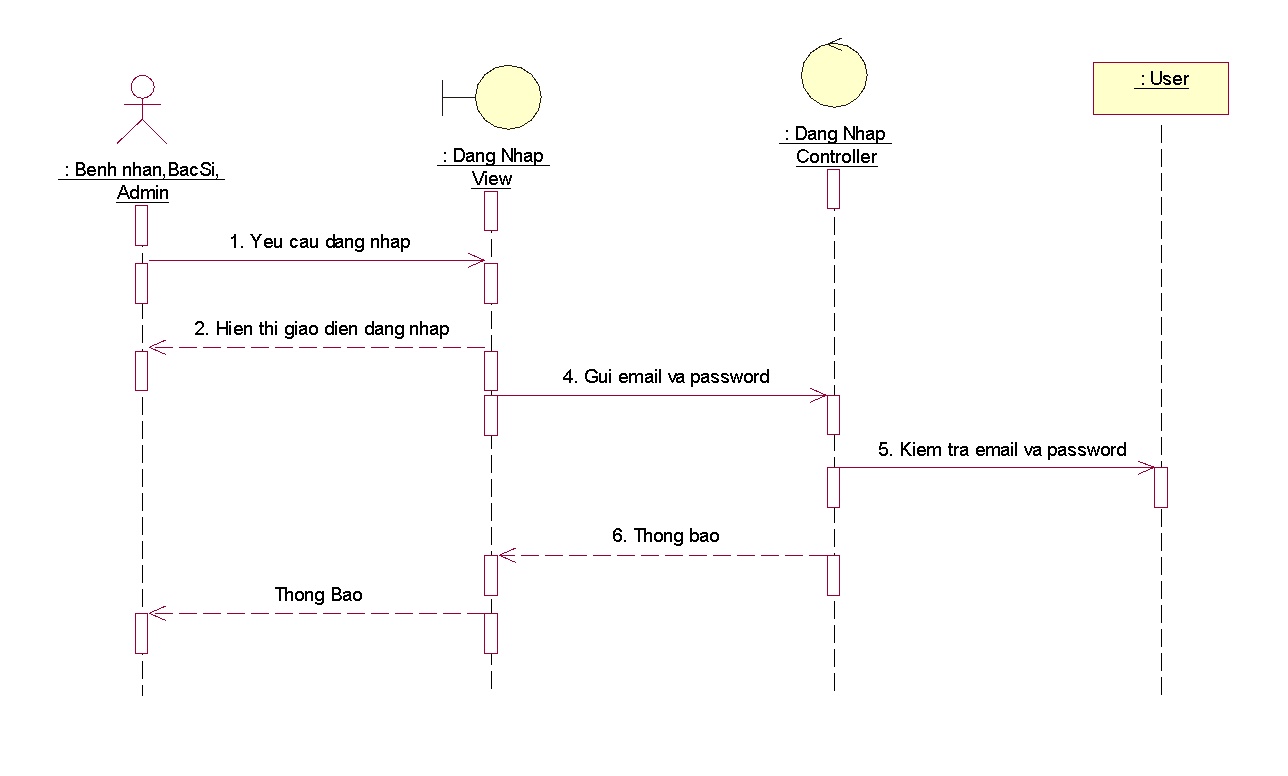
*Hình 2. 2 Biểu đồ của hệ thống*

### 2.4.1 Mô tả use case Đăng nhập

|  |  |
| --- | --- |
| Tên Use Case | Đăng nhập |
| Actor | Bệnh nhân, Bác sĩ, Admin |
| Mô tả | UC cho phép thành viên đăng nhập vào hệ thống. |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng nhập thành công và có thể sử dụng các chức năng của hệ thống. |
| Luồng sự kiện chính | Người dùng chọn chức năng đăng nhập.   1. Form đăng nhập hiển thị. 2. Nhập tên, mật khẩu vào form đăng nhập. 3. Hệ thống kiểm tra tên, mật khẩu của người dùng. 4. Nếu việc đăng nhập thành công thì người dùng có thể vào sử dụng các chức năng của hệ thống. Nếu nhập sai tên hoặc mật khẩu thì chuyển sang luồng nhánh A1. 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Luồng nhánh A1: Người dùng đăng nhập không thành công   1. Hệ thống thông báo quá trình đăng nhập không thành công. 2. Chọn đăng ký hoặc nhập lại. Nếu chọn đăng ký thì chuyển sang luồng đăng ký. 3. Hệ thống yêu cầu người dùng nhập lại tên và mật khẩu.   Nếu người dùng đồng ý thì quay về bước 2 của luồng sự kiện chính, nếu không đồng ý thì UC kết thúc. |

*Bảng 2. 1.Mô tả usecase Đăng nhập*

Biểu đồ trình tự :

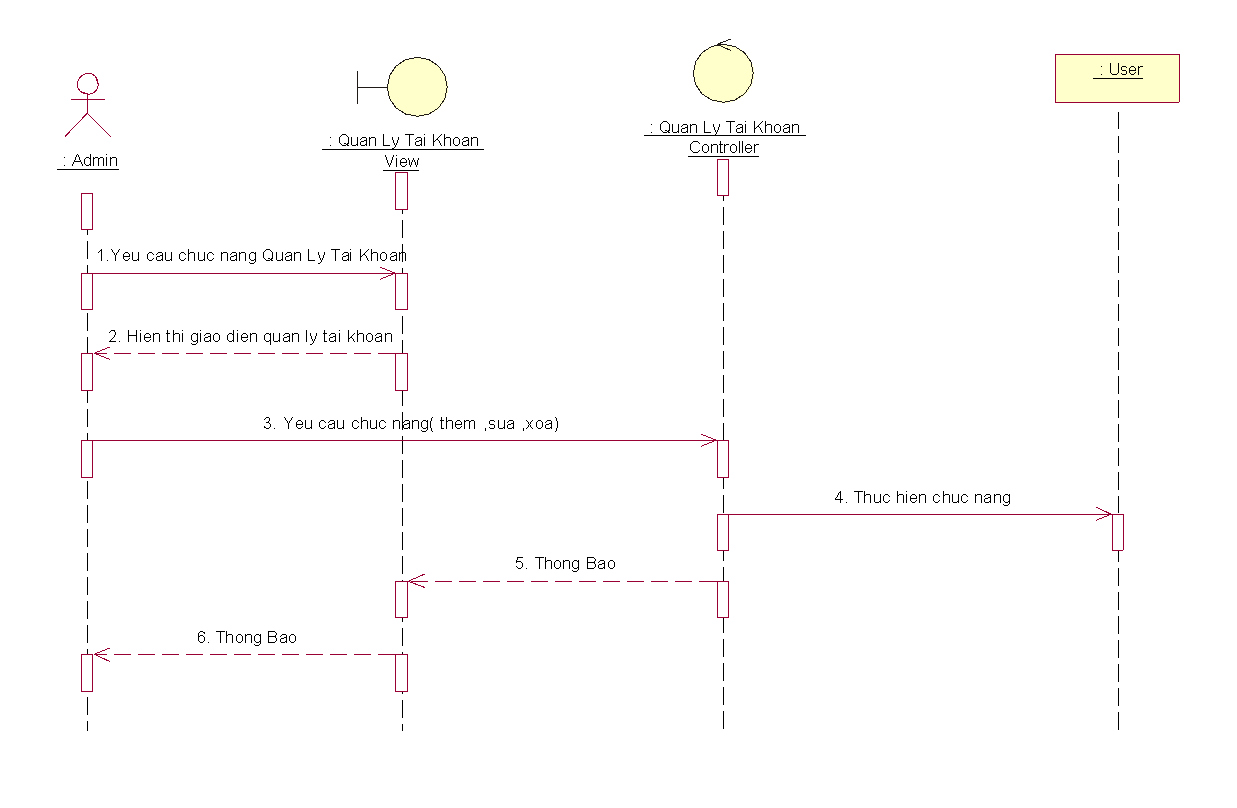


*Hình 2. 2..Biểu đồ trình tự đăng nhập*

### 2.4.2 Mô tả use case Quản lý tài khoản

|  |  |
| --- | --- |
| Tên Use Case | Quản lý tài khoản |
| Actor | Admin |
| Mô tả | UC cho phép quản trị viên quản lý tài khoản |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng nhập thành công và có thể sử dụng các chức năng của hệ thống. |
| Luồng sự kiện chính | Người dùng chọn chức năng quản lý tài khoản   1. Giao diện quản lý tài khoản hiển thị 2. Thực hiện các chức năng muốn ( thêm , sửa , xóa ) 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lưu thông tin báo lỗi thì quay lại bước 2 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 2. Mô tả use case Quản lý tài khoản*

Biểu đồ trình tự

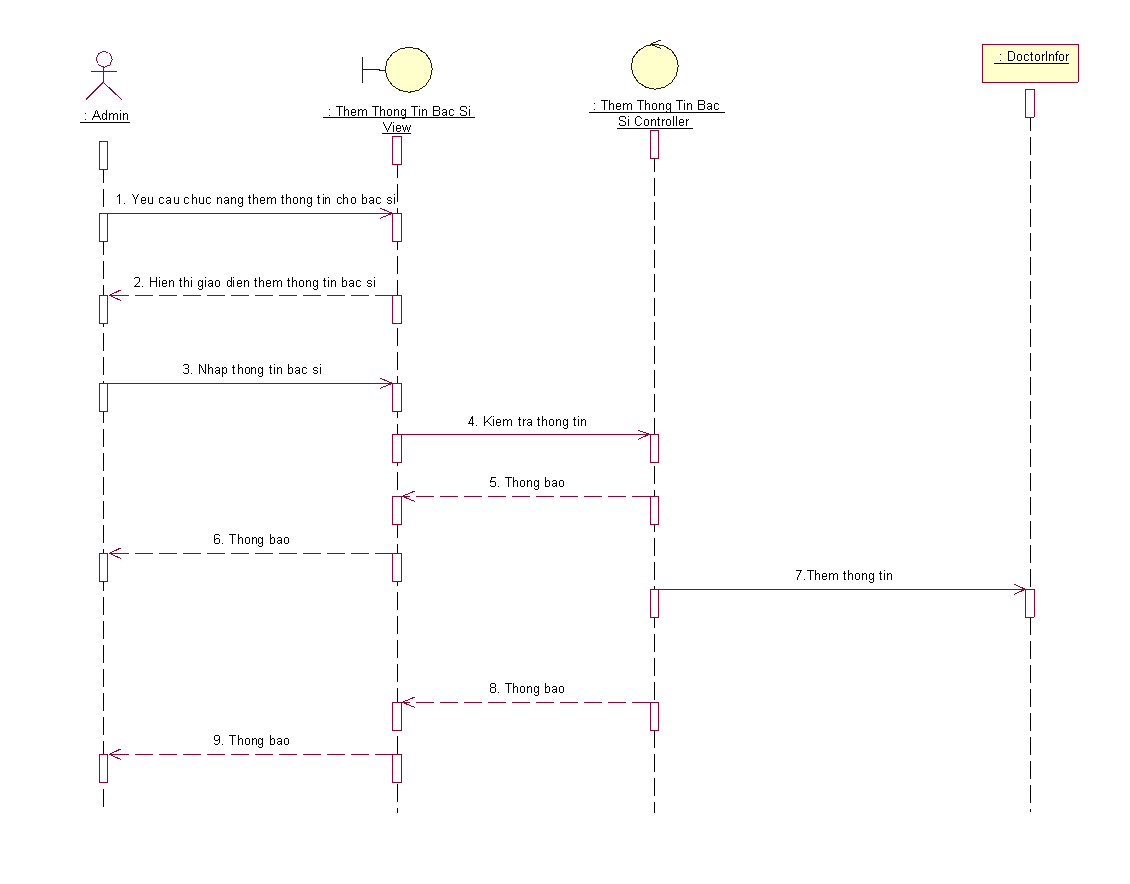
*Hình 2. 4.Biểu đồ trình tự quản lý tài khoản*

### 2.4.3 Mô tả use case Thêm thông tin bác sĩ

|  |  |
| --- | --- |
| Tên Use Case | Thêm thông tin bác sĩ |
| Actor | Admin, Bác sĩ |
| Mô tả | UC cho phép admin và bác sĩ thêm sửa thông tin bác sĩ |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng nhập thành công và có thể sử dụng các chức năng của hệ thống. |
| Luồng sự kiện chính | Người dùng chọn chức năng thêm thông tin bác sĩ   1. Giao diện thêm thông tin bác sĩ 2. Nhập thông tin cho bác sĩ 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lưu thông tin báo lỗi thì quay lại bước 2 . Nếu hợp lệ sẽ thêm dữ liệu và thông báo thành công 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 3. Mô tả use case Thêm thông tin bác sĩ*

Biểu đồ trình tự



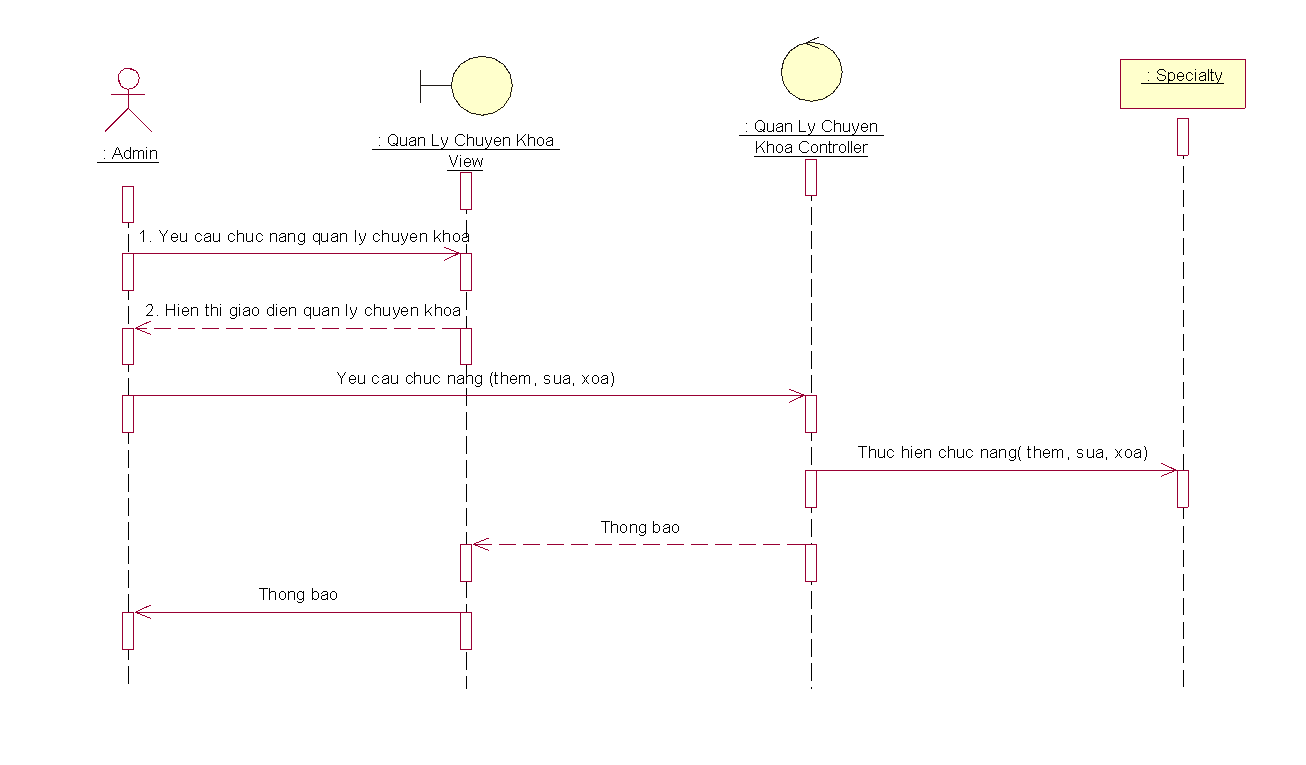
*Hình 2. 5. Biểu đồ trình tự hóa thêm thông tin bác sĩ*

### 2.4.4 Mô tả usecase Quản lý chuyên khoa

|  |  |
| --- | --- |
| Tên Use Case | Quản lý chuyên khoa |
| Actor | Admin |
| Mô tả | UC cho phép admin thực hiên thêm, sửa, xóa chuyên khoa |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng nhập thành công và có thể sử dụng các chức năng của hệ thống. |
|  | Người dùng chọn chức năng quản lý chuyên khoa   1. Giao diện thêm thông tin bác sĩ 2. Nhập thông tin cho bác sĩ 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lưu thông tin báo lỗi thì quay lại bước 2 . Nếu hợp lệ sẽ thêm dữ liệu và thông báo thành công 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 4. Mô tả use case Quản lý chuyên khoa*

Biểu đồ trình tự



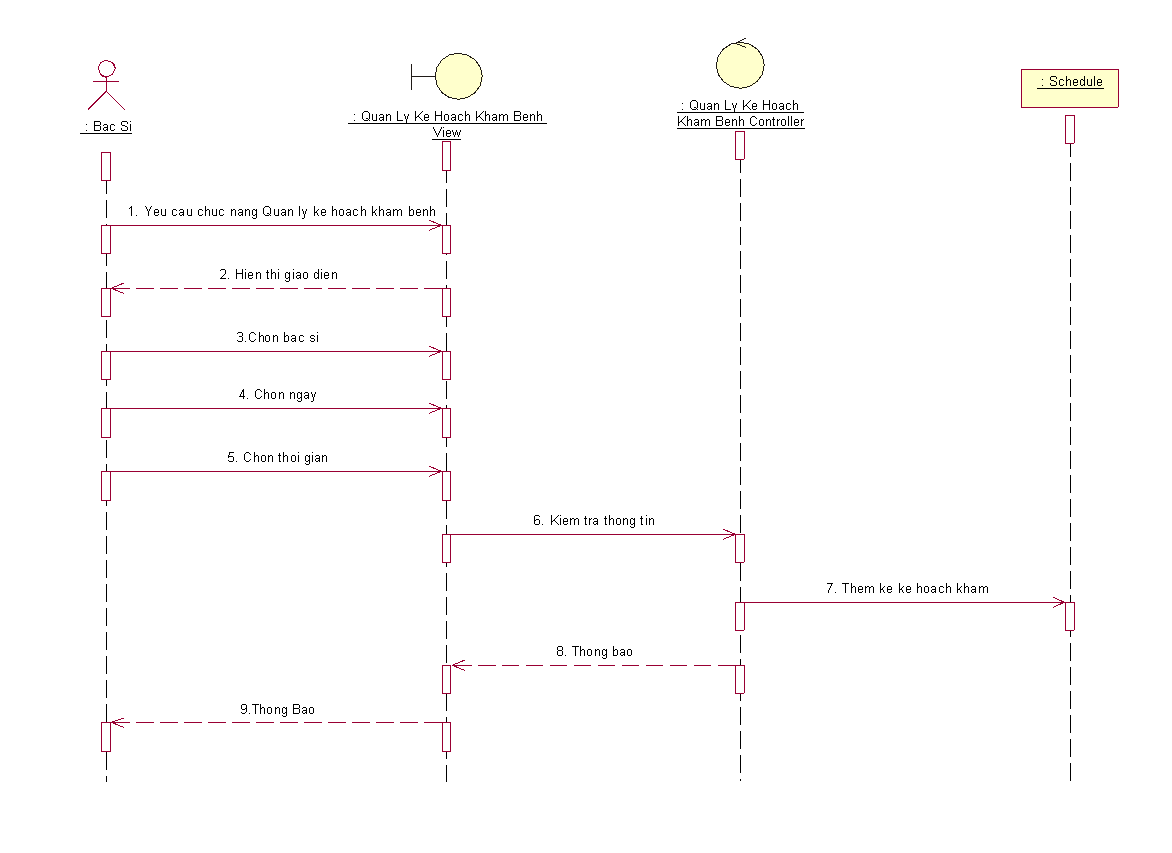
*Hình 2.6. Biểu đồ trình tự hóa quản lý chuyên khoa*

### 2.4.5 Mô tả usecase Quản lý kế hoạch khám bệnh

|  |  |
| --- | --- |
| Tên Use Case | Quản lý kế hoạch khám bệnh |
| Actor | Admin, Bác sĩ |
| Mô tả | UC cho phép admin và bác sĩ thêm kế hoạch khám bệnh cho từng ngày tiếp theo |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng nhập thành công và có thể sử dụng các chức năng của hệ thống. |
| Luồng sự kiện chính | Người dùng chọn chức năng thêm thông tin bác sĩ   1. Giao diện kế hoạch khám bệnh 2. Chọn các thông tin 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lưu thông tin báo lỗi thì quay lại bước 2 . Nếu hợp lệ sẽ thêm dữ liệu và thông báo thành công 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 5. Mô tả use case Thêm kế hoạch khám bệnh*

Biểu đồ trình tự



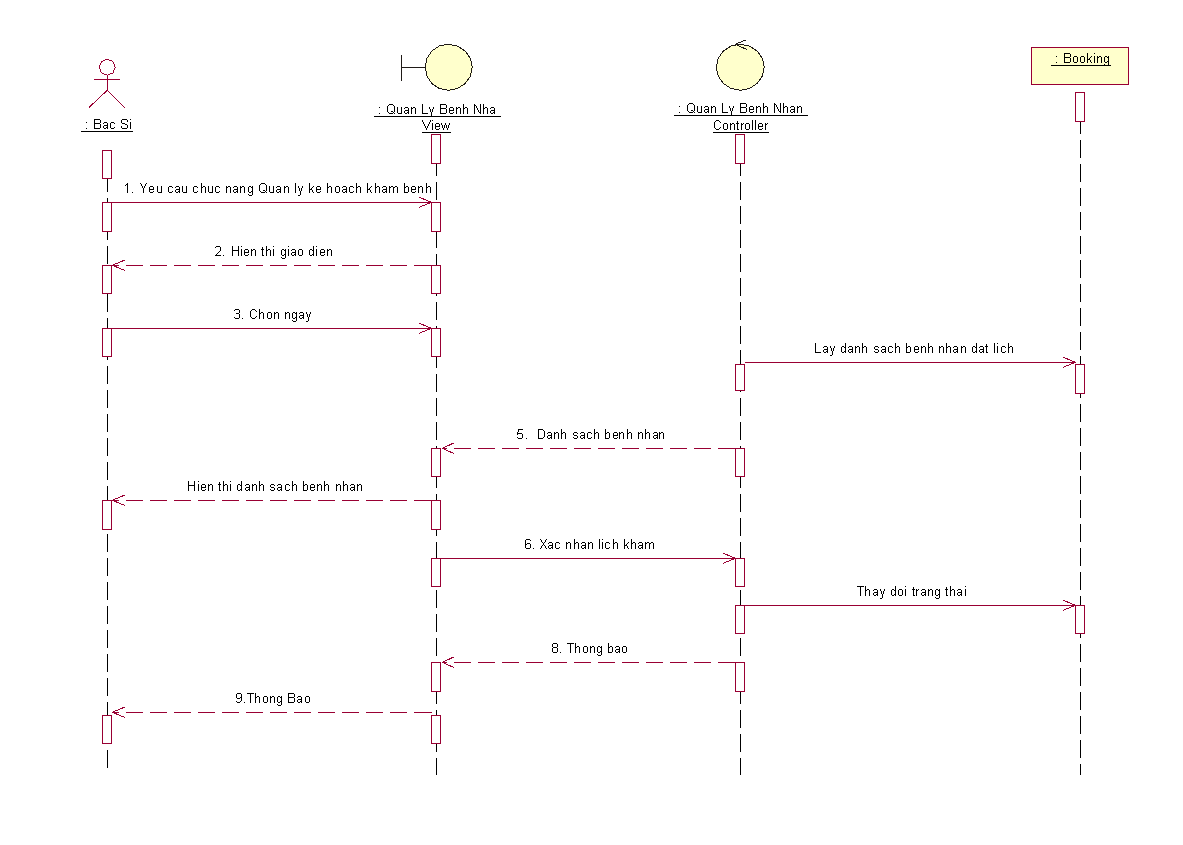
*Hình 2 .7. Thêm kế họạch khám*

### 2.4.6 Mô tả usecase Quản lý bệnh nhân

|  |  |
| --- | --- |
| Tên Use Case | Quản lý bệnh nhân |
| Actor | Bác sĩ |
| Mô tả | UC cho phép bác sĩ quản lý bệnh nhân |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Người dùng đăng nhập thành công và có thể sử dụng các chức năng của hệ thống. |
| Luồng sự kiện chính | Người dùng chọn chức năng thêm thông tin bác sĩ   1. Giao diện quản lý bệnh nhân 2. Chọn ngày 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lưu thông tin báo lỗi thì quay lại bước 2 . Nếu hợp lệ sẽ thêm dữ liệu và thông báo thành công 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 6. Mô tả use case Quản lý bệnh nhân*

Biểu đồ trình tự



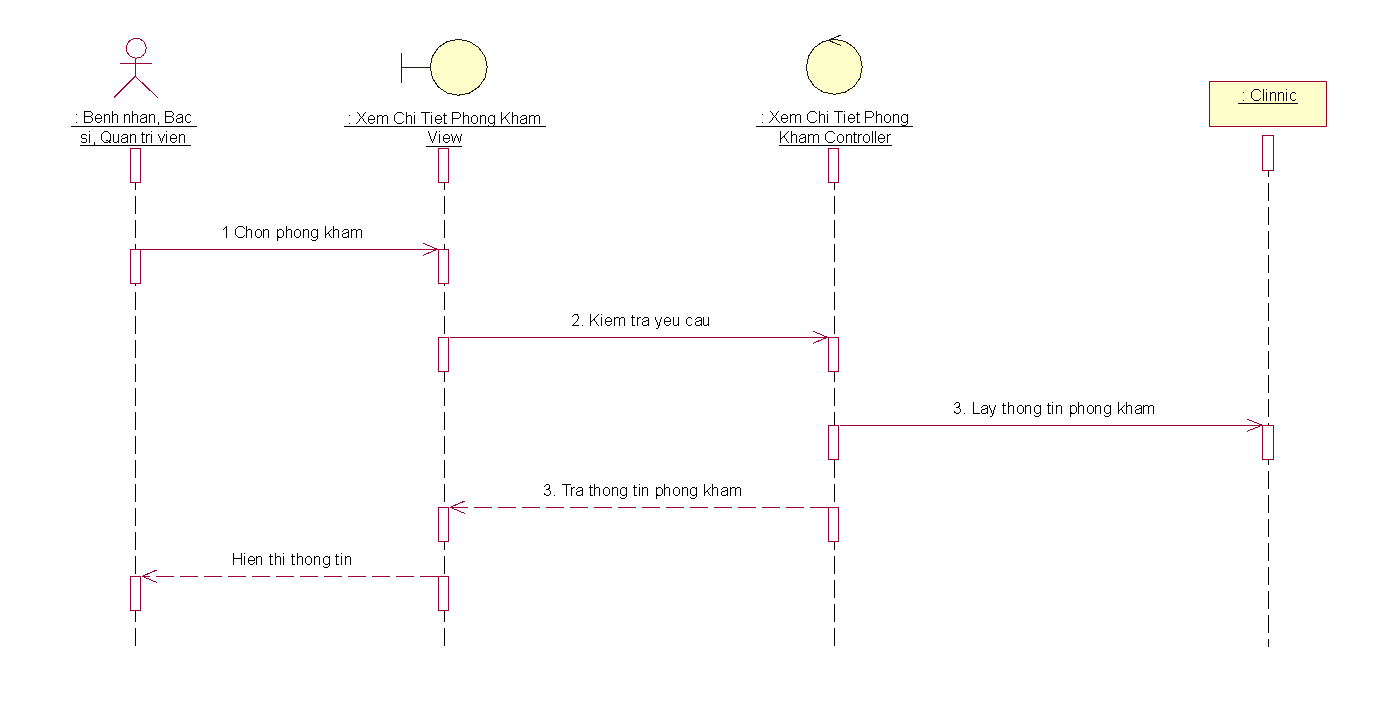
*Hình 2 .8. Quản lý bệnh nhân*

### 2.4.7 Mô tả usecase Xem chi tiết phòng khám

|  |  |
| --- | --- |
| Tên Use Case | Xem chi tiết phòng khám |
| Actor | Bác sĩ, Bệnh nhân , Quản trị viên |
| Mô tả | UC cho người dung xem chi tiết thông tin về phòng khám |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Không có |
| Luồng sự kiện chính | Người dùng chọn thông tin bất kì trên giao diện   1. Giao diện trang chính 2. Chọn phòng khám 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lấy thông tin thành công thì hiển thị cho người dùng. Ngược lại báo lỗi 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 7. Mô tả use case Xem chi tiết phòng khám*

Biểu đồ trình tự:



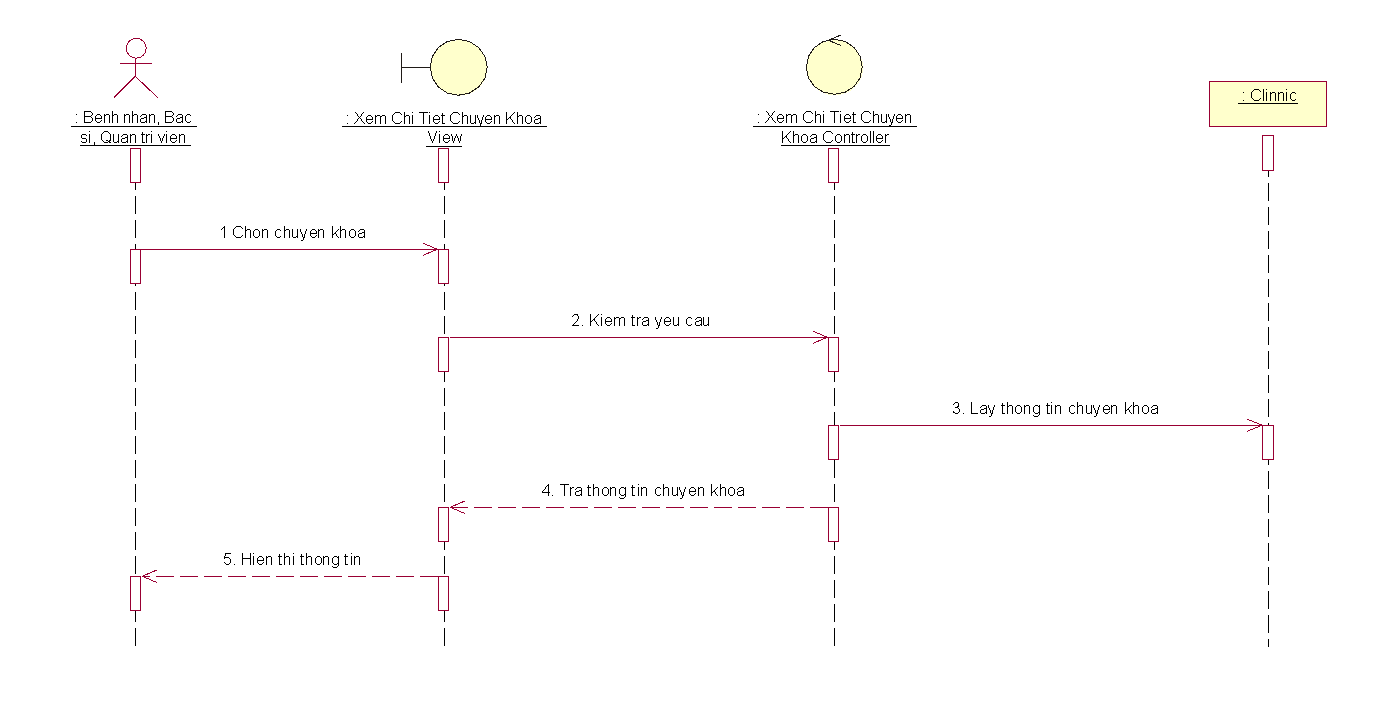
*Hình 2.9. Biểu đồ trình tự hóa xem chi tiết phòng khám*

### 2.4.8 Mô tả usecase Xem chi tiết chuyên khoa

|  |  |
| --- | --- |
| Tên Use Case | Xem chi tiết chuyên khoa |
| Actor | Bác sĩ, Bệnh nhân , Quản trị viên |
| Mô tả | UC cho người dung xem chi tiết thông tin về chuyên khoa |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Không có |
| Luồng sự kiện chính | Người dùng chọn thông tin bất kì trên giao diện   1. Giao diện trang chính 2. Chọn chuyên khoa 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lấy thông tin thành công thì hiển thị cho người dùng. Ngược lại báo lỗi 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 8. Mô tả use case Xem chi tiết chuyên khoa*

Biểu đồ trình tự



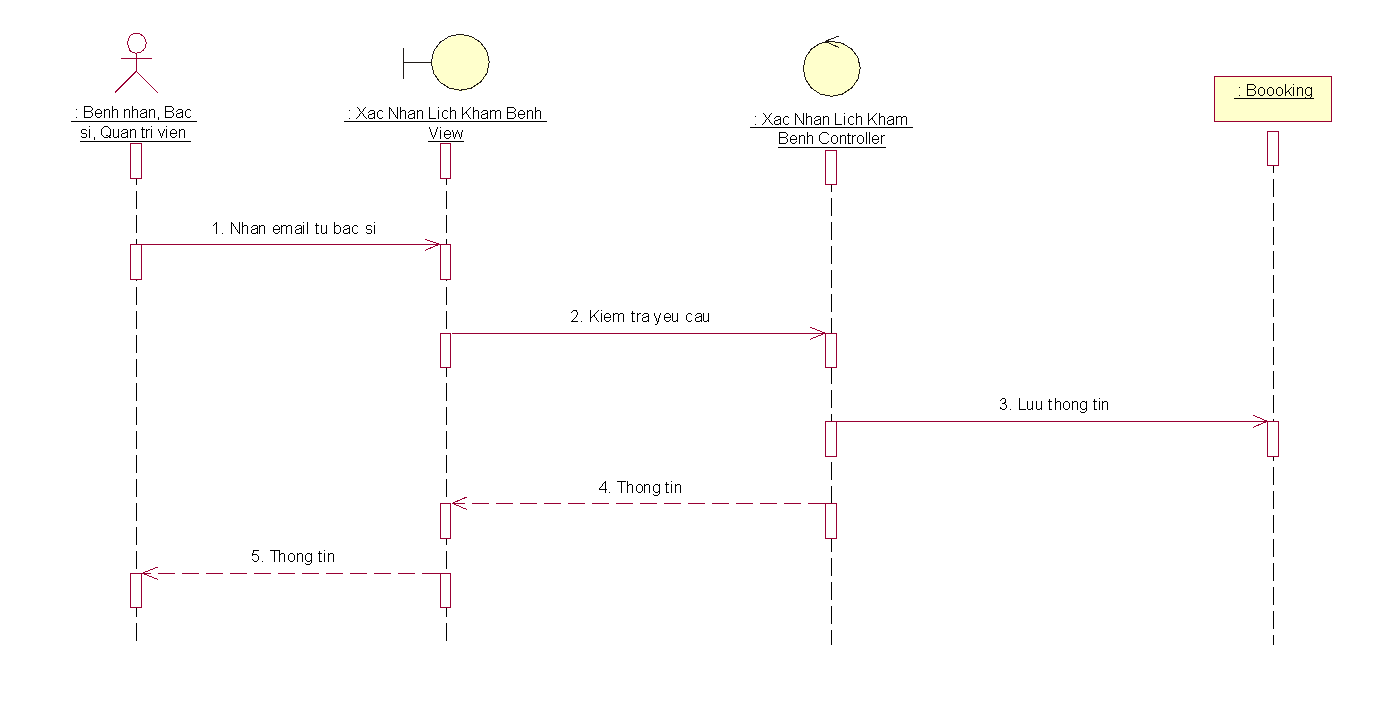
*Hình 2. 9. Biểu đồ tuần tự quy trình xem chi tiết chuyên khoa phòng khám*

### 2.4.9 Mô tả usecase Nhận email xác nhận lịch khám

|  |  |
| --- | --- |
| Tên Use Case | Nhân email xác nhận lịch khám |
| Actor | Bệnh nhân |
| Mô tả | UC cho bệnh nhân xác nhận chính xác lịch khám |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Không có |
| Luồng sự kiện chính | Người dùng chọn thông tin bất kì trên giao diện   1. Kiểm tra email cá nhân check mail nhận được từ bác sĩ 2. Chọn xác nhận 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lấy thông tin thành công thì hiển thị cho người dùng. Ngược lại báo lỗi 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 9. Mô tả ưu case Nhận email xác nhận lịch khám*

Biểu đồ trình tự



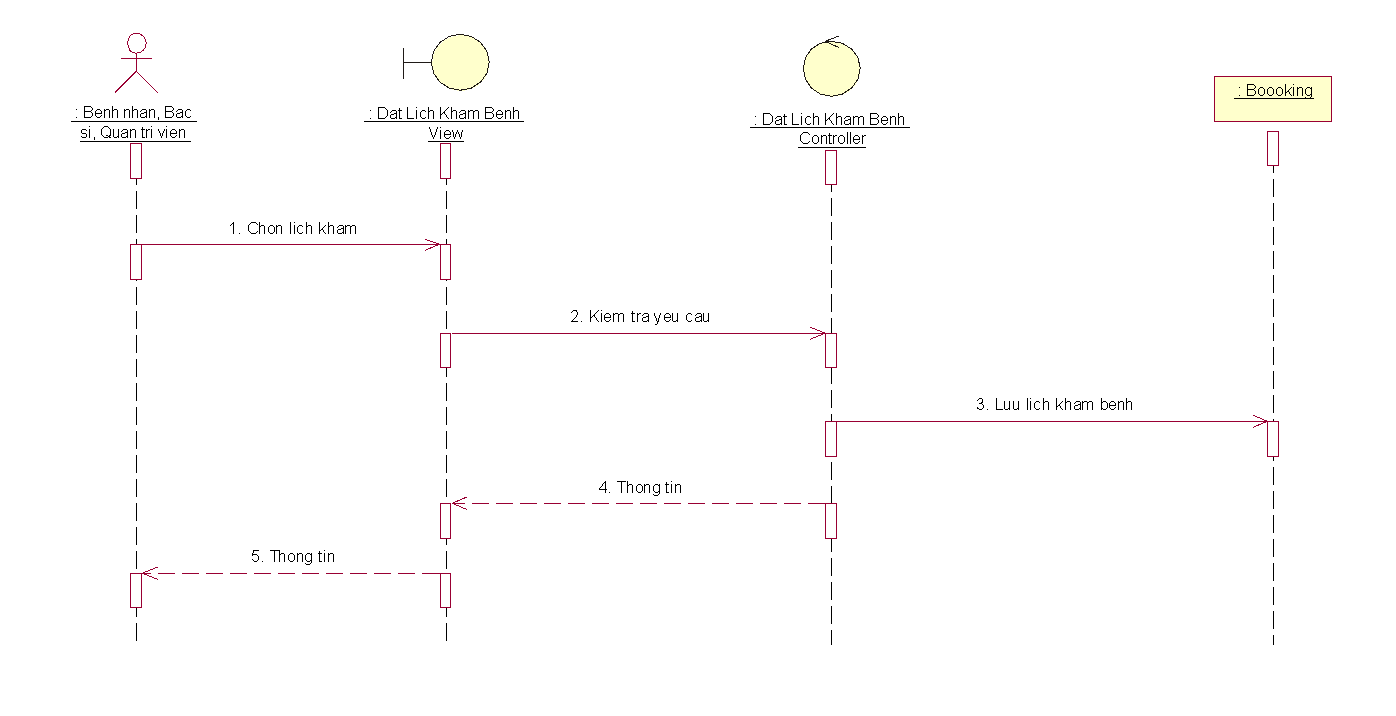
*Hình 2. 11. Biều đồ tuần tự quy trình xác nhận khám*

### 2.4.10 Mô tả usecase Đặt lịch khám bệnh

|  |  |
| --- | --- |
| Tên Use Case | Đặt lịch khám bệnh |
| Actor | Bác sĩ, Bệnh nhân , Quản trị viên |
| Mô tả | UC cho người dung xem chi tiết thông tin về phòng khám |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Không có |
| Luồng sự kiện chính | Người dùng chọn thông tin bất kì trên giao diện   1. Giao diện hiển thị bác sĩ 2. Chọn lịch khám 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lấy thông tin thành công thì hiển thị cho người dùng. Ngược lại báo lỗi 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 10. Mô tả use case Đặt lịch khám bệnh*

Biểu đồ trình tự :

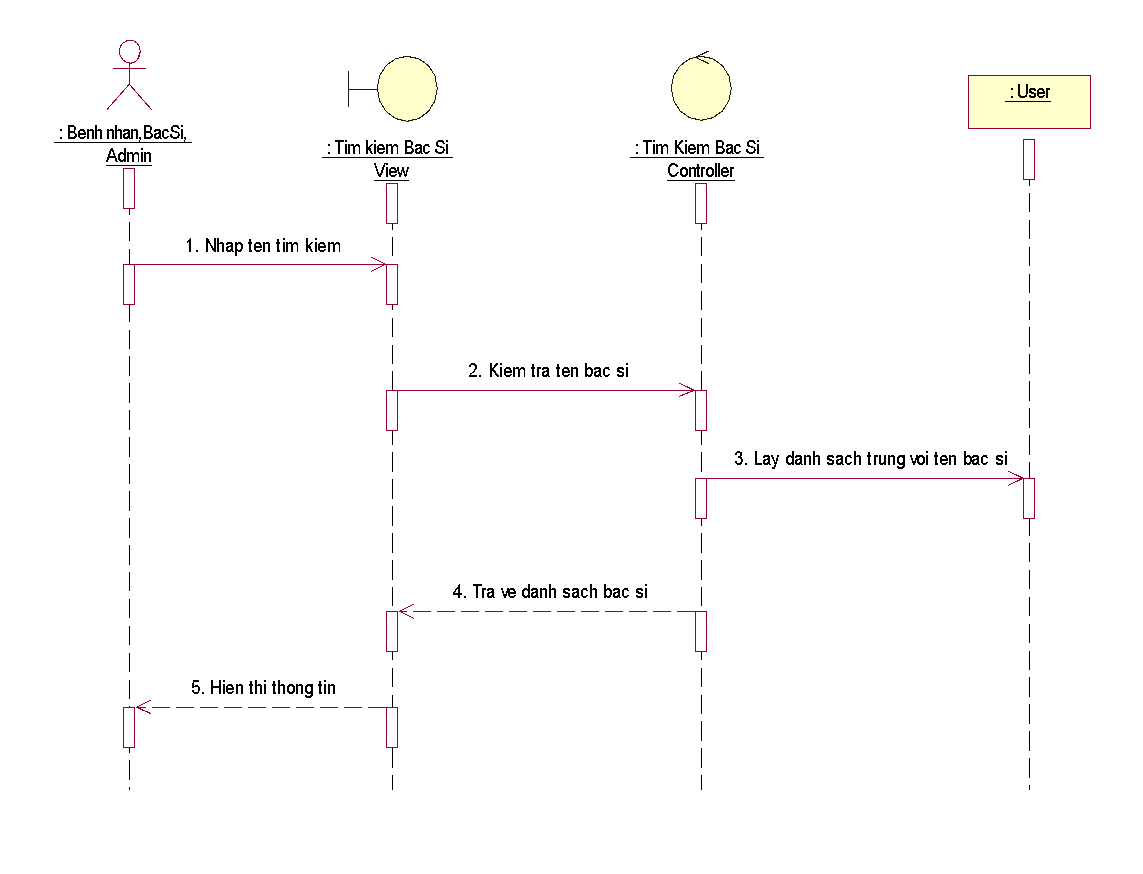


*Hình 2.12. Biểu đồ tuần tự đặt lịch khám bệnh*

### 2.4.11 Mô tả usecase Tìm kiếm bác sĩ

|  |  |
| --- | --- |
| Tên Use Case | Tìm kiếm bác sĩ |
| Actor | Bác sĩ, Bệnh nhân , Quản trị viên |
| Mô tả | UC cho người dùng tìm kiếm bác sĩ |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Không có |
| Luồng sự kiện chính | Người dùng chọn thông tin bất kì trên giao diện   1. Giao diện trang chính 2. Nhập tên bác sĩ 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lấy thông tin thành công thì hiển thị cho người dùng. Ngược lại báo lỗi 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 11.Mô tả use case tìm kiếm bác sĩ*

Biểu đồ trình tự 

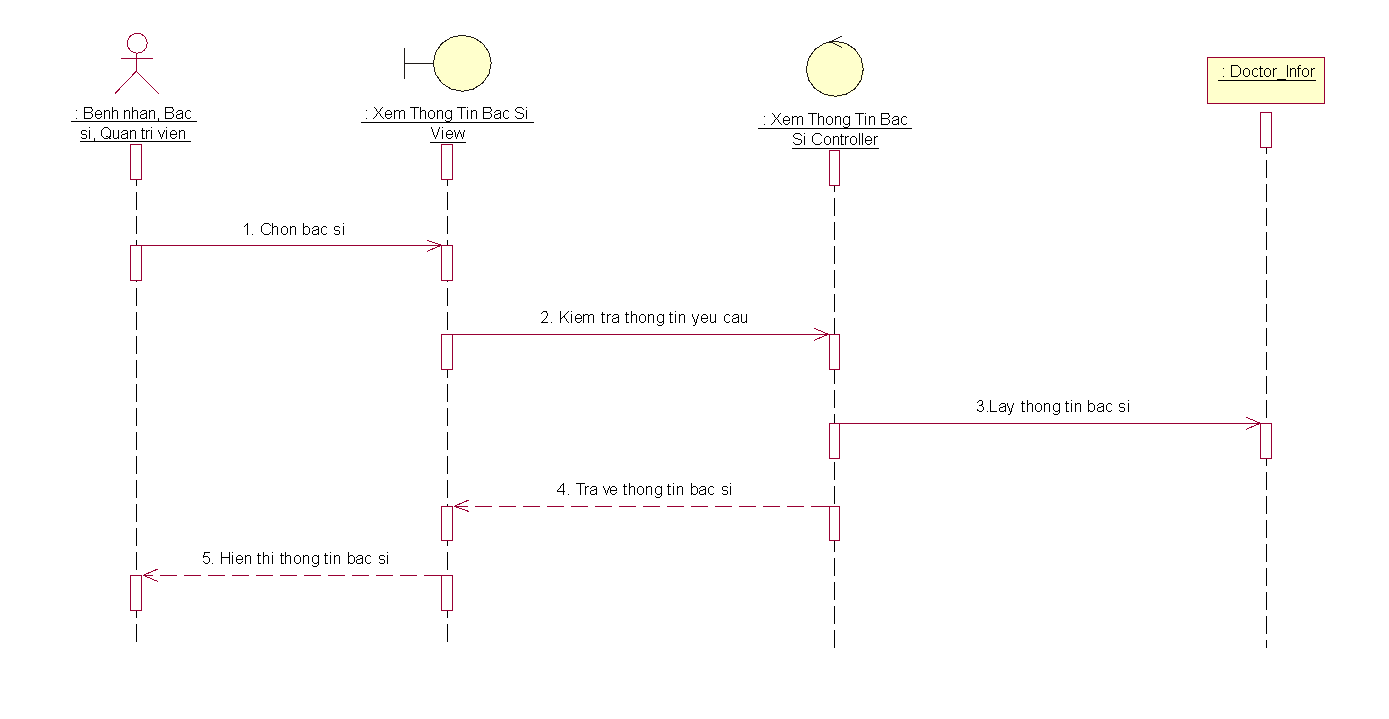
*Hình 2. 13. Biểu đồ tuần tự tìm kiếm bác sĩ*

### 2.4.12 Mô tả usecase Xem thông tin bác sĩ

|  |  |
| --- | --- |
| Tên Use Case | Xem thông tin bác sĩ |
| Actor | Bác sĩ, Bệnh nhân , Quản trị viên |
| Mô tả | UC cho người dung xem chi tiết thông tin về bác sĩ |
| Tiền điều kiện | Thành viên chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Không có |
| Luồng sự kiện chính | Người dùng chọn thông tin bất kì trên giao diện   1. Giao diện trang chính 2. Chọn vào một bác sĩ 3. Hệ thống kiểm tra và thực hiện chức năng 4. Nếu việc lấy thông tin thành công thì hiển thị cho người dùng. Ngược lại báo lỗi 5. UC kết thúc |
| Luồng sự kiện rẽ nhánh. | Không có |

*Bảng 2. 12. Mô tả use case xem thông tin bác sĩ*

Biểu đồ trình tự

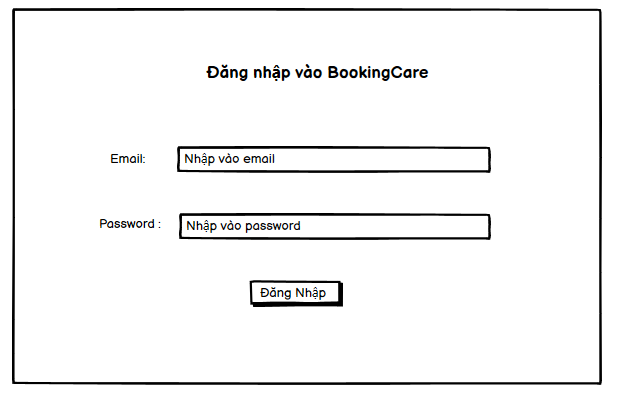


*Hình 2. 14. Biểu đồ tuần tự xem thông tin bác sĩ*

*Hình 2. SEQ Hình\_2. \\* ARABIC 14. Biểu đồ xem thông tin bác sĩ*

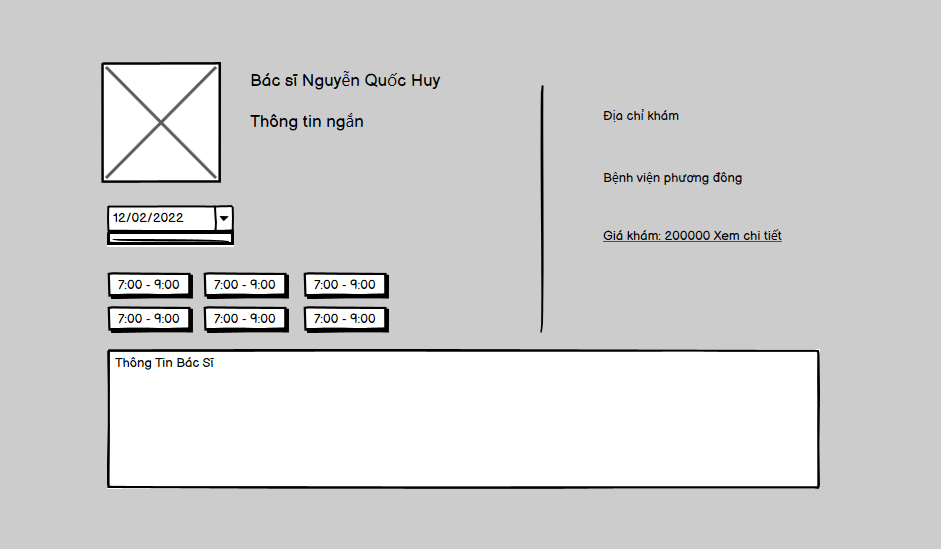
## Thiết kế giao diện

### 2.5.1 Màn hình Đăng nhập

****

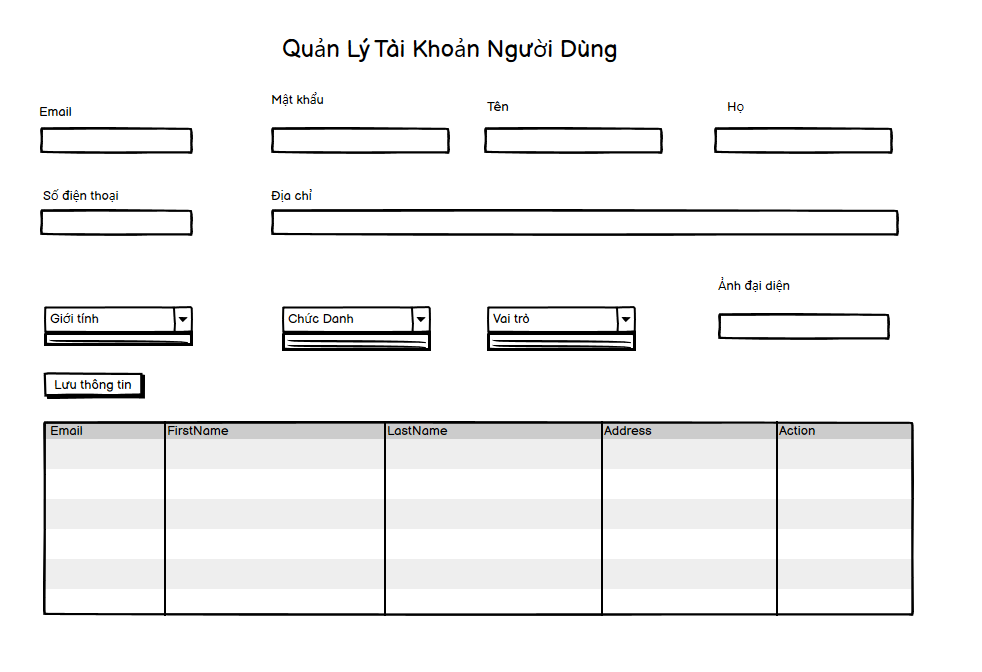
*Hình 2. 15. Thiết kế màn hình đăng nhập*

### 2.5.2 Màn hình thông tin bác sĩ

****

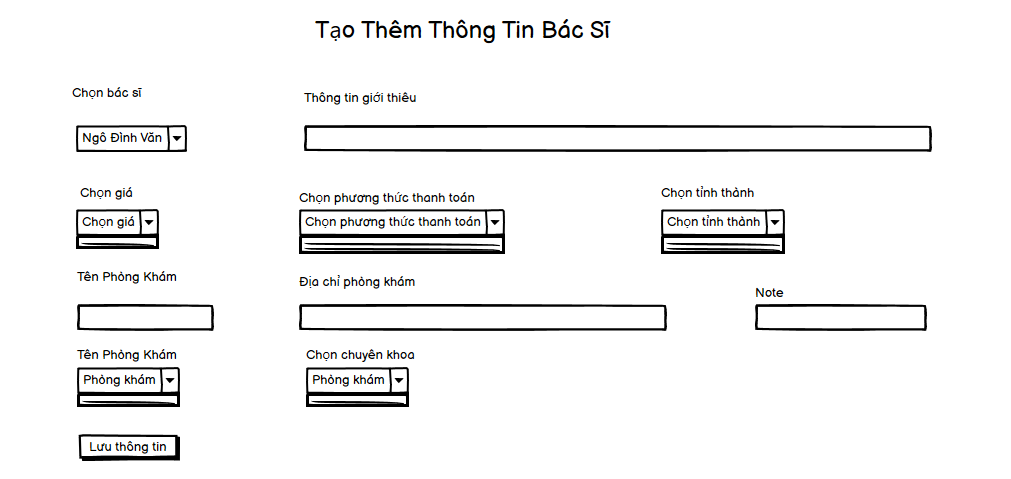
*Hình 2. 16. Màn hình xem thông tin bác sĩ*

### 2.5.3 Màn hình quản lý người dùng

****

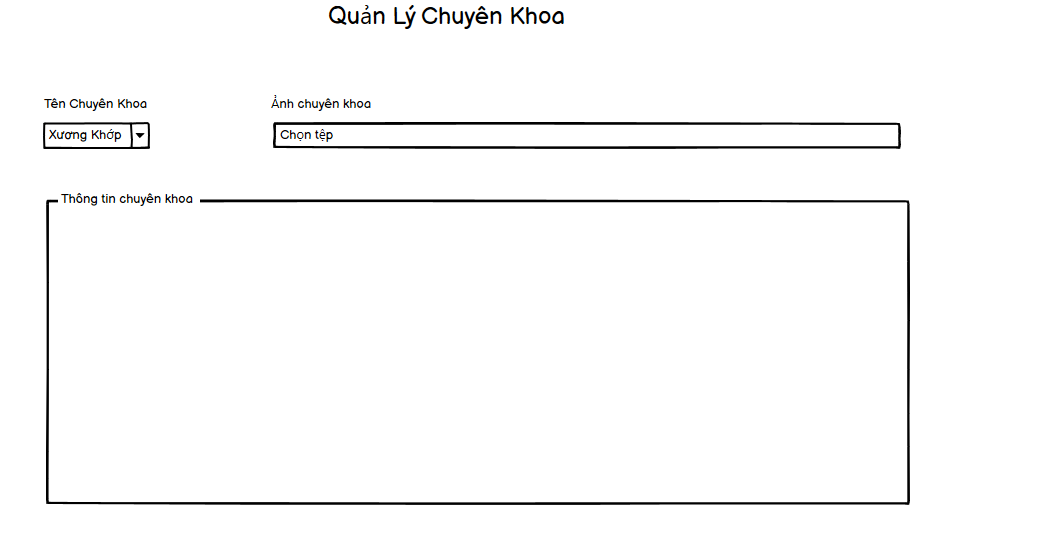
*Hình 2. 17. Thiết kế màn hình quản lý người dùng*

### 2.5.4 Màn hình Tạo thêm thông tin bác sĩ

****

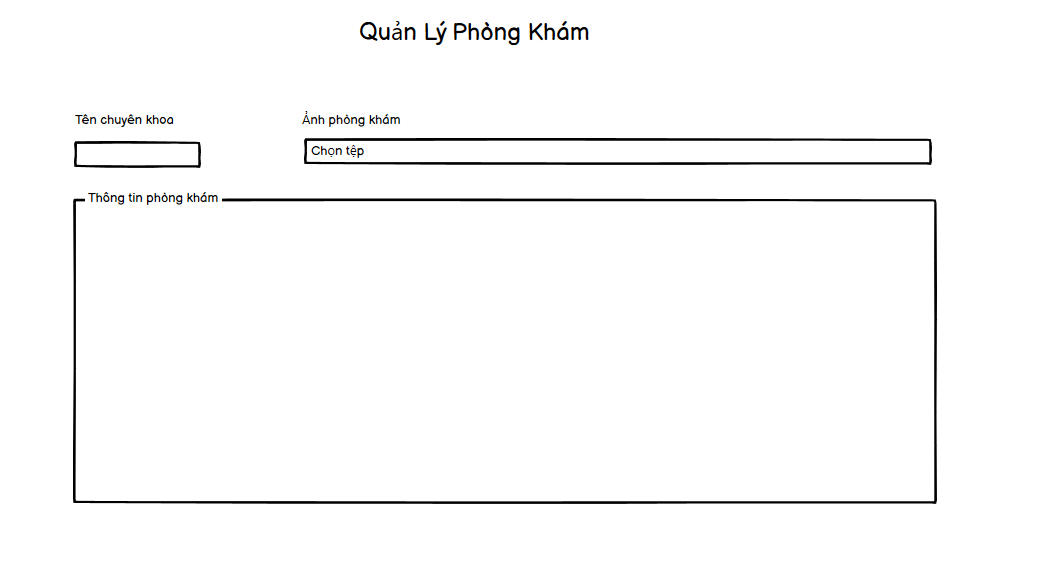
*Hình 2. 18. Thiết kế màn hình thêm thông tin bác sĩ*

### 2.5.5 Màn hình Quản Lý Chuyên Khoa

****

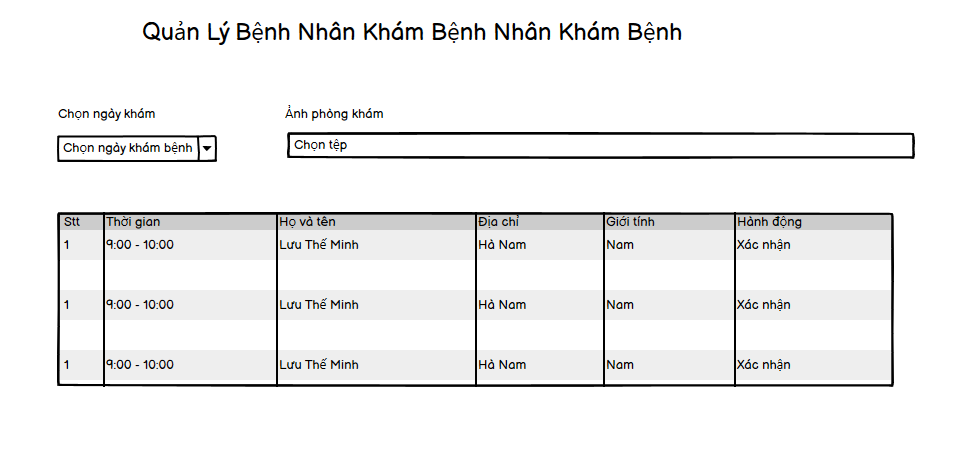
*Hình 2. 19. Thiết kế màn hình quản lý chuyên khoa*

### 2.5.6 Màn hình Xem chi tiết sản phẩm

****

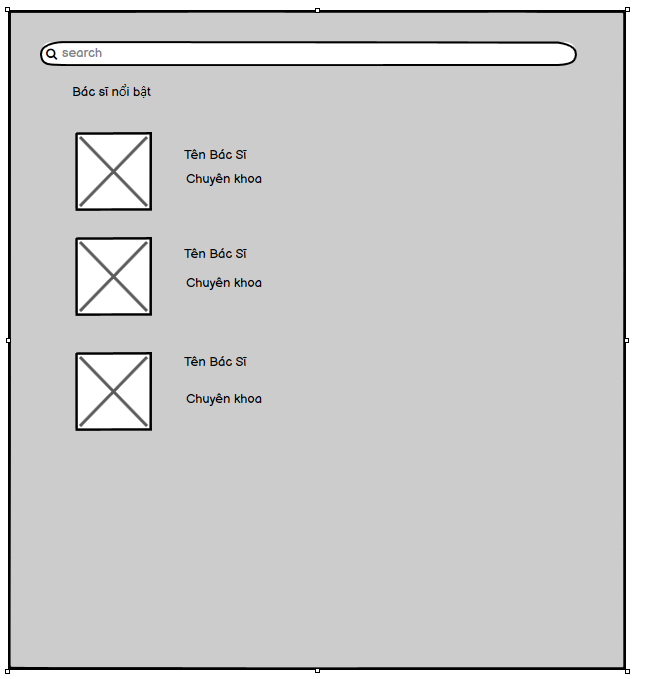
*Hình 2. 20. Thiết kế màn hình xem chi tiết sản phẩm*

### 2.5.7 Màn hình Quản lý bệnh nhân khám bệnh

****

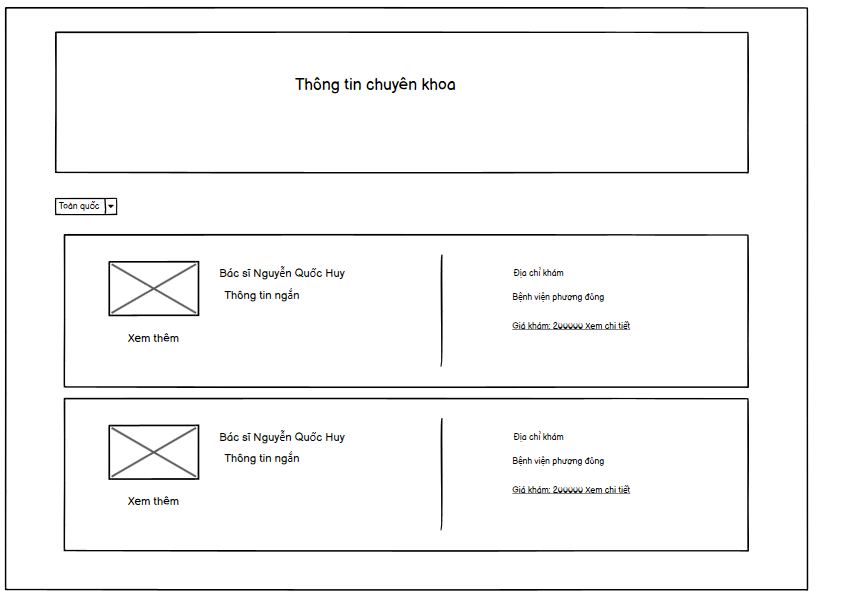
*Hình 2. 21. Thiết kế màn hình quản lý bệnh nhân khám bệnh*

### 2.5.8 Màn hình Tìm Kiếm Bác Sĩ

****

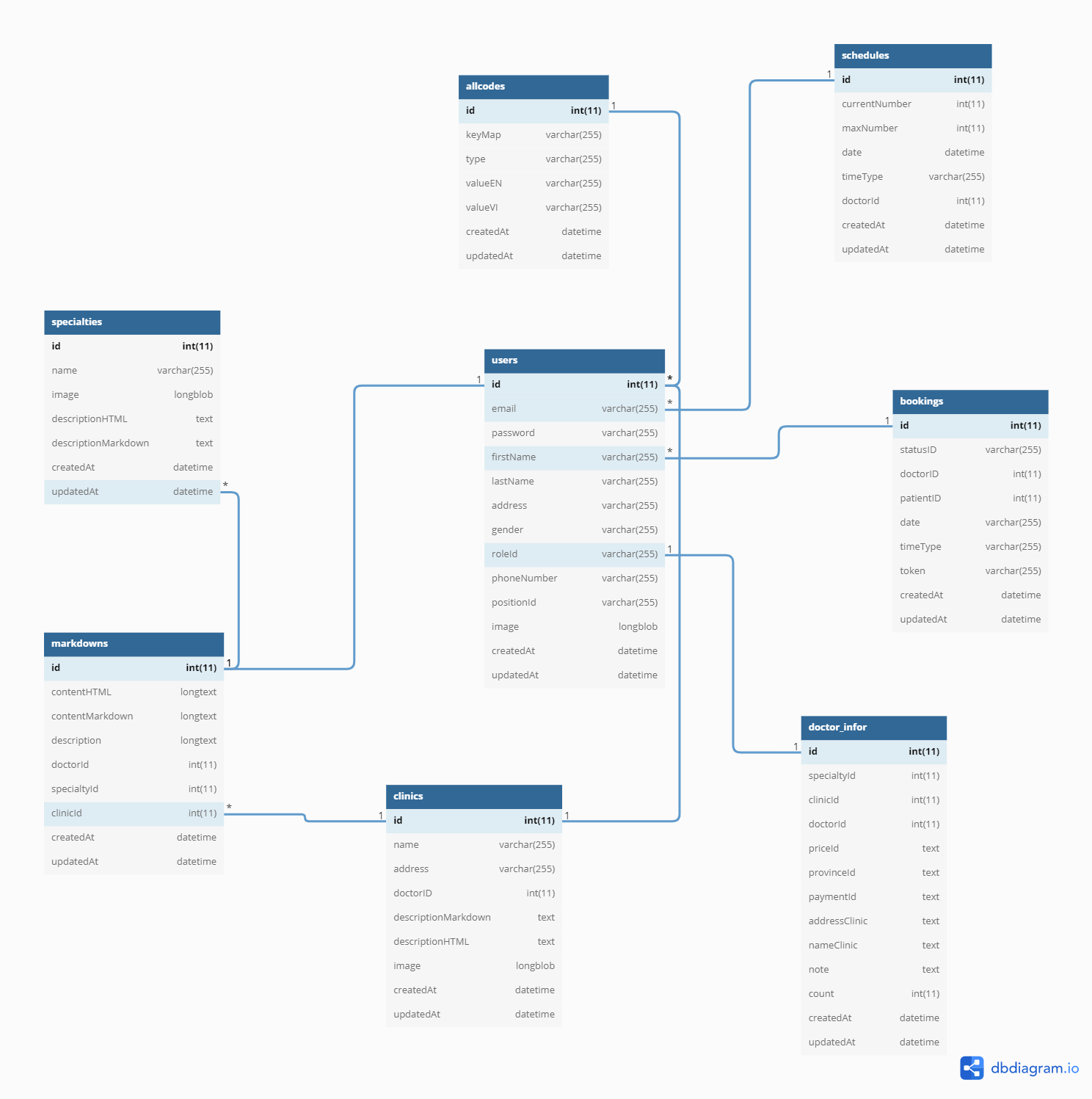
*Hình 2. 22. Thiết kế màn hình tìm kiếm bác sĩ*

### 2.5.9 Màn hình Chi tiết chuyên khoa



*Hình 2. 23. Thiết kế màn hình xem chi tiết thông tin chuyên khoa*

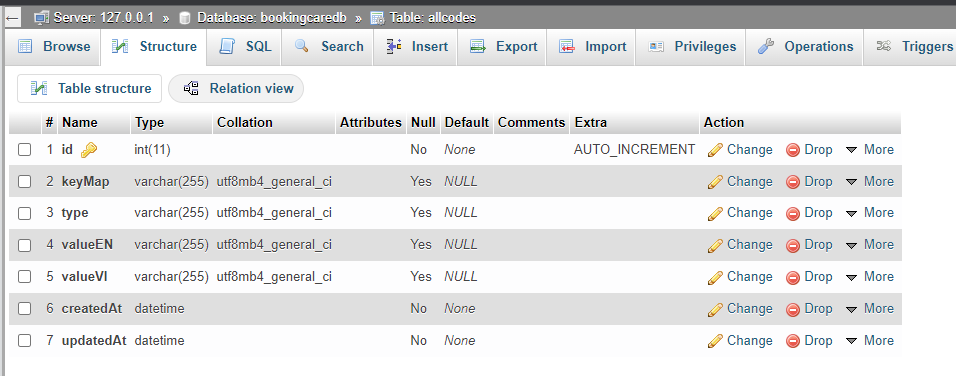
## Thiết kế cơ sở dữ liệu



*Hình 2. 24. Cơ sở dữ liệu*

### 2.6.1 Bảng All Code

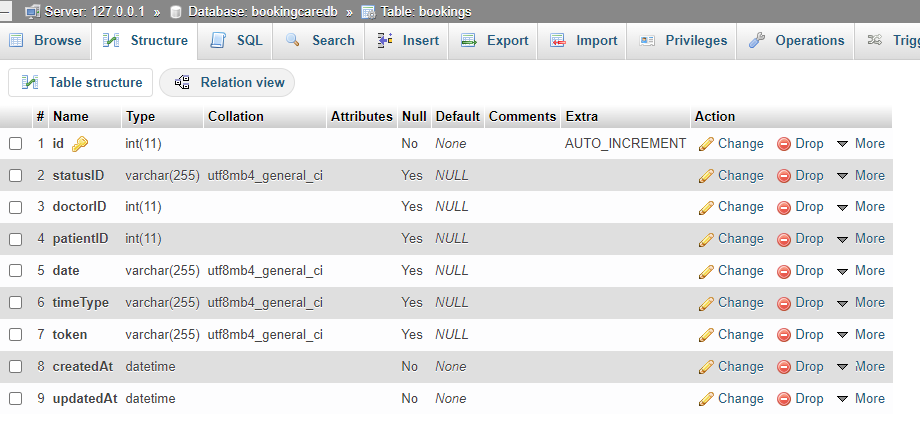
Đối với bảng mã code cần lưu trữ : Id, KeyMap, Type ,Value EN , Value VI



*Hình 2. 25. Bảng All Code*

### 2.6.2 Bảng Bookings

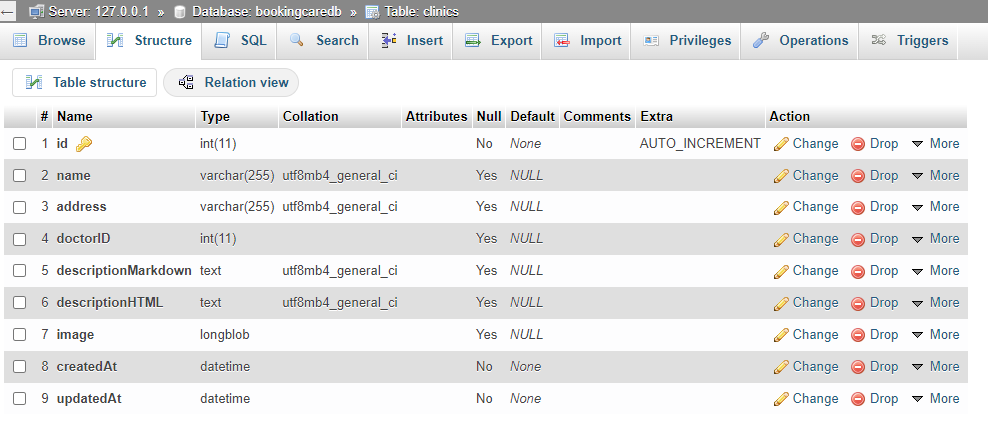
Đối với danh sách đặt lịch cần lưu trữ : Id, StatusId , Doctor Id , Patient Id , Date, Time Type



*Hình 2. 26. Bảng Booking*

### 2.6.3 Bảng Clinics

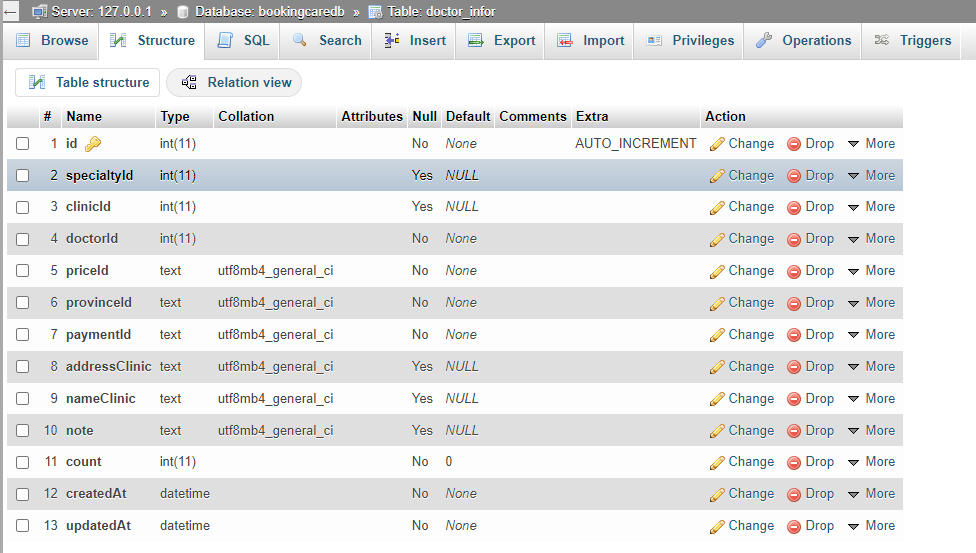
Đối với bảng chuyên khoa khám bệnh cần lưu trữ : Id , Name , Address , Doctor Id , DescriptionMarkdown , Description HTML, Image



*Hình 2. 27. Bảng Clinic*

### 2.6.4 Bảng Doctor Infor

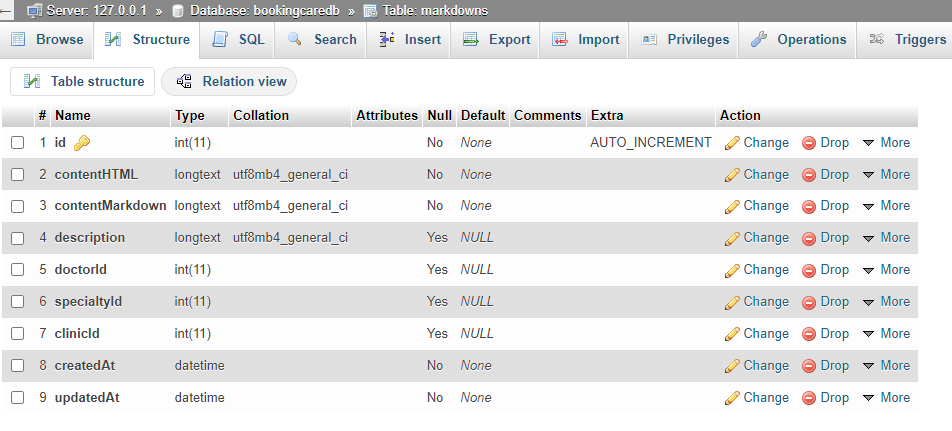
Với thông tin bác sĩ cần cần lưu trữ các thông tin như: Id, Specialty Id , Clinic Id , Doctor Id , price Id , Province Id ,Payment Id , Address Clinic , Name Clinic , Note , Count



*Hình 2. 28. Bảng Doctor\_Infor*

### 2.6.5 Bảng Markdown

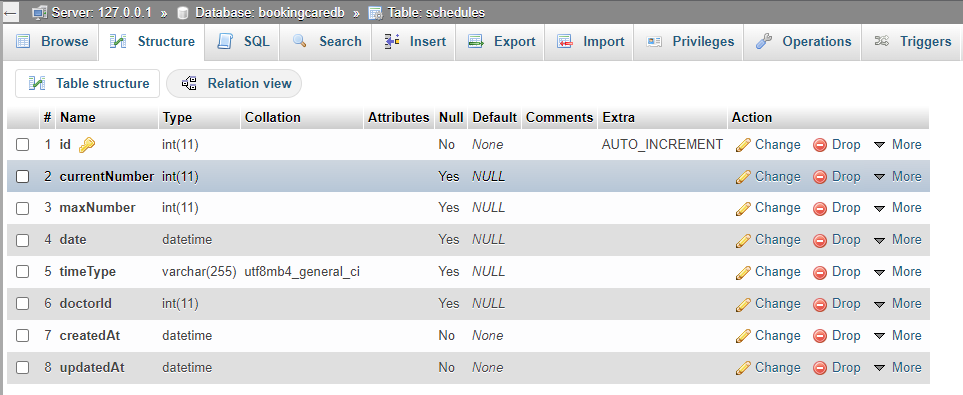
Đối với bảng đánh dấu cần lưu trữ các thông tin như: id , Content HTML , Content Markdown , Description , Doctor Id, Specialty Id , Clinic Id



*Hình 2. 29. Bảng Markdown*

### 2.6.6 Bảng Schedules

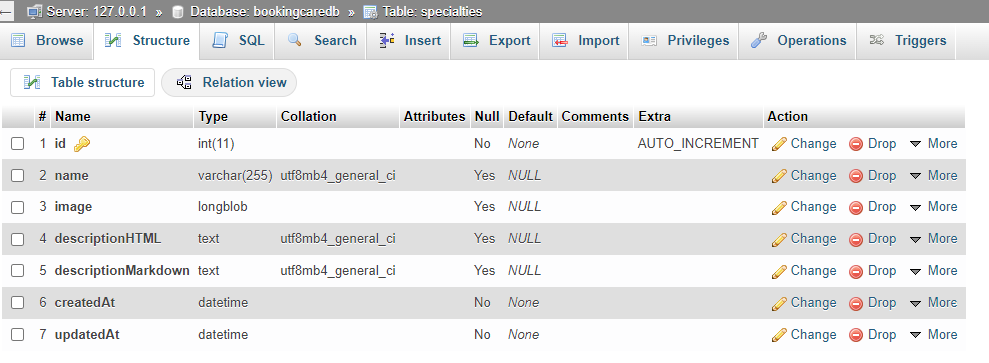
Đối với bảng lịch khám bệnh cần lưu trữ thông tin như : ID, Current Number , MaxNumber , Date , Time Type , DoctorID



*Hình 2. 30. Bảng Schedule*

### 2.6.7 Bảng Specialty

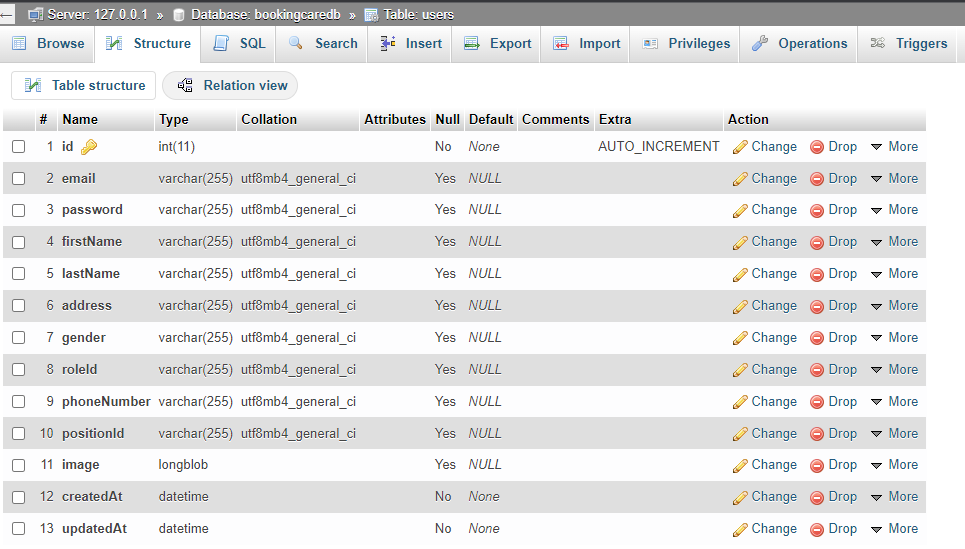
Ta cần lưu trữ thông tin chuyên khoa các thông tin như : id , Name , Image , Description HTML , Description Markdown



*Hình 2. 31. Bảng Specialty*

### 2.6.8 Bảng Users

Đối với bảng người dùng ta cần lưu trữ các thông tin như: id, email, password , First Name , Last Name , Address , Gender , Role Id , Phone Number , Position Id , Image .

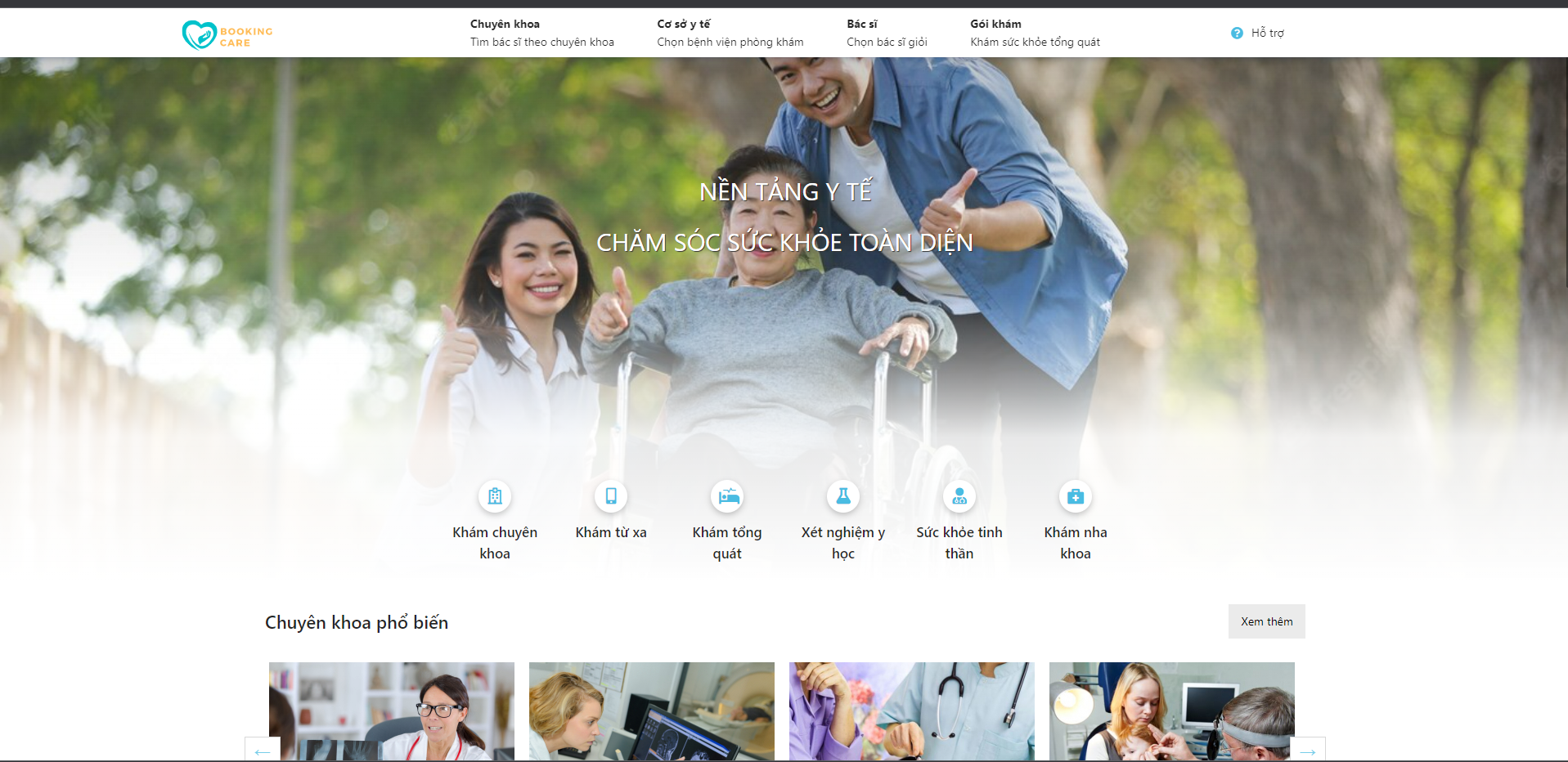


*Hình 2.32. Bảng User*

# Chương 3. KẾT QUẢ

## Kết quả thu được

* + - * Hệ thống với các chức năng:
    - Đăng nhập/ Đăng ký
    - Đặt lịch khám/ Xác nhận lịch khám
    - Quản lý chuyên khoa
    - Quản lý phòng khám
    - Tìm kiếm bác sĩ
    - Quản lý tài khoản người dùng
      * Một số giao diện tiêu biểu của hệ thống:



*Hình 2. 33. Giao diện trang chủ người dùng*



*Hình 2. SEQ Hình\_2. \\* ARABIC 34. Giao diện trang chủ admin*

*Hình 2. 1. Giao diện trang chủ admin*

**KẾT LUẬN**

## Kết quả đạt được

* + - * Nắm vững được kiến thức về thiết kế và xây dựng website với reactjs và nodejs
      * Xây dựng được hệ thống cung cấp đầy đủ các chức năng cơ bản của một website đặt lịch.
      * Giao diện thân thiện, dễ nhìn giúp người dùng dễ dàng lựa chọn được những sản phẩm theo yêu cầu.
      * Chức năng tạo lịch đặt khám chữa bệnh cho bác sĩ có thể xem được thông tin bệnh nhân đặt lịch giúp quản lý đầy đủ về người dùng.
      * Chức năng lịch chờ giúp quan sát tốt hơn về thông tin đặt lịch của bệnh nhân.

## 2. Hạn chế của đề tài

* + - * Ngoài những vấn đề đạt được vẫn còn những vấn đề vẫn chưa được giải quyết:
* Chưa xuất được hồ sơ qua thông tin bệnh nhân để giảm quá trình thủ tục.
* Hệ thống trang quản trị còn nhiều thiếu sót.
* Chưa cung cấp đầy đủ chức năng cho bệnh nhân và bác sĩ.
* Giao diện còn đơn giản và thiếu thu hút.
* Cơ sở dữ liệu còn nhiều thiếu sót.
* Hệ thống bảo mật chưa cao.

## 3. Hướng phát triển đề tài

* + - * Xây dựng thêm các chức năng còn thiếu:
* Xây dựng giao diện đẹp mắt, sinh động hơn, cơ sở dữ liệu nhất quán hơn.
* Xậy dựng thêm các tính năng phân loại bệnh nhân để gợi ý các bác sĩ và chuyên khoa tốt hơn.
* Xây dựng các tính năng thống kê cho bác sĩ và người quản trị.
* Xây dựng thêm tính năng chuẩn đoán bệnh cho bệnh nhân

# TÀI LIỆU THAM KHẢO

[1]. TopDev (2017),” RESTful API là gì?”, <https://topdev.vn/blog/restful-api-la-gi/>

[2]. Vũ Thị Dương, Phùng Đức Hòa, Nguyễn Thị Hương Lan (2015), “Giáo trình Phân tích thiết kế hướng đối tượng”, NXB Khoa học và Kỹ thuật.

[3]. Viblo (2018), “Giới thiệu về react js ”, <https://viblo.asia/p/gioi-thieu-ve-react-js-4dbZN7W8lYM>

[4] [Monamedia], “ Phần mềm Xampp là gì? Tổng quan về Xampp và hướng dẫn cài đặt” , <https://monamedia.co/phan-mem-xampp-la-gi-huong-dan-cai-dat-xampp/>

[5] [Monamedia], “ Phần mềm Xampp là gì? Tổng quan về Xampp và hướng dẫn cài đặt” , <https://monamedia.co/phan-mem-xampp-la-gi-huong-dan-cai-dat-xampp/>