# 

**Mục Lục**

[LỜI CẢM ƠN 2](#_Toc7869403)

[DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ 5](#_Toc7869404)

[CHƯƠNG I. ĐẶT VẤN ĐỀ VÀ ĐỊNH HƯỚNG GIẢI PHÁP 6](#_Toc7869405)

[1.1 Đặt vấn đề 6](#_Toc7869406)

[**1.1.1** **Lý do chọn đề tài** 6](#_Toc7869407)

[**1.1.2** **Mục đích của đề tài** 6](#_Toc7869408)

[1.2 Định hướng giải pháp 6](#_Toc7869409)

[**1.2.1** **Nội dung đề tài** 6](#_Toc7869410)

[**1.2.2** **Lựa chọn công nghệ** 7](#_Toc7869411)

[CHƯƠNG II. CƠ SỞ LÝ THUYẾT 8](#_Toc7869412)

[2.1 Giới thiệu Laravel framework 8](#_Toc7869413)

[**2.1.1** **Khái niệm** 8](#_Toc7869414)

[**2.1.2** **Tại sao chọn Laravel framework** 10](#_Toc7869415)

[2.2 Mô hình MVC 11](#_Toc7869416)

[2.3 jQuery, boostrap 13](#_Toc7869417)

[**2.3.1** **jQuery** 13](#_Toc7869418)

[**2.3.2** **Boostrap** 14](#_Toc7869419)

[**2.3.3** **Hệ quản trị cơ sở dữ liệu MySQL** 14](#_Toc7869420)

[CHƯƠNG III. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 16](#_Toc7869421)

[1 Xác định yêu cầu hệ thống 16](#_Toc7869422)

[2 Thiết kế ứng dụng web phía client 17](#_Toc7869423)

[**2.1** **Biểu đồ Use Case tổng quát** 17](#_Toc7869424)

[**2.2** **Đặc tả một số chức năng chính** 22](#_Toc7869425)

[**2.3** **Mô hình hóa hoạt động của hệ thống** 27](#_Toc7869435)

[**2.4** **Mô hình hóa sự tương tác** 33](#_Toc7869436)

[**2.5** **Thiết kế lớp (Class)** 39](#_Toc7869437)

[3 Thiết kế ứng dụng web quản trị 44](#_Toc7869438)

[**3.1** **Biểu đồ Use Case tổng quát** 44](#_Toc7869439)

[**3.2** **Đặc tả một số chức năng chính** 47](#_Toc7869440)

[**3.3** **Mô hình hóa hoạt động của hệ thống** 49](#_Toc7869444)

[**3.4** **Mô hình hóa sự tương tác** 51](#_Toc7869445)

[**3.5** **Thiết kế lớp (class)** 52](#_Toc7869446)

[4 Xây dựng ứng dụng và triển khai 53](#_Toc7869447)

[**4.1** **Thiết kế cơ sở dữ liệu** 53](#_Toc7869448)

[**4.2** **Đặc tả cơ sở dữ liệu** 53](#_Toc7869448)

[**5 Thiết kế giao diện** 65](#_Toc7869449)

[CHƯƠNG IV. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 76](#_Toc7869451)

[4.1 Các vấn đề được giải quyết 76](#_Toc7869453)

[4.2 Hướng phát triển trong tương lai 76](#_Toc7869454)

[TÀI LIỆU THAM KHẢO 77](#_Toc7869455)

# DANH MỤC CÁC TỪ VIẾT TẮT VÀ THUẬT NGỮ

|  |  |
| --- | --- |
| **Từ viết tắt, thuật ngữ** | **Cụm từ đầy đủ** |
| CSS | Cascading Style Sheets |
| HTML | HyperText Markup Language |
| HTTP | Hypertext Transfer Protocol |
| MVC | Model – View - Controller |
| CSDL | Cơ sở dữ liệu |
| PHP | Hypertext Preprocessor |
| XML | Extensible Markup Language |
| AJAX | Asynchronous JavaScript and XML |
| DOM | Document Object Model |
| JSON | JavaScript Object Notation |

# CHƯƠNG I. ĐẶT VẤN ĐỀ VÀ ĐỊNH HƯỚNG GIẢI PHÁP

## Đặt vấn đề

### **1.1.1 Lý do chọn đề tài**

Dựa trên vấn đề về nhu cầu đi du lịch của mọi người hiện nay, cùng với những ưu điểm vượt trội của Laravel framework nên em muốn xây dựng một website quản lý bệnh viên bằng Laravel. Website này giúp bệnh nhân có thể tìm kiếm, tham khảo, đặt các phòng của bệnh viện và giúp người quản trị dễ dàng quản lý các loại dịch vụ,phòng, dễ dàng cập nhật thay đổi để website của bệnh viện luôn luôn mới mẻ, thu hút bệnh nhân đến với website. Với việc xây dựng website bằng Laravel giúp em hiểu rõ hơn về Larvel framework.

### **Mục đích của đề tài**

Tìm hiểu và xây dựng hệ thống quản lý bệnh viện cho phép bệnh nhân có thể đăng ký khám bệnh, tra cứu thông tin hồ sơ y tế một cách dễ dàng và nhanh chóng. Hệ thống hỗ trợ nhân viên y tế nhập liệu, duyệt và xử lý các hồ sơ khám chữa bệnh, đồng thời cung cấp các công cụ thống kê chi tiết giúp ban quản lý bệnh viện theo dõi, quản lý và tối ưu hóa hoạt động của bệnh viện một cách hiệu quả hơn.

## Định hướng giải pháp

### **1.1 Định hướng giải pháp**

Hiện nay, nước ta có rất nhiều hệ thống quản lý bệnh viện, tuy nhiên phần lớn các hệ thống này có cấu trúc và tổ chức thông tin cố định, ít linh hoạt hoặc khó thay đổi. Đa số chỉ tập trung vào việc quản lý thông tin cơ bản và giới thiệu dịch vụ của bệnh viện mà chưa đáp ứng tốt các yêu cầu quản lý chuyên sâu và tùy chỉnh theo nhu cầu riêng biệt của từng cơ sở y tế.

Nhiều hệ thống hiện nay được thiết kế bằng ASP.NET, một framework mạnh mẽ của Microsoft. Dù ASP.NET mang lại nhiều ưu điểm vượt trội trong phát triển phần mềm, nhưng vẫn tồn tại hai hạn chế chính:

1. Không hỗ trợ chạy trên hệ điều hành Linux, làm giảm tính linh hoạt và khả năng triển khai trên nhiều môi trường khác nhau.
2. Việc phát triển và bảo trì gặp khó khăn do thiếu sự hỗ trợ toàn diện từ các công cụ như Visual Studio trong suốt quá trình phát triển.

### **Nội dung đề tài**

* Xây dựng các chức năng website phía client gồm:
  + Bệnh nhân có thể xem phòng , đặt phòng.
  + Bệnh nhân có thể thanh toán viện phí đã đặt.
  + Bệnh nhân có thể đăng ký làm thành viên và quản lý thông tin cá nhân của mình.
  + Bệnh nhân có thể gửi phản hồi về bệnh viện
* Xây dựng các chức năng website cho người quản trị gồm:
* Tìm kiếm các đơn đặt chỗ đã được xác nhận hay chưa xác nhận.
* Xác nhận đơn đặt chỗ hợp lệ, chỉnh sửa thông tin bệnh nhân và thông tin đặt chỗ trên từng đơn cụ thể.
* Thống kê tình hình đặt chỗ của từng phòng, xem danh sách bệnh nhân tham gia các phòng đó.
* Thay đổi, thêm mới hoặc xóa bỏ thông tin, hình ảnh về các loại dịch khác nhau mà bệnh viện muốn giới thiệu cho bệnh nhân.
* Theo dõi tình hình liên hệ thông qua website của bệnh nhân, để đáp ứng kịp thời nhu cầu của họ.

### **Lựa chọn công nghệ**

Website này em xây dựng bằng ASP.NET framework vì:

* Tổ chức file và cấu trúc code rõ ràng
* Hệ thống thư viện và công cụ phong phú
* Form Validation
* Code ít hơn, nhưng làm được nhiều việc hơn
* Mô hình MVC rõ ràng, dễ bảo trì
* Chức năng hỗ trợ bảo mật, validate dữ liệu, SQL Injection,…
* Template phong phú, thư viện form helper giúp xây dựng form nhanh và bảo mật
* Hỗ trợ đường dẫn thân thiện (Friendly Url)
* Bảo mật session, cookie,…

# CHƯƠNG II. CƠ SỞ LÝ THUYẾT

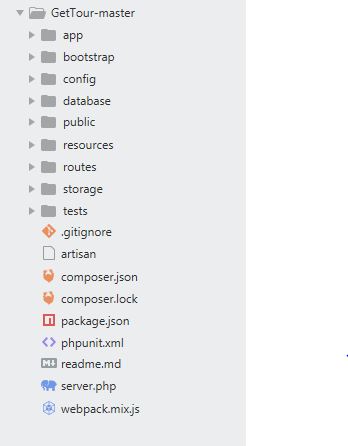
### **2.1 Giới thiệu ASP.NET Framework**

#### **2.1.1 Khái niệm**

ASP.NET là một framework mã nguồn mở do Microsoft phát triển, được thiết kế để xây dựng các ứng dụng web theo kiến trúc **Model-View-Controller (MVC)** hoặc Web API. ASP.NET cung cấp các tính năng mạnh mẽ như bảo mật, quản lý phiên, hỗ trợ kết nối cơ sở dữ liệu, và khả năng triển khai ứng dụng web một cách dễ dàng.

Hiện nay, ASP.NET là một trong những framework phổ biến nhất trên thế giới nhờ tận dụng tốt các kỹ thuật tiên tiến, hỗ trợ mạnh mẽ từ Microsoft và cộng đồng lập trình viên. Tuy nhiên, ASP.NET yêu cầu người dùng phải có kiến thức về môi trường phát triển của Microsoft, và có thể hạn chế với các dự án sử dụng các hệ điều hành không phải Windows.

#### **Cấu trúc thư mục trong một dự án ASP.NET**



**Hình 1: Cấu trúc thư mục ASP.NET Framework**

* **App\_Start**: Thư mục chứa các tệp khởi tạo ứng dụng, chẳng hạn như định tuyến, cấu hình bộ lọc, cấu hình Web API.
* **Controllers**: Chứa các tệp controller xử lý logic nghiệp vụ và điều hướng dữ liệu từ cơ sở dữ liệu đến giao diện người dùng.
* **Models**: Chứa các lớp định nghĩa dữ liệu và các logic liên quan đến xử lý dữ liệu.
* **Views**: Chứa các tệp giao diện người dùng (HTML, Razor).
* **Content**: Chứa các tệp CSS, hình ảnh, hoặc bất kỳ tài nguyên nào liên quan đến giao diện.
* **Scripts**: Chứa các tệp JavaScript hoặc thư viện front-end khác.
* **bin**: Chứa các tệp DLL được biên dịch khi xây dựng dự án.
* **Global.asax**: Tệp cấu hình toàn bộ ứng dụng, xử lý các sự kiện cấp ứng dụng như Application\_Start, Application\_Error.
* **web.config**: Tệp cấu hình chính cho ứng dụng ASP.NET, nơi định nghĩa các thiết lập bảo mật, kết nối cơ sở dữ liệu, và các cài đặt ứng dụng khác.

#### **2.1.2 Tại sao chọn ASP.NET Framework**

* **Hiệu năng cao**: ASP.NET cung cấp khả năng tối ưu hóa hiệu suất nhờ vào .NET runtime và các tính năng như caching, lazy loading, và hỗ trợ tối ưu hóa server-side.
* **Bảo mật mạnh mẽ**: Tích hợp sẵn các tính năng như xác thực người dùng (Authentication), ủy quyền (Authorization), và mã hóa dữ liệu.
* **Cộng đồng lớn**: Với sự hỗ trợ từ Microsoft, ASP.NET có tài liệu chi tiết, diễn đàn hỗ trợ chính thức và cộng đồng sôi động giúp giải quyết các vấn đề trong quá trình phát triển.
* **Hỗ trợ MVC pattern**: ASP.NET giúp tách biệt rõ ràng giữa logic nghiệp vụ, giao diện và dữ liệu, làm cho ứng dụng dễ bảo trì và phát triển.
* **Công cụ phát triển mạnh mẽ**: Visual Studio là IDE chính thức hỗ trợ ASP.NET, cung cấp các công cụ như IntelliSense, trình debug mạnh mẽ, và tích hợp kiểm thử.
* **Khả năng tích hợp tốt**: ASP.NET dễ dàng tích hợp với các dịch vụ đám mây của Microsoft (Azure), các hệ thống cơ sở dữ liệu (SQL Server), và các gói thư viện khác thông qua NuGet.

### **2.1.2 Tại sao chọn ASP.NET Framework**

* **ASP.NET Framework** có một kiến trúc thống nhất, dễ tiếp cận và được thiết kế khoa học, giúp những người mới làm quen với việc phát triển ứng dụng web có thể nắm bắt dễ dàng. Dù đơn giản nhưng ASP.NET hoàn toàn có khả năng xây dựng các hệ thống ứng dụng web phức tạp, với hiệu năng cao và khả năng mở rộng lớn.
* **Tài nguyên hỗ trợ phong phú**: Các tài liệu chính thức được cung cấp bởi Microsoft, cùng với sự hỗ trợ từ cộng đồng lập trình viên lớn. Ngoài ra, các nền tảng như Microsoft Learn và Stack Overflow cung cấp vô số bài viết, video hướng dẫn, và giải pháp cho các vấn đề mà bạn có thể gặp phải khi làm việc với ASP.NET.
* **Trực quan và dễ bảo trì**: Cách viết mã trong ASP.NET được đánh giá là rõ ràng, trực quan, giúp các nhà phát triển tập trung vào các nhiệm vụ chính. Đồng thời, mã nguồn dễ bảo trì và nâng cấp.
* **Công cụ Razor Pages Template**: Giúp việc tích hợp giữa HTML và C# trở nên đơn giản, sáng sủa, và đảm bảo sự tách biệt rõ ràng giữa giao diện và logic nghiệp vụ.
* **Hỗ trợ MVC pattern**: ASP.NET sử dụng mô hình MVC (Model-View-Controller), đảm bảo rằng mã nguồn được tổ chức logic, dễ hiểu và dễ quản lý. Điều này làm cho việc phát triển và bảo trì các dự án web trở nên đơn giản và hiệu quả hơn.
* **Tích hợp thư viện mạnh mẽ**: ASP.NET hỗ trợ việc tích hợp nhiều thư viện và gói phần mềm thông qua NuGet. Điều này mang lại những tính năng tuyệt vời và khả năng mở rộng không giới hạn.
* **Liên tục đổi mới và phát triển**: Với sự hỗ trợ mạnh mẽ từ Microsoft, ASP.NET luôn được cải tiến với các tính năng mới, đáp ứng nhu cầu của các nhà phát triển hiện đại.

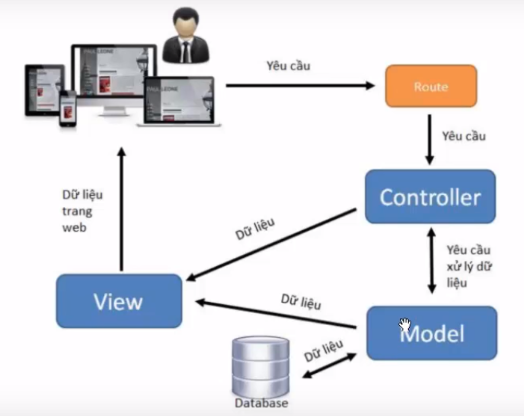
## 2.2 Mô hình MVC

Mô hình MVC (model, view, controller) là mô hình chuẩn cho ứng dụng web được sử dụng nhiều nhất ngày nay. Mô hình MVC được sử dụng lần đầu tiên trong Smalltalk, sau đó được sử dụng phổ biến trong ngôn ngữ lập trình Java. Hiện nay, đã có hơn hàng chục PHP framework dựa trên mô hình này. Mô hình MVC là viết tắt của 3 chữ Model, View, Controller. Mô hình này tách một ứng dụng web ra làm 3 thành phần đảm nhiệm chức năng tách biệt, thuận tiện cho việc xử lý và bảo trì.

**Model:** có nhiệm vụ thao tác với cơ sở dữ liệu, nghĩa là nó sẽ chứa tất cả các hàm, các phương thức truy vấn trực tiếp với dữ liệu và controller sẽ thông qua các hàm, phương thức đó để lấy dữ liệu rồi gửi qua View.

**View:** có nhiệm vụ tiếp nhận dữ liệu từ controller và hiển thị nội dung sang các đoạn mã HTML, có thể hiểu nôm na đây người ta còn gọi là thành phần giao diện.

**Controller:** đóng vài trò trung gian giữa Model và View. Nó có nhiệm vụ tiếp nhận yêu cầu từ người dùng sau đó xử lý request, load model tương ứng và gửi data qua view tương ứng rồi trả kết quả về cho người dùng.



Hình 2: Mô hình MVC trong Laravel

* Người dùng sử dụng trình duyệt web (Browser) yêu cầu truy cập vào địa chỉ website.
* **Route** sẽ phân tích xem đường dẫn (link) sẽ truy cập đến Controller nào.
* **Controller** sẽ truy cập vào Model để lấy dữ liệu từ cơ sở dữ liệu (Database), sau đó trả về dữ liệu cho Controller.
* **Controller** sẽ trả dữ liệu ra View hoặc trả dữ liệu dạng JSON cho người dùng.
* **View** sẽ xuất dữ liệu ra màn hình cho người dùng xem.

**Ưu điểm:**

* Hệ thống phân ra từng phần nên dễ dáng phát triển.
* Chia thành nhiều modun nhỏ nên nhiều người có thể làm chung dự án.
* Vấn đề bảo trì cũng tương đối tốt, dễ nâng cấp.
* Dễ dàng rà soát lỗi trong quá trình xây dựng.

**Nhược điểm:**

* Hệ thống sẽ chạy chậm hơn PHP thuần, tuy nhiên nó ko phải là vấn đề.
* Xây dựng cầu kì và mất thời gian để xây dựng thư viện, cấu trúc.

## 2.3 jQuery, boostrap

### **2.3.1 jQuery**

**jQuery** là một thư viện JavaScript nhỏ gọn, chạy nhanh. jQuery giải quyết các vấn đề tương tác với HTML, bắt sự kiện, hiệu ứng động… trở nên rất đơn giản.

Các tính năng chính jQuery cung cấp gồm:

* Tương tác với HTML/DOM
* Tương tác với CSS
* Bắt và xử lý sự kiện HTML
* Các hiệu ứng và chuyển động trong HTML
* AJAX (Asynchronous JavaScript and XML)
* JSON parsing
* Các tiện ích xây dựng sẵn

**Ưu điểm của jquery**

* Dễ sử dụng: Đây là lợi thế chính khi sử dụng jquery, nó dễ dàng hơn so với nhiều thư viện javascript chuẩn khác bởi cú pháp đơn giản và ta chỉ phải viết ít dòng lệnh để tạo ra các chức năng tương tự. Chỉ với 10 dòng lệnh JQuery ta có thể thay thế cả 20 dòng lệnh DOM javaScript, tiết kiệm thời gian của người lập trình.
* Là một thư viện lớn của javascript: Thực thi được nhiều chức năng hơn so với các thư viện jascript khác
* Cộng đồng mã nguồn mở mạnh mẽ (một số plugin jquery có sẵn.
* Có nhiều tài liệu và hướng dẫn chi tiết.
* Hỗ trợ ajax: JQuery cho phép ta phát triển các template Ajax một cách dễ dàng. Ajax cho phép một giao diện kiểu dáng đẹp trên website, các chức năng có thể được thực hiện trên các trang mà không đòi hỏi toàn bộ trang được reload lại.

**Nhược điểm của jquery**

Dù ưu điểm của jquery rất nhiều nhưng cũng tồn tại một vài nhược điểm phải lưu ý.

* Làm client trở nên chậm chạp: Client không những phải chỉ mình hiển thị nữa mà còn phải xử lý nhiều chức năng được tạo thành từ jquery. Nếu lạm dụng quá nhiều jquery sẽ làm cho client trở nên chậm chạp, đặc biệt những client yếu. Chính vì vậy mà lập trình viên phải dùng thêm cache.
* Chức năng có thể không có: JQuery đã có rất nhiều chức năng, tùy thuộc vào yêu cầu trên trang web của bạn. Nhưng nhiều chức năng vẫn chưa được phát triển, do đó bạn vẫn phải sử dụng javascript thuần để xây dựng chức năng này.

### **2.3.2 Boostrap**

**Bootstrap** là 1 framework HTML, CSS, và JavaScript cho phép người dùng dễ dàng thiết kế website theo 1 chuẩn nhất định, tạo các website thân thiện với các thiết bị cầm tay như mobile, ipad, tablet,...

Bootstrap bao gồm những cái cơ bản có sẵn như: typography, forms, buttons, tables, navigation, modals, image carousels và nhiều thứ khác. Trong bootstrap có thêm nhiều Component, Javascript hỗ trợ cho việc thiết kế reponsive của bạn dễ dàng, thuận tiện và nhanh chóng hơn.

**Tại sao chúng ta nên sử dụng Bootstrap?**

* Bootstrap là một trong những framework được sử dụng nhiều nhất trên thế giới để xây dựng nên một website. Bootstrap đã xây dựng nên 1 chuẩn riêng và rất được người dùng ưa chuộng. Chính vì thế, chúng ta hay nghe tới một cụm từ rất thông dụng "Thiết kế theo chuẩn Bootstrap".
* Từ cái "chuẩn mực" này, chúng ta có thể thấy rõ được những điểm thuận lợi khi sử dụng Bootstrap.
* Rất dễ để sử dụng: Nó đơn giản vì nó được base trên HTML, CSS và Javascript chỉ cẩn có kiến thức cơ bản về 3 cái đó là có thể sử dụng bootstrap tốt.
* Responsive: Bootstrap xây dựng sẵn reponsive css trên các thiết bị Iphones, tablets, và desktops. Tính năng này khiến cho người dùng tiết kiệm được rất nhiều thời gian trong việc tạo ra một website thân thiện với các thiết bị điện tử, thiết bị cầm tay.
* Tương thích với trình duyệt: Nó tương thích với tất cả các trình duyệt (Chrome, Firefox, Internet Explorer, Safari, and Opera). Tuy nhiên, với IE browser, Bootstrap chỉ hỗ trợ từ IE9 trở lên. Điều này vô cùng dễ hiểu vì IE8 không support HTML5 và CSS3.

### **2.3.3 Hệ quản trị cơ sở dữ liệu MySQL**

**MySQL** là hệ quản trị cơ sở dữ liệu mã nguồn mở miễn phí, được tích hợp sử dụng chung với apache, PHP. MySQL quản lý dữ liệu thông qua các cơ sở dữ liệu, mỗi cơ sở dữ liệu có thể có nhiều bảng quan hệ chứa dữ liệu. MySQL có cơ chế phân quyền người sử dụng riêng, mỗi người dùng có thể được quản lý một hoặc nhiều cơ sở dữ liệu khác nhau, mỗi người dùng có một tên truy cập (user name) và mật khẩu tương ứng để truy xuất đến cơ sở dữ liệu. Khi ta truy vấn tới cơ sở dữ liệu MySQL, ta phải cung cấp tên truy cập và mật khẩu của tài khoản có quyền sử dụng cơ sở dữ liệu đó. Nếu không, chúng ta sẽ không làm được gì cả.

**Ưu điểm của MySQL**

* Dễ dàng sử dụng: MySQL có thể dễ dàng cài đặt. Với các công cụ bên thứ 3 làm cho nó càng dễ đơn giản hơn để có thể sử dụng.
* Giàu tính năng: MySQL hỗ trợ rất nhiều chức năng SQL được mong chờ từ một hệ quản trị cơ sở dữ liệu quan hệ cả trực tiếp lẫn gián tiếp.
* Bảo mật: Có rất nhiều tính năng bảo mật, một số ở cấp cao đều được xây dựng trong MySQL.
* Khả năng mở rộng và mạnh mẽ: MySQL có thể xử lý rất nhiều dữ liệu và hơn thế nữa nó có thể được mở rộng nếu cần thiết.
* Nhanh: Việc đưa ra một số tiêu chuẩn cho phép MySQL để làm việc rất hiệu quả và tiết kiệm chi phí, do đó nó làm tăng tốc độ thực thi.

**Nhược điểm của MySQL**

* Giới hạn: Theo thiết kế, MySQL không có ý định làm tất cả và nó đi kèm với các hạn chế về chức năng mà một vào ứng dụng có thể cần.
* Độ tin cậy: Cách các chức năng cụ thể được xử lý với MySQL (ví dụ tài liệu tham khảo, các giao dịch, kiểm toán,...) làm cho nó kém tin cậy hơn so với một số hệ quản trị cơ sở dữ liệu quan hệ khác.
* Sự phát triển chậm: Mặc dù MySQL vẫn là một sản phẩm công nghệ mã nguồn mở phổ biến nhưng có những phàn nàn liên quan đến quá trình phát triển kể từ khi nó được mua lại.

# CHƯƠNG III. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## 1 Xác định yêu cầu hệ thống

* **Quản lý thông tin bệnh nhân**: Phần mềm được thiết kế để quản lý thông tin chi tiết của từng bệnh nhân, bao gồm họ tên, ngày sinh, giới tính, địa chỉ, số điện thoại, tiền sử bệnh lý, và hồ sơ khám chữa bệnh.
* **Lập lịch khám chữa bệnh**:
  + Phần mềm cho phép nhân viên y tế thiết lập lịch khám chữa bệnh cho từng bệnh nhân.
  + Hiển thị thông tin chi tiết về lịch trình, bác sĩ phụ trách, phòng khám, và các chỉ định liên quan.
  + Gửi thông báo nhắc nhở lịch khám qua email hoặc tin nhắn SMS cho bệnh nhân.
* **Tìm kiếm và theo dõi hồ sơ bệnh nhân**:
  + Bệnh nhân có thể tra cứu lịch sử khám chữa bệnh của mình trên hệ thống.
  + Nhân viên y tế có thể tìm kiếm bệnh nhân dựa trên các tiêu chí như họ tên, mã bệnh nhân, số điện thoại, hoặc tình trạng bệnh lý cụ thể.
* **Quản lý đơn thuốc và kết quả xét nghiệm**:
  + Lưu trữ thông tin đơn thuốc và kết quả xét nghiệm của từng bệnh nhân.
  + Cho phép bác sĩ chỉnh sửa, cập nhật thông tin đơn thuốc hoặc tải lên các kết quả xét nghiệm mới.
* **Phân quyền quản trị**:
  + Phân quyền cho các vai trò quản trị viên, bác sĩ, y tá, và nhân viên lễ tân.
  + Quản trị viên có thể thực hiện các công việc sau:
    - Tìm kiếm và xác nhận thông tin bệnh nhân mới đăng ký.
    - Xác nhận và chỉnh sửa thông tin hồ sơ bệnh nhân.
    - Thống kê tình hình khám chữa bệnh của từng bác sĩ và khoa phòng.
    - Quản lý thông tin, hình ảnh bệnh viện, phòng khám, và các dịch vụ y tế.
* **Thống kê và báo cáo**:
  + Hỗ trợ thống kê dữ liệu theo các tiêu chí như số lượng bệnh nhân theo thời gian, loại bệnh, phòng khám, hoặc bác sĩ phụ trách.
  + Tạo báo cáo chi tiết để phục vụ quản lý và cải tiến chất lượng dịch vụ.
* **Tích hợp và theo dõi liên hệ**:
  + Theo dõi tình trạng phản hồi của bệnh nhân thông qua hệ thống liên hệ trực tuyến.
  + Đáp ứng nhanh chóng các yêu cầu và thắc mắc của bệnh nhân.
* **Quản lý dữ liệu lưu trữ**:
  + Thêm, xóa, sửa thông tin về bệnh nhân, hồ sơ y tế, và thông tin khám chữa bệnh.
  + Lưu trữ dữ liệu an toàn và bảo mật, tuân thủ các quy định về bảo vệ thông tin cá nhân.

## 2 Thiết kế ứng dụng web phía client

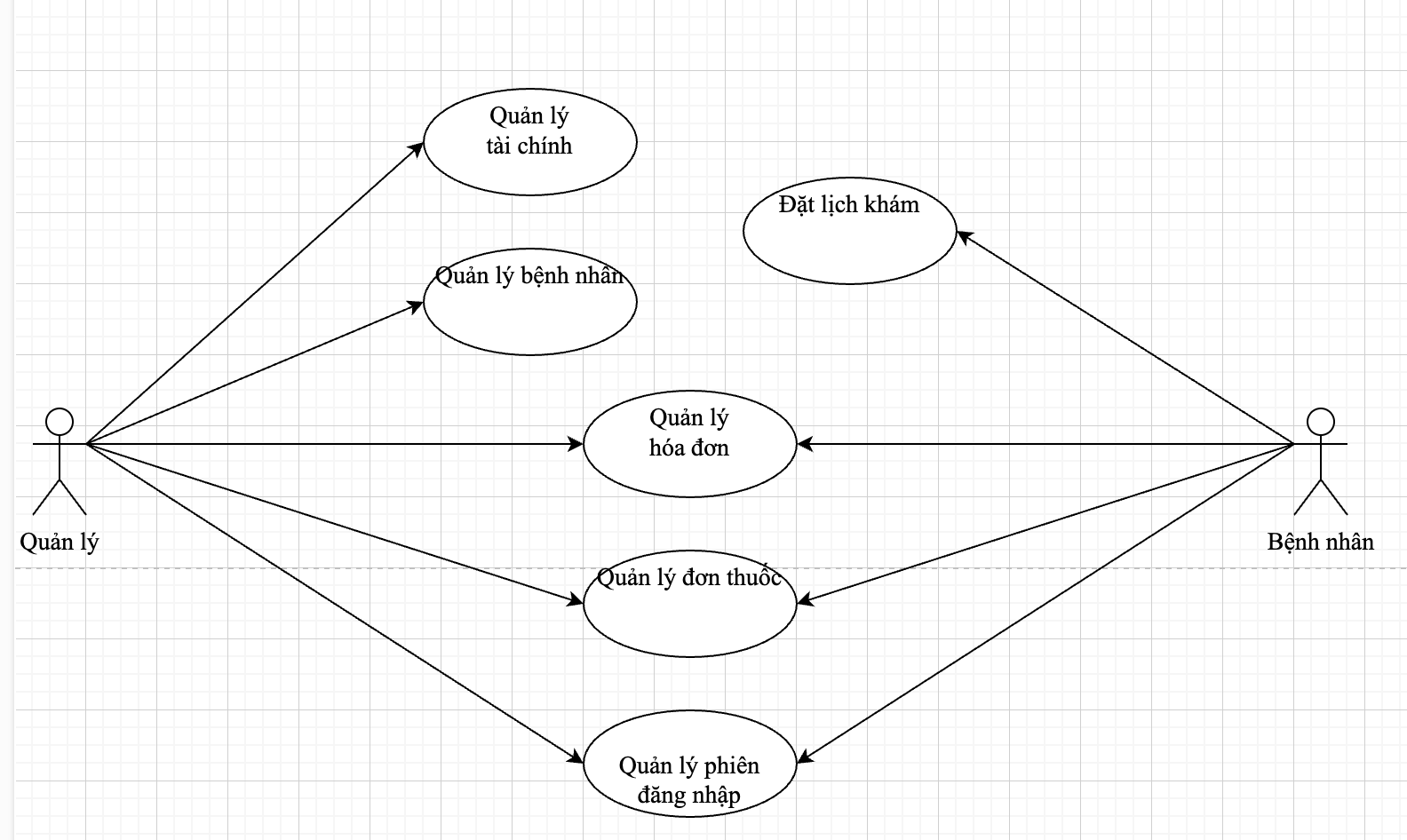
### **2.1 Biểu đồ Use Case tổng quát**

#### **1. Actor Nhân viên (chưa có tài khoản)**

* **Chức năng:**
  + Tìm kiếm thông tin bệnh nhân:
    - Theo mã bệnh nhân.
    - Theo họ tên.
    - Theo ngày nhập viện hoặc xuất viện.
  + Xem danh sách bệnh nhân hiện tại trong bệnh viện.
  + Đăng ký tài khoản để trở thành nhân viên chính thức và sử dụng các chức năng nâng cao.

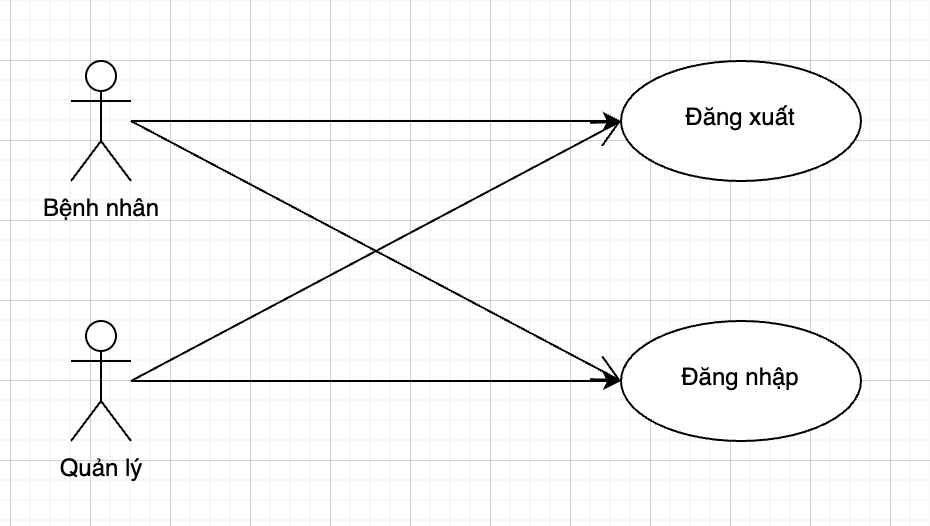
#### **2. Actor Nhân viên chính thức (đã có tài khoản và đăng nhập vào hệ thống)**

* **Chức năng nâng cao:**
  + Có đầy đủ chức năng của nhân viên chưa có tài khoản.
  + **Quản lý bệnh nhân:**
    - Thêm mới bệnh nhân với thông tin cá nhân, hồ sơ y tế và lịch sử khám chữa bệnh.
    - Chỉnh sửa thông tin bệnh nhân như địa chỉ, liên hệ, bệnh án, hoặc điều chỉnh lịch khám.
  + **Quản lý hồ sơ khám chữa bệnh:**
    - Cập nhật kết quả khám và dịch vụ y tế đã sử dụng.
    - Lập hóa đơn khám bệnh và các dịch vụ bổ sung.
  + **Quản lý nhập viện:**
    - Tạo hồ sơ nhập viện cho bệnh nhân cần điều trị nội trú.
    - Theo dõi tình trạng bệnh nhân lưu trú và cập nhật thời gian xuất viện.
  + **Thống kê:**
    - Báo cáo số lượng bệnh nhân khám bệnh theo ngày, tháng, hoặc năm.
    - Thống kê bệnh nhân nội trú và xuất viện.

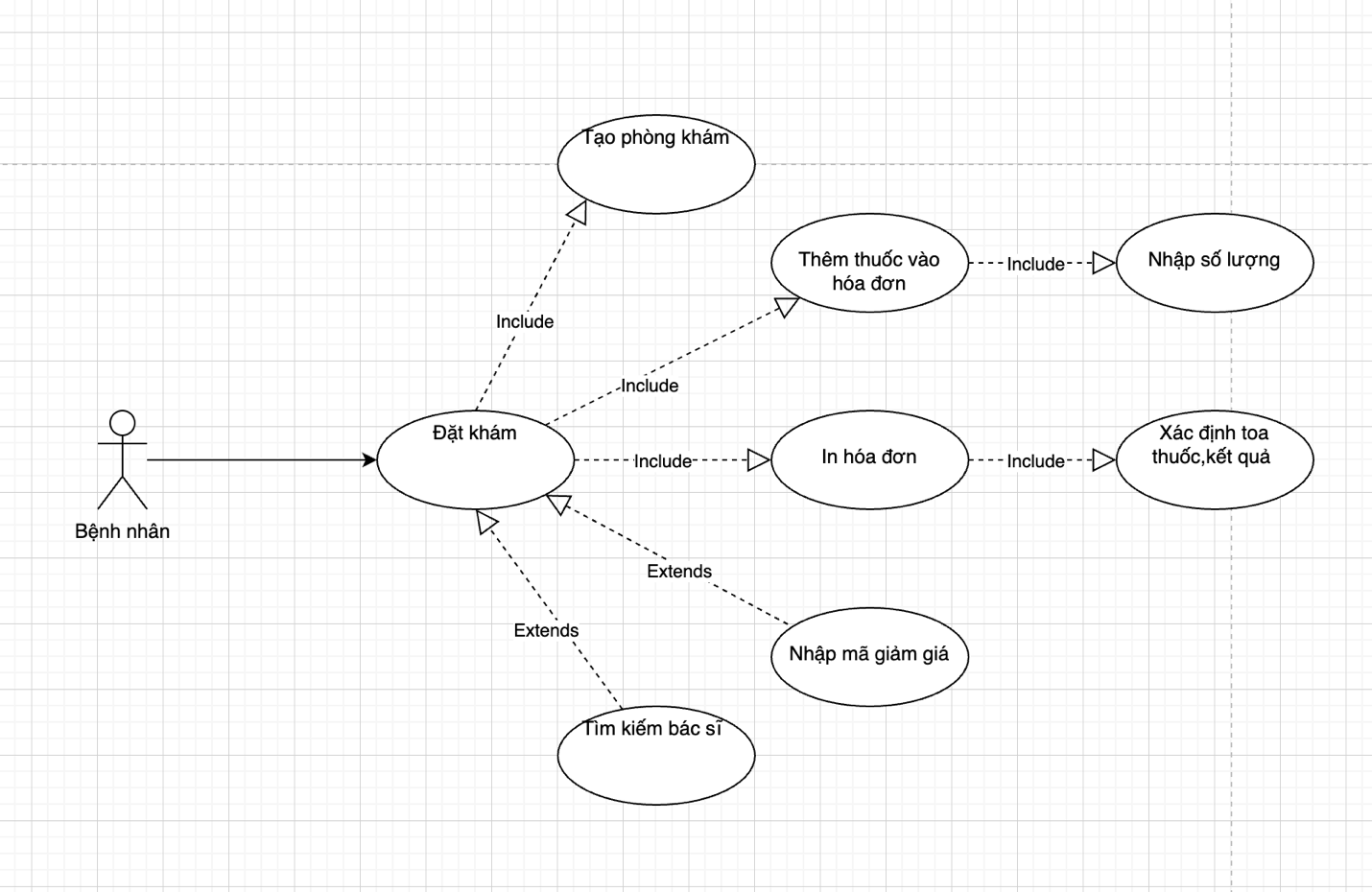


Hình 3: Biểu đồ Use Case tổng quát web client

1. ***Phân rã Use Case đăng nhập***

******

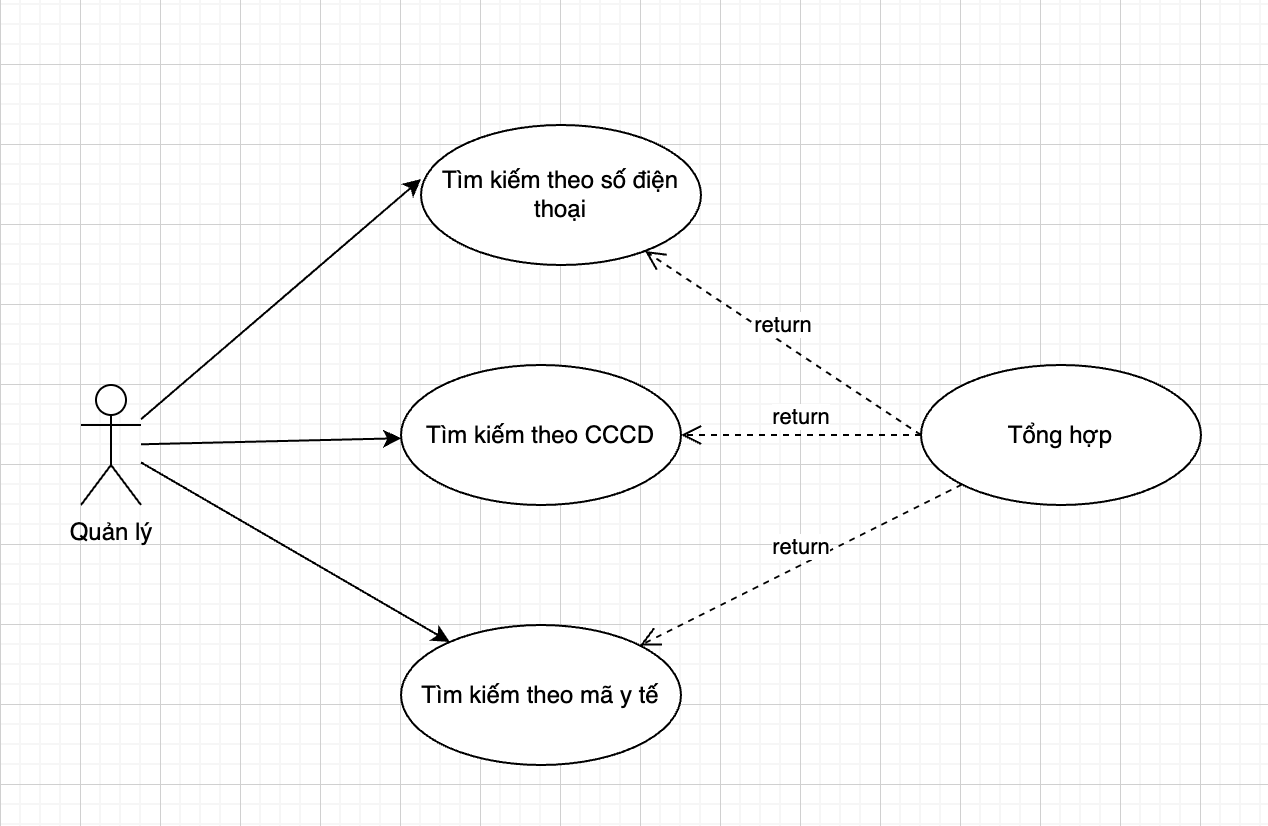
1. ***Phân rã Use Case đặt khám***

******

Hình 4: Phân rã Use Case đặt khám

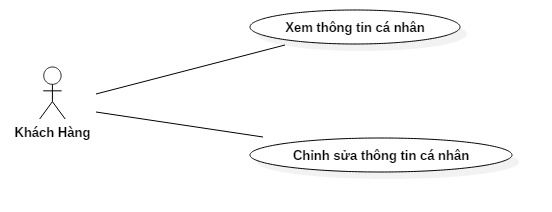
Hình 4 mô tả chức năng đặt khám

1. ***Phân rã Use Case tìm kiếm bệnh nâhn***



Hình 4: Phân rã Use Case tìm kiếm bệnh nhân

1. ***Phân rã Use Case quản lý thông tin cá nhân***

****

Hình 5: Phân rã UseCase quản lý thông tin cá nhân

### **2.2 Đặc tả một số chức năng chính**

**2.2.1 Đăng nhập**

#### Use case đăng nhập :

|  |  |
| --- | --- |
| Tên UC | Đăng nhập |
| Mô tả | User đăng nhập có thể thực hiện các chức năng đặt phòng khám ,chỉn sửa thông tin,quản lý ngày khám ở bệnh viện |
| Tác nhân | Khách |
| Sự kiện kích hoạt | Khách chọn “Đăng nhập” |
| Điều kiện tiên quyết | Không |
| Phương thức cơ bản | |  |  | | --- | --- | | Khách | Hệ thống | | 1. Chọn “Đăng nhập” |  | |  | 2. Hiển thị form đăng nhập | | 3. Nhập username & password và đăng nhập |  | |  | 4. Xác thực người dùng | |  | 5. Cho phép đăng nhập nếu hợp lệ | |
| Phương thức thay thế | Đăng nhập bằng CCCD,SĐT |
| Kết quả | Khách đăng nhập được vào hệ thống |
| Ngoại lệ | Nếu username & pass không hợp lệ, yêu cầu người dùng nhập lại |
| Tần suất sử dụng | Trung bình |

**2.2.2Bệnh nhân**

***a) Use Case: Tìm kiếm bệnh nhân***

|  |  |
| --- | --- |
| Tên Use Case | Tìm kiếm bệnh nhân |
| Tác nhân | Quản lý |
| Mục đích | Cho phép bệnh nhân tìm kiếm bệnh nhân |
| Điều kiện đầu vào | Quản lý truy cập hệ thống |
| Kịch bản chính | + Người sử dụng nhập thông tin cần thiết vào form tìm kiếm.  + Nhấn nút tìm kiếm.  + Hệ thống hiển thị kết quả.  . |

Bảng 1: Đặc tả Use Case Tìm kiếm bệnh nhân

***b)* *Use Case:***

|  |  |
| --- | --- |
| Tên Use Case | Xem chi tiết loại bệnh |
| Tác nhân | Bệnh nhân |
| Mục đích | Cho phép bệnh nhân xem thông tin chi tiết về loại bệnh |
| Điều kiện đầu vào | Bệnh nhân đã tìm kiếm loại bệnh |
| Kịch bản chính | + Người dùng chọn loại bệnh cần xem.  + Form xem thông tin chi tiết bệnh hiển thị. Hệ thống hiển thị thông tin bệnh đã chọn  + Người dùng xem thông tin chi tiết về bệnh đã chọn |

Bảng 2: Đặc tả Use Case xem chi tiết bệnh



## *c) Use Case: Đặt khám*

|  |  |
| --- | --- |
| Tên Use Case | Đặt khám |
| Tác nhân | Bệnh nhân |
| Mục đích | Cho phép bệnh nhân đặt phòng khám và tiến hành thanh toán |
| Điều kiện đầu vào | Không |
| Kịch bản chính | + Bệnh nhân nhấn thanh toán để tạo đơn đặt.  + Form thanh toàn xuất hiện, hệ thống hiển thị đơn đặt và một số thông tin mặc định của phòng  + Bệnh nhân nhập tiếp các thông tin.  + Chọn thanh toán.  + Hệ thống kiếm tra và thông báo. |

Bảng 3: Đặc tả Use Case Đặt khám

**2.2.3 Bệnh nhân**

***a) Use Case: Tìm kiếm lịch khám đã đặt***

|  |  |
| --- | --- |
| Tên Use Case | Tìm kiếm lịch khám đã đặt |
| Tác nhân | Bệnh nhân |
| Mục đích | Cho phép bệnh nhân tìm kiếm lịch khám đã đặt |
| Điều kiện đầu vào | Bệnh nhân truy cập hệ thống |
| Kịch bản chính | + Người sử dụng nhập thông tin cần thiết vào form tìm kiếm.  + Nhấn nút tìm kiếm.  + Hệ thống hiển thị kết quả.  . |

Bảng 1: Đặc tả Use Case Tìm kiếm lịch khám

***b)* *Use Case: Nhập viên***

|  |  |
| --- | --- |
| Tên Use Case | Nhập viên |
| Tác nhân | Quản lý |
| Mục đích | Cho phép quản lý có thể nhập thông tin bệnh nhân nhập viên |
| Điều kiện đầu vào | Bệnh nhân đã yêu cầu làm hồ sơ nhập viên |
| Kịch bản chính | + Người dùng chọn thêm hồ sơ  + Form xem thông tin chi tiết hiển thị để nhập.  + Người dùng xem thông tin chi tiết về thông.tin đã nhập  + Hiển thị kết quả và đưa ra hoá đơn nhập viên |

Bảng 2: Đặc tả Use Case Nhập viên

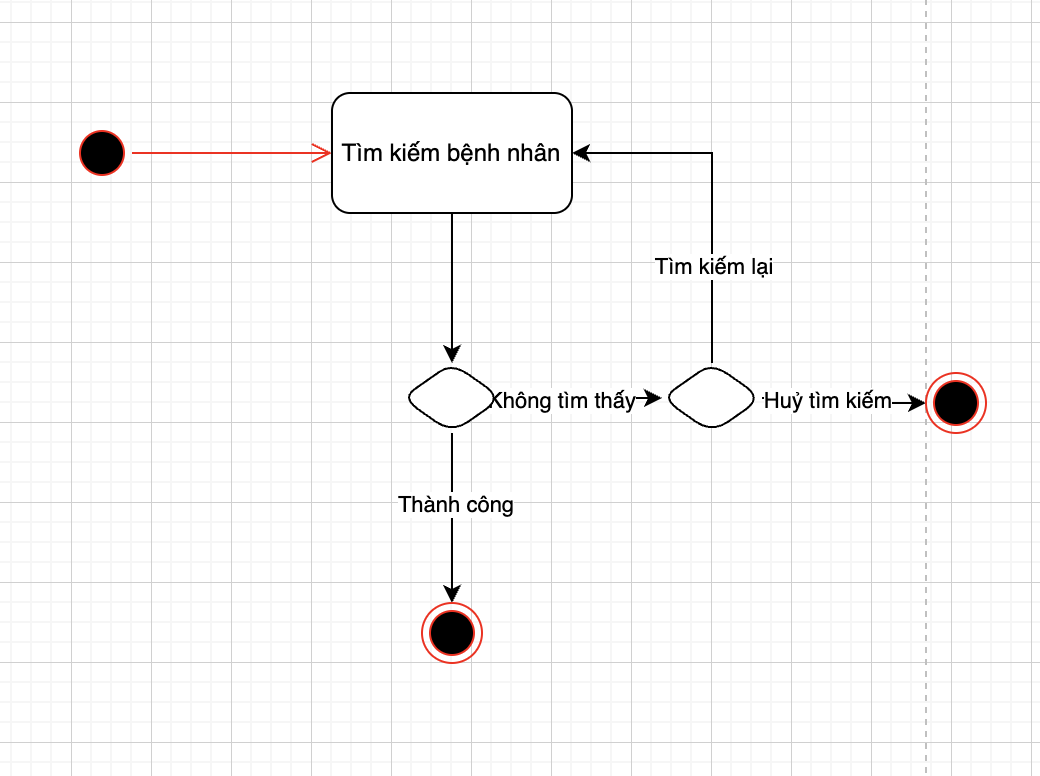


### **2.3 Mô hình hóa hoạt động của hệ thống**

**2.3.2 Quản lý**

1. ***Tìm kiếm thông tin bệnh nhân***

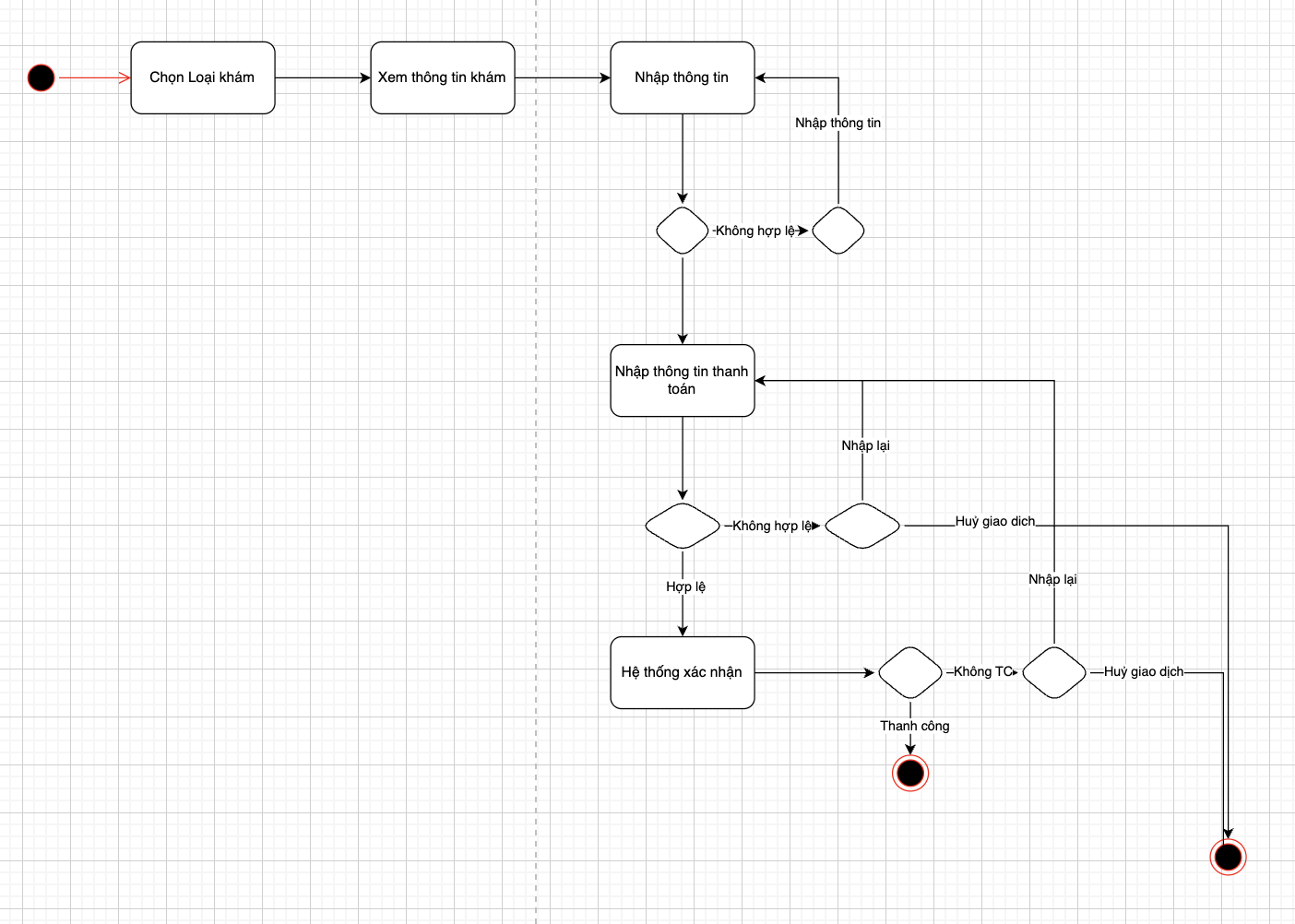
Hình 6 bên dưới là quy trình tìm kiếm Bệnh viên của bệnh nhân. Bệnh nhân sử dụng chức năng tìm kiếm Bệnh viên, hệ thống sẽ tìm kiếm trong cơ sở dữ liệu rồi trả về kết quả.



Hình 6: Biểu đồ hoạt động mô tả quá trình tìm kiếm

#### **Nhập viên**

Hình 7 là quy trình Nhập viên của bệnh nhân. Bệnh nhân chọn phòng mình muốn đặt, sau đó tiến hành nhập thông tin cho chức năng Nhập viên và tiến hành thanh toán.

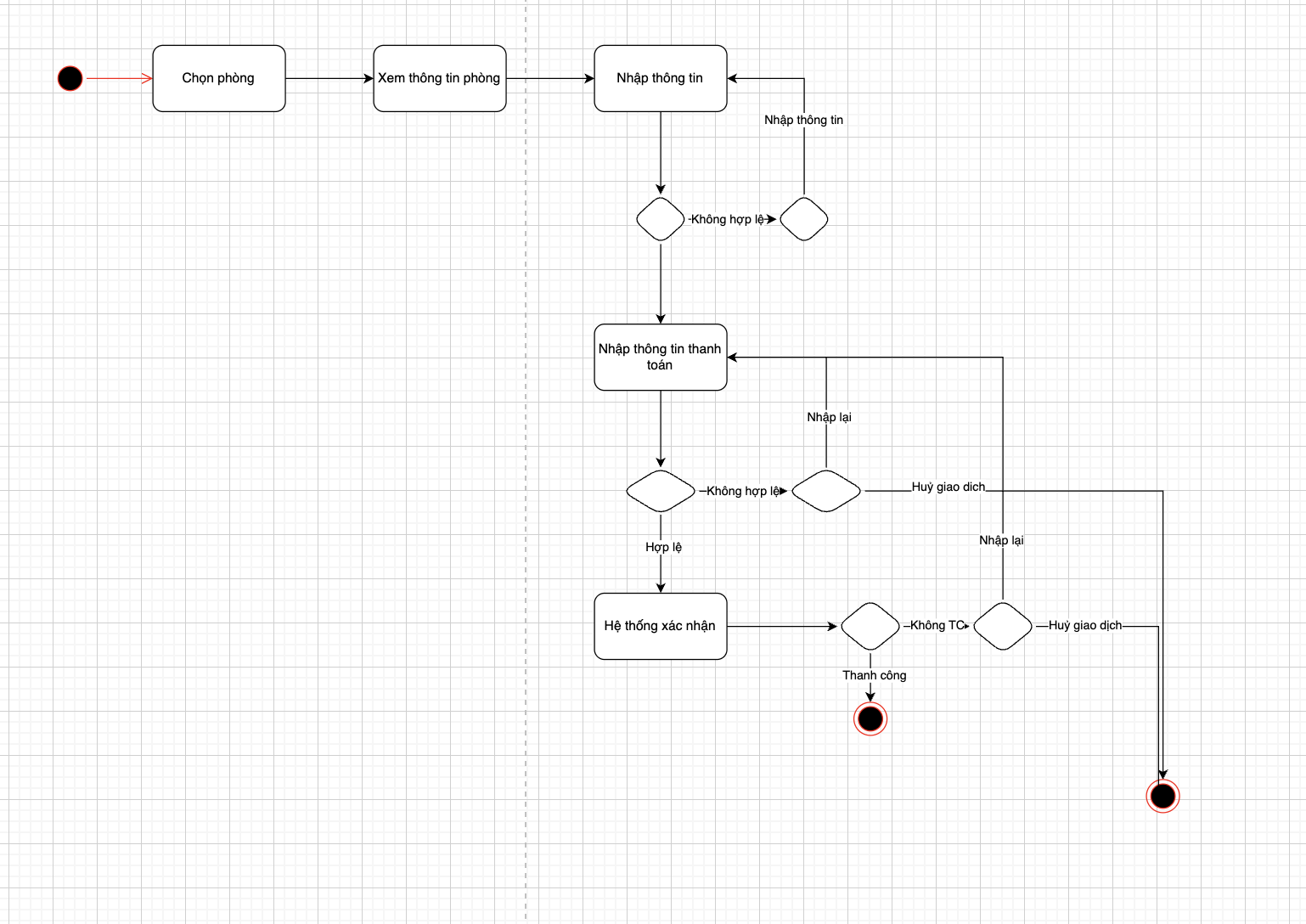


Hình 7: Biểu đồ hoạt động mô tả quá trình đặt phòng khám

**2.3.2 Nhập viên**

#### **Nhập viện**

Hình 7 là quy trình nhập viện của bệnh nhân. Bệnh nhân chọn loại phòng mình muốn đặt, sau đó tiến hành nhập thông tin cho chức năng đặt phòng và tiến hành thanh toán.



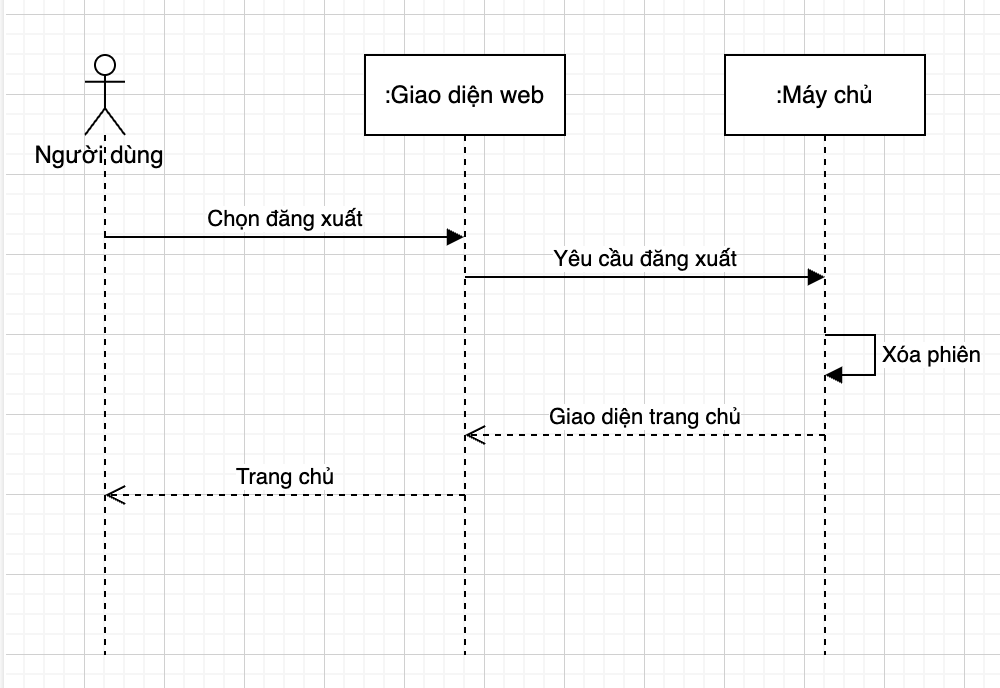
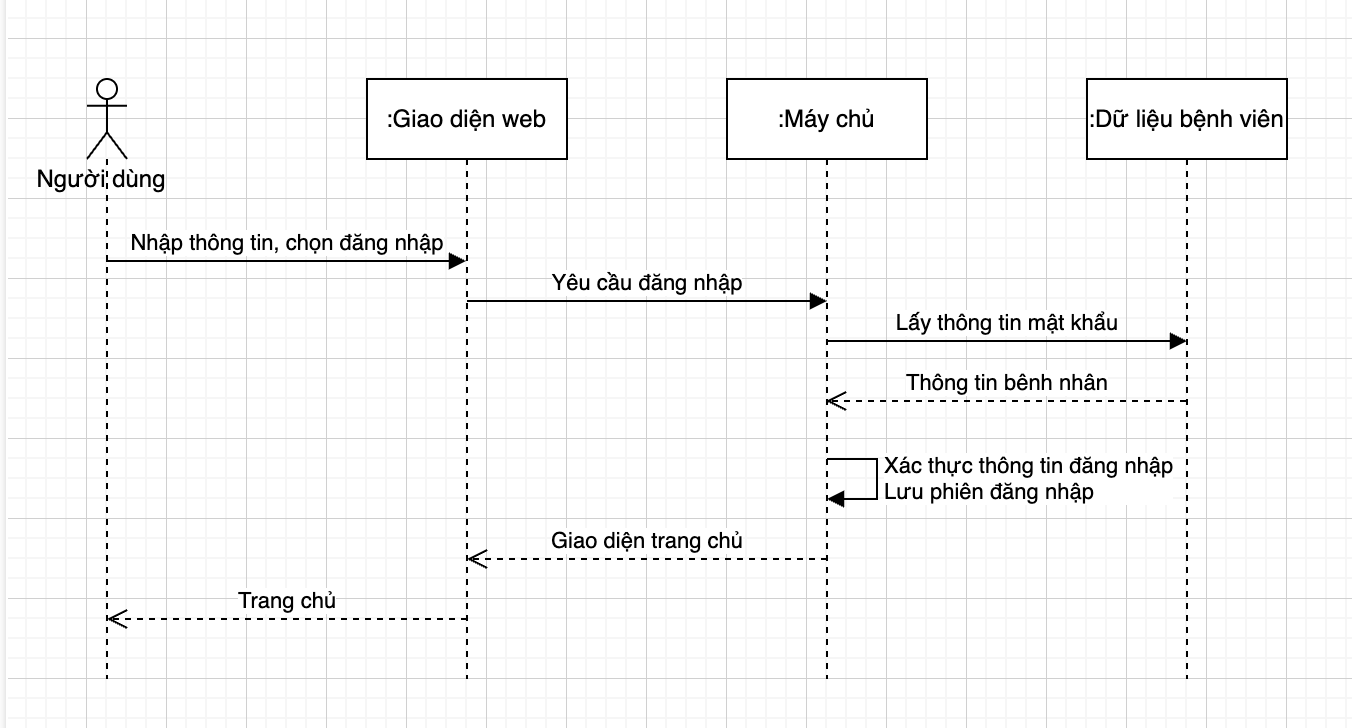
Hình 7: Biểu đồ hoạt động mô tả quá trình nhập viên

### **2.4 Mô hình hóa sự tương tác**

**2.4.1 Đăng nhập**

**a) Biểu đồ trình tự cho chức năng đăng nhập**

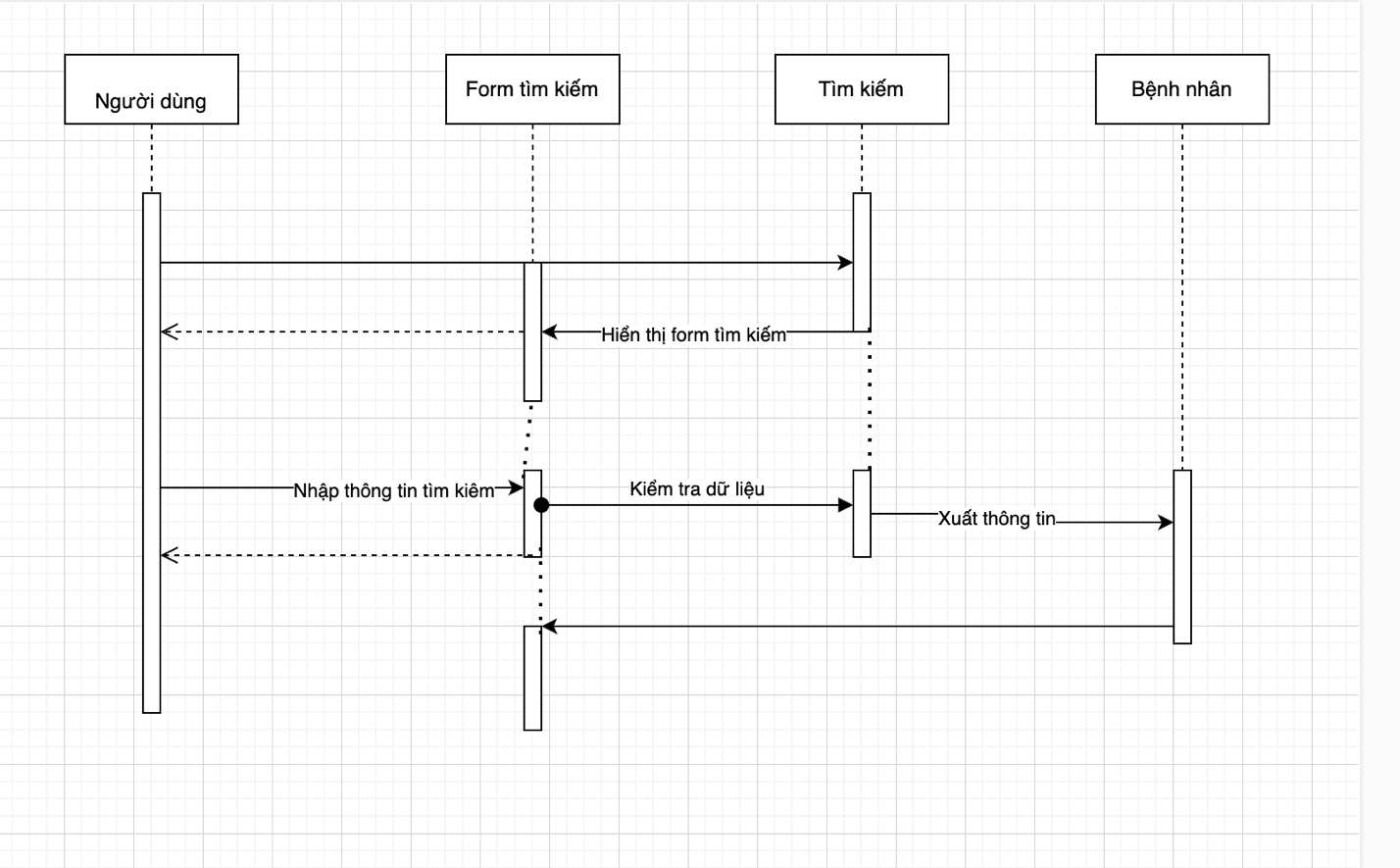
Hình 8 là biểu đồ trình tự đăng nhập. Bệnh nhân yêu cầu đăng nhập, hệ thống hiển thị form đăng nhập, bệnh nhân nhập account và password, hệ thống kiểm tra dữ liệu rồi trả về cho bệnh nhân message thông báo lỗi khi account mật không đúng còn di chuyển về trang chủ web.

**<**

**2.4.2 Bệnh nhân**

**a) Biểu đồ trình tự cho chức năng tìm kiếm**

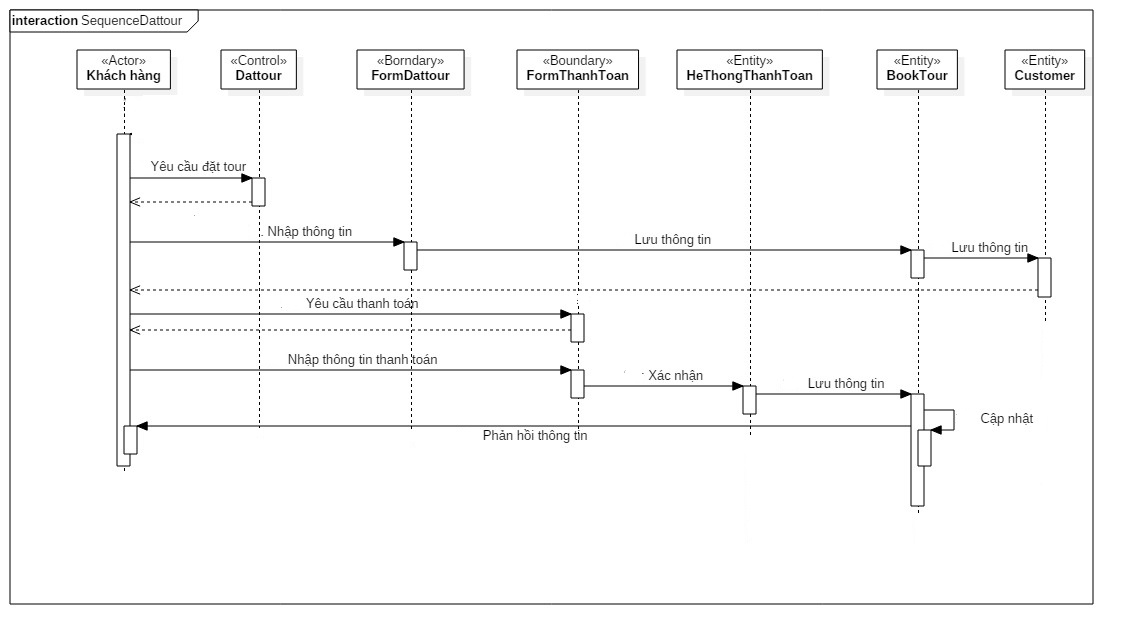
Hình 8 là biểu đồ trình tự tìm kiếm. Bệnh nhân yêu cầu tìm kiếm, hệ thống hiển thị form tìm kiếm, bệnh nhân nhập từ khóa tìm kiếm, hệ thống kiểm tra dữ liệu rồi trả về cho bệnh nhân trên màn hình hiển thị.



Hình 8: Biểu đồ trình tự tìm kiếm

**b) Biểu đồ trình tự cho chức năng đặt khám**

Hình 9 là biểu đồ trình tự cho chức năng đặt khám.

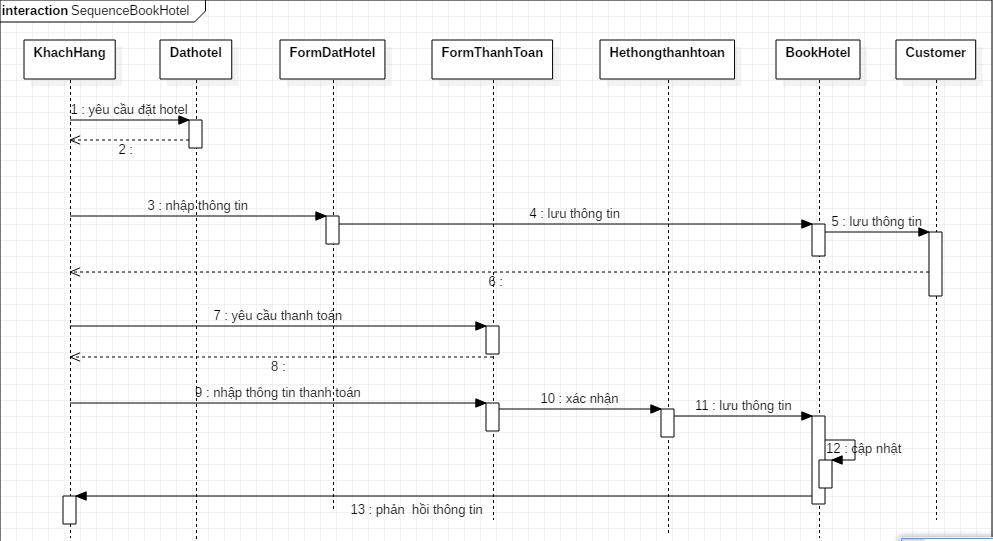


Hình 9: Biểu đồ trình tự chức năng đặt khám

**2.4.3 Quản lý**

**b) Biểu đồ trình tự cho chức năng Nhập viện**

Hình 9 là biểu đồ trình tự cho chức năng nhập viện



Hình 9: Biểu đồ trình tự chức năng Nhập viên

### **3.2 Đặc tả một số chức năng chính**

***a) Use Case: Đăng nhập***

|  |  |
| --- | --- |
| Tên Use Case | Đăng nhập |
| Tác nhân | Người quản trị (Quản lý chung, kế toán, nhân viên) |
| Mục đích | Cho phép người quản trị đăng nhập vào hệ thống để quản lý cơ sở dữ liệu |
| Điều kiện đầu vào | Người quản trị truy cập hệ thống |
| Kịch bản chính | + Người quản trị chọn chức năng đăng nhập.  + Hệ thống hiển thị form đăng nhập.  + Nhập tên, mật khẩu vào form đăng nhập  + Hệ thống kiểm tra tên, mật khẩu của người dùng. Nếu đúng thì đăng nhập vào hệ thống. Nếu sai thì yêu cầu đăng nhập lại. |

Bảng 4: Đặc tả Use Case đăng nhập

***b)* *Use Case: Quản lý nhân viên***

|  |  |
| --- | --- |
| Tên Use Case | Quản lý bệnh nhân |
| Tác nhân | Quản lý chung |
| Mục đích | Cho phép người quản lý chung thêm, sửa, xóa bệnh nhân |
| Điều kiện đầu vào | Người quản lý chung đăng nhập thành công, đang ở giao diện trang quản trị. |
| Kịch bản chính | + Thêm bệnh nhân  : Người quản lý chọn mục thêm bệnh nhân, sau đó nhập các trường thông tin, chọn chức năng thêm, hệ thống hiển thị kết quả.  + Xóa bệnh nhân: Người quản lý chọn bệnh nhân cần xóa, hệ thông hiển thị thông tin bệnh nhân, người quản lý chọn chức năng xóa, hệ thống thông báo kết quả.  + Cập nhật bệnh nhân: Người quản lý chọn nhân viên cần cập nhật, hệ thống hiển thị thông tin nhân viên, người quản lý cập nhật thông tin nhân viên, sau đó nhấn nút cập nhật, hệ thống hiển thị kết quả. |

Bảng 5: Đặc tả Use Case quản lý bệnh nhân



## *c) Use Case: Quản lý phòng bệnh*

|  |  |
| --- | --- |
| Tên Use Case | Quản lý phòng bệnh |
| Tác nhân | Quản lý |
| Mục đích | Cho phép nhân viên thêm, sửa, xóa phòng bệnh |
| Điều kiện đầu vào | Nhân viên đăng nhập thành công, đang ở giao diện trang quản lý. |
| Kịch bản chính | + Thêm phòng bệnh: Quản lý chọn mục thêm phòng bệnh, sau đó nhập các trường thông tin, chọn chức năng thêm, hệ thống hiển thị kết quả.  + Xóa phòng bệnh: Nhân viên chọn phòng cần xóa, hệ thống hiển thị thông tin nhân viên, nhân viên chọn chức năng xóa, hệ thống thông báo kết quả.  + Cập nhật phòng bệnh: Nhân viên chọn phòng bệnh cần cập nhật, hệ thống hiển thị thông tin phòng bệnh, nhân viên cập nhật thông tin, sau đó nhấn nút cập nhật, hệ thống hiển thị kết quả. |

Bảng 6: Đặc tả Use Case quản lý phòng bệnh

***d) Use Case: Quản lý đơn đặt khám của bệnh nhân***

|  |  |
| --- | --- |
| Tên Use Case | Quản lý đơn đặt khám |
| Tác nhân | Quản lý |
| Mục đích | Cho phép người dùng cập nhật, xóa đơn đặt của bệnh nhân |
| Điều kiện đầu vào | Nhân viên, kế toán đăng nhập thành công, đang ở giao diện trang quản lý. |
| Kịch bản chính | + Xóa đơn: Quản lý chọn đơn đặt cần xóa, hệ thống hiển thị thông tin, quản lý chọn chức năng xóa, hệ thống thông báo kết quả.  + Cập nhật đơn: Quản lý chọn đơn đặt cần cập nhật, hệ thống hiển thị thông tin, quản lý cập nhật thông tin, sau đó nhấn nút cập nhật, hệ thống hiển thị kết quả. |

Bảng 7: Đặc tả Use Case quản lý đơn đặt của bệnh nhân

***e) Use Case: Quản lý thanh toán***

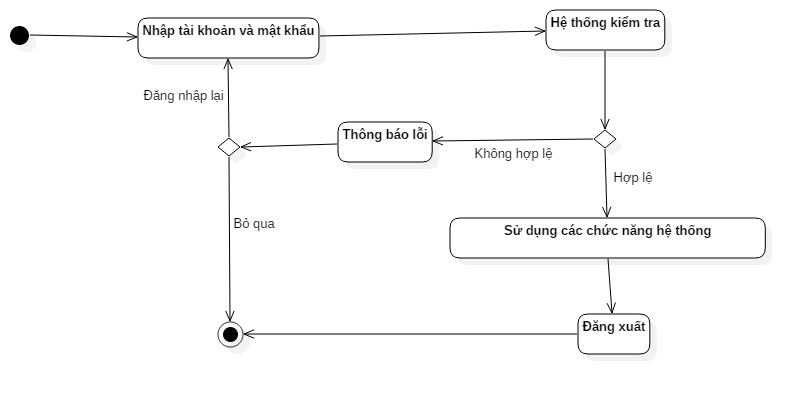
|  |  |
| --- | --- |
| Tên Use Case | Quản lý thanh toán |
| Tác nhân | Kế toán |
| Mục đích | Cho phép kế toán thanh toán các hóa đơn của bệnh nhân. |
| Điều kiện đầu vào | Nhân viên, kế toán đăng nhập thành công, có quyền thanh toán đơn đặt của bệnh nhân. |
| Kịch bản chính | + Kế toán yêu cầu thanh toán đơn đặt của bệnh nhân.  + Hệ thống hiển thị giao diện thanh toán đơn đặt của bệnh nhân.  + Kế toán nhập số tiền thanh toán, xác nhận.  + Hệ thống hiển thị thông báo. |

Bảng 8: Đặc tả Use Case quản lý thanh toán

### **3.3 Mô hình hóa hoạt động của hệ thống**

**a) Đăng nhập**

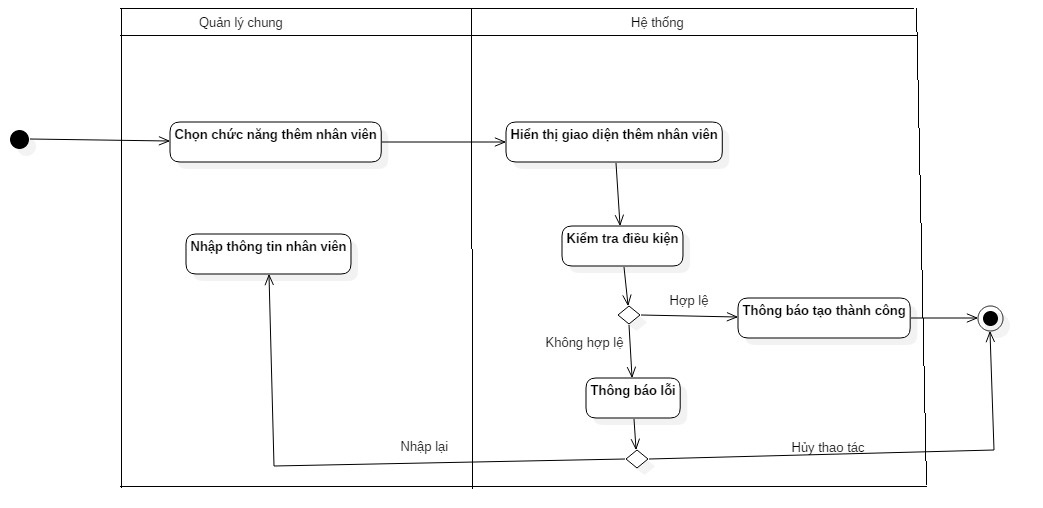
Hình 15 là biểu đồ hoạt động mô tả quy trình đăng nhập của nhân viên. Nhân viên truy cập website đăng nhập trang quản trị sau đó nhập tài khoản và mật khẩu. Hệ thống kiếm trả rồi trả kết quả lại cho người dùng.

****

Hình 15: Biểu đồ hoạt động mô tả quá trình đăng nhập

1. **Thêm nhân viên**

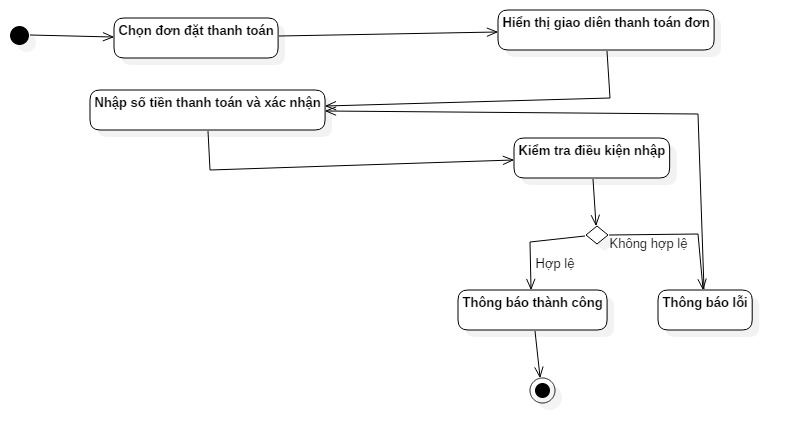
Hình 16 là biểu đồ hoạt động mô tả quy trình thêm nhân viên.



Hình 16: Biểu đồ hoạt động mô tả quá trình thêm nhân viên

1. **Thanh toán**

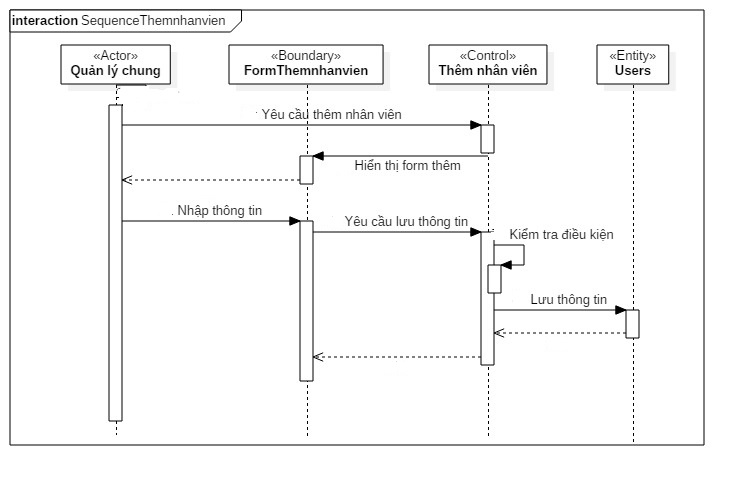
Hình 17 bên dưới là biểu đồ hoạt động mô tả quá trình nhân viên thanh toán đơn khám bệnh của bệnh nhân.

****

Hình 17: Biểu đồ hoạt động Thanh toán đơn đặt bệnh nhân

### **3.4 Mô hình hóa sự tương tác**

**a) Biểu đồ trình tự cho chức năng thêm nhân viên**

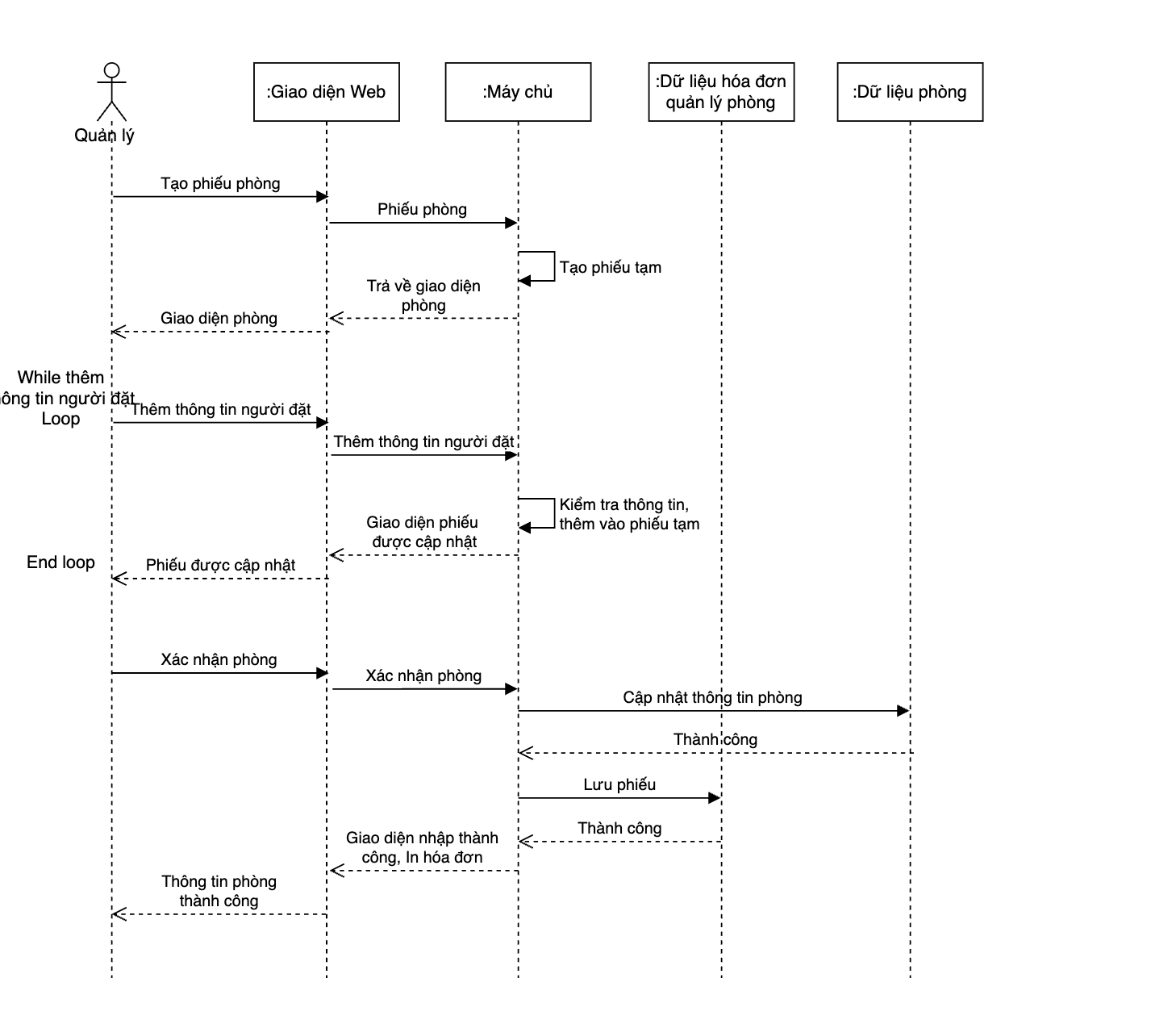
****

Hình 18: Biểu đồ trình tự thêm nhân viên

Hình 18 là biểu đồ trình tự cho chức năng thêm nhân viên của Actor quản lý chung.

**b) Biểu đồ trình tự cho chức năng thêm phòng**

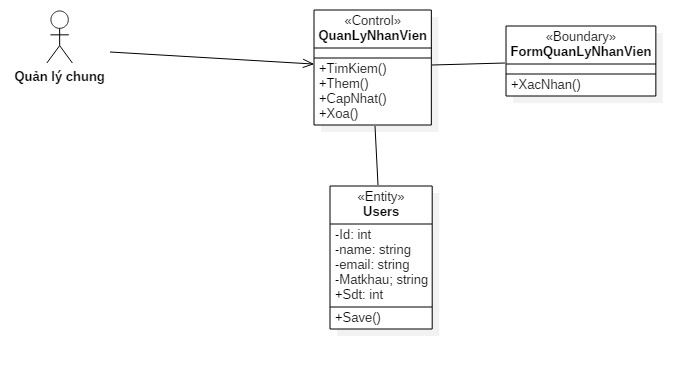
Hình 19 bên dưới là biểu đồ trình tự cho chức năng thêm phòng



Hình 19: Biểu đồ trình tự chức năng thêm

### **3.5 Thiết kế lớp (class)**

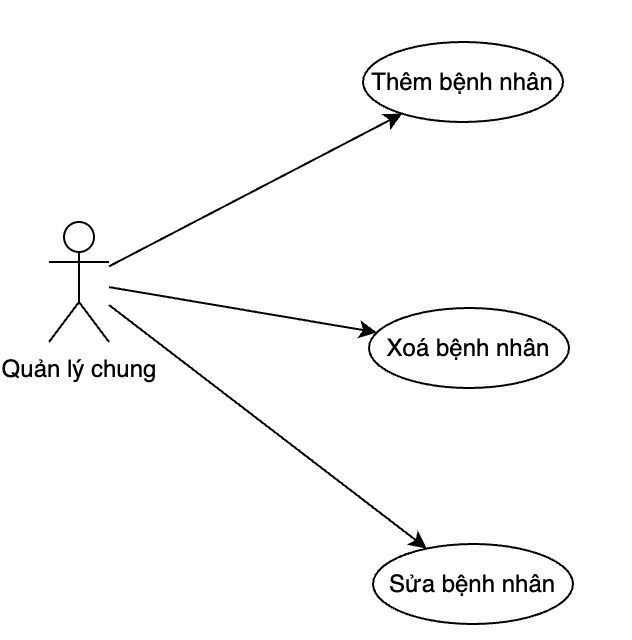
**a) Biểu đồ lớp cho chức năng quản lý nhân viên**

****

Hình 20: Biểu đồ lớp cho chức năng quản lý nhân viên

Hình 20 là biểu đồ lớp cho chức năng quản lý nhân viên của người quản lý chung. Biểu đồ này mô tả các phương thức và thuộc tính của các lớp mà người quản lý chung tương tác đến khi dùng chức năng quản lý nhân viên.

1. **Phân rã Use Case quản lý admin**



Hình 13: Phân rã Use Case quản lý nhân viên

Hình 13 mô tả chức năng Quản lý nhân viên. Quản lý chung có thể tạo, xem, chỉnh sửa và xóa các thông tin về nhân viên.

### **3.2 Đặc tả một số chức năng chính**

***a) Use Case: Đăng nhập***

|  |  |
| --- | --- |
| Tên Use Case | Đăng nhập |
| Tác nhân | Người quản trị (Quản lý chung, kế toán, nhân viên) |
| Mục đích | Cho phép người quản trị đăng nhập vào hệ thống để quản lý cơ sở dữ liệu |
| Điều kiện đầu vào | Người quản trị truy cập hệ thống |
| Kịch bản chính | + Người quản trị chọn chức năng đăng nhập.  + Hệ thống hiển thị form đăng nhập.  + Nhập tên, mật khẩu vào form đăng nhập  + Hệ thống kiểm tra tên, mật khẩu của người dùng. Nếu đúng thì đăng nhập vào hệ thống. Nếu sai thì yêu cầu đăng nhập lại. |

Bảng 4: Đặc tả Use Case đăng nhập

***b)* *Use Case: Quản lý admin***

|  |  |
| --- | --- |
| Tên Use Case | Quản lý ***admin*** |
| Tác nhân | Quản lý hệ thống |
| Mục đích | Cho phép người quản lý hệ thống thêm, sửa, xóa,phân quyền admin |
| Điều kiện đầu vào | Người quản lý hệ thống đăng nhập thành công, đang ở giao diện trang quản trị. |
| Kịch bản chính | + Thêm admin: Người quản lý chọn mục thêm admin, sau đó nhập các trường thông tin, chọn chức năng thêm, hệ thống hiển thị kết quả.  + Vô hiệu hóa nhân viên: Người quản lý chọn admin cần Vv hiệ, hệ thông hiển thị thông tin admin, người quản lý chọn chức năng vô hiệu hóa, hệ thống thông báo kết quả.  + Cập nhật admin: Người quản lý chọn nhân viên cần cập nhật, hệ thống hiển thị thông tin admin, người quản lý cập nhật thông tin admin, sau đó nhấn nút cập nhật, hệ thống hiển thị kết quả. |

Bảng 5: Đặc tả Use Case quản lý admin

## *c) Use Case: Quản lý phòng*

|  |  |
| --- | --- |
| Tên Use Case | Quản lý phòng |
| Tác nhân | Admin |
| Mục đích | Cho phép admin thêm, sửa, xóa phòng |
| Điều kiện đầu vào | Nhân viên đăng nhập thành công, đang ở giao diện trang quản lý. |
| Kịch bản chính | + Thêm phòng: Nhân viên chọn mục thêm phòng, sau đó nhập các trường thông tin, chọn chức năng thêm, hệ thống hiển thị kết quả.  + Xóa phòng: Nhân viên chọn phòng cần xóa, hệ thống hiển thị thông tin nhân viên, nhân viên chọn chức năng xóa, hệ thống thông báo kết quả.  + Cập nhật phòng: Nhân viên chọn phòng cần cập nhật, hệ thống hiển thị thông tin phòng, nhân viên cập nhật thông tin nhân viên, sau đó nhấn nút cập nhật, hệ thống hiển thị kết quả. |

Bảng 6: Đặc tả Use Case quản lý phòng

***d) Use Case: Quản lý đơn đặt của bệnh nhân***

|  |  |
| --- | --- |
| Tên Use Case | Quản lý đơn đặt |
| Tác nhân | Bệnh nhân |
| Mục đích | Cho phép người dùng cập nhật, xóa đơn đặt của bệnh nhân |
| Điều kiện đầu vào | Nhân viên, kế toán đăng nhập thành công, đang ở giao diện trang quản lý. |
| Kịch bản chính | + Xóa đơn: Nhân viên chọn đơn đặt cần xóa, hệ thống hiển thị thông tin, nhân viên chọn chức năng xóa, hệ thống thông báo kết quả.  + Cập nhật đơn: Nhân viên chọn đơn đặt cần cập nhật, hệ thống hiển thị thông tin, nhân viên cập nhật thông tin, sau đó nhấn nút cập nhật, hệ thống hiển thị kết quả. |

Bảng 7: Đặc tả Use Case quản lý đơn đặt của bệnh nhân

***e) Use Case: Quản lý thanh toán***

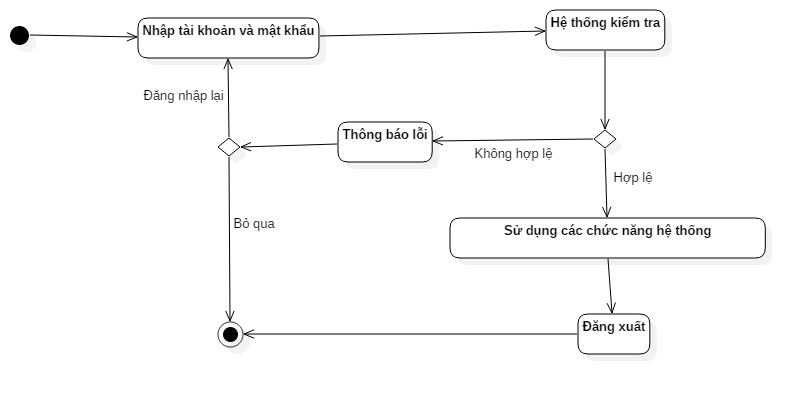
|  |  |
| --- | --- |
| Tên Use Case | Quản lý đơn đặt |
| Tác nhân | Kế toán |
| Mục đích | Cho phép kế toán thanh toán các hóa đơn của bệnh nhân. |
| Điều kiện đầu vào | Nhân viên, kế toán đăng nhập thành công, có quyền thanh toán đơn đặt của bệnh nhân. |
| Kịch bản chính | + Kế toán yêu cầu thanh toán đơn đặt của bệnh nhân.  + Hệ thống hiển thị giao diện thanh toán đơn đặt của bệnh nhân.  + Kế toán nhập số tiền thanh toán, xác nhận.  + Hệ thống hiển thị thông báo. |

Bảng 8: Đặc tả Use Case quản lý thanh toán

### **3.3 Mô hình hóa hoạt động của hệ thống**

**a) Đăng nhập**

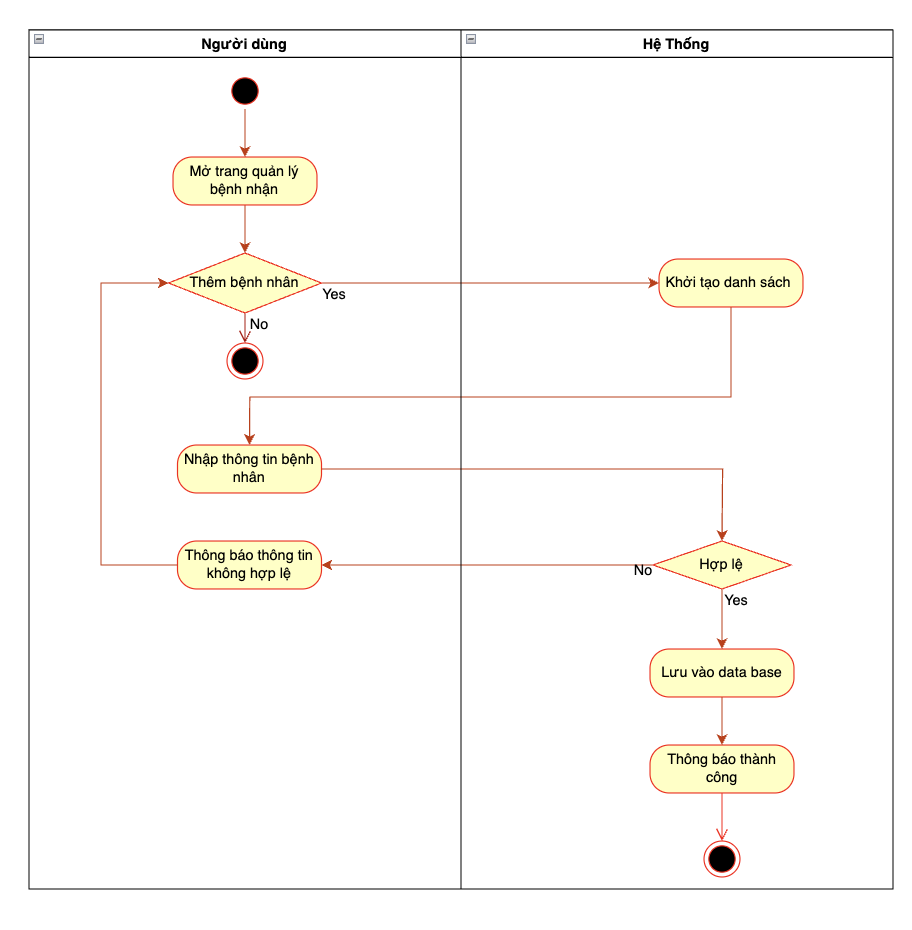
Hình 15 là biểu đồ hoạt động mô tả quy trình đăng nhập của nhân viên. Nhân viên truy cập website đăng nhập trang quản trị sau đó nhập tài khoản và mật khẩu. Hệ thống kiếm trả rồi trả kết quả lại cho người dùng.

****

Hình 15: Biểu đồ hoạt động mô tả quá trình đăng nhập

1. **Thêm nhân viên**

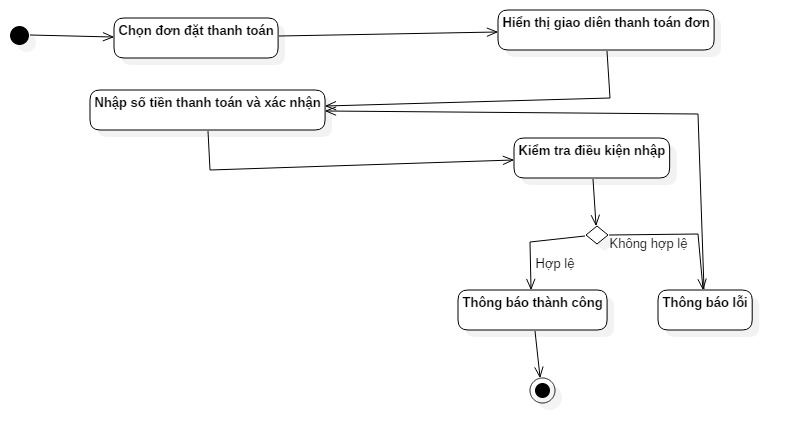
Hình 16 là biểu đồ hoạt động mô tả quy trình thêm nhân viên.



Hình 16: Biểu đồ hoạt động mô tả quá trình thêm nhân viên

1. **Thanh toán**

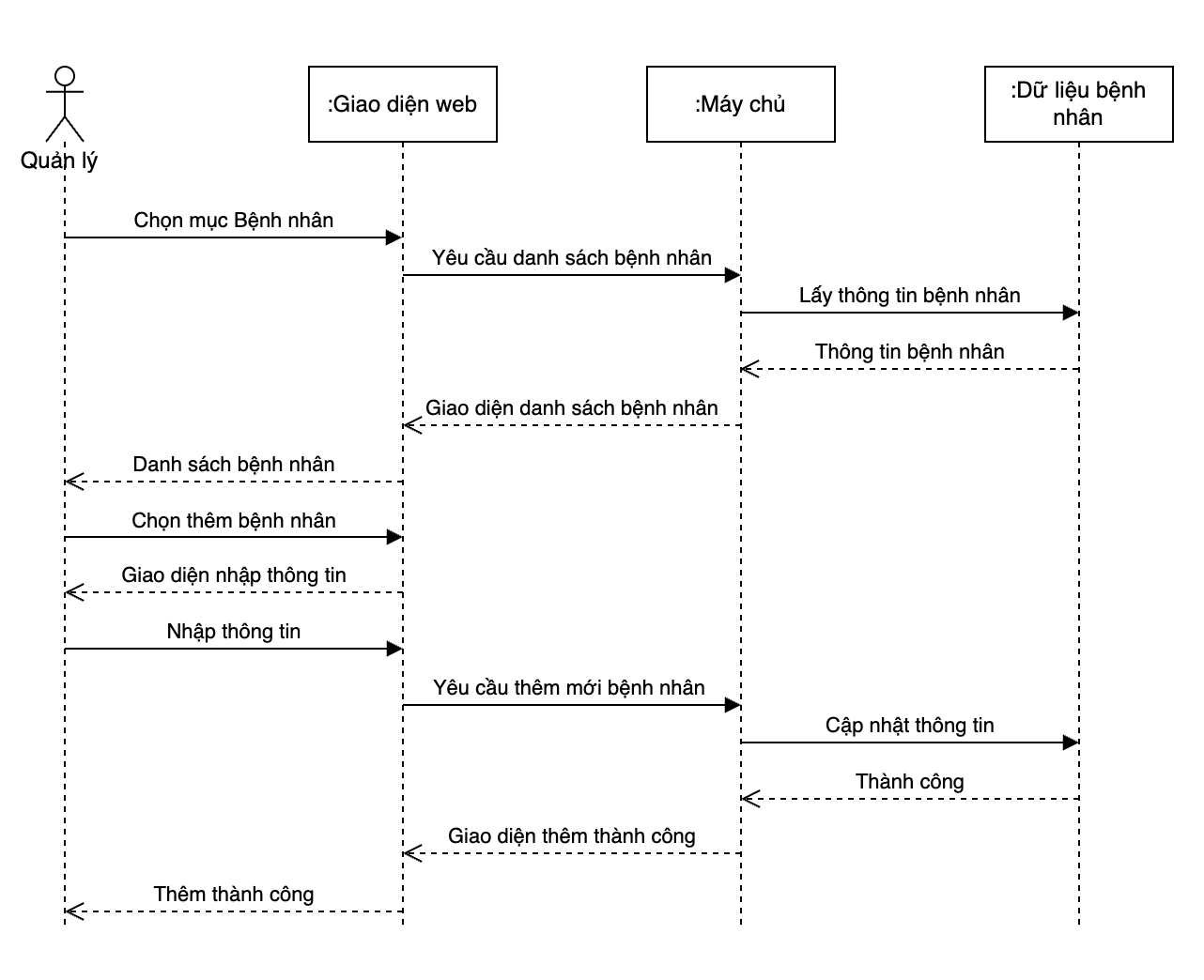
Hình 17 bên dưới là biểu đồ hoạt động mô tả quá trình nhân viên thanh toán đơn đặt tour của bệnh nhân.

****

Hình 17: Biểu đồ hoạt động Thanh toán đơn đặt bệnh nhân

### **3.4 Mô hình hóa sự tương tác**

**a) Biểu đồ trình tự cho chức năng thêm nhân viên**

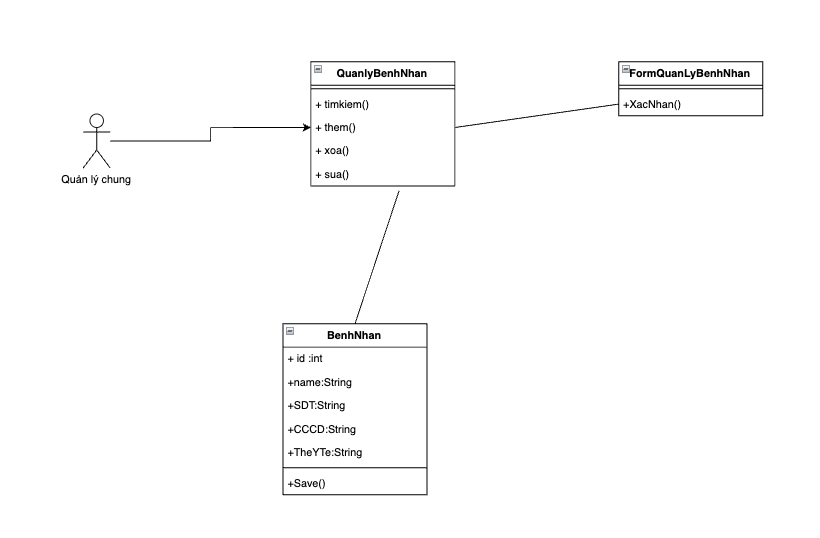


Hình 18: Biểu đồ trình tự thêm nhân viên

Hình 18 là biểu đồ trình tự cho chức năng thêm nhân viên của Actor quản lý chung.

### **3.5 Thiết kế lớp (class)**

**a) Biểu đồ lớp cho chức năng quản lý bệnh nhân)**



Hình 20: Biểu đồ lớp cho chức năng quản lý nhân viên

Hình 20 là biểu đồ lớp cho chức năng quản lý nhân viên của người quản lý chung. Biểu đồ này mô tả các phương thức và thuộc tính của các lớp mà người quản lý chung tương tác đến khi dùng chức năng quản lý nhân viên.

## 5 Xây dựng ứng dụng và triển khai

## 5 Thiết kế giao diện

**a) Giao diện phía client**

## b) Giao diện trang quản trị

# CHƯƠNG IV. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN



## 4.1 Các vấn đề được giải quyết

Đồ án đã hoàn thành được mục tiêu đề ra, bao gồm:

* Giao diện: Sử dụng HTML, CSS cùng với Boostrap và Javascript để tạo ra các trang giao diện sống động, thu hút bệnh nhân.
* Bệnh nhân có thể đăng ký thành viên, quản lý thông tin cá nhân, có thể tìm kiếm và đặt khám hoặc nhập viên Bệnh viên.
* Bệnh nhân có thể liên hệ hoặc phản hồi tới công ty
* Bệnh nhân đặt chỗ trực truyến trên website, ứng với phòng được chọn.
* Người quản trị có thể thêm, sửa, xóa bỏ các chương trình phòng,
* Người quản trị có thể thêm, sửa, xóa quyền của nhân viên để truy cập web quản trị

Website mà em xây dựng đã giải quyết được những yêu cầu cơ bản của một website quản lý bệnh viên. Tuy nhiên nếu đưa vào thực tế sử dụng thì vẫn phải cần nâng cấp nhiều trong tương lai.

## Hướng phát triển trong tương lai

* Bổ sung và hoàn thiện các trang đã có.
* Tăng cường chế độ bảo mật cho trang web.
* Xây dựng chức năng chat trực tiếp với bệnh nhân.
* Xây dựng trên đa nền tảng, đa thiết bị.

# TÀI LIỆU THAM KHẢO

[1]. Tài liệu hướng dẫn sử dụng Laravel Framework

<https://laravel.com/docs/5.4>

[2]. Tài liệu về HTML,CSS, Boostrap, JS

<https://w3school.com>

[3]. Mô hình MVC trong Laravel

<http://phpcoban.com/cai-dat-laravel-5-va-gioi-thieu-mo-hinh-mvc-trong-laravel/>

[4]. Tài liệu “ Tích hợp ngân lượng tiêu chuẩn version 2.0” – Nguyễn Cẩm Huế