TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

--------------------------------

HUỲNH LINH KHÔI

**BÁO CÁO THỰC TẬP TỐT NGHIỆP**

**CÔNG TY CỔ PHẦN DU LỊCH BẠC LIÊU**

**BÁO CÁO THỰC TẬP TỐT NGHIỆP**

Ngành: Công Nghệ Thông Tin

Hướng chuyên ngành: Công Nghệ Thông Tin

Lớp: DH18IT82

NGƯỜI HƯỚNG DẪN KHOA HỌC

ThS. Dương Hữu Thành

Thành phố Hồ Chí Minh – Năm 2021

Chương 1. Giới thiệu cơ quan thực tập

Giới thiệu

Cơ cấu tổ chức

Lịch làm việc

Những việc được phân công sthực tập tại cty

Chương 2. Nội dung lý thuyết

2.1.Cơ sở lý thuyết

2.2.Những việc phân công

……….

Chương 3. Kết luận

TP. Hồ Chí Minh, ngày 09 tháng 10 năm 2021

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với các thầy cô khoa Công nghệ thông tin của trường Đại học Mở TP.HCM, đặc biệt là thày Dương Hữu Thành đã giúp đỡ để em có thể hoàn thành tốt Đồ án ngành, và em cũng xin chân thành cám ơn thầy đã nhiệt tình hướng dẫn hướng dẫn em trong việc lựa chọn đề tài nghiên cứu, hướng tiếp cận và giúp em chỉnh sửa những thiếu sót trong quá trình nghiên cứu. Trong quá trình nghiên cứu, khó tránh khỏi sai sót, rất mong các Thầy, Cô bỏ qua và giúp em hoàn thiện hơn cho em. Đồng thời do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên bài luận văn không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp từ phía Thầy, Cô để em học thêm được nhiều kinh nghiệm và sẽ hoàn thành tốt hơn.

Em xin chân thành cảm ơn!

TÓM TẮT ĐỒ ÁN

**Đề tài: Phát triển hệ thống phòng mạch tư dựa trên Spring Framework**

1. **Lý do chọn đề tài**

Xã hội hiện nay đang trên đà phát triển và việc ứng dụng công nghệ thông tin là điều cần thiết. Chính vì vậy em chọn đề tài “Phát triển hệ thống phòng mạch tư dựa trên Spring Framework” nhằm giúp cho các phòng mạch tư hiện nay có thể ứng dụng CNTT vào kinh doanh nhằm tăng doanh thu. Ngoài ra trang web còn có chức năng đặt phiếu khám online nhằm giúp cho bệnh nhân không phí thời gian mà vẫn điều trị được bệnh của mình.

1. **Các vấn đề nghiên cứu**

Ngôn ngữ lập trình Java: lịch sử hình thành Java, các khái nhiệm, lập trình hướng đối tượng,…

Spring Framework: lịch sử hình thành Spring Framework, các khái niệm, tìm hiểu các module của Spring, cách tạo một dự án Spring MVC.

Hibernate: lịch sử hình thành Hibernate, các ưu điểm của hibernate và ứng dụng vào dự án Spring MVC

MySQL: khái niệm về MySQL, kết hợp MySQL và hibernate vào một dự án.

Cloudinary: khái niệm về Cloudinary, ứng dụng Cloudinary vào lưu trữ và sử dụng hình ảnh.

1. **Quá trình thực hiện**

Phân tích hệ thống phòng mạch tư: tạo các sơ đồ use case, đặc tả hệ thống, sơ đồ tuần tự, sơ đồ hoạt động, sơ đồ lớp, sơ đồ cơ sở dữ liệu để hỗ trợ quá trình tạo dự án

Thiết kế giao diện: hình thành nên dự án web phòng mạch tư sau khi đã xây dựng được các nền tảng đã có.

1. **Kết quả và hướng phát triển**

Kết quả của việc xây dựng hệ thống phòng mạch tư dựa trên nền tảng Spring Framework.

Xem xét các thiếu sót và từ đó có hướng phát triển của đề tài.

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

**MỤC LỤC**

[DANH MỤC TỪ VIẾT TẮT 7](#_Toc84778449)

[DANH MỤC HÌNH VẼ 8](#_Toc84778450)

[DANH MỤC BẢNG 9](#_Toc84778451)

[MỞ ĐẦU 10](#_Toc84778452)

[Chương 1. TỔNG QUAN 11](#_Toc84778453)

[1.1. Lý do chọn đề tài 11](#_Toc84778454)

[1.2. Mục đích nghiên cứu 11](#_Toc84778455)

[1.3. Mục tiêu đề tài 11](#_Toc84778456)

[1.3.1. Admin: 11](#_Toc84778457)

[1.3.2. Bác sĩ: 11](#_Toc84778458)

[1.3.3. Y tá: 12](#_Toc84778459)

[1.3.4. Bệnh nhân: 12](#_Toc84778460)

[Chương 2. TÌM HIỂU VỀ CƠ SỞ LÝ THUYẾT 13](#_Toc84778461)

[2.1. Tổng quan về ngôn ngữ Java 13](#_Toc84778462)

[2.1.1. Lịch sử phát triển của Java 13](#_Toc84778463)

[2.2. Tổng quan về Spring Framework 13](#_Toc84778464)

[2.2.1. Tổng quan về Spring 13](#_Toc84778465)

[2.2.2. Tổng quan về Spring MVC 14](#_Toc84778466)

[2.2.3. Các khái nhiệm trong Spring 14](#_Toc84778467)

[2.2.4. Ưu điểm của Spring MVC 15](#_Toc84778468)

[2.2.5. Tạo dự án với Spring MVC với NetBeans 15](#_Toc84778469)

[2.3. Tổng quan về Hibernate 20](#_Toc84778470)

[2.3.1. Lịch sử Hibernate 20](#_Toc84778471)

[2.3.2. Giới thiệu Hibernate Framework 20](#_Toc84778472)

[2.3.3. Lợi ích của việc sử dụng Hibernate 20](#_Toc84778473)

[2.4. Tổng quan về MySQL 21](#_Toc84778474)

[2.4.1. Giới thiệu về MySQL 21](#_Toc84778475)

[2.4.2. Lịch sử của MySQL 21](#_Toc84778476)

[2.4.3. Ưu điểm khi sử dụng MySQL 21](#_Toc84778477)

[2.4.4. Kết nối MySQL vào dự án Spring thông qua Hibernate 22](#_Toc84778478)

[2.5. Tổng quan về Cloudinary 25](#_Toc84778479)

[2.5.1. Giới thiệu Cloudinary 25](#_Toc84778480)

[2.5.2. Cài đặt Cloudinary cho dự án Spring 25](#_Toc84778481)

[2.6. Tổng quan về hệ thống ví điện tử ZaloPay 26](#_Toc84778482)

[2.6.1. Giới thiệu ZaloPay 26](#_Toc84778483)

[2.6.2. Tích hợp ZaloPay vào dự án Spring MVC 27](#_Toc84778484)

[Chương 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 32](#_Toc84778485)

[3.1. Phân tích thiết kế các chức năng ứng dụng 32](#_Toc84778486)

[3.1.1. Mô tả chức năng ứng dụng 32](#_Toc84778487)

[3.1.2. Sơ đồ Use Case: 33](#_Toc84778488)

[3.1.3. Sơ đồ tuần tự: 42](#_Toc84778489)

[3.1.4. Sơ đồ hoạt động: 46](#_Toc84778490)

[3.1.5. Sơ đồ lớp: 51](#_Toc84778491)

[3.1.6. Mô hình cơ sở dữ liệu: 51](#_Toc84778492)

[3.2. Thiết kế giao diện 56](#_Toc84778493)

[3.2.1. Giao diện Admin: 56](#_Toc84778494)

[3.2.2. Giao diện Bác sĩ: 59](#_Toc84778495)

[3.2.3. Giao diện y tá: 62](#_Toc84778496)

[3.2.4. Giao diện bệnh nhân: 65](#_Toc84778497)

[Chương 4. TỔNG KẾT 68](#_Toc84778498)

[4.1. Kết quả đạt được: 68](#_Toc84778499)

[4.2. Hoàn thành: 69](#_Toc84778500)

[4.3. Hướng phát triển: 70](#_Toc84778501)

[TÀI LIỆU THAM KHẢO 71](#_Toc84778502)

DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Ký hiệu chữ viết tắt | Chữ viết đầy đủ |
| 1 | CSDL | Cơ sở dữ liệu |
| 2 | CNTT | Công nghệ thông tin |
| 3 | SQL | Structured Query Language |
| 4 | MVC | Model - View - Controller |

DANH MỤC HÌNH VẼ

[Hình 2.1: Hình module của Spring framework 10](#_Toc367742554)

[Hình 2.2: Hình sơ đồ luồng xử lý của Spring MVC 11](#_Toc367742554)

[Hình 3.1: Hình sơ đồ use-case của Admin 16](#_Toc367742554)

[Hình 3.2: Hình sơ đồ use-case của bác sĩ 17](#_Toc367742554)

[Hình 3.3: Hình sơ đồ use-case của y tá 17](#_Toc367742554)

[Hình 3.4: Hình sơ đồ use-case của bệnh nhân 18](#_Toc367742554)

[Hình 3.5: Hình sơ đồ tuần tự Đăng nhập 25](#_Toc367742554)

[Hình 3.6: Hình sơ đồ tuần tự Đăng ký 26](#_Toc367742554)

[Hình 3.7: Hình sơ đồ tuần tự chọn phiếu khám vào toa thuốc 26](#_Toc367742554)

[Hình 3.8: Hình sơ đồ tuần tự Thêm thuốc vào toa 27](#_Toc367742554)

[Hình 3.9: Hình sơ đồ tuần tự Thêm toa thuốc 27](#_Toc367742554)

[Hình 3.10: Hình sơ đồ tuần tự Thống kê doanh thu năm 28](#_Toc367742554)

[Hình 3.11: Hình sơ đồ hoạt động Đăng nhập 29](#_Toc367742554)

[Hình 3.12: Hình sơ đồ hoạt động Xác nhận phiếu khám 30](#_Toc367742554)

[Hình 3.13: Hình sơ đồ hoạt động Thêm toa thuốc 31](#_Toc367742554)

[Hình 3.14: Hình sơ đồ hoạt động Thanh toán hoá đơn 32](#_Toc367742554)

[Hình 3.15: Hình sơ đồ hoạt động Thống kê doanh thu tháng 33](#_Toc367742554)

[Hình 3.16: Hình sơ đồ lớp hệ thống 34](#_Toc367742554)

[Hình 3.17: Hình sơ đồ CSDL hệ thống 38](#_Toc367742554)

[Hình 3.18: Hình giao diện quản lý thuốc 39](#_Toc367742554)

[Hình 3.19: Hình giao diện thêm thuốc 40](#_Toc367742554)

[Hình 3.20: Hình giao diện thống kê số lượng bệnh nhân 41](#_Toc367742554)

[Hình 3.21: Hình giao diện xem danh sách bệnh nhân 42](#_Toc367742554)

[Hình 3.22: Hình giao diện thêm thuốc vào toa 42](#_Toc367742554)

[Hình 3.23: Hình giao diện check toa thuốc 43](#_Toc367742554)

[Hình 3.24: Hình giao diện chọn toa thuốc tạo hoá đơn 43](#_Toc367742554)

[Hình 3.25: Hình giao diện thêm hoá đơn 44](#_Toc367742554)

[Hình 3.26: Hình giao diện thanh toán zalopay 45](#_Toc367742554)

[Hình 3.27: Hình giao diện đăng nhập 46](#_Toc367742554)

[Hình 3.28: Hình giao diện đăng ký 47](#_Toc367742554)

[Hình 3.29: Hình giao diện tạo phiếu khám 48](#_Toc367742554)

DANH MỤC BẢNG

[Bảng 3.1: Bảng danh sách use-case 19](#_Toc367742567)

[Bảng 3.2: Bảng danh sách tác nhân 20](#_Toc367742568)

[Bảng 3.3: Bảng đặc tả use case “Đăng nhập” 21](#_Toc367742568)

[Bảng 3.4: Bảng đặc tả use case “Đăng ký” 21](#_Toc367742568)

[Bảng 3.5: Bảng đặc tả use case “Đặt phiếu khám” 22](#_Toc367742568)

[Bảng 3.6: Bảng đặc tả use case “Lên toa thuốc” 22](#_Toc367742568)

[Bảng 3.7: Bảng đặc tả use case “Thêm hoá đơn” 23](#_Toc367742568)

[Bảng 3.8: Bảng đặc tả use case “Thanh toán” 24](#_Toc367742568)

[Bảng 3.9: Bảng đặc tả use case “Thống kê doanh thu theo năm” 24](#_Toc367742568)

[Bảng 3.10: Bảng CSDL “User” 34](#_Toc367742568)

[Bảng 3.11: Bảng CSDL “Doctor” 35](#_Toc367742568)

[Bảng 3.12: Bảng CSDL “Nurse” 35](#_Toc367742568)

[Bảng 3.13: Bảng CSDL “Patient” 35](#_Toc367742568)

[Bảng 3.14: Bảng CSDL “Sick” 35](#_Toc367742568)

[Bảng 3.15: Bảng CSDL “Medicine” 3](#_Toc367742568)5

[Bảng 3.16: Bảng CSDL “Appointment” 36](#_Toc367742568)

[Bảng 3.17: Bảng CSDL “Presscription” 36](#_Toc367742568)

[Bảng 3.18: Bảng CSDL “PrescriptionDetail” 36](#_Toc367742568)

[Bảng 3.19: Bảng CSDL “Order” 37](#_Toc367742568)

[Bảng 3.20: Bảng CSDL “MedicalRecord” 37](#_Toc367742568)

[Bảng 3.20: Bảng CSDL “ScheduleDoctor” 37](#_Toc367742568)

[Bảng 3.20: Bảng CSDL “ScheduleNurse” 37](#_Toc367742568)

MỞ ĐẦU

Hiện nay tình hình dịch covid-19 vẫn còn diễn biến khó lường, việc di chuyển khám chữa bệnh cũng rất khó khăn. Tuy nhiên nhu cầu khám chữa bệnh của mọi người tăng cao, cùng với sự phát triển vượt bậc của công nghệ thông tin. Nên em chọn đề tài “Phát triển hệ thống phòng mạch tư dựa trên Spring Framework” nhằm để ứng dụng công nghệ thông tin vào lĩnh vực y tế. Em tin đề tài này sẽ giúp ích được mọi người trong việc khám bệnh và giúp các phòng mạch tư có thể quản lý tốt hơn.

# TỔNG QUAN

## Lý do chọn đề tài

Xã hội hiện nay đang trên đà phát triển và việc ứng dụng công nghệ thông tin là điều cần thiết. Chính vì vậy em chọn đề tài “Phát triển hệ thống phòng mạch tư dựa trên Spring Framework” nhằm giúp cho các phòng mạch tư hiện nay có thể ứng dụng CNTT vào kinh doanh nhằm tăng doanh thu. Ngoài ra trang web còn có chức năng đặt phiếu khám online nhằm giúp cho bệnh nhân không phí thời gian mà vẫn điều trị được bệnh của mình.

## Mục đích nghiên cứu

Đồ án này sẽ tập trung trình bày nghiên cứu về các nội dung:

Tìm hiểu và ứng dụng công nghệ Spring framework.

Tìm hiểu và ứng dụng công nghệ Hibernate trên Java để tương tác CSDL.

Tìm hiểu và ứng dụng hệ quản trị cơ sở dữ liệu MySQL.

Ứng dụng thanh toán trực tuyến qua ví điện tử ZaloPay.

Ứng dụng nền tảng lưu trữ Cloudinary trong việc lưu trữ ảnh đám mây.

## Mục tiêu đề tài

Đề tài sẽ tập trung xây dựng trang web cho cả 4 đối tượng người dùng:

### Admin:

Admin có quyền quản lý (thêm/xoá/sửa/tra cứu) các thông tin của bác sĩ, y tá, sắp lịch trực cho từng đối tượng, thuốc, bệnh.

Thống kê báo cáo số lượng bệnh nhân đến khám, doanh thu theo tháng, quý và năm.

### Bác sĩ:

Bác sĩ có thể lên toa thuốc cho bệnh nhân, trong quá trình này, bác sĩ sẽ tra cứu danh sách thuốc để thêm vào toa.

Xem lịch sử khám và bệnh của bệnh nhân để tiện có việc chuẩn đoán bệnh.

### Y tá:

Xác nhận phiếu khám được bệnh nhân đặt trên trang web đồng thời sẽ tự động gửi mail khi xác nhận thành công.

Y tá sẽ thanh toán tiền khám của bệnh nhân dựa trên toa thuốc và phiếu khám, y tá sẽ sử dụng thanh toán online bằng ví điện tử ZaloPay cho khách hàng thanh toán.

### Bệnh nhân:

Bệnh nhân có thể đặt phiếu khám bệnh trực tuyến trên trang web của bệnh viện, bệnh nhân sẽ nhận được mail sau khi xác nhận thành công

Bệnh nhân có thể thanh toán phí khám bằng ví điện tử ZaloPay

# TÌM HIỂU VỀ CƠ SỞ LÝ THUYẾT

## Tổng quan về ngôn ngữ Java

### Lịch sử phát triển của Java

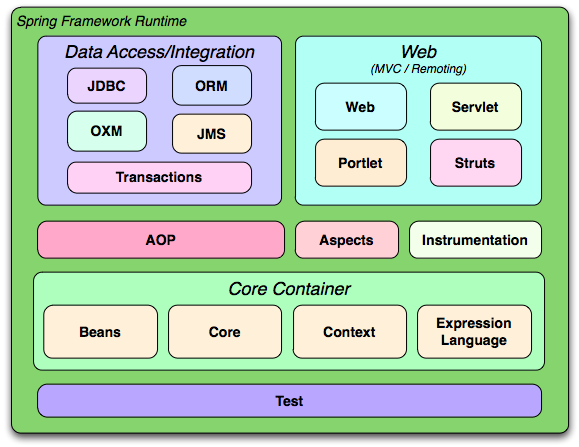
Java là một ngôn ngữ bậc cao theo thuần hướng đối tượng được phát triển bởi Sun Microsystems (Sau này được Oracle mua lại). Java được tạo ra bởi James Gosling và các đồng nghiệp, ban đầu Java được gọi là Oak (nghĩa là cây sồi do ngoài cơ quan ông Gosling trồng nhiều cây này). Java được phát hành vào năm 1994.

Java được lập trình với tiêu chí “Viết một lần, thực thi mọi nơi” nên Java có thể chạy trên mọi nên tảng dựa trên bộ máy ảo của Java.

## Tổng quan về Spring Framework

### Tổng quan về Spring

Spring Framework là một Framework dựa trên Java rất phổ biến. Spring Framework được Rod Johnson phát triển vào năm 2003. Spring Framework là một khung mã nguồn mở sử dụng để phát triển các ứng dụng Java nhanh chóng và dễ dàng [1].

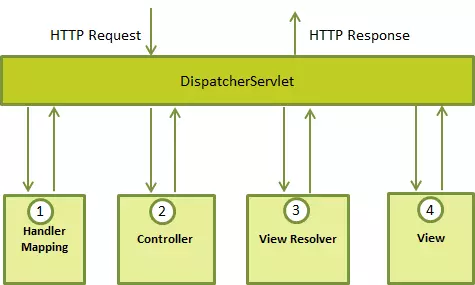


Hình .: Module của Spring framework (nguồn: docs.spring.io)

### Tổng quan về Spring MVC

Spring MVC là một module con của Spring framework, cung cấp kiến trúc Model-View-Controller và các components có sẵn để sử dụng và phát triển ứng dụng web.

Spring MVC được thiết kế xung quanh DispatcherServlet để xử lý các HTTP Request và HTTP Response . Luồng xử lý các yêu cầu của DispatcherServlet được minh hoạ như hình:



Hình .: Sơ đồ luồng xử lý của Spring MVC framework (nguồn: docs.spring.io)

### Các khái nhiệm trong Spring

Core Container:

Core module là thành phần quan trọng nhất trong Spring vì nó cung cấp các đặc trưng chính như Dependency Injection và Inversion of Control.

Bean module cung cấp Bean Factory, Bean được coi là phần trung tâm của Spring Core .

Context module: ApplicationContext sẽ nạp các bean của Spring và mốc nối chúng lại với nhau.

Spring Expression Language: là một ngôn ngữ biểu diễn rất mạnh trong việc hỗ trợ các đặc trưng truy vấn và thao tác với đối tượng thực thi

### Ưu điểm của Spring MVC

Các tầng có trong Spring MVC thường độc lập nên việc unit test sẽ trở nên dễ dàng hơn rất nhiều.

Phần view của chúng sẽ được tích hợp với nhiều Framework về UI như là: JSF, Freemarker hoặc Themeleaf,…

Có thể hỗ trợ được cả Annotation và XML config giúp cho quá trình phát triển trở nên sạch hơn và nhanh hơn rất nhiều.

Có thể cung cấp giúp việc phân chia trở nên rõ ràng, linh hoạt hơn giữa các controller, service, data access layer.

### Tạo dự án với Spring MVC với NetBeans

#### Đầu tiên ta vào NetBeans và tạo một project Maven

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình .: Hình quá trình tạo dự án Spring MVC

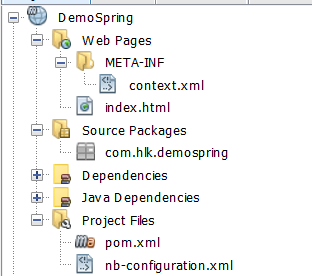
Ta đặt tên project

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình .: Hình đặt tên cho dự án Spring MVC

Ta được cấu trúc trong một dự án:



Hình .:Hình cấu trúc dự án của Spring MVC

Để sử dụng được Spring Framework trước tiên ta vào file pom.xml để thêm các dependencies.

<dependencies>

<dependency>

<groupId>javax</groupId>

<artifactId>javaee-web-api</artifactId>

<version>8.0.1</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.3.9</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

</dependencies>

Tiếp đến ta tạo một package com.exam.configs trong source packages, sau đó tạo 1 file tên WebApplicationContextConfig.java.

@Configuration

@EnableWebMvc

@ComponentScan(basePackages = {

"com.exam.controllers"

})

public class WebApplicationContextConfig implements WebMvcConfigurer {

@Override

public void configureDefaultServletHandling(DefaultServletHandlerConfigurer configurer) {

configurer.enable();

}

@Bean

public InternalResourceViewResolver getInternalResourceViewResolver() {

InternalResourceViewResolver resolver

= new InternalResourceViewResolver();

resolver.setViewClass(JstlView.class);

resolver.setPrefix("/WEB-INF/layout/");

resolver.setSuffix(".jsp");

return resolver;

}

}

Ta tạo thêm 1 class DispatcherServletInitializer ngay trong chính package này.

public class DispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer{

@Override

protected Class<?>[] getRootConfigClasses() {

return null;

}

@Override

protected Class<?>[] getServletConfigClasses() {

return new Class[] {

WebApplicationContextConfig.class

};

}

@Override

protected String[] getServletMappings() {

return new String[] {"/"};

}

}

Ta tạo 1 thư mục WEB-INF/jsp trong thư mục webapp, trong thư mục đó tạo tập tin welcome.jsp như sau:

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<title>Welcome Page</title>

</head>

<body>

<h1>${message}</h1>

</body>

</html>

Ta tạo 1 package controller và tạo tập tin HomeController.java

@Controller

public class HomeController {

@RequestMapping(value = "/")

public String index(Model model) {

model.addAttribute("message",

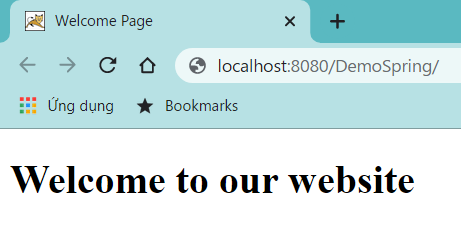
"Welcome to our Website!!!");

return "welcome";

}

}

Sau cùng ta build và run sẽ được 1 trang như sau:



Hình .: Hình giao diện trang web của Spring MVC

## Tổng quan về Hibernate

### Lịch sử Hibernate

Hibernate ra đời năm 2001 bởi Gavin King và được coi là thay thế cho EJB2 kiểu thực thể bean. Hiện nay, phiên bản mới nhất của Hibernate là 5.2.10.Final.

### Giới thiệu Hibernate Framework

Hibernate framework là một giải pháp ORM mã nguồn mở, gọn, nhẹ. Hibernate giúp đơn giản hoá sự phát triển của ứng dụng java để tương tác với cơ sở dữ liệu [2].

### Lợi ích của việc sử dụng Hibernate

**Mã nguồn mở và nhẹ:** Hibernate là mã nguồn mở có giấy phép LGPL và rất nhẹ khi sử dụng.

**Hiệu suất nhanh:** Hiệu suất của Hibernate Framework là nhanh bởi vì bộ nhớ cache được sử dụng trong nội bộ Hibernate Framework. Có hai loại bộ nhớ cache trong Hibernate Framework, gồm bộ nhớ cache cấp một và bộ nhớ cache cấp hai. Bộ nhớ cache cấp một được bật bằng lệnh mặc định.

**Truy vấn cơ sở dữ liệu dễ dàng:** Hibernate cung cấp các Criteria API để thực hiện các thao tác tương tác với cơ sở dữ liệu dễ dàng, nhanh chóng và không cần biết quá nhiều cú pháp truy vấn SQL.

**Tạo bảng tự động:** Hibernate framework cung cấp phương tiện để tạo ra các bảng cơ sở dữ liệu tự động. Vì vậy, không cần phải tạo ra các bảng trong cơ sở dữ liệu bằng tay.

## Tổng quan về MySQL

### Giới thiệu về MySQL

MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở hoạt động theo mô hình client-server. MySQL đứng thứ hai về số người sử dụng nhiều nhất chỉ sau Oracle. MySQL cũng có cùng một cách truy xuất và mã lệnh tương tự với ngôn ngữ SQL khác [3].

### Lịch sử của MySQL

MySQL được công ty MySQL AB ở Thuỵ Điển phát triển vào năm 1994. Phiên bản đầu tiên được phát hành vào 1995. Vào năm 2008 MySQL AB bị Công ty Sun Microsystems mua lại. Đến năm 2010, tập đoàn Oracle thâu tóm Sun Microsystems.

### Ưu điểm khi sử dụng MySQL

Dễ sử dụng: Do sử dụng các mã lệnh tương tự như các ngôn ngữ SQL khác nên việc bắt đầu sử dụng MySQL rất dễ dàng.

Độ bảo mật cao: MySQL đảm bảo tính bảo mật cao khi truy cập CSDL trên internet.

Đa tính năng: MySQL hỗ trợ rất nhiều chức năng trong việc quản trị CSDL.

Khả năng mở rộng và mạnh mẽ: MySQL cho phép khả năng mở rộng nếu người dùng mong muốn.

Nhanh chóng: Tốc độ thực thi truy vấn của MySQL là một trong những điểm mạnh mà người dùng tin tưởng.

### Kết nối MySQL vào dự án Spring thông qua Hibernate

Đầu tiên ta thêm các dependencies liên quan vào file pom.xml

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.5.5.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.26</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>5.3.9</version>

</dependency>

Tạo tập tin database.properties trong thư mục src/main/resources để cấu hình hibernate

hibernate.dialect=org.hibernate.dialect.MySQLDialect

hibernate.showSql=true

hibernate.connection.driverClass=com.mysql.cj.jdbc.Driver

hibernate.connection.url=jdbc:mysql://localhost:3306/data

hibernate.connection.username=root

hibernate.connection.password=1234

Ở packages configs tạo một tập tin HibernateConfigs.java

@Configuration

@PropertySource("classpath:database.properties")

public class HibernateConfig {

@Autowired

private Environment env;

@Bean

public LocalSessionFactoryBean getSessionFactory() {

LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();

sessionFactory.setPackagesToScan(new String[] {

"com.hlk.pojos"

});

sessionFactory.setDataSource(dataSource()); sessionFactory.setHibernateProperties(hibernateProperties());

return sessionFactory;

}

@Bean

public DataSource dataSource() {

DriverManagerDataSource dataSource = new DriverManagerDataSource();

dataSource.setDriverClassName( env.getProperty("hibernate.connection.driverClass")); dataSource.setUrl(env.getProperty("hibernate.connection.url"));

dataSource.setUsername(

env.getProperty("hibernate.connection.username"));

dataSource.setPassword(

env.getProperty("hibernate.connection.password"));

return dataSource;

}

private Properties hibernateProperties() {

Properties props = new Properties();

props.put(DIALECT, env.getProperty("hibernate.dialect"));

props.put(SHOW\_SQL, env.getProperty("hibernate.showSql"));

return props;

}

@Bean

public HibernateTransactionManager transactionManager() {

HibernateTransactionManager transactionManager = new HibernateTransactionManager();

transactionManager.setSessionFactory(getSessionFactory().getObject());

return transactionManager;

}

}

Sau đó ta cập nhật hàm getRootConfigClasses trong DispatcherServletInitializer.java

@Override

protected Class<?>[] getRootConfigClasses() {

return new Class[] {

HibernateConfig.class

};

}

Ở tập tin WebApplicationContextConfig.java ta thêm annotations để sử dụng hibernate quản lý giao tác

@Configuration

@EnableWebMvc

**@EnableTransactionManagement**

@ComponentScan(basePackages = "com.dht.controller")

public class WebApplicationContextConfig

implements WebMvcConfigurer {

}

## **Tổng quan về Cloudinary**

### Giới thiệu Cloudinary

Cloudinary là một dịch vụ đám mây cho phép quản lý hình ảnh bao gồm tải ảnh lên cloud, lưu trữ và tối ưu hoá. Cloudinary cung cấp API để cho phép người dùng upload ảnh và sử dụng để hiển thị trên giao diện.

### Cài đặt Cloudinary cho dự án Spring

Ta cài đặt dependency ở file pom.xml:

<dependency>

<groupId>com.cloudinary</groupId>

<artifactId>cloudinary-http44</artifactId>

<version>1.29.0</version>

</dependency>

Ở file WebApplicationContextConfig ta thêm 1 bean cloudinary:

@Bean

public Cloudinary cloudinary() {

Cloudinary cloudinary = new Cloudinary(ObjectUtils.asMap(

"cloud\_name", "linhkhoi",

"api\_key", "528136112873553",

"api\_secret", "tB11NGtTeSS51dsaaFXgTRH19OA",

"secure", true

));

return cloudinary;

}

Vơi thông tin cloud\_name, api\_key, api\_secret từ tài khoản cloudinary đã đăng ký từ trước đó:

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình .: Hình thông tin cloudinary

Cuối cùng ta tạo hàm trả về chuỗi url từ file ảnh đã upload:

@Autowired

private Cloudinary cloudinary;

public String uploadCloudinary(File image) throws UnsupportedEncodingException, IOException {

Map r = this.cloudinary.uploader().upload(image getBytes(), ObjectUtils.asMap("resource\_type", "auto"));

String img = (String) r.get("secure\_url");

return img;

}

## Tổng quan về hệ thống ví điện tử ZaloPay

### Giới thiệu ZaloPay

ZaloPay là một hệ thống ví điện tử rất thịnh hành tại Việt Nam với các chức năng thanh toán online tiền điện thoại, hoá đơn, vé phim,… rất nhanh chóng và an toàn.

ZaloPay hỗ trợ tích hợp nhiều nền tảng như web, ứng dụng di động, máy pos,… rất dễ dàng.

### Tích hợp ZaloPay vào dự án Spring MVC

Đầu tiên ta thêm các dependency cần thiết:

<dependency>

<groupId>org.json</groupId>

<artifactId>json</artifactId>

<version>20210307</version>

</dependency>

<dependency>

<groupId>org.apache.httpcomponents</groupId>

<artifactId>httpclient</artifactId>

<version>4.5.13</version>

</dependency>

Để tạo được 1 hoá đơn ZaloPay ta cần tạo thông tin về app\_id, key1,key2 và endpoint:

private static Map<String, String> config = new HashMap<String, String>(){{

put("app\_id", "2553");

put("key1", "PcY4iZIKFCIdgZvA6ueMcMHHUbRLYjPL");

put("key2", "kLtgPl8HHhfvMuDHPwKfgfsY4Ydm9eIz");

put("endpoint", "https://sb-openapi.zalopay.vn/v2/create");

}};

Sau đó ta tạo một hàm dùng để tạo đơn hàng từ thông tin trên:

public String urlToPay(final BigDecimal total, final int prescriptionId) throws UnsupportedEncodingException, IOException{

String url = "";

Random rand = new Random();

final int random\_id = rand.nextInt(1000000);

final String appTransId = getCurrentTimeString("yyMMdd") +"\_"+ random\_id;

final Map embed\_data = new HashMap(){{

put("redirecturl", "http://localhost:8080/SpringClinic/nurse/check-zalo/?appId="+appTransId+"&preId="+prescriptionId);

}};

final Map[] item = {

new HashMap(){{}}

};

Map<String, Object> order = new HashMap<String, Object>(){{

put("app\_id", config.get("app\_id"));

put("app\_trans\_id", appTransId);

put("app\_time", System.currentTimeMillis());

put("app\_user", "user123");

put("amount", total);

put("description", "Clinic - Payment for the order #"+random\_id);

put("bank\_code", "zalopayapp");

put("item", "[{}]");

put("embed\_data", new JSONObject(embed\_data).toString());

}};

String data = order.get("app\_id") +"|"+ order.get("app\_trans\_id") +"|"+ order.get("app\_user") +"|"+ order.get("amount")

+"|"+ order.get("app\_time") +"|"+ order.get("embed\_data") +"|"+ order.get("item");

order.put("mac", HMACUtil.HMacHexStringEncode(HMACUtil.HMACSHA256, config.get("key1"), data));

CloseableHttpClient client = HttpClients.createDefault();

HttpPost post = new HttpPost(config.get("endpoint"));

List<NameValuePair> params = new ArrayList<>();

for (Map.Entry<String, Object> e : order.entrySet()) {

params.add(new BasicNameValuePair(e.getKey(), e.getValue().toString()));

}

post.setEntity(new UrlEncodedFormEntity(params));

CloseableHttpResponse res = client.execute(post);

BufferedReader rd = new BufferedReader(new InputStreamReader(res.getEntity().getContent()));

StringBuilder resultJsonStr = new StringBuilder();

String line;

while ((line = rd.readLine()) != null) {

resultJsonStr.append(line);

}

JSONObject result = new JSONObject(resultJsonStr.toString());

if(result.getInt("return\_code")==1){

url = result.getString("order\_url");

}

return url;

}

Trong đó thông tin redirecturl chính là redirect về url này sau khi thanh toán trên cổng ZaloPay.

Mỗi lần gọi hàm tạo đơn hàng thì sẽ nhận về thông tin bao gồm

{

"return\_code": 1,

"return\_message": "Giao dịch thành công",

"sub\_return\_code": 1,

"sub\_return\_message": "Giao dịch thành công",

"zp\_trans\_token": "20090400000112548Y3z18",

"order\_url": "https://sbgateway.zalopay.vn/openinapp?order=eyJ6cHRyYW5zdG9rZW4iOiIyMDA5MDQwMDAwMDExMjU0OFkzejE4IiwiYXBwaWQiOjI1NTN9",

"order\_token": "ptazBLb128DJ6MSe4d-I2okWQpO7FdUwK9VA4MNqFliUIO1SM3E8ElOsum-iie61rou4A1lblWwddvCwKObzFpo0xJRX4AgP6moSsVqKsxM8K-Duix5ZdH3HFNN07fk7"

}

Trong đó order\_url chính là thông tin url dùng để quét mã QR thanh toán.

Sau khi thanh toán xong ta dùng hàm kiểm tra trạng thái đơn hàng:

public static int getStatus(String app\_trans\_id) throws URISyntaxException, UnsupportedEncodingException, IOException{

String data = config.get("app\_id") +"|"+ app\_trans\_id +"|"+ config.get("key1"); // appid|app\_trans\_id|key1

String mac = HMACUtil.HMacHexStringEncode(HMACUtil.HMACSHA256, config.get("key1"), data);

List<NameValuePair> params = new ArrayList<>();

params.add(new BasicNameValuePair("app\_id", config.get("app\_id")));

params.add(new BasicNameValuePair("app\_trans\_id", app\_trans\_id));

params.add(new BasicNameValuePair("mac", mac));

URIBuilder uri = new URIBuilder(config.get("endpoint"));

uri.addParameters(params);

CloseableHttpClient client = HttpClients.createDefault();

HttpPost post = new HttpPost(uri.build());

post.setEntity(new UrlEncodedFormEntity(params));

CloseableHttpResponse res = client.execute(post);

BufferedReader rd = new BufferedReader(new InputStreamReader(res.getEntity().getContent()));

StringBuilder resultJsonStr = new StringBuilder();

String line;

while ((line = rd.readLine()) != null) {

resultJsonStr.append(line);

}

JSONObject result = new JSONObject(resultJsonStr.toString());

return result.getInt("return\_code");

}

Sau khi gọi hàm này ta sẽ được lấy được thông tin thanh toán:

|  |  |  |
| --- | --- | --- |
| **Tham số** | **Kiểu dữ liệu** | **Ý nghĩa** |
| return\_code | int | 1 : Thành công 2 : Thất bại 3 : Đơn hàng chưa thanh toán hoặc giao dịch đang xử lý |
| return\_message | String | Thông tin trạng thái đơn hàng |
| sub\_return\_code | int | Mã trạng thái chi tiết |
| sub\_return\_message | String | Thông tin chi tiết trạng thái đơn hàng |
| is\_processing | boolean | Trạng thái xử lý |
| amount | long | Số tiền ứng dụng nhận được (chỉ có ý nghĩa khi thanh toán thành công) |
| zp\_trans\_id | long | Mã giao dịch của ZaloPay |

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Phân tích thiết kế các chức năng ứng dụng

### Mô tả chức năng ứng dụng

Mục đích đề tài xây dựng hệ thống phần mềm hoạt động trên môi trường web để quản lý phòng mạch tư được thuận tiện, dễ quản lý. Người quản lý sẽ dễ dàng trong việc quản lý bác sĩ, y tá, thuốc, lịch trực. Bác sĩ sẽ dễ dàng trong việc ra toa thuốc, y tá sẽ thanh toán hoá đơn dễ dàng hơn, còn khách hàng đặt phiếu khám trên web sẽ được thoải mải hơn không tốn thời gian để đến phòng mạch chờ đợi khám bệnh.

Tuỳ vào từng vai trò sẽ có những chức năng khác nhau:

Admin:

Admin có quyền quản lý (thêm/xoá/sửa/tra cứu) các thông tin của bác sĩ, y tá, sắp lịch trực cho từng đối tượng, thuốc, bệnh.

Thống kê báo cáo số lượng bệnh nhân đến khám, doanh thu theo tháng, quý và năm.

Bác sĩ:

Bác sĩ có thể lên toa thuốc cho bệnh nhân, trong quá trình này, bác sĩ sẽ tra cứu danh sách thuốc để thêm vào toa.

Xem lịch sử khám và bệnh của bệnh nhân để tiện có việc chuẩn đoán bệnh.

Y tá:

Xác nhận phiếu khám được bệnh nhân đặt trên trang web đồng thời sẽ tự động gửi mail khi xác nhận thành công.

Y tá sẽ thanh toán tiền khám của bệnh nhân dựa trên toa thuốc và phiếu khám, y tá sẽ sử dụng thanh toán online bằng ví điện tử ZaloPay cho khách hàng thanh toán.

Bệnh nhân:

Bệnh nhân có thể đặt phiếu khám bệnh trực tuyến trên trang web của bệnh viện, bệnh nhân sẽ nhận được mail sau khi xác nhận thành công

Bệnh nhân có thể thanh toán phí khám bằng ví điện tử ZaloPay

### Sơ đồ Use Case:

#### Actor hệ thống:

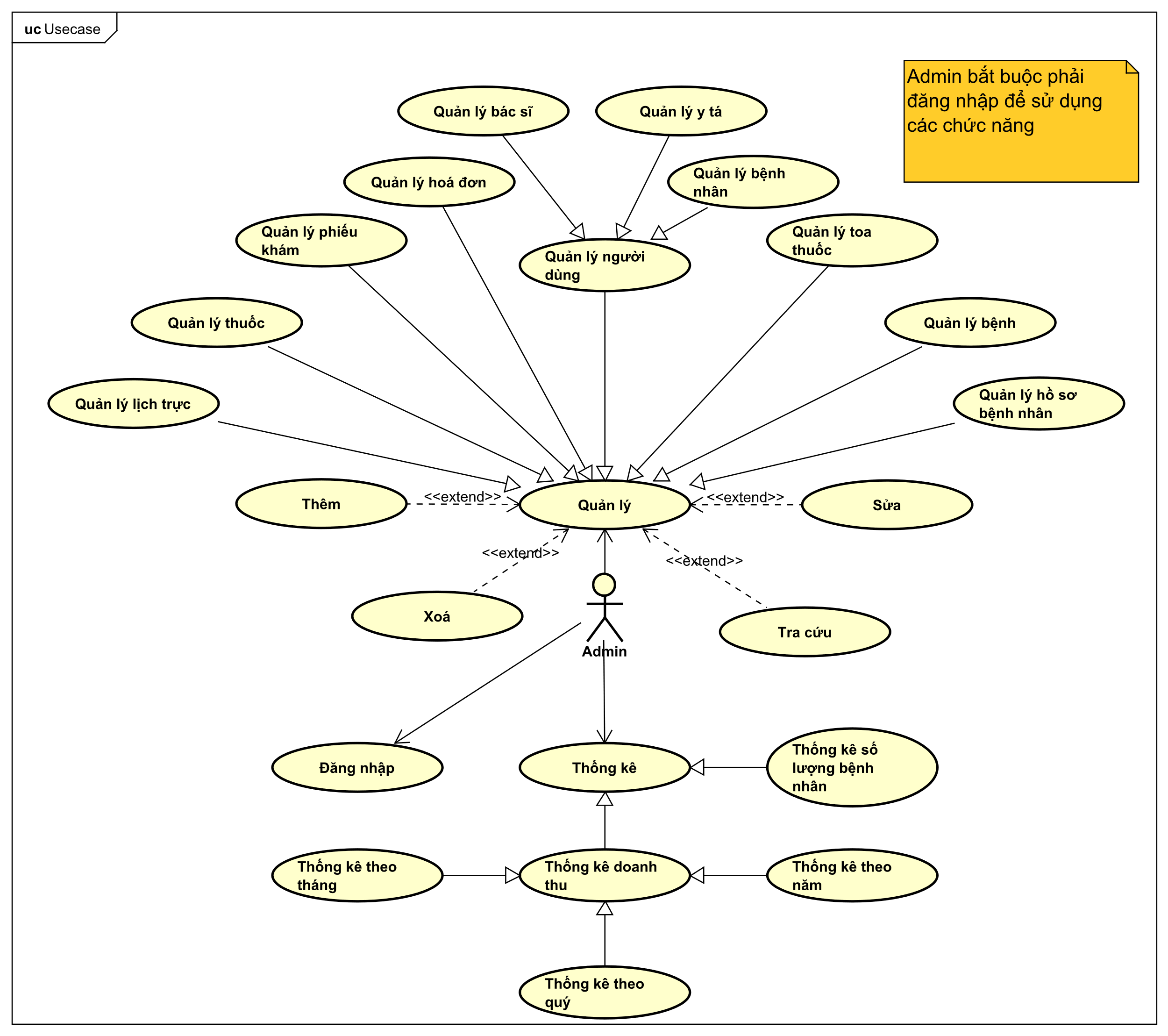
Admin

Bác sĩ

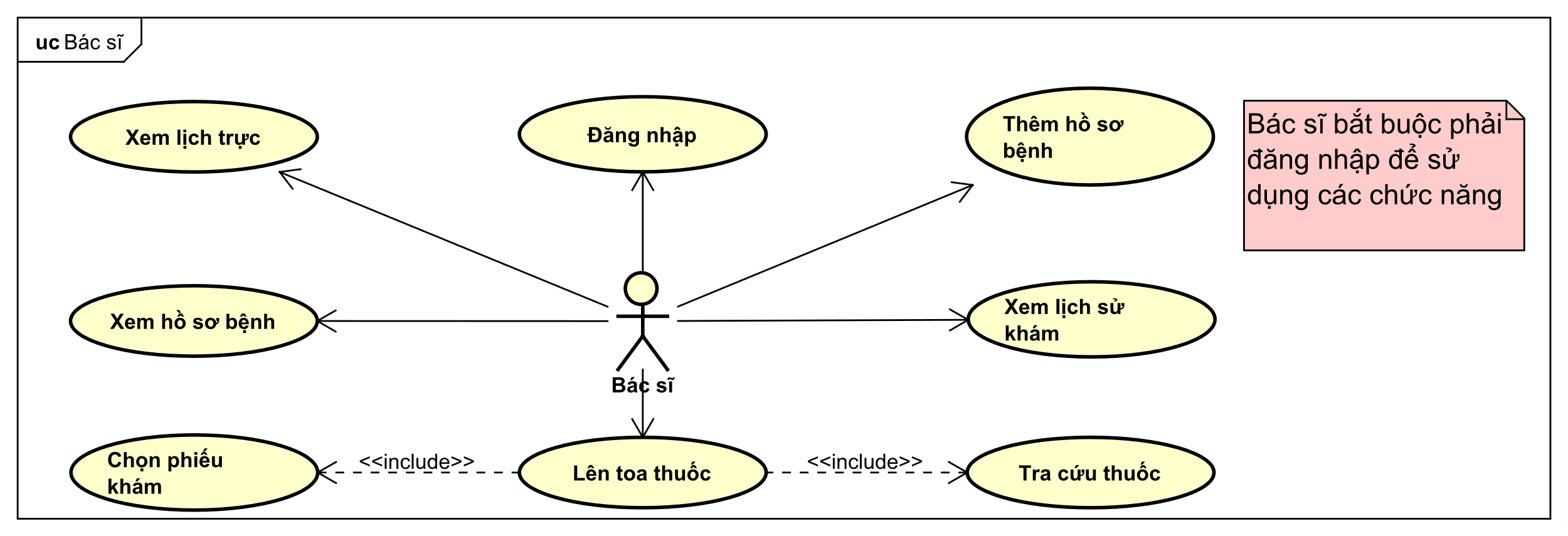
Y tá

Bệnh nhân

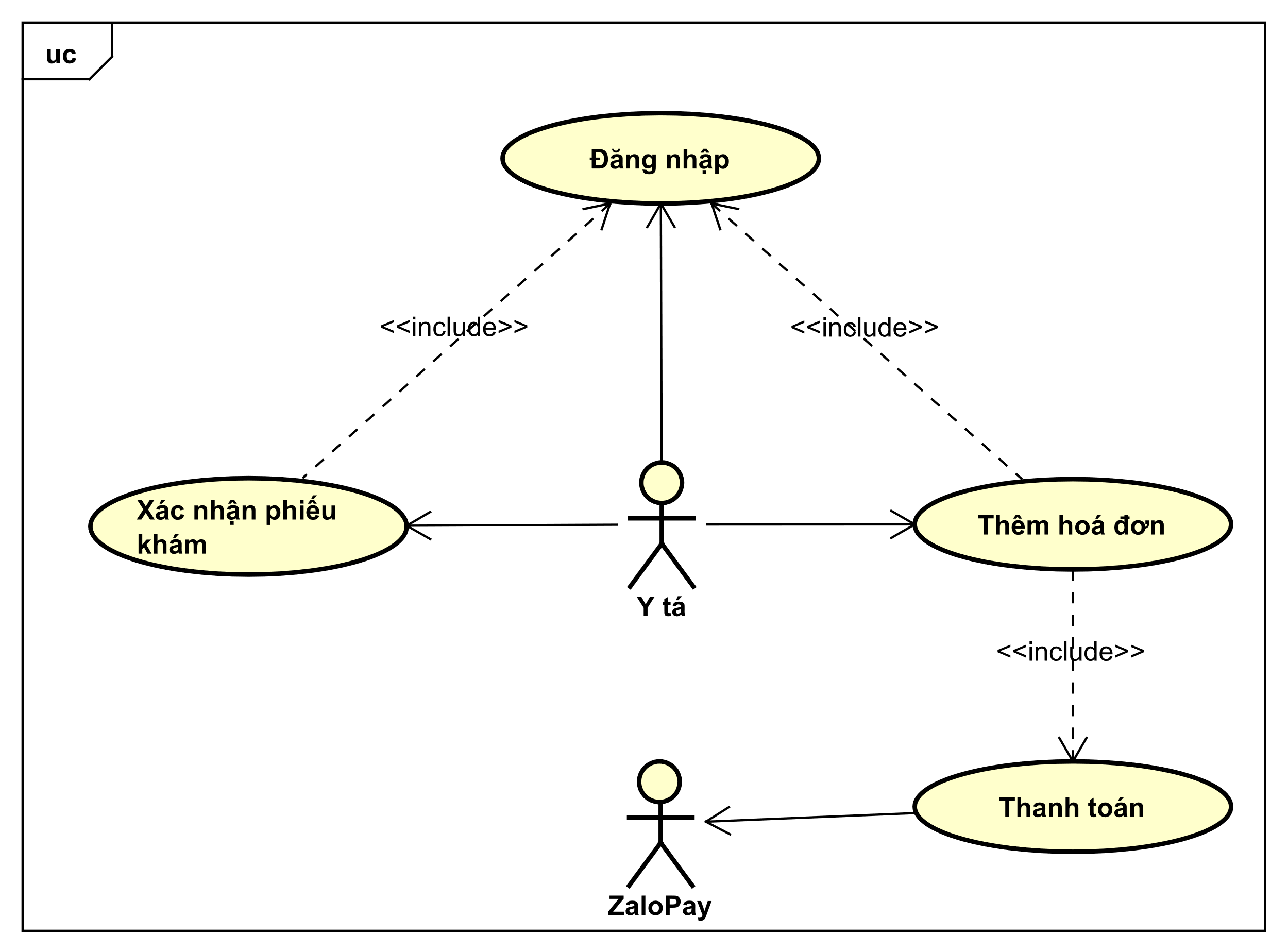
#### Sơ đồ use-case của hệ thống:



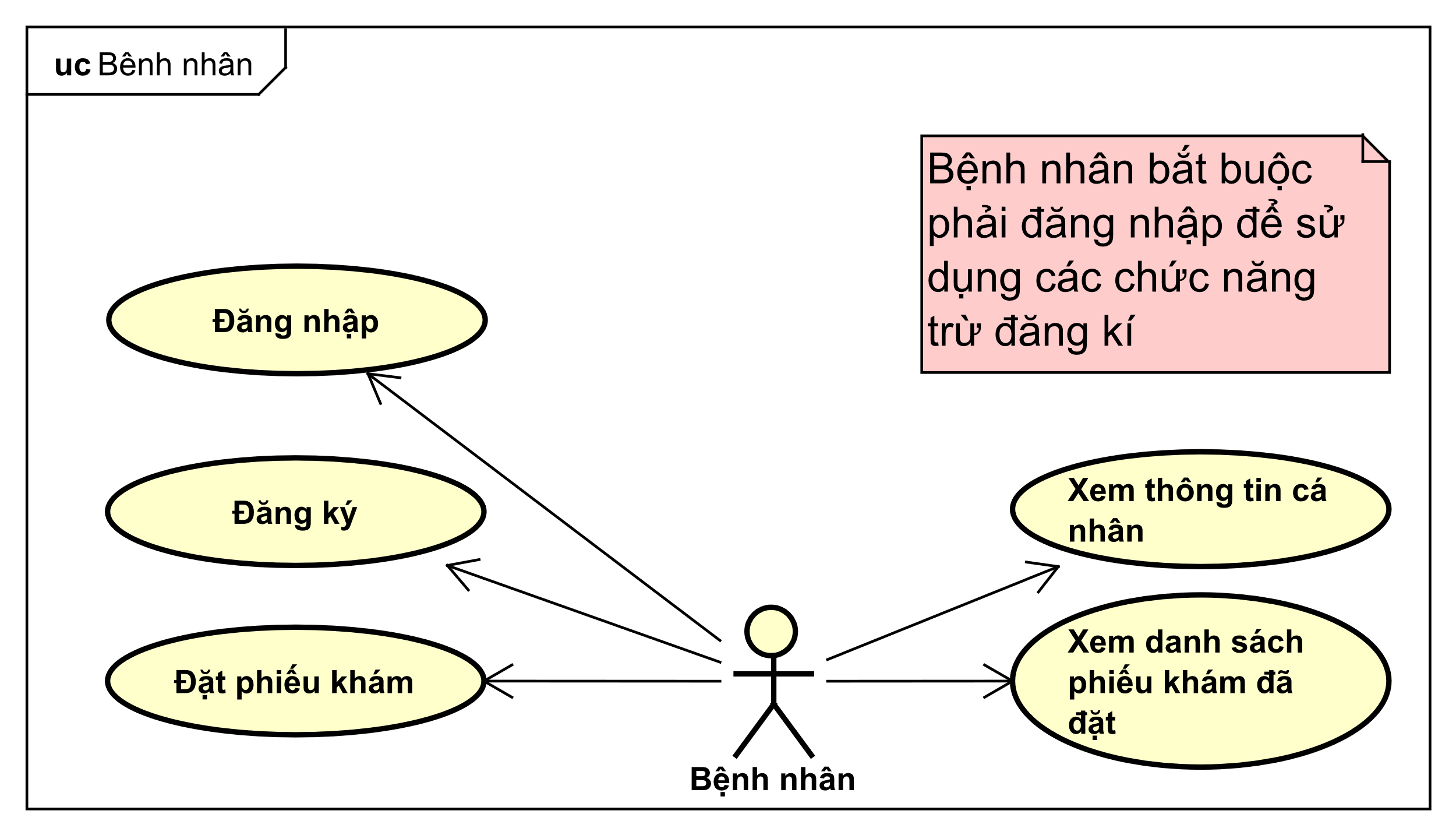
Hình .: Sơ đồ use-case của Admin



Hình .: Sơ đồ use-case của bác sĩ



Hình .: Sơ đồ use-case của y tá



Hình .: Sơ đồ use-case của bệnh nhân

#### Danh sách các use-case của hệ thống:

|  |  |
| --- | --- |
| Mã use-case | Tên use-case |
| 1 | Quản lý |
| 2 | Quản lý bác sĩ |
| 3 | Quản lý y tá |
| 4 | Quản lý bệnh nhân |
| 5 | Quản lý thuốc |
| 6 | Quản lý bệnh |
| 7 | Quản lý hồ sơ bệnh |
| 8 | Quản lý lịch trực |
| 9 | Quản lý người dùng |
| 10 | Quản lý phiếu khám |
| 11 | Quản lý toa thuốc |
| 12 | Quản lý hoá đơn |
| 13 | Thêm |
| 14 | Sửa |
| 15 | Xoá |
| 16 | Tra Cứu |
| 17 | Thống kê doanh thu |
| 18 | Thống kê theo năm |
| 19 | Thống kê theo quý |
| 20 | Thống kê theo tháng |
| 21 | Đăng nhập |
| 22 | Đăng ký |
| 23 | Lên toa thuốc |
| 24 | Chọn phiếu khám |
| 25 | Tra cứu thuốc |
| 26 | Thêm hồ sơ bệnh |
| 27 | Xem hồ sơ bệnh |
| 28 | Xem lịch sử khám |
| 29 | Xem lịch trực |
| 30 | Xác nhận phiếu khám |
| 31 | Thêm hoá đơn |
| 32 | Thanh toán |
| 33 | Đặt phiếu khám |
| 34 | Xem thông tin cá nhân |
| 35 | Xem danh sách phiếu khám đã đặt |

Bảng .: Bảng danh sách use-case

#### Danh sách các tác nhân:

|  |  |  |
| --- | --- | --- |
| Mã tác nhân | Các tác nhân | Mã use case |
| 1 | Admin | 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 |
| 2 | Bác sĩ | 21, 23, 24, 25, 26, 27, 28 |
| 3 | Y tá | 21, 28, 29, 30, 31 |
| 4 | Bệnh nhân | 21, 22, 32, 33, 34 |
| 5 | ZaloPay | 31 |

Bảng .: Bảng danh sách tác nhân

#### Đặc tả use case:

##### Đặc tả use case “Đăng nhập”

|  |  |
| --- | --- |
| Mô tả | use case này cho phép người dùng và người quản trị đăng nhập vào hệ thống. |
| Actor chính | Admin, bác sĩ, y tá, bệnh nhân |
| Actor phụ | Không có. |
| Tiền điều kiện | Người sử dụng có tên truy cập và mật khẩu hợp lệ. |
| Hậu điều kiện | Người dùng đăng nhập thành công sẽ sử dụng được các chức năng của hệ thống. |
| Luồng hoạt động | Bước 1 - Người dùng vào trang chính của hệ thống và nhấn vào button “Đăng nhập”.  Bước 2 - Hệ thống sẽ hiển thị lên màn hình đăng nhập.  Bước 3 - Người dùng nhập username và password sau đó nhấn vào nút đăng nhập.  Bước 4 - Hệ thống xác thực người dùng qua username và password. |
| Luồng thay thế | Ở bước 4, nếu hệ thống xác thực thành công thì sẽ cho người dùng sử dụng hệ thống. |
| Luồng ngoại lệ | Ở bước 4, nếu hệ thống xác thực thất bại thì sẽ thông báo cho người dùng và yêu cầu nhập lại username và password. |

Bảng .: Bảng đặc tả use case “Đăng nhập”

##### Đặc tả use case “Đăng ký”

|  |  |
| --- | --- |
| Mô tả | use case này cho phép bệnh nhân đăng kí tài khoản |
| Actor chính | Bệnh nhân |
| Actor phụ | không có. |
| Tiền điều kiện | không có. |
| Hậu điều kiện | Bệnh nhân đăng ký thành công và có thể đăng nhập vào hệ thống. |
| Luồng hoạt động | Bước 1 - Nhân viên vào chức năng “Đăng ký”.  Bước 2 - Hệ thống yêu cầu bệnh nhân nhập thông tin cá nhân cần.  Bước 3 - Bệnh nhân nhập các thông tin cần thiết.  Bước 4 - Hệ thống sẽ kiểm tra các thông tin của bệnh nhân và tiến hành đăng ký. |
| Luồng thay thế | Ở bước 4, nếu hệ thống đăng ký thành công thì sẽ hiện thị trang đăng nhập cho bệnh nhân. |
| Luồng ngoại lệ | Ở bước 4, nếu hệ thống đăng ký thất bại hệ thống sẽ hiển thị thông báo “lỗi” và quay lại bước 2. |

Bảng .: Bảng đặc tả use case “Đăng ký”

##### Đặc tả use case “Đặt phiếu khám”

|  |  |
| --- | --- |
| Mô tả | use case này cho phép bệnh nhân đặt phiếu khám |
| Actor chính | Bệnh nhân |
| Actor phụ | không có. |
| Tiền điều kiện | không có. |
| Hậu điều kiện | Bệnh nhân đặt thành công và chờ đợi mail xác nhận từ y tá. |
| Luồng hoạt động | Bước 1 - Bệnh nhân vào chức năng “Đặt phiếu khám”.  Bước 2 - Hệ thống yêu cầu bệnh nhân nhập thông tin ngày giờ khám.  Bước 3 - Bệnh nhân nhập các thông tin cần thiết.  Bước 4 - Hệ thống sẽ kiểm tra các thông tin của bệnh nhân và tiến hành tạo phiếu khám. |
| Luồng thay thế | Ở bước 4, nếu hệ thống tạo phiếu thành công thì sẽ thông báo cho bệnh nhân. |
| Luồng ngoại lệ | Ở bước 4, nếu hệ thống tạo phiếu thất bại thì hệ thống sẽ hiển thị thông báo “lỗi” và quay lại bước 2. |

Bảng .: Bảng đặc tả use case “Đặt phiếu khám”

##### Đặc tả use case “Lên toa thuốc”

|  |  |
| --- | --- |
| Mô tả | use case này cho phép bác sĩ lên toa thuốc cho bệnh nhân |
| Actor chính | Bác sĩ |
| Actor phụ | không có. |
| Tiền điều kiện | không có. |
| Hậu điều kiện | Bác sĩ lên toa thuốc thành công. |
| Luồng hoạt động | Bước 1 - Bác sĩ vào chức năng “Danh sách bệnh nhân”.  Bước 2 - Hệ thống hiện danh sách.  Bước 3 - Bác sĩ sẽ chọn vào bệnh nhân cần lên toa.  Bước 4 - Hệ thống sẽ hiện phiếu khám của bệnh nhân.  Bước 5 - Bác sĩ chọn phiếu khám cần ra toa của bệnh nhân  Bước 6 - Hệ thống sẽ hiện danh sách thuốc  Bước 7 - Bác sĩ chọn thuốc để lên toa cho bệnh nhận và chọn vào xem toa thuốc  Bước 8 - Hệ thống sẽ hiện ra toa thuốc sau khi bác sĩ đã lên toa xong  Bước 9 - Bác sĩ kiểm tra thông tin và chọn xác nhận |
| Luồng thay thế | Ở bước 9, nếu hệ thống tạo toa thuốc thành công thì sẽ thông báo cho bác sĩ. |
| Luồng ngoại lệ | Ở bước 9, nếu hệ thống tạo toa thuốc thất bại thì hệ thống sẽ hiển thị thông báo “lỗi” và quay lại bước 8. |

Bảng .: Bảng đặc tả use case “Lên toa thuốc”

##### Đặc tả use case thêm hoá đơn

|  |  |
| --- | --- |
| Mô tả | use case này cho phép y tá thêm hoá đơn cho bệnh nhân |
| Actor chính | Y tá |
| Actor phụ | Không có |
| Tiền điều kiện | Không có. |
| Hậu điều kiện | Y tá thêm hoá đơn thành công và chuyển sang bước thanh toán. |
| Luồng hoạt động | Bước 1 - Y tá vào chức năng “Danh sách toa thuốc”.  Bước 2 - Hệ thống hiện danh sách.  Bước 3 - Y tá sẽ chọn toa thuốc cần thanh toán.  Bước 4 - Hệ thống sẽ tiến hành tạo hoá đơn Hệ thống sẽ hiện màn hình thông tin hoá đơn. |
| Luồng thay thế | Ở bước 4, nếu hệ thống tạo hoá đơn thành công thì sẽ thông báo cho y tá. |
| Luồng ngoại lệ | Ở bước 4, nếu hệ thống tạo hoá đơn thất bại thì hệ thống sẽ hiển thị thông báo “lỗi” và quay lại bước 2. |

Bảng .: Bảng đặc tả use case “Thêm hoá đơn”

##### Đặc tả use case “Thanh toán”

|  |  |
| --- | --- |
| Mô tả | use case này cho phép y tá thêm hoá đơn cho bệnh nhân |
| Actor chính | Y tá |
| Actor phụ | Hệ thống ZaloPay, bệnh nhân |
| Tiền điều kiện | Hoá đơn đã được tạo |
| Hậu điều kiện | Y tá thêm hoá đơn thành công và chuyển sang bước thanh toán. |
| Luồng hoạt động | Bước 1 - Y tá sau khi hoàn thành quá trình thêm hoá đơn sẽ được hệ thống hiển thị hoá đơn.  Bước 2 - Y tá chọn thanh toán bằng hệ thống ZaloPay.  Bước 3 - Hệ thống sẽ hiện mã QR của ZaloPay thanh toán.  Bước 4 - Y tá sẽ yêu cầu bệnh nhân quét mã QR bằng điện thoại để thanh toán. |
| Luồng thay thế | Ở bước 4, nếu ZaloPay thanh toán thành công thì sẽ thông báo cho y tá. |
| Luồng ngoại lệ | Ở bước 4, nếu ZaloPay thanh toán thất bại thì hệ thống sẽ hiển thị thông báo “lỗi” và quay lại bước 2. |

Bảng .: Bảng đặc tả use case “Thanh toán”

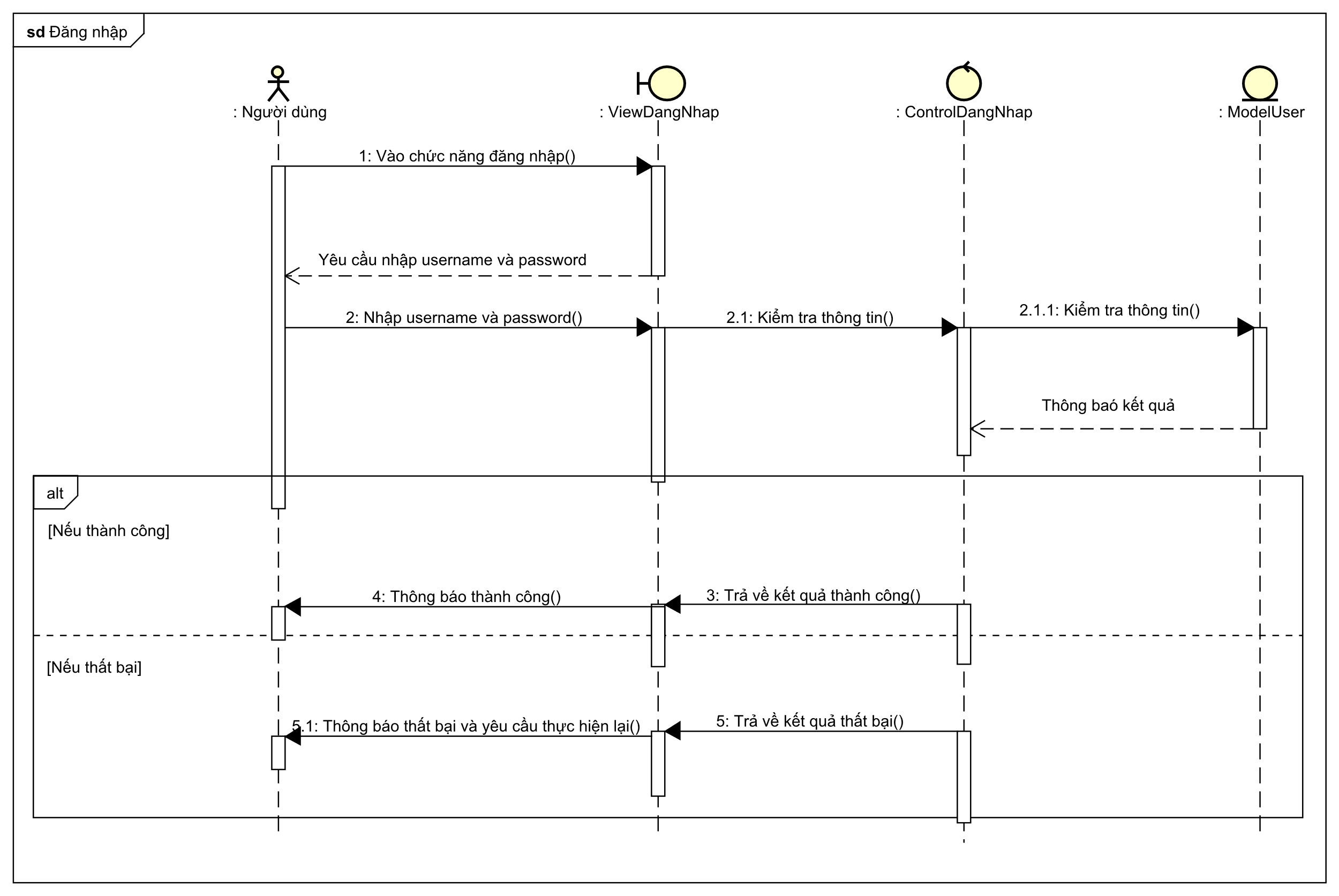
##### Đặc tả use case “Thống kê doanh thu theo năm”

|  |  |
| --- | --- |
| Mô tả | use case này cho phép admin thống kê doanh thu của phòng mạch theo năm |
| Actor chính | Admin |
| Actor phụ | Không có |
| Tiền điều kiện | Không có |
| Hậu điều kiện | Admin sẽ xem được báo cáo thống kê doanh thu theo năm. |
| Luồng hoạt động | Bước 1 - Admin chọn chức năng thống kê doanh thu theo năm.  Bước 2 - Hệ thống sẽ yêu cầu nhập năm cần thống kê  Bước 3 - Admin nhập năm cần thống kê  Bước 4 - Hệ thống sẽ tiến hành thống kê doanh thu theo năm. |
| Luồng thay thế | Ở bước 4, nếu hệ thống sẽ hiện ra báo cáo thống kê theo năm. |
| Luồng ngoại lệ | Không có |

Bảng .: Bảng đặc tả use case “Thống kê doanh thu theo năm”

### Sơ đồ tuần tự:

#### Sơ đồ đăng nhập:



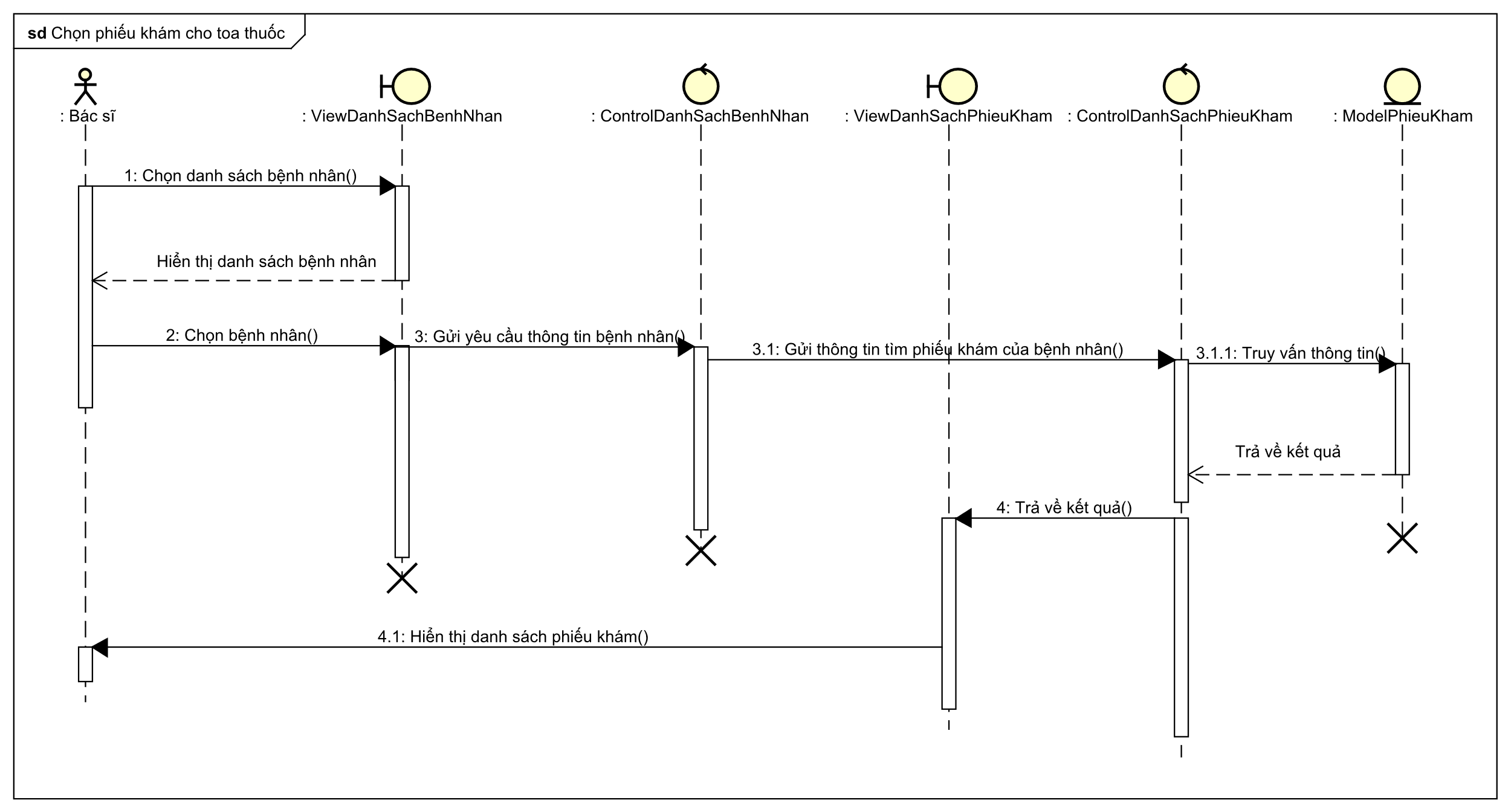
Hình .: Hình sơ đồ tuần tự Đăng nhập

#### Sơ đồ đăng ký:



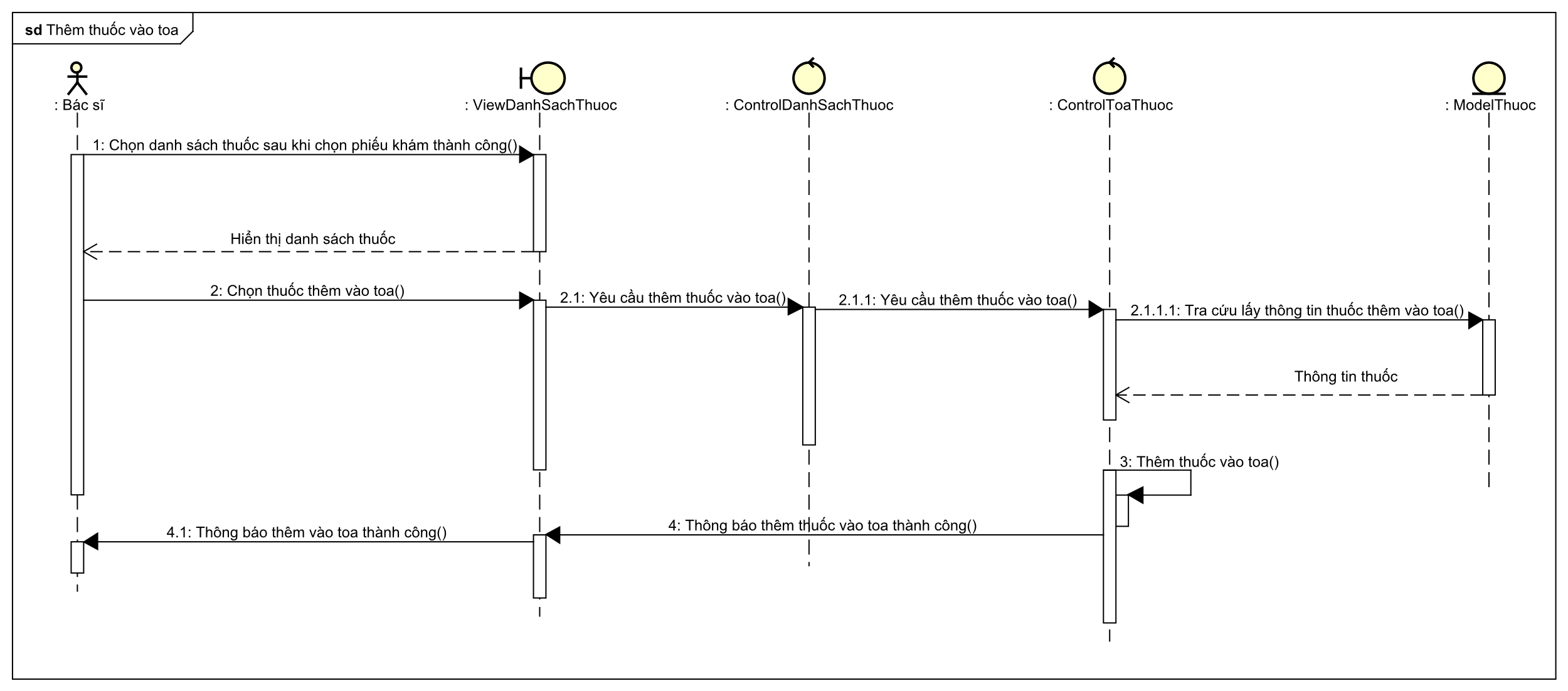
Hình .: Hình sơ đồ tuần tự Đăng ký

#### Sơ đồ chọn phiếu khám vào toa thuốc:



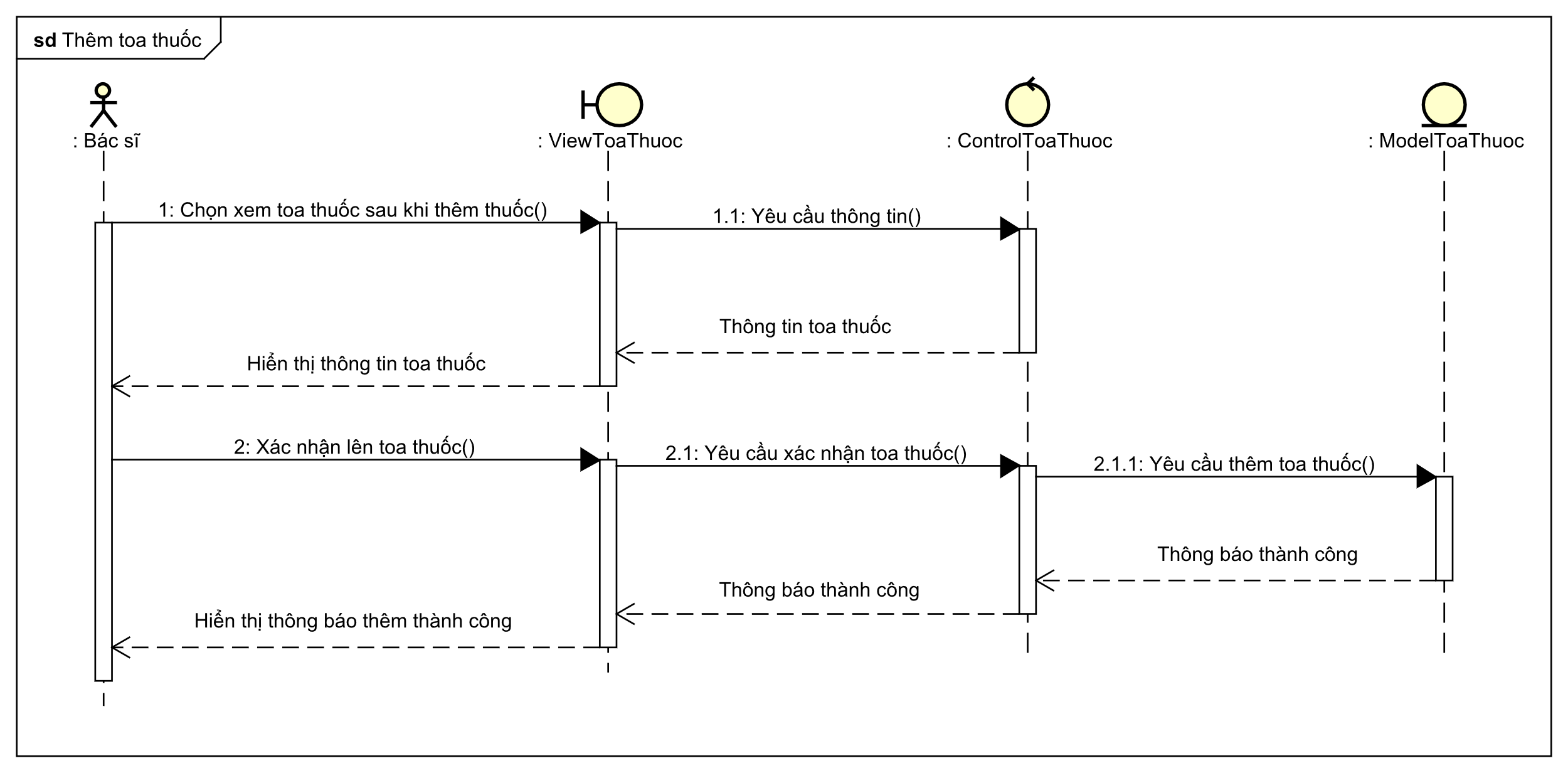
Hình .: Hình sơ đồ tuần tự chọn phiếu khám vào toa thuốc

#### Sơ đồ thêm thuốc vào toa:



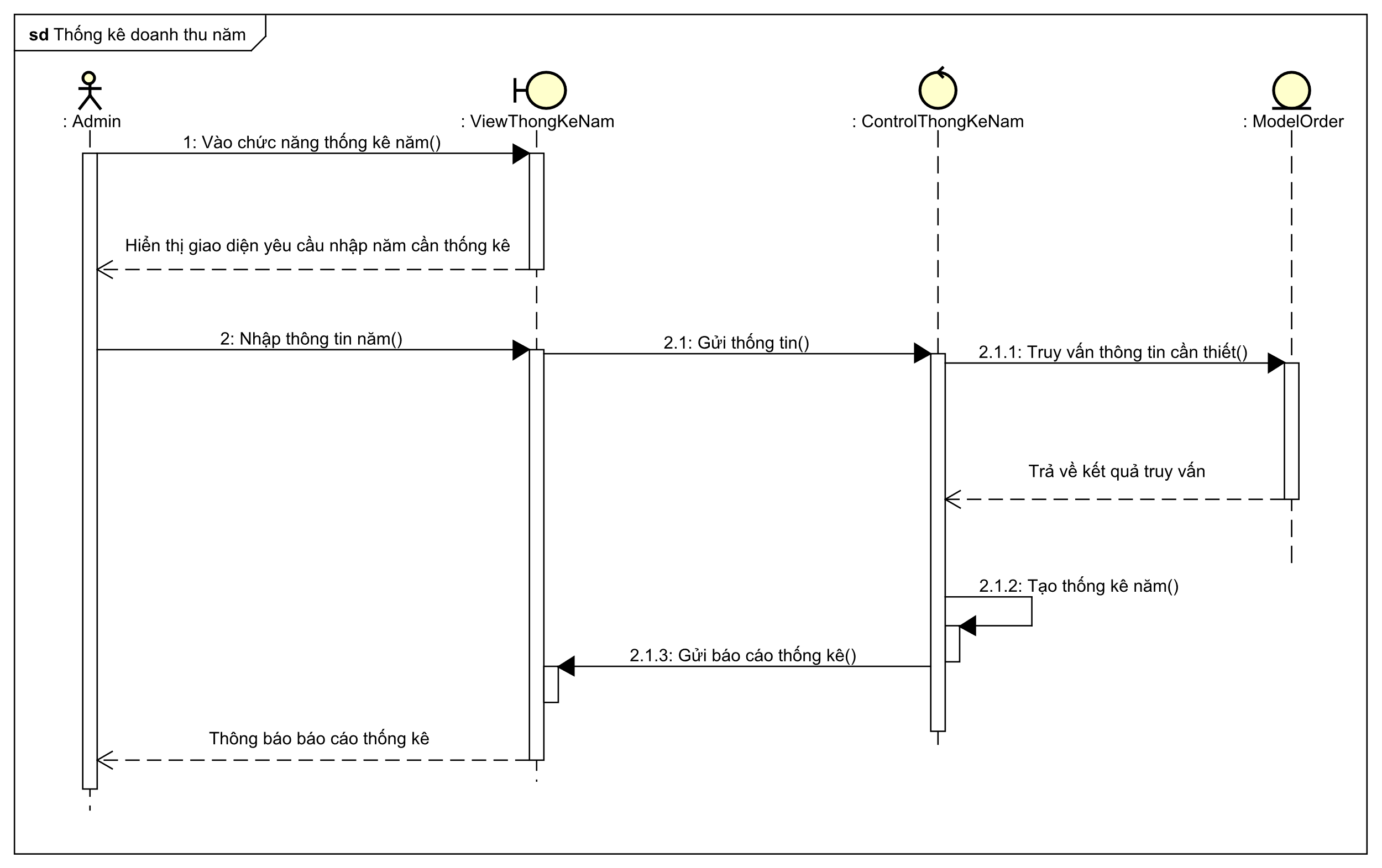
Hình .: Hình sơ đồ tuần tự Thêm thuốc vào toa

#### Sơ đồ thêm toa thuốc:



Hình .: Hình sơ đồ tuần tự Thêm toa thuốc

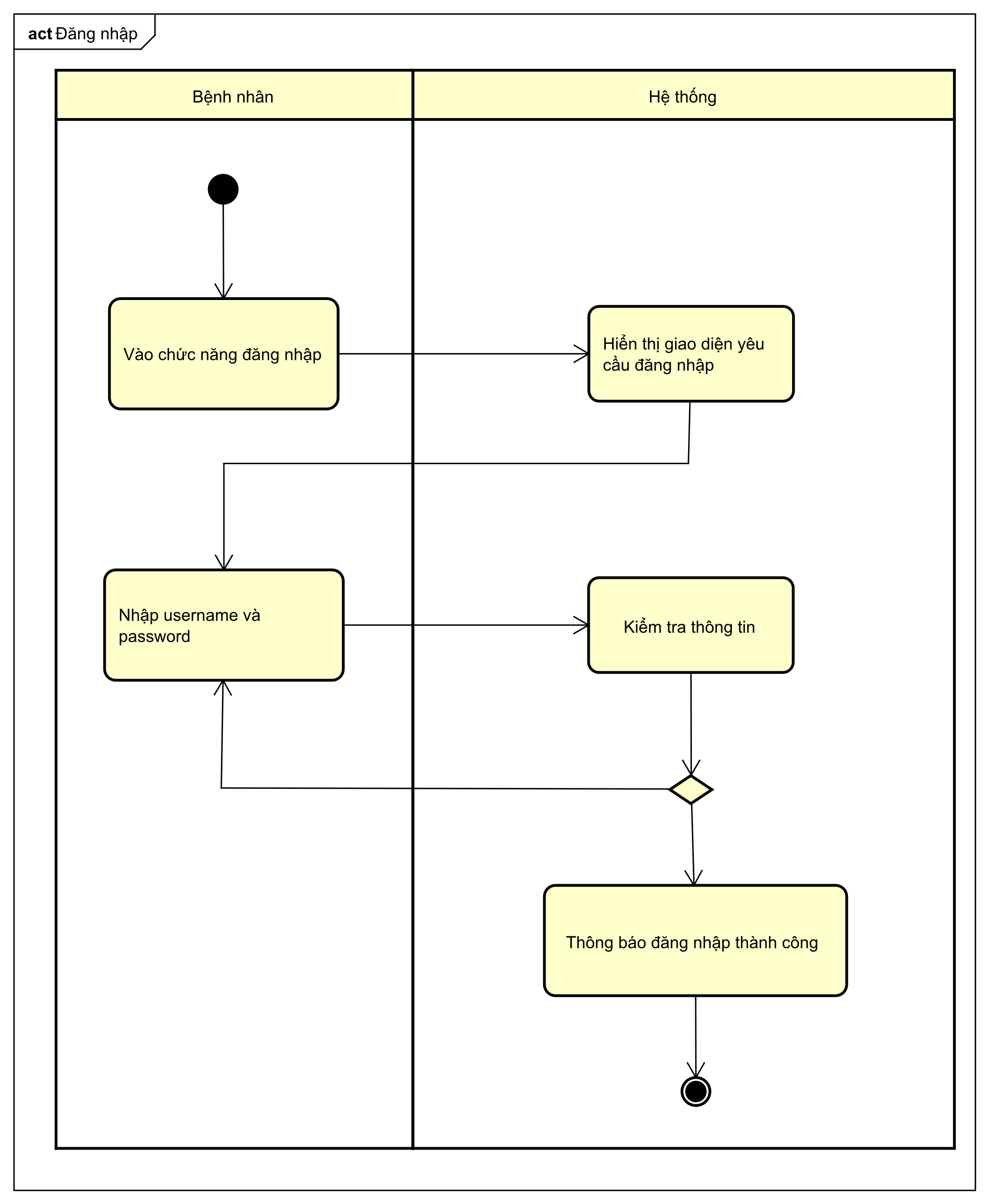
#### Sơ đồ thống kê doanh thu năm:



Hình .: Hình sơ đồ tuần tự Thống kê doanh thu năm

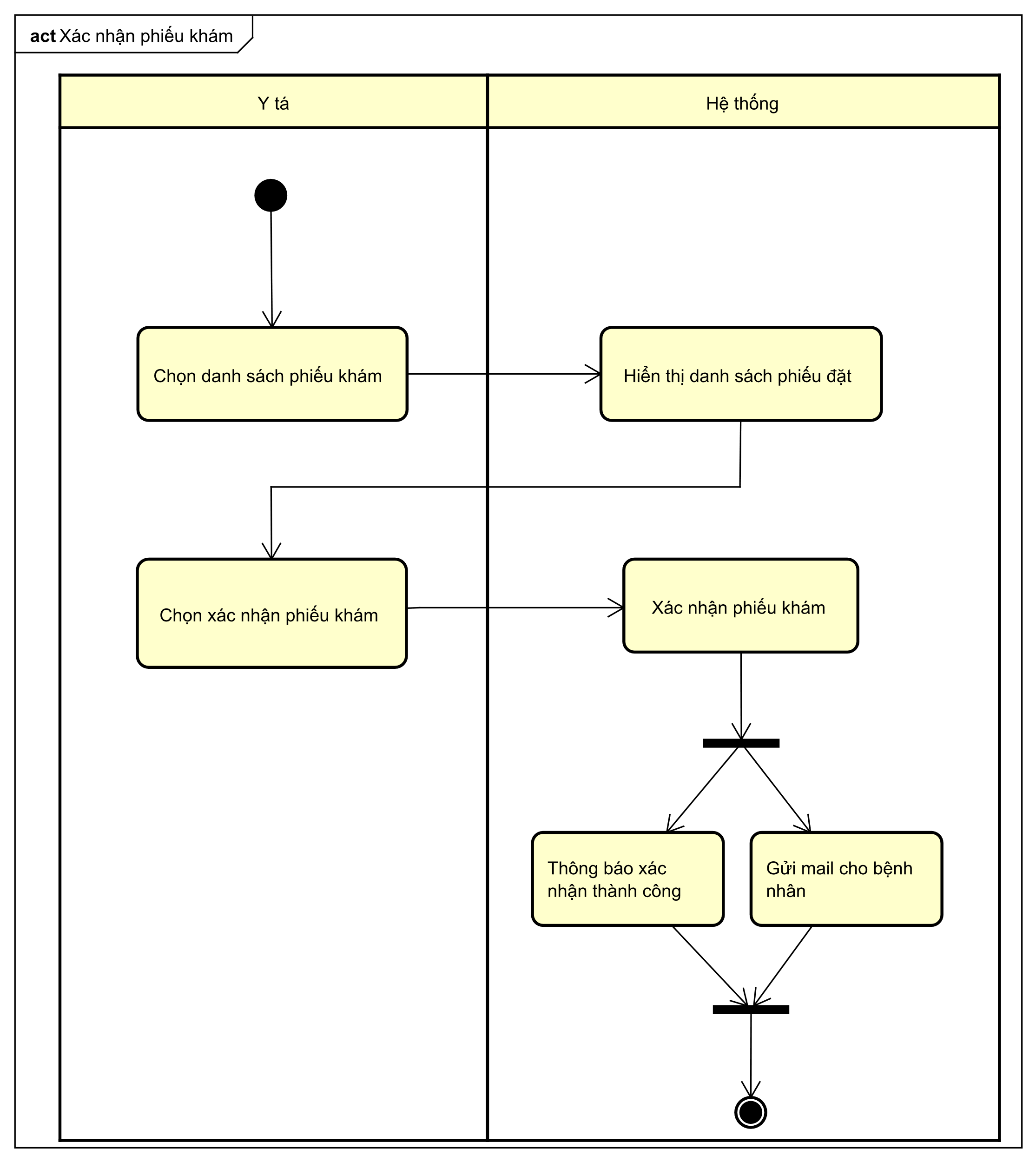
### Sơ đồ hoạt động:

#### Sơ đồ hoạt động “Đăng nhập”:



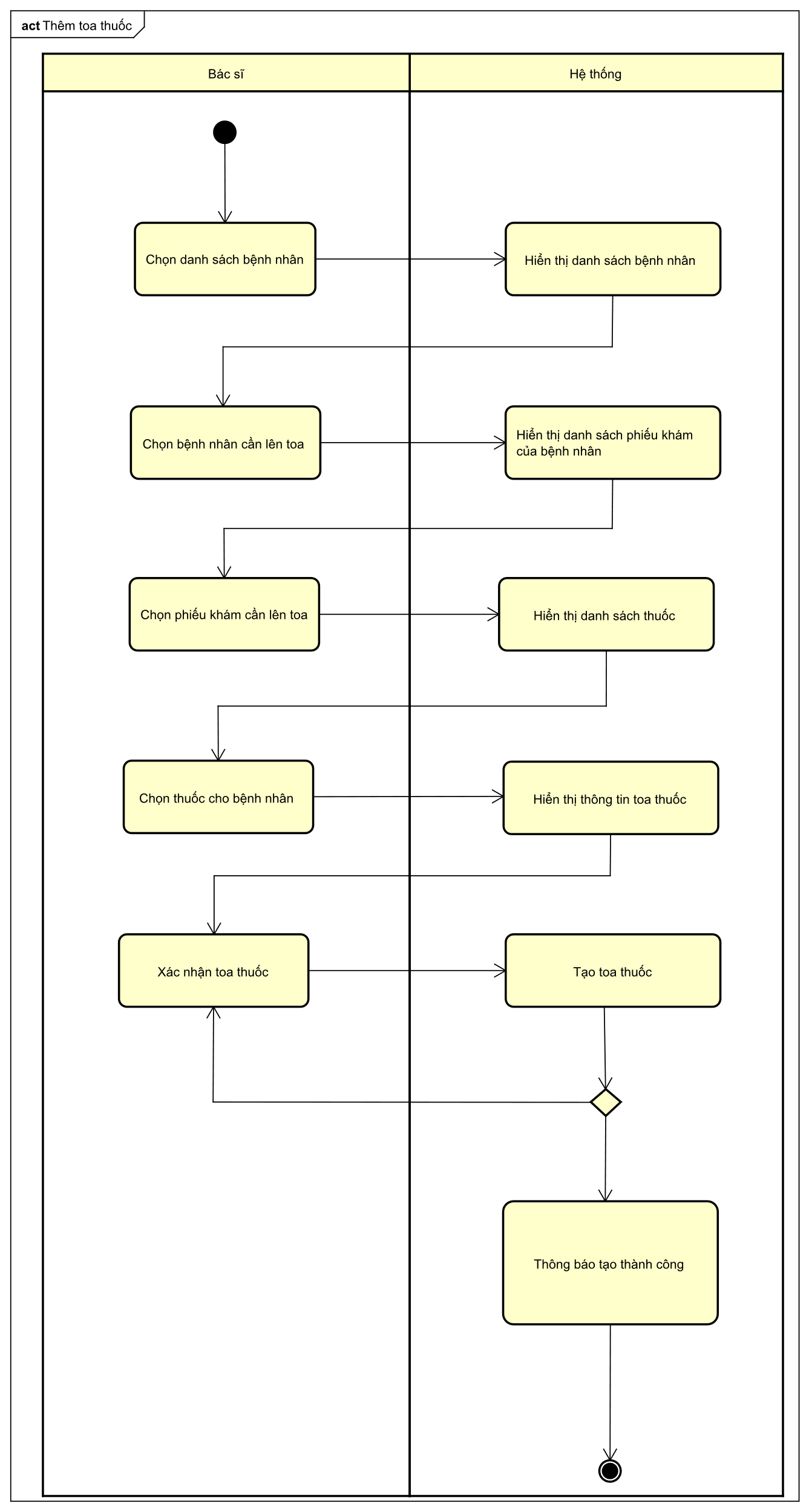
Hình .: Hình sơ đồ hoạt động Đăng nhập

#### Sơ đồ hoạt động “Xác nhận phiếu khám”:



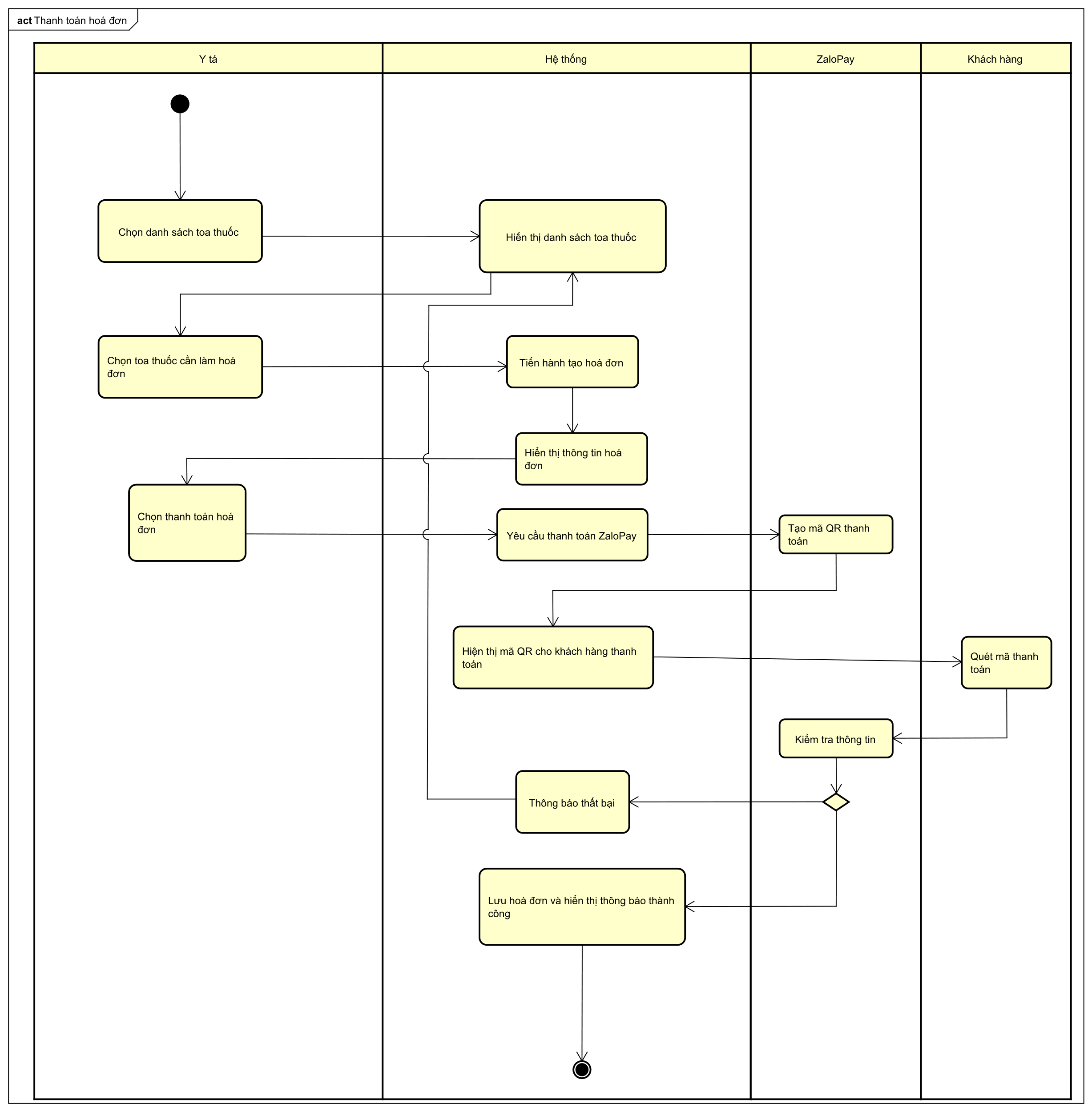
Hình .: Hình sơ đồ hoạt động Xác nhận phiếu khám

#### Sơ đồ hoạt động “Thêm toa thuốc”:



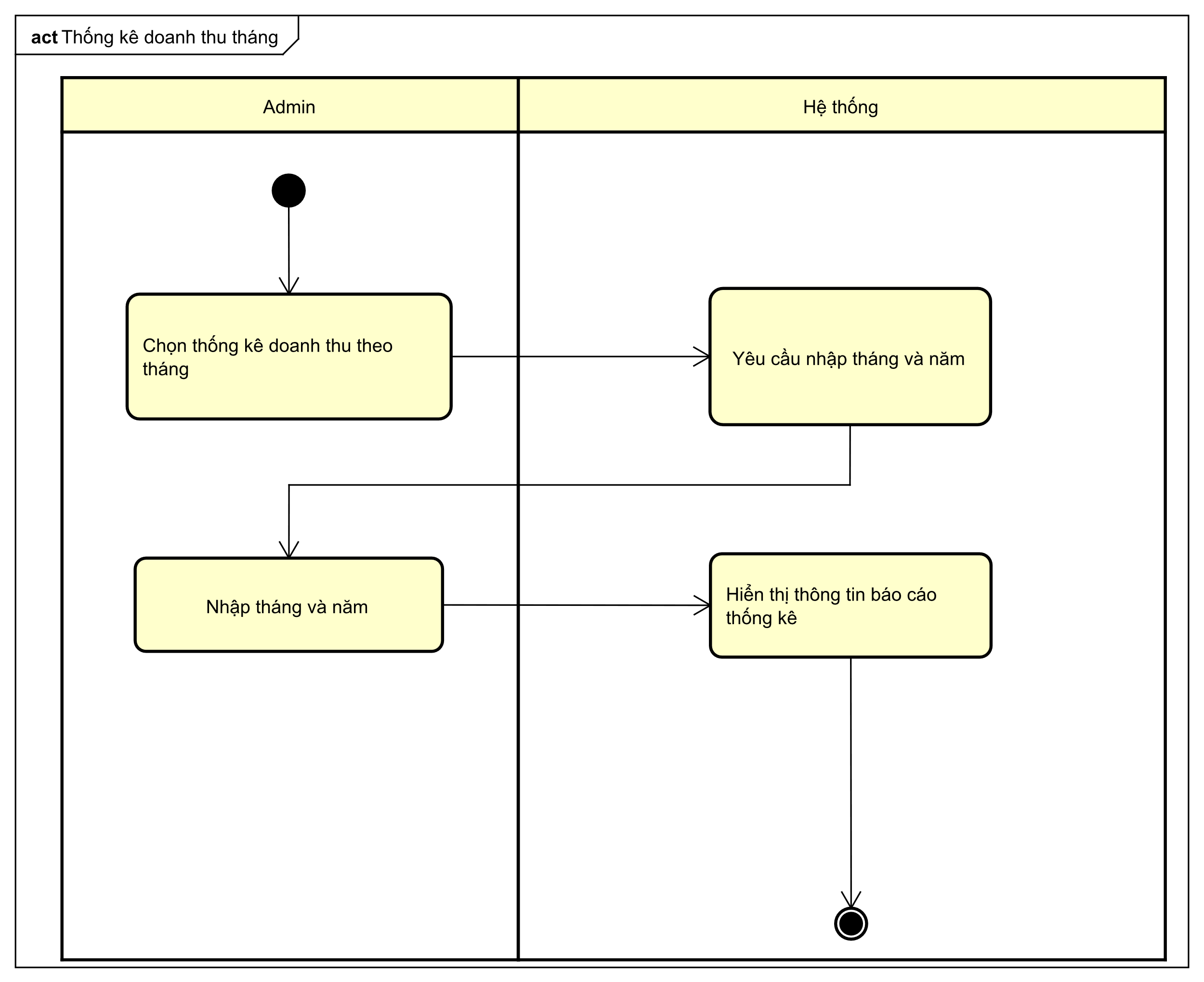
Hình .: Hình sơ đồ hoạt động Thêm toa thuốc

#### Sơ đồ hoạt động “Thanh toán hoá đơn”



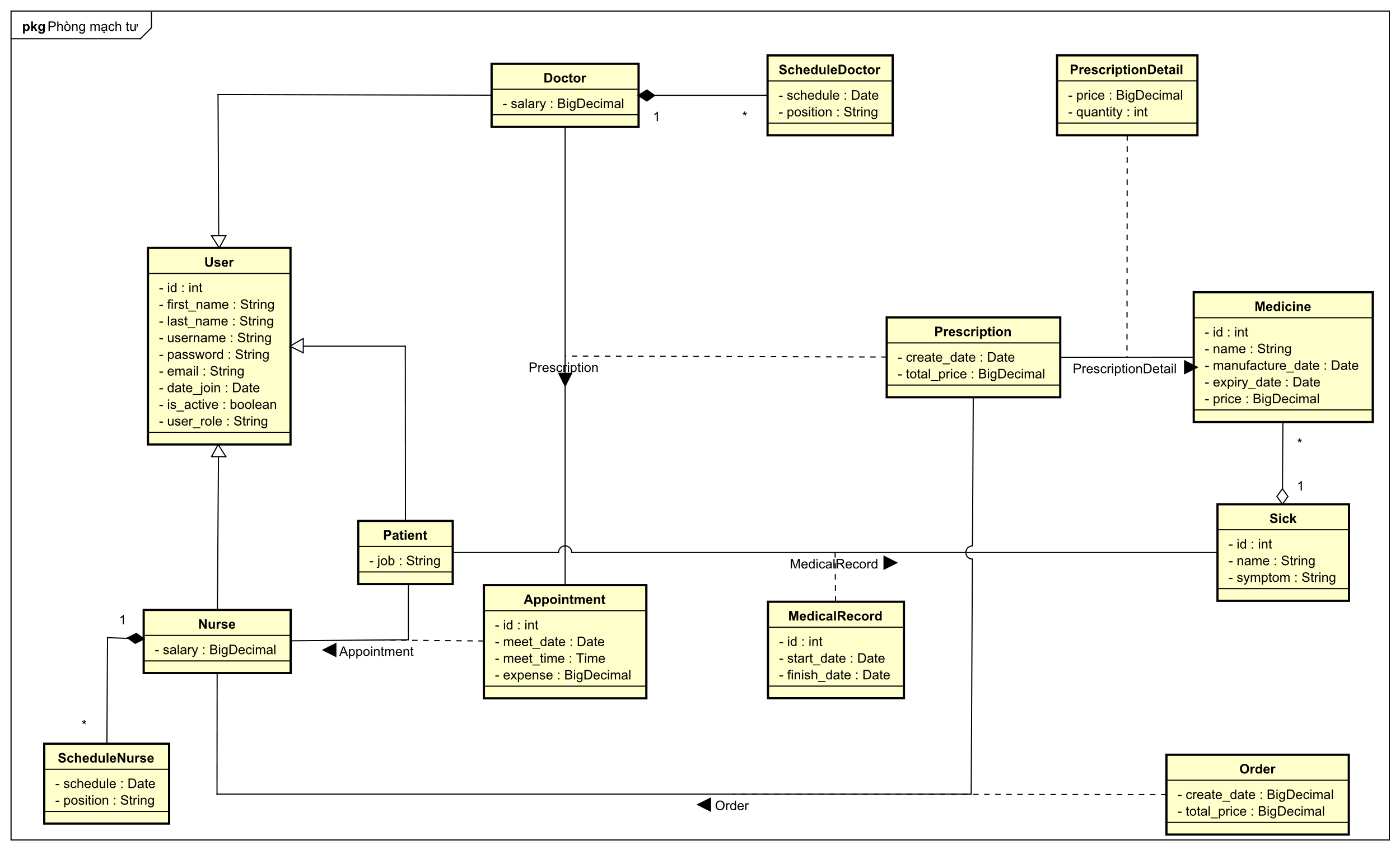
Hình .: Hình sơ đồ hoạt động Thanh toán hoá đơn

#### Sơ đồ thống kê doanh thu tháng:



Hình .: Hình sơ đồ hoạt động Thống kê doanh thu tháng

### Sơ đồ lớp:



Hình .: Hình sơ đồ lớp hệ thống

### Mô hình cơ sở dữ liệu:

#### Các bảng dữ liệu:

###### Bảng User:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã người dùng |
| username | Varchar(250) |  | Tên đăng nhập |
| password | Varchar(250) |  | Mật khẩu |
| first\_name | Varchar(250) |  | Tên |
| last\_name | Varchar(250) |  | Họ |
| email | Varchar(250) |  | Email |
| user\_role | Varchar(250) |  | Quyền người dùng |
| is\_active | Bit |  | Trạng thái hoạt động |
| date\_joined | DateTime |  | Ngày tham gia |
| avatar | Varchar(250) |  | ảnh đại diện |

Bảng .: Bảng CSDL “User”

###### Bảng Doctor:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| User\_id | Int | Khóa chính và khoá ngoại | Mã bác sĩ tham chiếu đến mã người dùng |
| salary | BigDecimal |  | Lương bác sĩ |

Bảng .: Bảng CSDL “Doctor”

###### Bảng Nurse:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| User\_id | Int | Khóa chính và khoá ngoại | Mã y tá tham chiếu đến mã người dùng |
| salary | BigDecimal |  | Lương y tá |

Bảng .: Bảng CSDL “Nurse”

###### Bảng Patient:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| User\_id | Int | Khóa chính và khoá ngoại | Mã bác sĩ tham chiếu đến mã người dùng |
| job | Varchar(250) |  | Nghề nghiệp |

Bảng .: Bảng CSDL “Patient”

###### Bảng Sick:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã bệnh |
| name | Varchar(250) |  | Tên bệnh |
| symptom | Varchar(250) |  | Triệu chứng |

Bảng .: Bảng CSDL “Sick”

###### Bảng Medicine:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã thuốc |
| name | Varchar(250) |  | Tên thuốc |
| Matufacturing\_date | DateTime |  | Ngày sản xuất |
| Expiry\_date | DateTime |  | Hạn sử dụng |
| Price | BigDecimal |  | Giá thuốc |
| count\_in\_stock | Int |  | Số lượng |
| Sick\_id | Int | Khoá ngoại | Mã bệnh |
| image | Varchar(250) |  | Hình ảnh |

Bảng .: Bảng CSDL “Medicine”

###### Bảng Appointment:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã phiếu hẹn |
| Meet\_date | Date |  | Ngày khám |
| Meet\_time | Time |  | Giờ khám |
| expense | BigDecimal |  | Phí khám |
| Nurse\_id | Int | Khoá ngoại | Mã y tá |
| Patient\_id | Int | Khoá ngoại | Mã bệnh nhân |

Bảng .: Bảng CSDL “Appointment”

###### Bảng Prescription:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã toa thuốc |
| created\_date | DateTime |  | Ngày tạo |
| Total\_price | BigDecimal |  | Tổng tiền |
| Doctor\_id | Int | Khoá ngoại | Mã bác sĩ |
| appointment\_id | Int | Khoá ngoại | Mã phiếu hẹn |
| Is\_paid | Bit |  | Trạng thái thanh toán |

Bảng .: Bảng CSDL “Presscription”

###### Bảng PrescriptionDetail:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã chi tiết toa thuốc |
| Quantity | Int |  | Số lượng |
| Price | BigDecimal |  | Đơn giá |
| medicine\_id | Int | Khoá ngoại | Mã thuốc |
| prescription\_id | Int | Khoá ngoại | Mã toa thốc |

Bảng .: Bảng CSDL “PrescriptionDetail”

###### Bảng Order:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã hoá đơn |
| created\_date | DateTime |  | Ngày tạo |
| Total\_price | BigDecimal |  | Tổng tiền |
| prescription \_id | Int | Khoá ngoại | Mã toa thuốc |
| Nurse \_id | Int | Khoá ngoại | Mã y tá |

Bảng .: Bảng CSDL “Order”

###### Bảng MedicalRecord:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã hồ sơ bệnh |
| start\_date | DateTime |  | Ngày bắt đầu |
| end\_price | BigDecimal |  | Ngày kết thúc |
| patient \_id | Int | Khoá ngoại | Mã bệnh nhân |
| sick \_id | Int | Khoá ngoại | Mã bệnh |

Bảng .: Bảng CSDL “MedicalRecord”

###### Bảng ScheduleDoctor:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã lịch trực |
| position | Varchar(250) |  | Vị trí |
| schedule | DateTime |  | Giờ trực |
| doctor \_id | Int | Khoá ngoại | Mã bác sĩ |

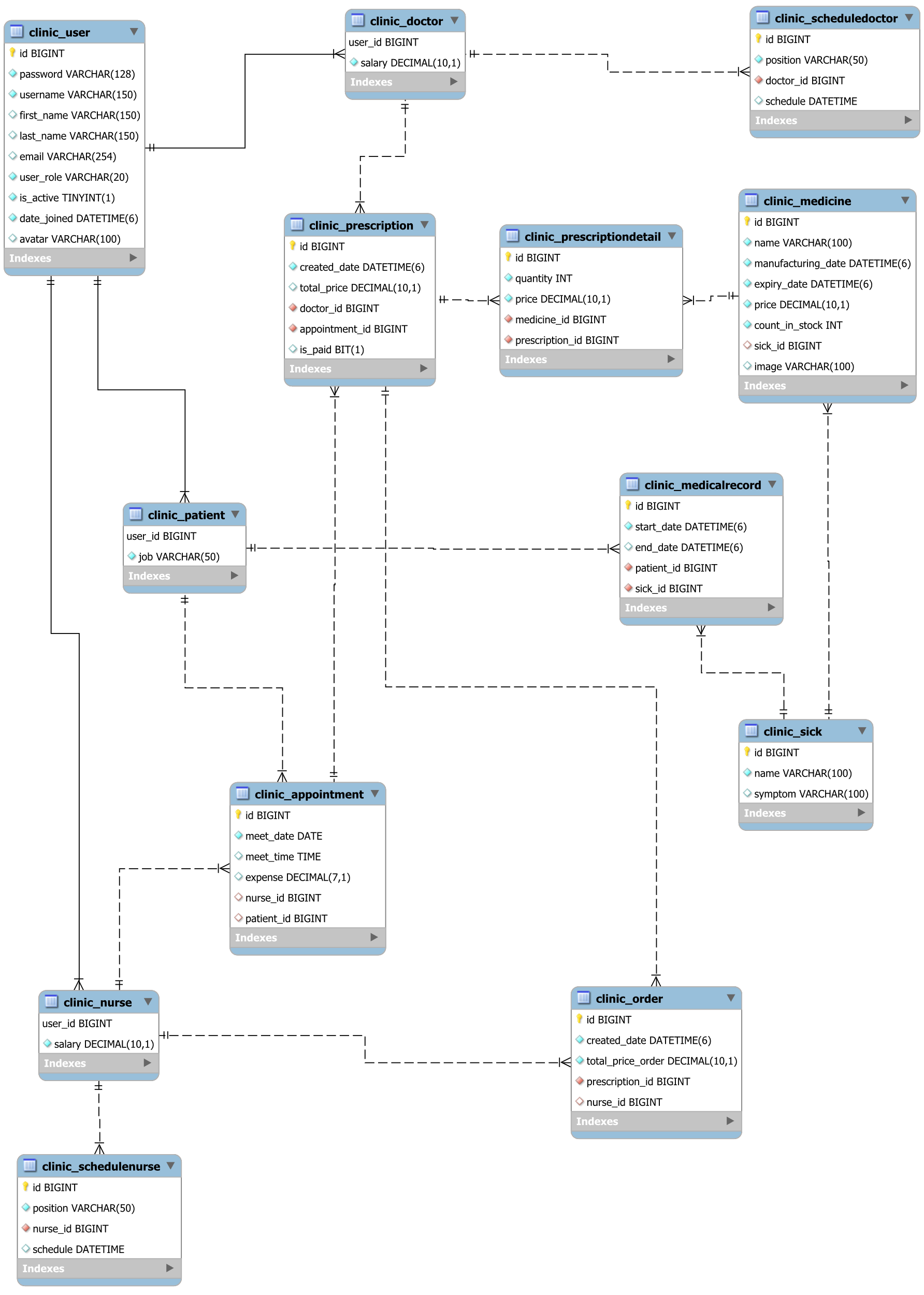
Bảng .: Bảng CSDL “ScheduleDoctor”

###### Bảng ScheduleNurse:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên trường** | **Kiểu dữ liệu** | **Khóa** | **Ghi chú** |
| id | Int | Khóa chính | Mã lịch trực |
| position | Varchar(250) |  | Vị trí |
| schedule | DateTime |  | Giờ trực |
| Nurse \_id | Int | Khoá ngoại | Mã y tá |

Bảng .: Bảng CSDL “ScheduleNurse”

#### Sơ đồ CSDL:



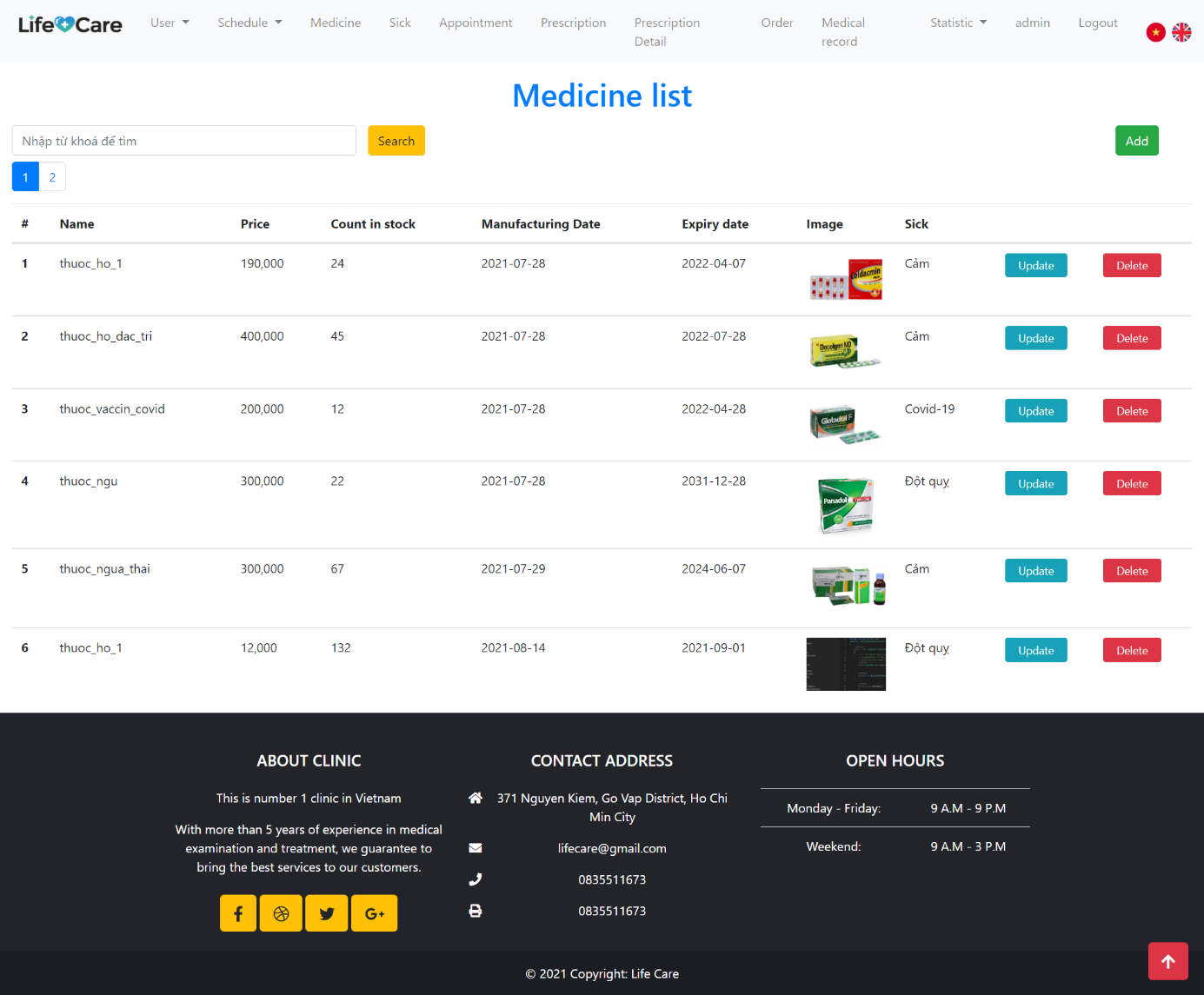
Hình .: Hình sơ đồ CSDL hệ thống

## Thiết kế giao diện

### Giao diện Admin:

#### Giao diện Quản lý thuốc:

Giao diện cho phép Admin có thể quản lý danh sách thuốc của phòng mạch tư bao gồm thêm, sửa, xoá, tra cứu.



Hình .: Hình giao diện quản lý thuốc

#### Giao diện Thêm thuốc:

Giao diện cho phép Admin thêm các thông tin cần thiết cho việc thêm dữ liệu thuốc vào hệ thống.

Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình .: Hình giao diện thêm thuốc

#### Giao diện thống kê số lượng bệnh nhân đến khám:

Giao diện cho phép Admin thống kê số lượng bệnh nhân đến khám trong 1 năm theo từng tháng.



Hình .: Hình giao diện thống kê số lượng bệnh nhân

### Giao diện Bác sĩ:

#### Giao diện xem danh sách bệnh nhân:

Giao diện cho phép bác sĩ xem danh sách bệnh nhân trong hệ thống với các chức năng tra cứu, xem hồ sơ bệnh và xem toa thuốc.

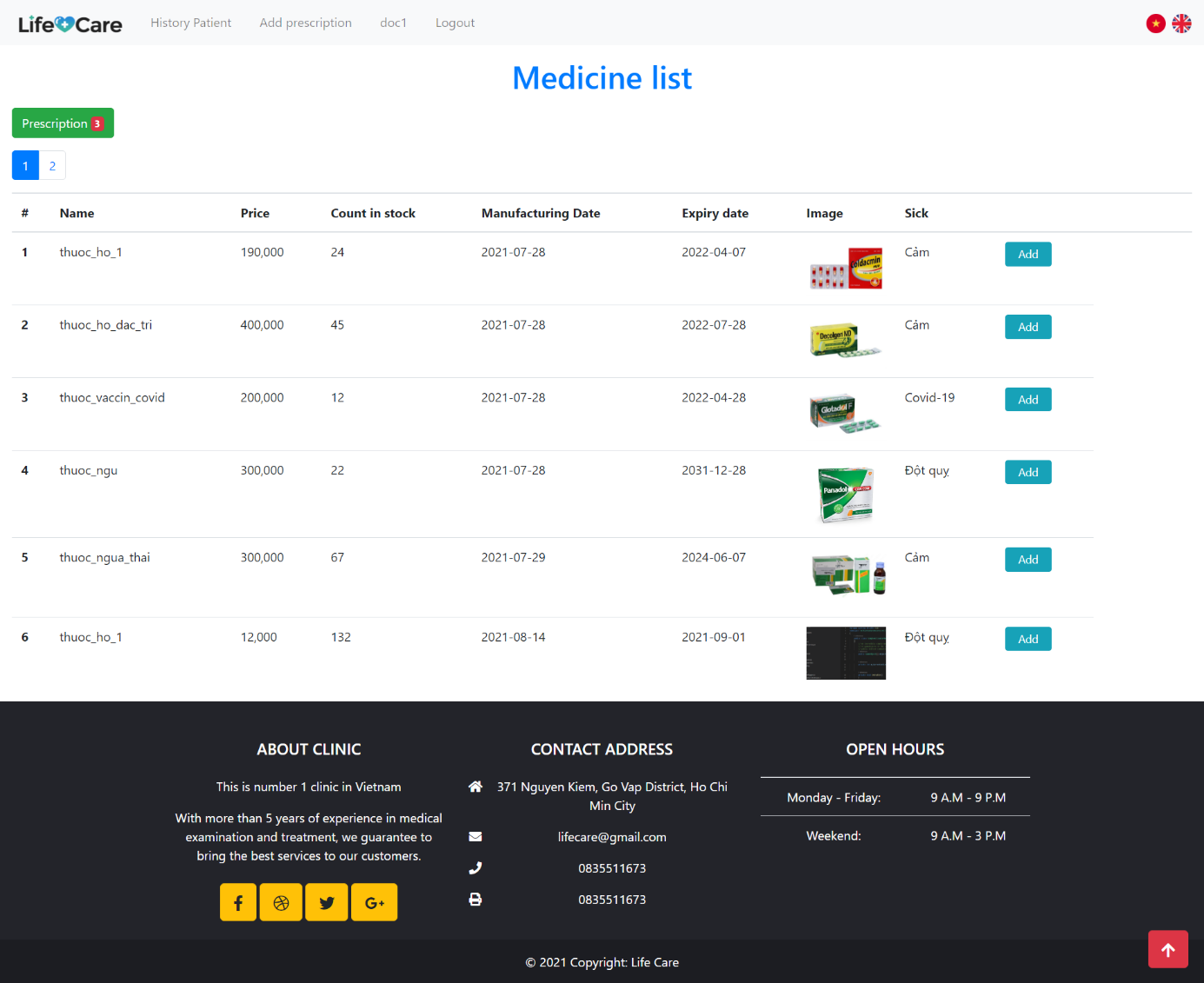
Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, đen

Mô tả được tạo tự động

Hình .: Hình giao diện xem danh sách bệnh nhân

#### Giao diện thêm thuốc vào toa:

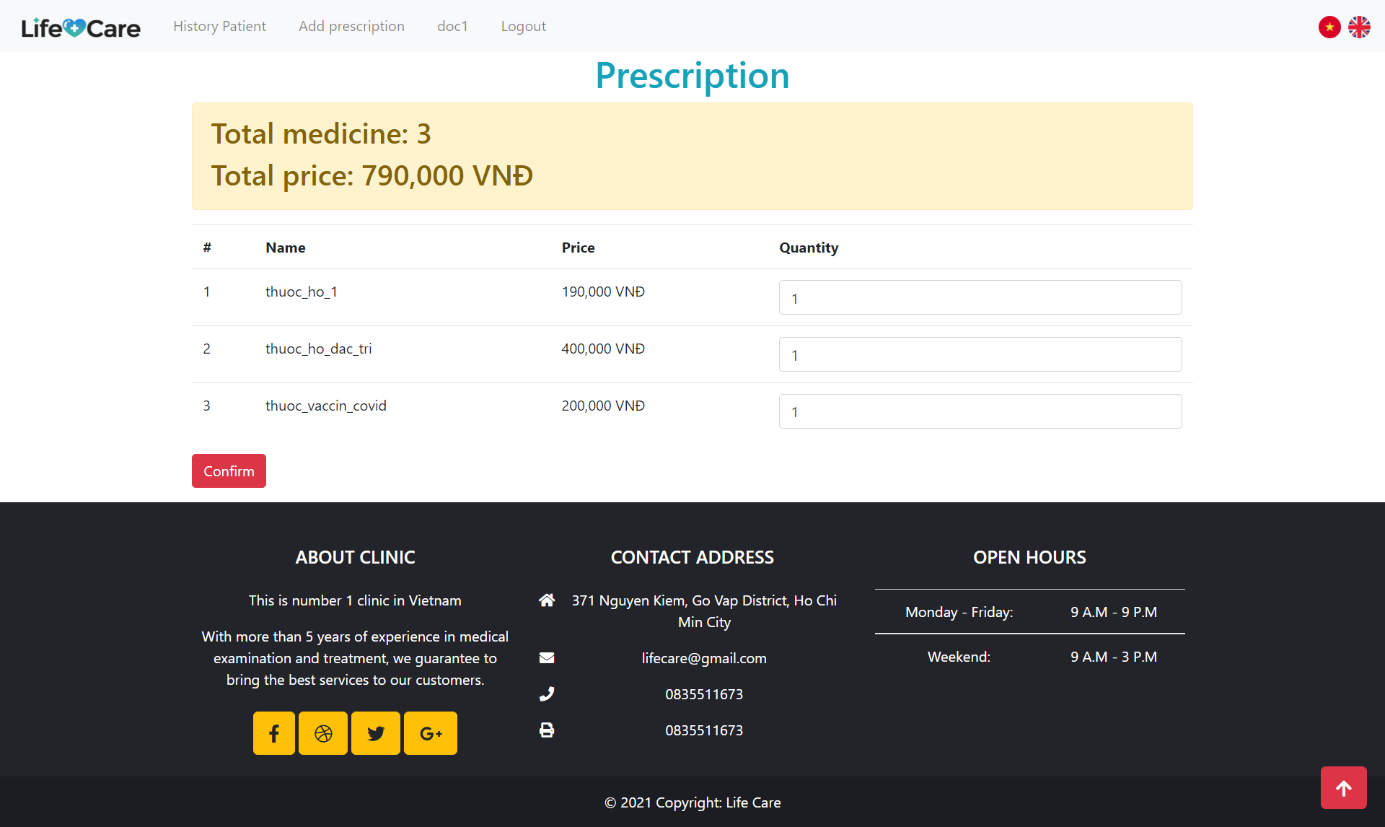
Giao diện cho phép bác sĩ thêm thuốc vào toa cho bệnh nhân.



Hình .: Hình giao diện thêm thuốc vào toa

#### Giao diện check toa thuốc:

Giao diện cho phép bác sĩ xem lại số thuốc đã thêm trong toa để bác sĩ có thể điều chỉnh theo mong muốn.



Hình .: Hình giao diện check toa thuốc

### Giao diện y tá:

#### Giao diện chọn toa thuốc để tạo hoá đơn:

Giao diện cho phép y tá xem danh sách toa thuốc và chọn toa thuốc cần lên hoá đơn.

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, đen

Mô tả được tạo tự động

Hình .: Hình giao diện chọn toa thuốc tạo hoá đơn

#### Giao diện thêm hoá đơn:

Giao diện cho phép y tá kiểm tra hoá đơn để thanh toán

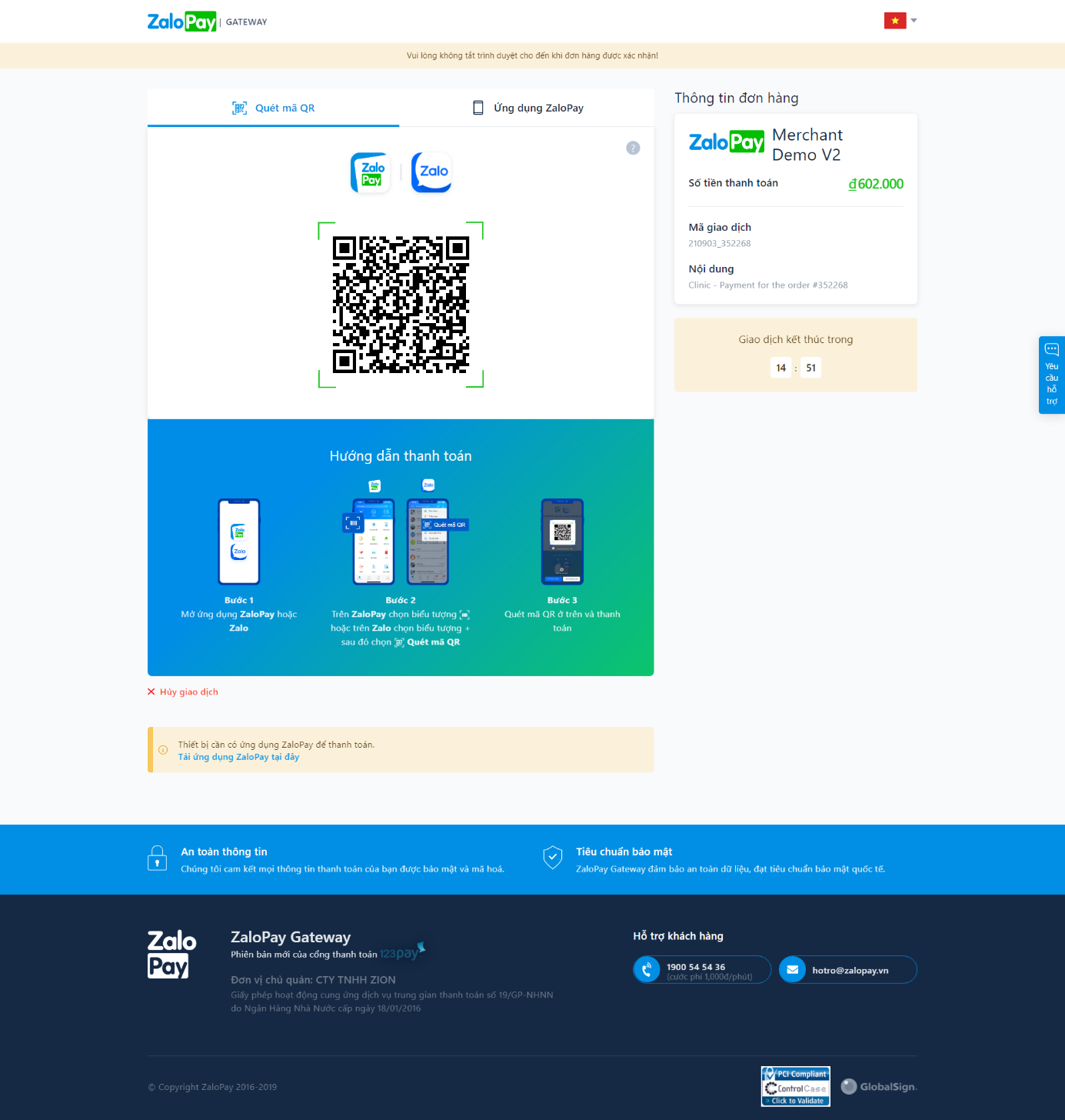
Ảnh có chứa văn bản

Mô tả được tạo tự động

Hình .: Hình giao diện thêm hoá đơn

#### Giao diện thanh toán ZaloPay:

Giao diện thanh toán online bằng ví điện tử ZaloPay giúp bệnh nhân có thể thanh toán không dùng tiền mặt bằng cách quét mã QR trên app ZaloPay điện thoại.



Hình .: Hình giao diện thanh toán zalopay

### Giao diện bệnh nhân:

#### Giao diện đăng nhập:

Giao diện giúp cho người dùng đăng nhập vào hệ thống để sử dụng các chức năng.

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình

Mô tả được tạo tự động

Hình .: Hình giao diện đăng nhập

#### Giao diện đăng ký:

Giao diện cho phép bệnh nhân đăng kí tài khoản để đặt lịch khám và tra cứu thông tin cần thiết.

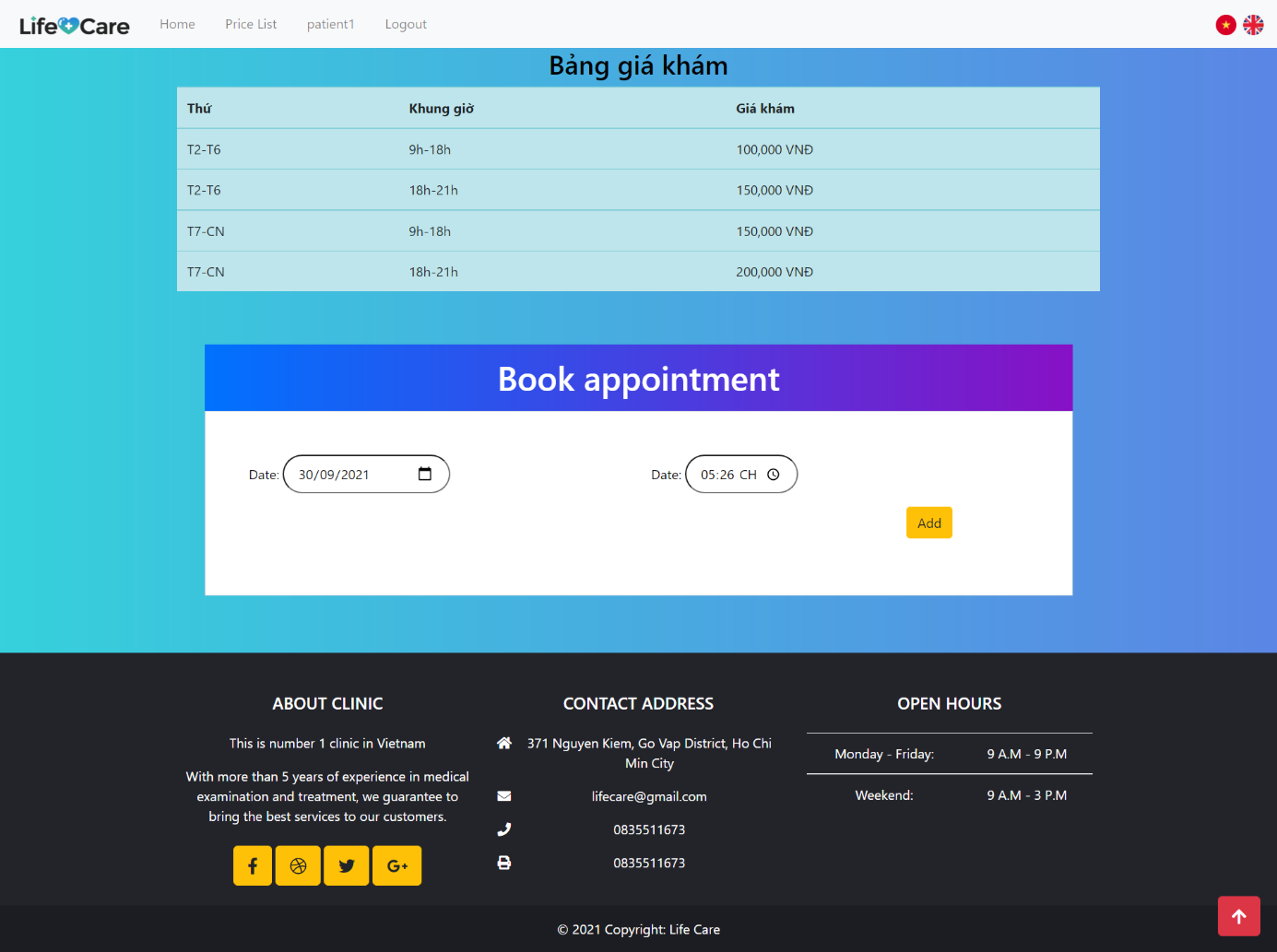
Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, đen

Mô tả được tạo tự động

Hình .: Hình giao diện đăng ký

#### Giao diện tạo phiếu khám:

Giao diện cho phép bệnh nhân xem bảng giá khám và đặt phiếu khám theo ngày giờ mong muốn.



Hình .: Hình giao diện tạo phiếu khám

# TỔNG KẾT

## Kết quả đạt được:

Qua quá trình thực hiện đồ án môn học này, em đã thu về được một số kết quả như sau:

* Hiểu biết thêm về ứng dụng web công nghệ Spring Framework.
* Hiểu biết về ngôn ngữ lập trình Java ứng dụng vào Web của Oracle, cụ thể là Spring MVC.
* Hiển thêm một số ngôn ngữ như JSP, Tiles, CSS, …
* Hiểu biết về ngôn ngữ truy vấn CSDL là Hibernate, một ngôn ngữ truy vấn CSDL theo dạng hướng đối tượng, gần gũi với người lập trình từ trước tới nay tương tác với các CSDL khác như XML, CSDL quan hệ,…
* Đồ án giúp em tổng hợp các kiến thức đã học từ các môn cơ sở của trường đã trang bị cho sinh viên từ các học kỳ trước như Lập trình Java, Thiết kế web, CSDL, Hệ quản trị CSDL, Công nghệ phần mềm, Kỹ thuật lập trình, Lập trình hướng đối tượng (OOP), Phân tích thiết kế hệ thống thông tin, ….
* Nâng cao khả năng lập trình, cũng như được biết thêm mốt số công nghệ mới như: thanh toán trực tuyến bằng ZaloPay, FaceBook API, Google API, ...
* Trên cơ sở kiến thức đã học em đã phát họa cơ bản một ứng web mang tính nhu cầu phục vụ cho việc học. Tuy ứng dụng web còn khá đơn giản về nội dung cũng chưa đầy đủ nhiều chức năng cần thiết, chưa tối ưu về mặt giao diện, giải thuật hay xử lý lỗi, ... nhưng đây là bước làm nền tảng cho em có thể phát triển, xây dựng các ứng dụng web thực tế hơn.

## Hoàn thành:

|  |  |  |
| --- | --- | --- |
| STT | Chức năng | Mức độ hoàn thành |
| 1 | Quản lý của Admin | Hoàn thành |
| 17 | Báo cáo thống kê | Hoàn thành |
| 21 | Đăng nhập | Hoàn thành |
| 22 | Đăng ký | Hoàn thành |
| 23 | Lên toa thuốc | Hoàn thành |
| 24 | Chọn phiếu khám | Hoàn thành |
| 25 | Tra cứu thuốc | Hoàn thành |
| 26 | Thêm hồ sơ bệnh | Hoàn thành |
| 27 | Xem hồ sơ bệnh | Hoàn thành |
| 28 | Xem lịch sử khám | Hoàn thành |
| 29 | Xem lịch trực | Hoàn thành |
| 30 | Xác nhận phiếu khám | Hoàn thành |
| 31 | Thêm hoá đơn | Hoàn thành |
| 32 | Thanh toán | Hoàn thành |
| 33 | Đặt phiếu khám | Hoàn thành |
| 34 | Xem thông tin cá nhân | Hoàn thành |
| 35 | Xem danh sách phiếu khám đã đặt | Hoàn thành |
| 36 | Y tá gửi mail xác nhận phiếu khám | Hoàn thành |
| 37 | Đăng nhập bằng facebook | Hoàn thành |
| 38 | Đăng nhập bằng google | Hoàn thành |

Bảng .: Bảng danh sách các chức năng hoành thành

## Hướng phát triển:

Ở trang người dùng, chúng ta cần nên hoàn thiện hơn về bố cục trang web, trang trí cũng như thiết kế cho trang web mang tính thân thiện cho người dùng hơn nữa.

Cần phát triển việc triển khai hệ thống trên các thiết bị di động

Với tình hình dịch Covid-19 hiện tại cần phát triển thêm tính năng video call để bệnh nhân có thể trao đổi với bác sĩ mà không cần gặp trực tiếp.

Cần phải làm thêm về việc các lỗi 404, 500 và thêm các công nghệ Web Service,…

TÀI LIỆU THAM KHẢO

[1] R.Johnson, J.Hoeller, A.Arendsen, T.Risberg and D.Kopylenko, *Professional Java Development with the Spring Framework*. United Kingdom: Wrox Press Ltd., 2005.

[2] Vibrant Publishers, *Hibernate, Spring & Struts Interview Questions You'll Most Likely Be Asked*. Scotts Valley, CA: CreateSpace, Scotts Valley, 2011.

[3] E.Foster and S.Godbole, “Overview of MySQL”, In *Database Systems: A Pragmatic Approach*, Second Edition. New York: Apress, 2016, pp.451-460.