**MỤC LỤC**

[CHƯƠNG 1. TỔNG QUAN 8](#_Toc154441707)

[1.1. Phân tích thực trạng 8](#_Toc154441708)

[1.2. Yêu cầu bài toán 11](#_Toc154441709)

[1.3. Yêu cầu bài toán 11](#_Toc154441710)

[1.4. Công cụ sử dụng 13](#_Toc154441711)

[1.5. Khảo sát hệ thống 13](#_Toc154441712)

[1.5.1. Giới thiệu sơ lược hệ thống 13](#_Toc154441713)

[1.5.2. Trình bày cơ cấu tổ chức 14](#_Toc154441714)

[1.6. Các quy trình nghiệp vụ 15](#_Toc154441715)

[1.6.1. Khách hàng 15](#_Toc154441716)

[1.6.2. Nhân viên chăm sóc y tế 18](#_Toc154441717)

[1.6.3. Quản trị viên 21](#_Toc154441718)

[1.7. Kết chương 23](#_Toc154441719)

[CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 25](#_Toc154441720)

[2.1. Mô hình hóa nghiệp vụ 25](#_Toc154441721)

[2.1.1. Sơ đồ Usecase nghiệp vụ 25](#_Toc154441722)

[2.2. Mô hình hóa quy trình nghiệp vụ 26](#_Toc154441723)

[2.3. Mô hình hóa chức năng 26](#_Toc154441724)

[2.3.1. Đăng nhập 26](#_Toc154441725)

[2.3.2. Đăng ký tài khoản 28](#_Toc154441726)

[2.3.3. Tìm kiếm dịch vụ chăm sóc y tế 30](#_Toc154441727)

[2.3.4. Đánh giá dịch vụ 32](#_Toc154441728)

[2.3.5. Đăng ký khám bệnh 34](#_Toc154441729)

[2.3.6. Hủy đăng ký khám bệnh 36](#_Toc154441730)

[2.3.7. Gửi đánh giá sức khỏe và phản hồi từ nhân viên chăm sóc y tế 38](#_Toc154441731)

[2.3.8. Đăng ký lịch làm việc 40](#_Toc154441732)

[2.3.9. Xác nhận/ hủy đăng ký khám bệnh 42](#_Toc154441733)

[2.3.10. Cập nhật trạng thái làm việc 44](#_Toc154441734)

[2.3.11. Cung Cấp dịch vụ chăm sóc y tế 46](#_Toc154441735)

[2.3.12. Phê duyệt tài khoản 48](#_Toc154441736)

[2.4. Thiết kế cơ sở dữ liệu 50](#_Toc154441737)

[2.5. Mô hình dữ liệu quan hệ 50](#_Toc154441738)

[2.5.1. Ý nghĩa của các bảng 51](#_Toc154441739)

[2.6. Kết chương 62](#_Toc154441740)

[CHƯƠNG 3. CỞ SỞ LÝ THUYẾT 64](#_Toc154441741)

[3.1. Giới thiệu về React Native 64](#_Toc154441742)

[3.1.1. Định nghĩa về React Native 64](#_Toc154441743)

[3.1.2. Lịch sử của React Native 66](#_Toc154441744)

[3.1.3. Cách thức hoạt động React Native 67](#_Toc154441745)

[3.1.4. Lợi ích của việc sử dụng React Native 68](#_Toc154441746)

[3.1.5. Ứng dụng của React Native 71](#_Toc154441747)

[3.2. Giới thiệu về .NET Core 76](#_Toc154441748)

[3.2.1. Định nghĩa về .NET Core 76](#_Toc154441749)

[3.2.2. Lịch sử của .NET Core 76](#_Toc154441750)

[3.2.3. Thành phần của .NET Core 77](#_Toc154441751)

[3.2.4. Đặc điểm .NET Core 78](#_Toc154441752)

[3.2.5. Ưu điểm của .NET Core 79](#_Toc154441753)

[3.2.6. NET Framework và .NET Core 80](#_Toc154441754)

[3.3. Giới thiệu về ASP.NET Core 81](#_Toc154441755)

[3.3.1. Tổng quan về ASP.NET Core 81](#_Toc154441756)

[3.3.2. Cấu trúc của ASP.NET Core 82](#_Toc154441757)

[3.3.3. Cách ASP.NET Core làm việc 86](#_Toc154441758)

[3.3.4. Đặc điểm của ASP.NET Core 87](#_Toc154441759)

[3.3.5. Ưu điểm của ASP.NET Core 88](#_Toc154441760)

[3.3.6. ASP.NET và ASP.NET Core 88](#_Toc154441761)

[3.3.7. Lợi ích khi sử dụng ASP.NET Core 90](#_Toc154441762)

[3.4. Kết chương 92](#_Toc154441763)

[CHƯƠNG 4. XÂY DỰNG ỨNG DỤNG DI ĐỘNG 93](#_Toc154441764)

[4.1. Thiết kế giao diện 93](#_Toc154441765)

[4.1.1. Giao diện giới thiệu 93](#_Toc154441766)

[4.1.2. Giao diện đăng nhập 94](#_Toc154441767)

[4.1.3. Giao diện đăng ký 95](#_Toc154441768)

[4.1.4. Giao diện trang chủ người dùng 96](#_Toc154441769)

[4.1.5. Giao diện cập nhật thông tin tài khoản 97](#_Toc154441770)

[4.1.6. Giao diện chi tiết dịch vụ 98](#_Toc154441771)

[4.1.7. Giao diện đăng ký khám bệnh 99](#_Toc154441772)

[4.1.8. Giao diện quản lý dịch vụ 100](#_Toc154441773)

[4.1.9. Giao diện cung cấp thông tin dịch vụ 101](#_Toc154441774)

[4.1.10. Giao diện quản lý danh sách đăng ký khám bệnh 102](#_Toc154441775)

[4.1.11. Giao diện cập nhật trạng thái đăng ký khám bệnh 103](#_Toc154441776)

[4.2. Xây dựng API 104](#_Toc154441777)

[4.2.1. API Quản lý tài khoản 104](#_Toc154441778)

[4.2.2. API Quản lý đăng ký dịch vụ 104](#_Toc154441779)

[4.2.3. API Quản lý dịch vụ 105](#_Toc154441780)

[4.2.4. API Quản lý địa điểm 106](#_Toc154441781)

[4.2.5. API Quản lý nhân viên 107](#_Toc154441782)

[4.2.6. API quản lý lịch làm việc 108](#_Toc154441783)

[4.2.7. API Quản lý thông tin người dùng 108](#_Toc154441784)

[4.3. Kết chương 109](#_Toc154441785)

[CHƯƠNG 5. KẾT LUẬN 110](#_Toc154441786)

[5.1. Kết luận 110](#_Toc154441787)

[5.2. Hướng phát triển tương lai 110](#_Toc154441788)

[TÀI LIỆU THAM KHẢO 113](#_Toc154441789)

**DANH MỤC CÁC HÌNH VẼ VÀ ĐỒ THỊ**

[Hình 1.1 Sơ đồ cơ cấu tổ chức hệ thống 15](#_Toc154441616)

[Hình 2.1 Sơ đồ cơ Use case tổng quát 26](#_Toc154441617)

[Hình 2.2 Use case đăng nhập 27](#_Toc154441618)

[Hình 2.3 Sơ đồ hoạt động Use case đăng nhập 28](#_Toc154441619)

[Hình 2.4 Use case đăng ký 29](#_Toc154441620)

[Hình 2.5 Sơ đồ hoạt động Use case đăng ký tài khoản 30](#_Toc154441621)

[Hình 2.6 Use case tìm kiếm dịch vụ chăm sóc y tế 31](#_Toc154441622)

[Hình 2.7 Sơ đồ hoạt động Use case tìm kiếm dịch vụ khám bệnh 32](#_Toc154441623)

[Hình 2.8 Use case đánh giá dịch vụ 33](#_Toc154441624)

[Hình 2.9 Sơ đồ hoạt động Use case đánh giá dịch vụ 34](#_Toc154441625)

[Hình 2.10 Use case Đăng ký khám bệnh 35](#_Toc154441626)

[Hình 2.11 Sơ đồ hoạt động Use case đăng ký khám bệnh 36](#_Toc154441627)

[Hình 2.12 Use case Hủy đăng ký khám bệnh 37](#_Toc154441628)

[Hình 2.13 Sơ đồ hoạt động Use case hủy đăng ký khám bệnh 38](#_Toc154441629)

[Hình 2.14 Use case gửi bảng đánh giá sức khỏe 39](#_Toc154441630)

[Hình 2.15 Sơ đồ hoạt động Use case gửi bảng đánh giá sức khỏe 40](#_Toc154441631)

[Hình 2.16 Use case Đăng ký lịch làm việc 41](#_Toc154441632)

[Hình 2.17 Sơ đồ hoạt động Use case Đăng ký lịch làm việc 42](#_Toc154441633)

[Hình 2.18 Use case Xác nhận/ Hủy đăng ký lịch làm việc 43](#_Toc154441634)

[Hình 2.19 Sơ đồ hoạt động Use case Xác nhận/ Hủy đăng ký lịch làm việc 44](#_Toc154441635)

[Hình 2.20 Use case cập nhật trạng thái làm việc 45](#_Toc154441636)

[Hình 2.21 Sơ đồ hoạt động Use case cập nhật trạng thái làm việc 46](#_Toc154441637)

[Hình 2.22 Use case cung cấp dịch vụ chăm sóc y tế 47](#_Toc154441638)

[Hình 2.23 Sơ đồ hoạt động Use case cung cấp dịch vụ chăm sóc y tê 48](#_Toc154441639)

[Hình 2.24 Use case phên duyệt tài khoản 49](#_Toc154441640)

[Hình 2.25 Sơ đồ hoạt động Use case phê duyệt tài khoản 50](#_Toc154441641)

[Hình 2.26 Cơ sở dữ liệu hệ thống 51](#_Toc154441642)

[Hình 3.1 Giới thiệu React Ntive 65](#_Toc154441643)

[Hình 3.2 React.js và React Native 66](#_Toc154441644)

[Hình 3.3 Cách thức hoạt dộng React Native 68](#_Toc154441645)

[Hình 3.4 Ứng dụng di động Facebook được phát biển bởi React Native 72](#_Toc154441646)

[Hình 3.5 Ứng dụng di động Skype được phát biển bởi React Native 73](#_Toc154441647)

[Hình 3.6 Ứng dụng di động Instagram được phát biển bởi React Native 74](#_Toc154441648)

[Hình 3.7 Ứng dụng di động Uber được phát biển bởi React Native 75](#_Toc154441649)

[Hình 3.8 Ứng dụng di động Facebook được phát biển bởi React Native 76](#_Toc154441650)

[Hình 3.9 Lịch sử phát triển Net Core 78](#_Toc154441651)

[Hình 3.10 Thành phần của .Net Core 79](#_Toc154441652)

[Hình 3.11 Net core và Net framework 82](#_Toc154441653)

[Hình 3.12 Cấu trúc của ASP.NET core 83](#_Toc154441654)

[Hình 3.13 Cơ chế hoạt động ASP.Net core 87](#_Toc154441655)

[Hình 4.1 Giao diện giới thiệu ứng dụng 94](#_Toc154441656)

[Hình 4.2 Giao diện đăng nhập 95](#_Toc154441657)

[Hình 4.3 Giao diện đăng ký 96](#_Toc154441658)

[Hình 4.4 Giao diện trang chủ người dùng 97](#_Toc154441659)

[Hình 4.5 Giao diện cập nhật thông tin tài khoản 98](#_Toc154441660)

[Hình 4.6 Giao diện chi tiết dịch vụ 99](#_Toc154441661)

[Hình 4.7 Giao diện đăng ký khám bệnh 100](#_Toc154441662)

[Hình 4.8 Giao diện quản lý dịch vụ 101](#_Toc154441663)

[Hình 4.9 Giao diện cung cấp thông tin dịch vụ 102](#_Toc154441664)

[Hình 4.10 Giao diện quản lý danh sách đăng ký khám bệnh 103](#_Toc154441665)

[Hình 4.11 Giao diện cập nhật trạng thái đăng ký khám bệnh 104](#_Toc154441666)

**MỞ ĐẦU**

Trong những năm gần đây, nhu cầu chăm sóc sức khỏe của người dân ngày càng tăng cao. Điều này dẫn đến sự gia tăng của các cơ sở y tế, từ bệnh viện, phòng khám đến các dịch vụ chăm sóc sức khỏe tại nhà. Tuy nhiên, việc tìm kiếm và đăng ký các dịch vụ chăm sóc y tế vẫn còn nhiều bất tiện. Người dân thường phải trực tiếp đến cơ sở y tế để đăng ký hoặc gọi điện thoại để đặt lịch. Điều này gây tốn thời gian và công sức.

Ứng dụng công nghệ thông tin (CNTT) vào lĩnh vực y tế là một giải pháp hiệu quả để giải quyết vấn đề này. Xây dựng hệ thống đăng dịch vụ chăm sóc y tế trực tuyến sẽ giúp người dân có thể dễ dàng tìm kiếm và đăng ký các dịch vụ chăm sóc y tế mà không cần phải trực tiếp đến cơ sở y tế.

Bố cục đề tài nghiên cứu khóa luận tốt nghiệp này bao gồm các chương  
**Chương 1: Tổng quan**

Trình bày rõ các nghiệp vụ của hệ thống đăng dịch vụ chăm sóc y tế trực tuyến, các phần mềm được sử dụng trong quá trình phát triển đề tài, chỉ ra mục tiêu hướng phát triển của đề tài.

**Chương 2: Phân tích và thiết kế hệ thống**

Phân tích và thiết kế hệ thống, phân tách các chức năng của hệ thống. Thiết kế các sơ đồ use case nghiệp vụ, sơ đồ hoạt động, sơ đồ tuần tự của từng loại nghiệp vụ và mô hình hóa các nghiệp vụ bằng bảng. Thiết kế các use case hệ thống, mô hình hóa các hệ thống bằng bảng, thiết kế sơ đồ lớp ở mức phân tích và thiết kế sau đó chuyển đổi thành sơ đồ diagrams trong SQL. Từ các chức năng trong use case hệ thống đặc tả thành sơ đồ lớp ba tầng và sơ đồ tuần tự của chúng. Sau đó kết luận những gì đạt được trong chương.

**Chương 3: Cơ sở lý thuyết**  
Giới thiệu cách thức hoạt động, ưu điểm và nhược điểm, ứng dụng của React Native  
Giới thiệu cách thức hoạt động, ưu điểm và nhược điểm, ứng dụng của ASP.Net core Web API

**Chương 4: Xây dựng ứng dụng của hệ thống**

Tổng quản về website, mô tả phần API và giao diện của website. Cách cài đặt chương trình

**Chương 5: Kết luận**

Tổng quát về hệ thống và đưa các giải pháp phát triển trong tương lai

# TỔNG QUAN

## Phân tích thực trạng

Hiện nay, việc đăng ký dịch vụ chăm sóc y tế vẫn còn gặp nhiều khó khăn, bất cập, cụ thể như sau:

* Khách hàng phải đến trực tiếp cơ sở y tế để đăng ký dịch vụ: Điều này gây mất thời gian, công sức và chi phí đi lại cho khách hàng.
* Khách hàng phải chờ đợi lâu để được đăng ký dịch vụ: Điều này gây tâm lý bức xúc cho khách hàng, đặc biệt là trong những trường hợp cấp cứu.
* Quy trình đăng ký dịch vụ còn nhiều thủ tục rườm rà: Điều này khiến khách hàng cảm thấy mệt mỏi và chán nản.
* Từ góc độ khách hàng:
  + Khách hàng ngày càng có nhu cầu sử dụng các dịch vụ chăm sóc y tế một cách nhanh chóng, thuận tiện và hiệu quả.
  + Khách hàng có xu hướng sử dụng các dịch vụ trực tuyến, trong đó có dịch vụ đăng ký dịch vụ chăm sóc y tế.
* Từ góc độ nhân viên y tế:
  + Nhân viên y tế cũng mong muốn có thể tiếp cận với khách hàng một cách nhanh chóng và hiệu quả.
  + Nhân viên y tế có thể sử dụng các công nghệ thông tin để nâng cao chất lượng phục vụ khách hàng.

Giải pháp xây dựng hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến

Xây dựng hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến là giải pháp hiệu quả để giải quyết các khó khăn, bất cập trong việc đăng ký dịch vụ chăm sóc y tế hiện nay. Hệ thống này sẽ mang lại những lợi ích sau:

* Tiện lợi cho khách hàng: Khách hàng có thể đăng ký dịch vụ chăm sóc y tế mọi lúc, mọi nơi, mà không cần phải đến trực tiếp cơ sở y tế.
* Nhanh chóng và hiệu quả: Quy trình đăng ký dịch vụ được đơn giản hóa, giúp khách hàng tiết kiệm thời gian và công sức.
* Tăng cường tính minh bạch: Hệ thống giúp khách hàng dễ dàng theo dõi tình trạng đăng ký dịch vụ của mình.

Yêu cầu cụ thể của hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến

Hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cần đáp ứng các yêu cầu cụ thể sau:

* Cung cấp đầy đủ thông tin về các dịch vụ chăm sóc y tế: Khách hàng cần được cung cấp đầy đủ thông tin về các dịch vụ chăm sóc y tế, bao gồm tên dịch vụ, mô tả dịch vụ, giá dịch vụ,...
* Cho phép khách hàng đăng ký dịch vụ một cách nhanh chóng và thuận tiện: Quy trình đăng ký dịch vụ cần được đơn giản hóa, giúp khách hàng dễ dàng thực hiện.
* Cung cấp thông tin về tình trạng đăng ký dịch vụ: Khách hàng cần được cung cấp thông tin về tình trạng đăng ký dịch vụ của mình, bao gồm trạng thái đăng ký, thời gian và địa điểm thực hiện dịch vụ,...

Phân hệ của hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến

Hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến được chia thành 3 phân hệ chính:

* Ứng dụng di động dành cho nhân viên y tế: Phân hệ này cho phép nhân viên y tế công bố lịch làm việc, chấp nhận hoặc từ chối đăng ký của khách hàng, lưu kết quả thực hiện.
* Ứng dụng di động dành cho khách hàng: Phân hệ này cho phép khách hàng đăng ký tài khoản, đăng ký dịch vụ chăm sóc, xác nhận kết quả thực hiện.
* Ứng dụng web dùng để quản lý thông tin: Phân hệ này cho phép quản trị viên quản lý thông tin nhân viên y tế, khách hàng sử dụng dịch vụ y tế, quá trình hoạt động, thống kê kết quả hoạt động.

Xây dựng hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến là giải pháp hiệu quả để giải quyết các khó khăn, bất cập trong việc đăng ký dịch vụ chăm sóc y tế hiện nay. Hệ thống này sẽ mang lại nhiều lợi ích cho cả khách hàng và nhân viên y tế.

## Yêu cầu bài toán

## Yêu cầu bài toán

* Hệ thống đăng dịch vụ chăm sóc y tế trực tuyến, có 3 tác nhân chính:
* Khách hàng: là người sử dụng hệ thống để tìm kiếm và đăng ký các dịch vụ chăm sóc y tế.
* Nhân viên chăm sóc y tế: là người cung cấp các dịch vụ chăm sóc y tế trên hệ thống.
* Quản trị viên: là người quản lý hệ thống, bao gồm duyệt cấp tài khoản người dùng, phê duyệt profile người dùng, tìm kiếm lịch làm việc của nhân viên chăm sóc y tế, tìm kiếm dịch vụ đăng ký của khách hàng, và thống kê số liệu.

1. Khách hàng có thể thực hiện các chức năng sau:

* Tìm kiếm dịch vụ chăm sóc y tế: khách hàng có thể tìm kiếm các dịch vụ chăm sóc y tế theo các tiêu chí như loại dịch vụ, địa điểm, giá cả,...
* Đăng ký dịch vụ chăm sóc y tế: khách hàng có thể đăng ký sử dụng các dịch vụ chăm sóc y tế mà họ tìm thấy.
* Đặt lịch khám/chữa bệnh trực tuyến: khách hàng có thể đặt lịch khám/chữa bệnh trực tuyến với nhân viên chăm sóc y tế.
* Đánh giá chất lượng dịch vụ chăm sóc y tế: khách hàng có thể đánh giá chất lượng dịch vụ chăm sóc y tế mà họ đã sử dụng.

1. Nhân viên chăm sóc y có thể thực hiện các chức năng sau:

* Đăng ký tài khoản: nhân viên chăm sóc y tế có thể đăng ký tài khoản trên hệ thống để cung cấp các dịch vụ chăm sóc y tế.
* Cung cấp thông tin dịch vụ: nhân viên chăm sóc y tế có thể cung cấp thông tin về các dịch vụ chăm sóc y tế mà họ cung cấp.
* Nhận/huỷ đăng ký dịch vụ: nhân viên chăm sóc y tế có thể nhận hoặc huỷ đăng ký dịch vụ của khách hàng.

1. Quản trị viên là tác nhân quản lý hệ thống có thể thực hiện các chức năng sau:

* Duyệt cấp tài khoản người dùng: quản trị viên có thể duyệt cấp tài khoản người dùng cho khách hàng và nhân viên chăm sóc y tế.
* Phê duyệt profile người dùng: quản trị viên có thể phê duyệt profile người dùng cho khách hàng và nhân viên chăm sóc y tế.
* Tìm kiếm lịch làm việc của nhân viên chăm sóc y tế: quản trị viên có thể tìm kiếm lịch làm việc của nhân viên chăm sóc y tế.
* Tìm kiếm dịch vụ đăng ký của khách hàng: quản trị viên có thể tìm kiếm dịch vụ đăng ký của khách hàng.
* Thống kê số liệu: quản trị viên có thể thống kê số liệu về hoạt động của hệ thống.

## Công cụ sử dụng

* Draw.io: Vẽ các sơ đồ use case nghiệp vụ, use case hệ thống, sơ đồ tuần tự, sơ đồ hoạt động, sơ đồ lớp ở mức phân tích, sơ đồ lớp ở mức thiết kế, sơ đồ ba tầng thiết kế.
* Github: Lưu trữ các source code về API, UI mobile của nhóm.
* Microsoft SQL Server 2022: Tạo database, lưu trữ các dữ liệu, lập sơ đồ cơ sở dữ liệu của hệ thống.
* Microsoft Visual Studio 2022: Tạo API dùng ASP.NET Core 7.0.
* Visual Studio Code: Tạo UI mobile sử dụng thư viện React Native.
* Ngôn ngữ lập trình: C#, Javascript

## Khảo sát hệ thống

### Giới thiệu sơ lược hệ thống

Hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cho ứng dụng di động là một giải pháp công nghệ giúp người dân đăng ký khám chữa bệnh tại các cơ sở y tế một cách thuận tiện và nhanh chóng. Hệ thống này được cài đặt trên điện thoại di động, giúp người dân có thể truy cập và sử dụng mọi lúc, mọi nơi.

Hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cho ứng dụng di động mang lại nhiều lợi ích cho người dân và các cơ sở y tế, bao gồm:

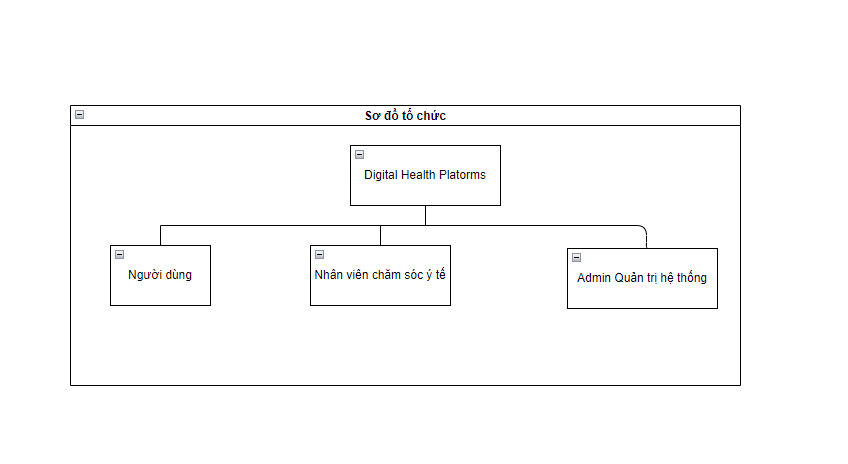
* Đăng ký khám chữa bệnh: Người dân có thể đăng ký khám chữa bệnh tại các cơ sở y tế, bao gồm thông tin về bác sĩ, chuyên khoa, thời gian khám,...
* Tìm kiếm thông tin về các cơ sở y tế: Người dân có thể tìm kiếm thông tin về các cơ sở y tế, bao gồm địa chỉ, số điện thoại, chuyên khoa,...
* Lưu trữ hồ sơ sức khỏe điện tử: Hệ thống lưu trữ hồ sơ sức khỏe điện tử của người dân, bao gồm thông tin về bệnh án, tiền sử bệnh,...
* Thông báo và nhắc lịch khám: Hệ thống gửi thông báo và nhắc lịch khám cho người dân.
* Thanh toán dịch vụ y tế trực tuyến: Hệ thống hỗ trợ thanh toán dịch vụ y tế trực tuyến.

Hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cho ứng dụng di động mang lại nhiều lợi ích cho người dân và các cơ sở y tế, bao gồm:

* Tiện lợi cho người dân: Người dân có thể đăng ký khám chữa bệnh, tìm kiếm thông tin về các cơ sở y tế và lưu trữ hồ sơ sức khỏe điện tử một cách thuận tiện và nhanh chóng.
* Tiết kiệm thời gian và chi phí: Người dân không cần phải đến trực tiếp cơ sở y tế để đăng ký khám chữa bệnh. Điều này giúp tiết kiệm thời gian và chi phí đi lại.
* Nâng cao chất lượng dịch vụ y tế: Hệ thống giúp các cơ sở y tế quản lý hiệu quả lịch khám, thông báo và nhắc lịch khám cho người dân. Điều này giúp nâng cao chất lượng dịch vụ y tế.

Hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cho ứng dụng di động là một giải pháp công nghệ hiện đại, góp phần cải thiện chất lượng chăm sóc sức khỏe cho người dân.

### Trình bày cơ cấu tổ chức



Hình 1.1 Sơ đồ cơ cấu tổ chức hệ thống

Sơ đồ cơ cấu tổ chức hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cho ứng di động gồm ba nhóm chính là người dùng, nhân viên chăm sóc y tế và admin quản trị hệ thống.

Người dùng là nhóm đối tượng sử dụng hệ thống để đăng ký khám chữa bệnh, tìm kiếm thông tin về các cơ sở y tế và lưu trữ hồ sơ sức khỏe điện tử. Người dùng có thể là cá nhân hoặc tổ chức.

Nhân viên chăm sóc y tế là nhóm đối tượng cung cấp dịch vụ chăm sóc y tế cho người dân, bao gồm bác sĩ, y tá, dược sĩ,... Nhân viên chăm sóc y tế có thể làm việc tại các cơ sở y tế công lập, tư nhân hoặc tự do hành nghề.

* Người dùng
* Đăng ký khám chữa bệnh tại các cơ sở y tế
* Tìm kiếm thông tin về các cơ sở y tế
* Lưu trữ hồ sơ sức khỏe điện tử
* Nhân viên chăm sóc y tế
* Quản lý lịch khám
* Thông báo và nhắc lịch khám cho người dân
* Admin quản trị hệ thống
* Xây dựng và triển khai hệ thống
* Bảo trì và nâng cấp hệ thống
* Đảm bảo an ninh và bảo mật hệ thống
* Giải quyết các vấn đề phát sinh trong quá trình sử dụng hệ thống

## Các quy trình nghiệp vụ

### Khách hàng

#### 1.6.1.1 Tìm Kiếm dịch vụ chăm sóc y tế

Cho phép người dùng tìm kiếm các dịch vụ chăm sóc y tế theo nhiều tiêu chí khác nhau như loại dịch vụ, địa điểm, giá cả, v.v.

Bước 1: Nhập thông tin tìm kiếm

Người dùng truy cập chức năng tìm kiếm trên ứng dụng.

Họ nhập các thông tin tìm kiếm như loại dịch vụ mong muốn, địa điểm, khoảng giá cả, v.v.

Bước 2: Xử lý thông tin tìm kiếm

Hệ thống nhận và xử lý thông tin tìm kiếm từ người dùng.

Dựa trên các tiêu chí như loại dịch vụ, địa điểm, giá cả, hệ thống truy vấn vào cơ sở dữ liệu để lấy thông tin liên quan.

Bước 3: Trả về kết quả tìm kiếm

Hệ thống trả về danh sách các dịch vụ chăm sóc y tế phù hợp với thông tin tìm kiếm.

Mỗi kết quả sẽ bao gồm các chi tiết như tên dịch vụ, địa điểm, giá cả, và các thông tin khác liên quan.

Bước 4: Chọn dịch vụ chăm sóc y tế

Người dùng xem qua danh sách kết quả và chọn một dịch vụ chăm sóc y tế mà họ quan tâm.

Bước 5: Đăng ký dịch vụ

Người dùng chọn dịch vụ cụ thể và tiến hành đăng ký hoặc liên hệ trực tiếp theo hướng dẫn cung cấp.

Kết quả:

Người dùng có thể dễ dàng tìm kiếm và xác định được các dịch vụ chăm sóc y tế phù hợp với nhu cầu của họ thông qua quy trình này.

#### 1.6.1.2 Đăng ký dịch vụ chăm sóc y tế

Mục đích:

Cho phép người dùng đăng ký sử dụng các dịch vụ chăm sóc y tế mà họ đã tìm thấy trong quá trình tìm kiếm.

Bước 1: Chọn dịch vụ chăm sóc y tế

Người dùng sau khi tìm kiếm chọn một dịch vụ cụ thể mà họ muốn đăng ký.

Bước 2: Hiển thị thông tin chi tiết về dịch vụ

Hệ thống hiển thị thông tin chi tiết về dịch vụ chăm sóc y tế, bao gồm các thông tin như mô tả dịch vụ, giá cả, địa điểm, và thông tin liên hệ.

Bước 3: Nhập thông tin cá nhân và lịch hẹn

Người dùng điền thông tin cá nhân cần thiết như họ tên, số điện thoại, địa chỉ, và thời gian mong muốn sử dụng dịch vụ.

Bước 4: Xác nhận đăng ký

Người dùng xác nhận thông tin đã nhập và yêu cầu đăng ký dịch vụ chăm sóc y tế.

Bước 5: Gửi yêu cầu đăng ký đến nhân viên chăm sóc y tế

Hệ thống gửi yêu cầu đăng ký, bao gồm thông tin chi tiết và lịch hẹn, đến cho nhân viên chăm sóc y tế hoặc bộ phận liên quan.

Kết quả:

Người dùng đã hoàn tất quá trình đăng ký dịch vụ chăm sóc y tế mà họ mong muốn, và thông tin đã được chuyển đến nhân viên chăm sóc y tế để xử lý tiếp theo.

#### 1.6.1.3 Đặt lịch khám/chữa bệnh trực tuyến

Mục đích:

Cho phép người dùng đặt lịch khám/chữa bệnh trực tuyến với nhân viên chăm sóc y tế.

Bước 1: Chọn dịch vụ chăm sóc y tế và đặt lịch

Người dùng chọn dịch vụ chăm sóc y tế mà họ muốn đặt lịch.

Bước 2: Hiển thị lịch làm việc của nhân viên chăm sóc y tế

Hệ thống hiển thị lịch làm việc của nhân viên chăm sóc y tế, bao gồm các khung giờ trống và đã đặt lịch.

Bước 3: Chọn lịch hẹn phù hợp

Người dùng chọn một khung giờ trống trong lịch làm việc của nhân viên chăm sóc y tế mà họ mong muốn.

Bước 4: Xác nhận đặt lịch

Người dùng xác nhận thông tin đặt lịch, bao gồm dịch vụ chăm sóc y tế và thời gian đặt lịch.

Bước 5: Gửi yêu cầu đặt lịch đến nhân viên chăm sóc y tế

Hệ thống gửi yêu cầu đặt lịch, bao gồm thông tin chi tiết và thời gian lịch hẹn, đến cho nhân viên chăm sóc y tế.

Kết quả:

Người dùng đã đặt lịch khám/chữa bệnh trực tuyến thành công, và thông tin đã được chuyển đến nhân viên chăm sóc y tế để chuẩn bị cho lịch hẹn.

#### 1.6.1.4 Đánh giá chất lượng dịch vụ chăm sóc y tế

Mục đích:

Cho phép người dùng đánh giá chất lượng dịch vụ chăm sóc y tế mà họ đã sử dụng.

Bước 1: Truy cập trang đánh giá

Người dùng truy cập trang đánh giá trên ứng dụng hoặc trang web.

Bước 2: Nhập thông tin đánh giá

Người dùng nhập thông tin đánh giá, bao gồm các mục như đánh giá chất lượng dịch vụ, đội ngũ nhân viên, thời gian phục vụ, và các ý kiến, ghi chú khác.

Bước 3: Lưu thông tin đánh giá

Hệ thống lưu thông tin đánh giá vào cơ sở dữ liệu, kèm theo thông tin về người đánh giá và dịch vụ chăm sóc y tế cụ thể.

Kết quả:

Đánh giá chất lượng dịch vụ đã được người dùng nhập và lưu trữ thành công trong hệ thống. Thông tin này có thể được sử dụng để cung cấp phản hồi cho nhân viên chăm sóc y tế và cải thiện chất lượng dịch vụ trong tương lai.

### Nhân viên chăm sóc y tế

#### 1.6.2.1 Đăng ký tài khoản cho nhân viên chăm sóc y tế

Mục đích:

Cho phép nhân viên chăm sóc y tế đăng ký tài khoản trên hệ thống để cung cấp các dịch vụ chăm sóc y tế.

Bước 1: Nhập thông tin đăng ký

Nhân viên chăm sóc y tế truy cập phần đăng ký trên ứng dụng di động hoặc trang web.

Họ nhập thông tin đăng ký bao gồm tên, địa chỉ email, số điện thoại, và thông tin định danh như mã nhân viên….

Bước 2: Gửi yêu cầu đăng ký đến quản trị viên

Hệ thống gửi yêu cầu đăng ký, bao gồm thông tin chi tiết của nhân viên chăm sóc y tế, đến cho quản trị viên hoặc bộ phận liên quan để xác nhận và phê duyệt.

Kết quả:

Yêu cầu đăng ký của nhân viên chăm sóc y tế đã được gửi đến quản trị viên để xử lý. Sau khi được phê duyệt, nhân viên sẽ có tài khoản trên hệ thống và có thể bắt đầu cung cấp các dịch vụ chăm sóc y tế.

#### 1.6.2.2 Cung cấp thông tin dịch vụ cho nhân viên chăm sóc y tế

Mục đích:

Cho phép nhân viên chăm sóc y tế cung cấp thông tin về các dịch vụ chăm sóc y tế mà họ cung cấp.

Bước 1: Nhập thông tin dịch vụ

Nhân viên chăm sóc y tế đăng nhập vào hệ thống và truy cập phần quản lý dịch vụ.

Họ nhập thông tin chi tiết về các dịch vụ chăm sóc y tế mà họ cung cấp, bao gồm tên dịch vụ, mô tả, loại hình, giá cả, địa điểm, và thông tin khác liên quan.

Bước 2: Lưu thông tin dịch vụ

Hệ thống lưu thông tin chi tiết về dịch vụ vào cơ sở dữ liệu, kết hợp với thông tin của nhân viên chăm sóc y tế để liên kết với tài khoản của họ.

Kết quả:

Thông tin chi tiết về các dịch vụ chăm sóc y tế đã được nhân viên chăm sóc y tế nhập và lưu trữ thành công trong hệ thống. Các thông tin này sẽ được hiển thị cho người dùng khi họ tìm kiếm và đặt lịch hẹn cho các dịch vụ này.

#### 1.6.2.3 Nhận/hủy đăng ký dịch vụ cho nhân viên chăm sóc y tế

Mục đích:

Cho phép nhân viên chăm sóc y tế nhận hoặc huỷ đăng ký dịch vụ của khách hàng.

Bước 1: Nhận hoặc huỷ đăng ký dịch vụ

Nhân viên chăm sóc y tế đăng nhập vào hệ thống và truy cập phần quản lý đăng ký dịch vụ.

Họ xem danh sách các đăng ký dịch vụ chờ xử lý và có thể chọn nhận hoặc huỷ đăng ký dịch vụ từ khách hàng.

Bước 2: Cập nhật trạng thái đăng ký dịch vụ

Hệ thống cập nhật trạng thái của đăng ký dịch vụ tương ứng theo quyết định của nhân viên chăm sóc y tế.

Nếu nhân viên chọn nhận đăng ký, trạng thái sẽ được chuyển sang "Đã nhận."

Nếu nhân viên chọn huỷ đăng ký, trạng thái sẽ được chuyển sang "Đã huỷ."

Kết quả:

Trạng thái của đăng ký dịch vụ được cập nhật dựa trên quyết định của nhân viên chăm sóc y tế. Thông báo về thay đổi này có thể được gửi đến khách hàng và nhân viên cùng các bên liên quan.

#### 1.6.2.4 Gửi bản đánh giá sức khỏe và phản hồi từ nhân viên chăm sóc y tế

Mục đích:

Cho phép khách hàng gửi bản đánh giá sức khỏe và nhân viên y tế đưa mô tả và các biện pháp phòng tránh.

Bước 1: Khách hàng gửi bản đánh giá sức khỏe

Khách hàng đăng nhập vào hệ thống và truy cập phần đánh giá sức khỏe.

Họ hoàn thành bản đánh giá với các thông tin như triệu chứng, tình trạng sức khỏe, và các yếu tố khác liên quan.

Bước 2: Nhân viên y tế đưa mô tả và các biện pháp phòng tránh

Nhân viên y tế đăng nhập vào hệ thống và truy cập phần đánh giá sức khỏe đã được gửi.

Họ đưa ra mô tả về tình trạng sức khỏe của khách hàng và đề xuất các biện pháp phòng tránh hoặc hướng dẫn tiếp theo.

Bước 3: Cập nhật trạng thái đánh giá

Hệ thống cập nhật trạng thái của đánh giá sức khỏe, bao gồm cả mô tả và biện pháp phòng tránh từ nhân viên y tế.

Thông báo về nội dung mới có thể được gửi đến khách hàng để họ có thể theo dõi và thực hiện các hướng dẫn cần thiết.

Kết quả:

Bản đánh giá sức khỏe được gửi từ khách hàng và phản hồi từ nhân viên y tế đã được lưu trữ và cập nhật trong hệ thống. Cả hai bên có thể theo dõi thông tin này để đảm bảo quản lý sức khỏe hiệu quả.

### Quản trị viên

#### 1.6.3.1 Duyệt cấp tài khoản người dùng

Mục đích:

Cho phép quản trị viên duyệt cấp tài khoản người dùng cho khách hàng và nhân viên chăm sóc y tế.

Bước 1: Duyệt hoặc từ chối yêu cầu đăng ký tài khoản

Quản trị viên đăng nhập vào hệ thống và kiểm tra danh sách yêu cầu đăng ký tài khoản từ khách hàng và nhân viên chăm sóc y tế.

Quản trị viên có thể duyệt hoặc từ chối yêu cầu dựa trên thông tin được cung cấp và các tiêu chí xác nhận.

Bước 2: Gửi thông báo đến người dùng

Hệ thống gửi thông báo đến người dùng về kết quả của yêu cầu đăng ký tài khoản.

Nếu yêu cầu được duyệt, thông báo sẽ chứa thông tin đăng nhập và hướng dẫn cụ thể.

Nếu yêu cầu bị từ chối, thông báo sẽ cung cấp lý do từ chối (nếu có) và hướng dẫn về các bước tiếp theo.

Kết quả:

Quản trị viên đã duyệt hoặc từ chối yêu cầu đăng ký tài khoản, và người dùng đã nhận được thông báo tương ứng với kết quả của yêu cầu.

#### 1.6.3.2 Phê duyệt profile người dùng

Mục đích:

Cho phép quản trị viên phê duyệt profile người dùng cho khách hàng và nhân viên chăm sóc y tế.

Bước 1: Phê duyệt hoặc từ chối yêu cầu phê duyệt profile

Quản trị viên đăng nhập vào hệ thống và kiểm tra danh sách yêu cầu phê duyệt profile từ khách hàng và nhân viên chăm sóc y tế.

Quản trị viên có thể phê duyệt hoặc từ chối yêu cầu dựa trên thông tin đã được cung cấp và các tiêu chí xác nhận.

Bước 2: Gửi thông báo đến người dùng

Hệ thống gửi thông báo đến người dùng về kết quả của yêu cầu phê duyệt profile.

Nếu yêu cầu được phê duyệt, thông báo sẽ chứa thông tin về việc profile đã được chấp nhận và có thể sử dụng.

Nếu yêu cầu bị từ chối, thông báo sẽ cung cấp lý do từ chối (nếu có) và hướng dẫn về các bước tiếp theo.

Kết quả:

Quản trị viên đã phê duyệt hoặc từ chối yêu cầu phê duyệt profile, và người dùng đã nhận được thông báo tương ứng với kết quả của yêu cầu.

#### 1.6.3.3 Tìm kiếm lịch làm việc của nhân viên chăm sóc y tế

Mục đích:

Cho phép quản trị viên tìm kiếm lịch làm việc của nhân viên chăm sóc y tế.

Bước 1: Nhập thông tin tìm kiếm

Quản trị viên đăng nhập vào hệ thống và truy cập phần tìm kiếm lịch làm việc.

Họ nhập thông tin tìm kiếm, bao gồm tên nhân viên, khoảng thời gian, hoặc các tiêu chí khác liên quan đến lịch làm việc.

Bước 2: Hệ thống trả về danh sách lịch làm việc của nhân viên chăm sóc y tế

Hệ thống xử lý thông tin tìm kiếm và trả về danh sách lịch làm việc của nhân viên chăm sóc y tế phù hợp với các tiêu chí đã nhập.

Danh sách có thể bao gồm các thông tin như tên nhân viên, thời gian làm việc, các dịch vụ được cung cấp, và trạng thái của lịch làm việc.

Kết quả:

Quản trị viên đã tìm kiếm thành công và nhận được danh sách lịch làm việc của nhân viên chăm sóc y tế dựa trên thông tin đã nhập. Thông tin này có thể được sử dụng để quản lý và điều chỉnh lịch làm việc của nhân viên.

## Kết chương

Mục tiêu của dự án "Hệ thống đăng dịch vụ chăm sóc y tế trực tuyến cho ứng dụng di động" là xây dựng một nền tảng hiệu quả, linh hoạt và tiện lợi để kết nối giữa người dùng và nhân viên chăm sóc y tế thông qua ứng dụng di động. Mục tiêu chính của hệ thống là cung cấp một trải nghiệm đơn giản và thuận tiện cho người dùng khi tìm kiếm, đăng ký, và sử dụng các dịch vụ chăm sóc y tế.

1. Tối ưu hóa Quy trình Đăng ký và Sử dụng Dịch vụ:

* Đơn giản hóa và tối ưu hóa quy trình đăng ký dịch vụ chăm sóc y tế cho người dùng qua ứng dụng di động.
* Cung cấp khả năng tìm kiếm dịch vụ chăm sóc y tế theo nhiều tiêu chí khác nhau để đáp ứng nhanh chóng nhu cầu của người dùng.

1. Tích hợp Đặt lịch và Gửi Đánh giá:

* Cho phép người dùng đặt lịch trực tuyến với nhân viên chăm sóc y tế một cách thuận tiện.
* Tích hợp chức năng đánh giá dịch vụ để người dùng có thể chia sẻ trải nghiệm và giúp cải thiện chất lượng dịch vụ.

1. Đảm bảo Bảo mật và Quản lý Thông tin:

* Bảo vệ thông tin cá nhân của người dùng và đảm bảo an toàn dữ liệu y tế theo các tiêu chuẩn bảo mật cao.
* Cung cấp cho quản trị viên khả năng quản lý thông tin lịch làm việc và dịch vụ của nhân viên chăm sóc y tế.

1. Hỗ trợ Tương tác Thời gian thực:

* Tích hợp chức năng trò chuyện trực tuyến giữa người dùng và nhân viên chăm sóc y tế để giải đáp thắc mắc và cung cấp hỗ trợ ngay lập tức.

1. Phát triển Ứng dụng Di động Linh hoạt:

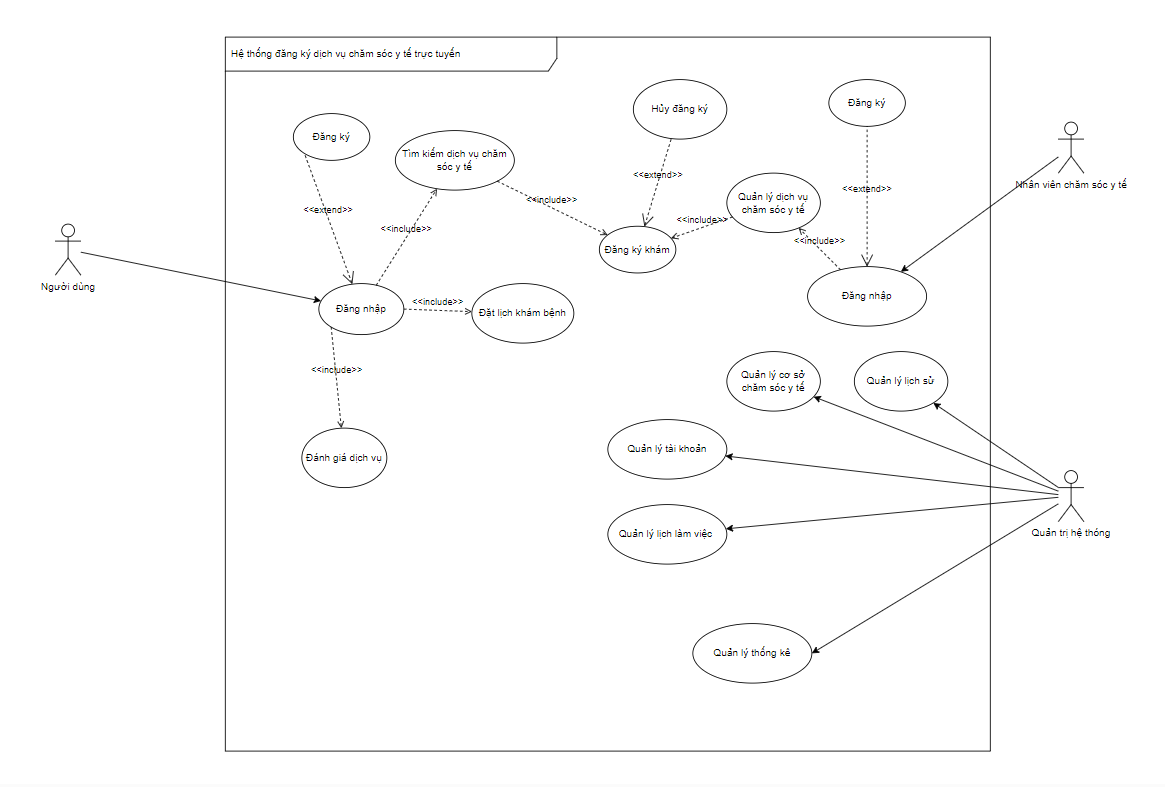
* Xây dựng ứng dụng di động có khả năng tương thích trên nhiều nền tảng và thiết bị để đảm bảo sự tiện lợi và tiếp cận rộng rãi cho người dùng.

Mục tiêu chung là tạo ra một hệ thống chăm sóc y tế trực tuyến toàn diện, nơi mà người dùng có thể dễ dàng tương tác, đặt lịch và đánh giá dịch vụ, đồng thời nhân viên chăm sóc y tế có thể quản lý thông tin và tương tác với người dùng một cách hiệu quả.

# PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## Mô hình hóa nghiệp vụ

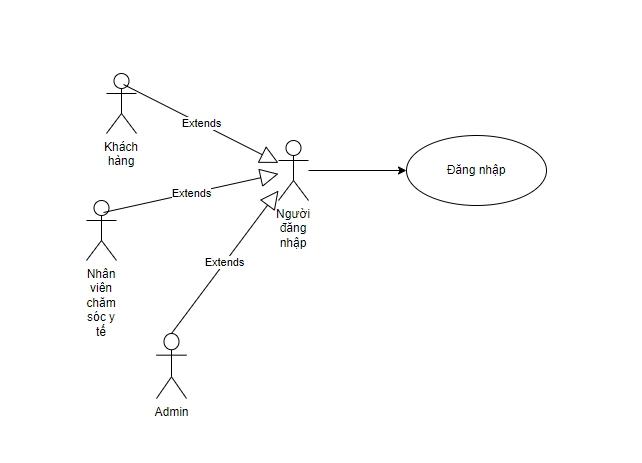
### Sơ đồ Usecase nghiệp vụ

  
Hình 2.1 Sơ đồ cơ Use case tổng quát

## Mô hình hóa quy trình nghiệp vụ

## Mô hình hóa chức năng

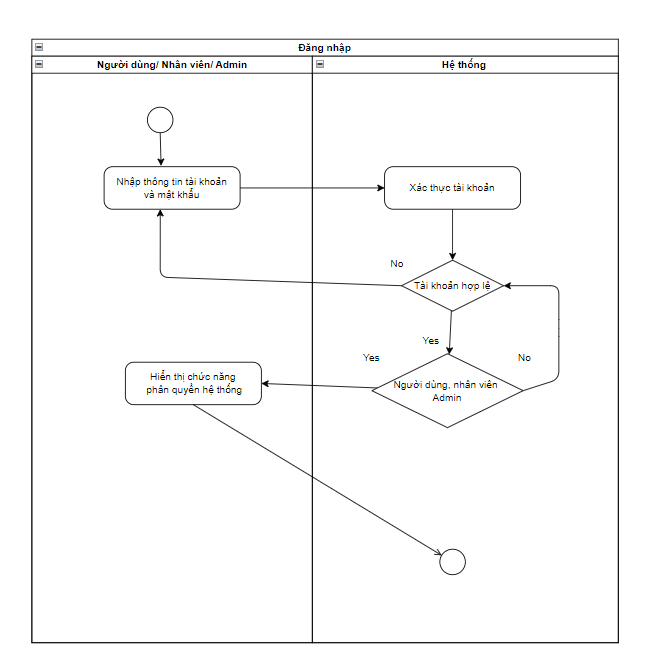
### Đăng nhập

  
Hình 2.2 Use case đăng nhập

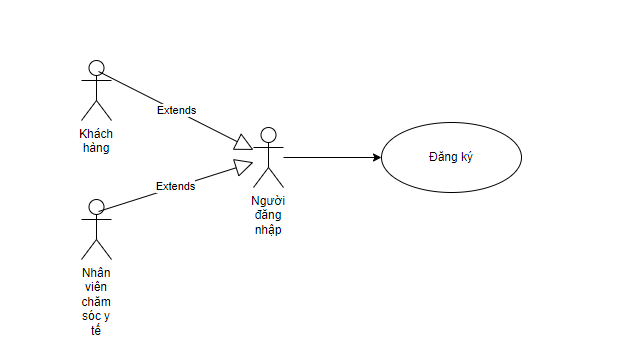
#### 2.3.1.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Đăng nhập |
| Tóm tắt | Người dùng nhập thông tin đăng nhập để truy cập ứng dụng di động. |
| Tác nhân | Admin, Nhân viên chăm sóc y tế, Khách hàng |
| Use case liên quan | Quên mật khẩu, Đăng ký tài khoản mới |
| Dòng sự kiện chính | Người dùng mở ứng dụng -> Chọn mục đăng nhập -> Nhập thông tin đăng nhập -> Ấn Đăng nhập |
| Dòng điều kiện phụ | Nếu thông tin đăng nhập không chính xác, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và đã có tài khoản đăng nhập |
| Hậu điều kiện | Đăng nhập thành công -> người dùng được chuyển đến màn hình chính của ứng dụng theo vai trò  * Đăng nhập không thành công ***->*** người dùng được nguyên màng hình đăng nhập và thông báo lỗi |

#### 2.3.1.2 Sơ đồ hoạt động

  
Hình 2.3 Sơ đồ hoạt động Use case đăng nhập

### Đăng ký tài khoản

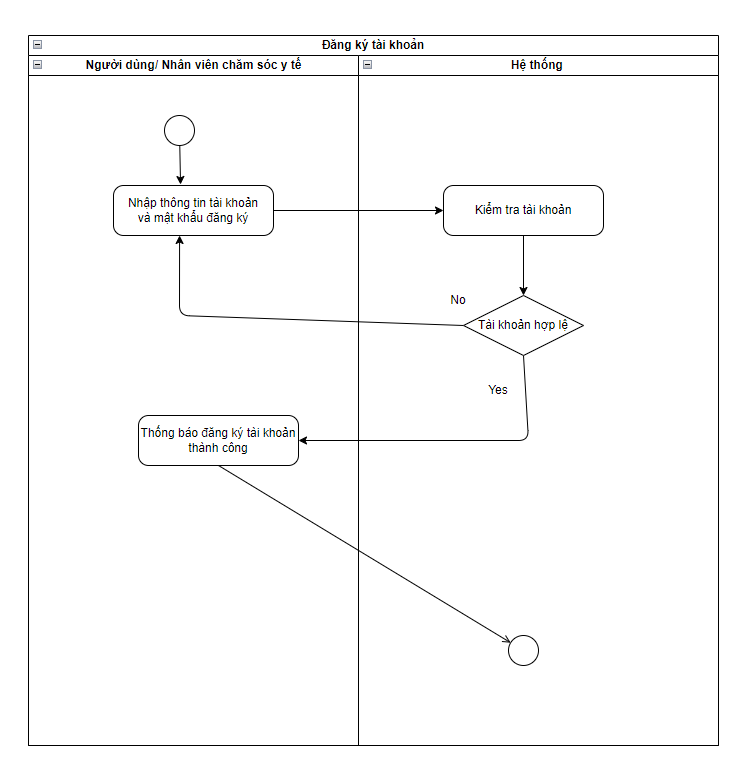


Hình 2.4 Use case đăng ký

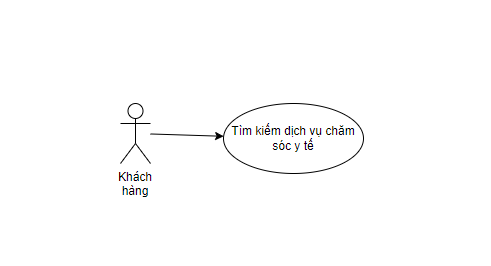
#### 2.3.2.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Đăng ký tài khoản |
| Tóm tắt | Người dùng tạo tài khoản mới để truy cập ứng dụng di động. |
| Tác nhân | Khách hàng, nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập trên ứng dụng di động |
| Dòng sự kiện chính | Người dùng mở ứng dụng -> Chọn mục đăng ký -> Nhập thông tin cá nhân -> Ấn Đăng ký |
| Dòng điều kiện phụ | Nếu thông tin cá nhân không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và chưa có tài khoản |
| Hậu điều kiện | * Đăng ký thành công -> người dùng được chuyển đến màn hình chính của ứng dụng theo vai trò * Đăng ký không thành công -> người dùng được nguyên màng hình đăng nhập và thông báo lỗi |

#### 2.3.2.2 Sơ đồ hoạt động

  
Hình 2.5 Sơ đồ hoạt động Use case đăng ký tài khoản

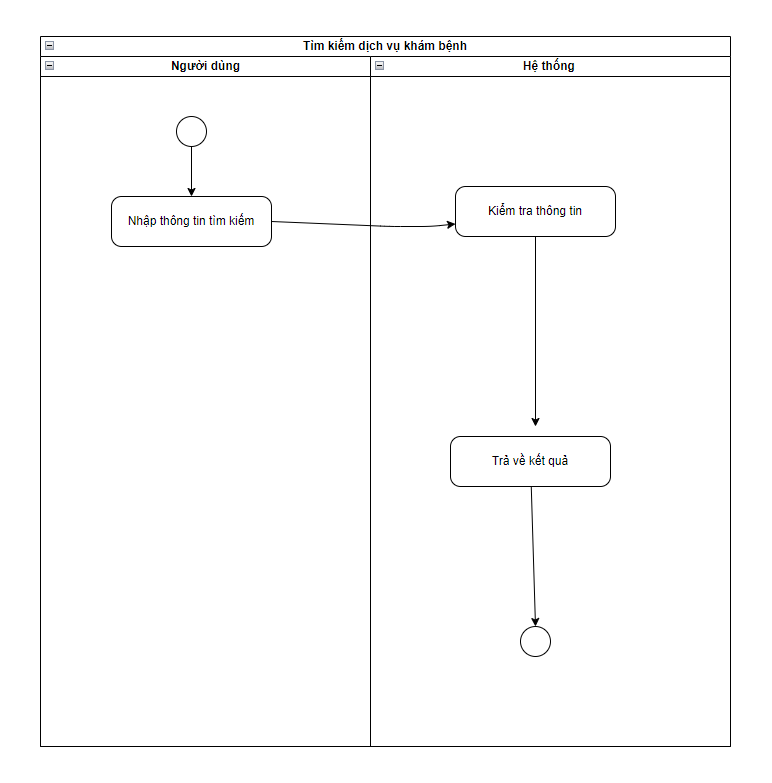
### Tìm kiếm dịch vụ chăm sóc y tế

  
Hình 2.6 Use case tìm kiếm dịch vụ chăm sóc y tế

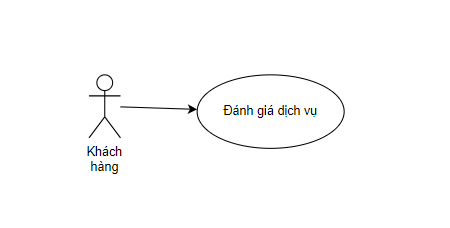
#### 2.3.3.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Tìm kiếm dịch vụ chăm sóc y tế |
| Tóm tắt | Người dùng tìm kiếm dịch vụ chăm sóc y tế trên ứng dụng di động |
| Tác nhân | Khách hàng |
| Use case liên quan | Đăng nhập |
| Dòng sự kiện chính | Người dùng mở ứng dụng -> Chọn mục tìm kiếm -> Nhập từ khóa tìm kiếm -> Ấn Tìm kiếm |
| Dòng điều kiện phụ | Nếu không tìm thấy dịch vụ phù hợp, hiển thị thông báo “Không tìm thấy dịch vụ” |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và đã có tài khoản đăng nhập |
| Hậu điều kiện | • Tìm kiếm thành công -> người dùng được chuyển đến màn hình hiển thị kết quả tìm kiếm• Tìm kiếm không thành công -> người dùng được nguyên màng hình tìm kiếm và thông báo “Không tìm thấy dịch vụ” |

#### 2.3.3.2 Sơ đồ hoạt động

  
Hình 2.7 Sơ đồ hoạt động Use case tìm kiếm dịch vụ khám bệnh

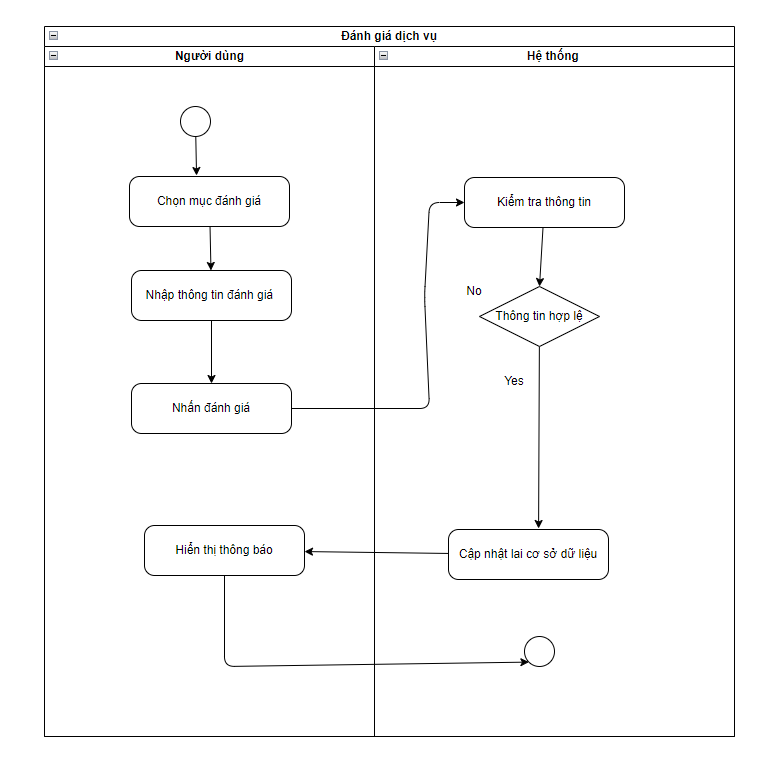
### Đánh giá dịch vụ

  
Hình 2.8 Use case đánh giá dịch vụ

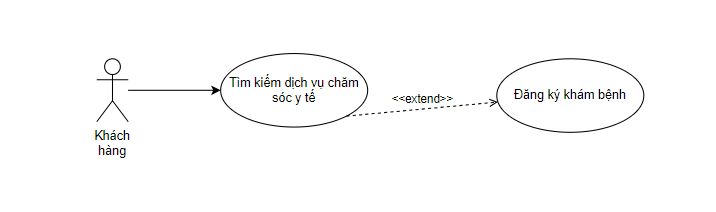
#### 2.3.4.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Đánh giá dịch vụ |
| Tóm tắt | Người dùng đánh giá chất lượng dịch vụ chăm sóc y tế mà họ đã sử dụng. |
| Tác nhân | Khách hàng |
| Use Case liên quan | Đăng nhập |
| Dòng sự kiện chính | Người dùng mở ứng dụng -> Chọn mục đánh giá -> Nhập thông tin đánh giá -> Ấn Đánh giá |
| Dòng điều kiện phụ | Nếu thông tin đánh giá không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và người dùng đã đăng nhập |
| Hậu điều kiện | * Đánh giá thành công -> người dùng được chuyển đến màn hình chính của ứng dụng theo vai trò * Đánh giá không thành công -> người dùng được nguyên màng hình đánh giá và thông báo lỗi |

#### 2.3.4.2 Sơ đồ hoạt động

  
Hình 2.9 Sơ đồ hoạt động Use case đánh giá dịch vụ

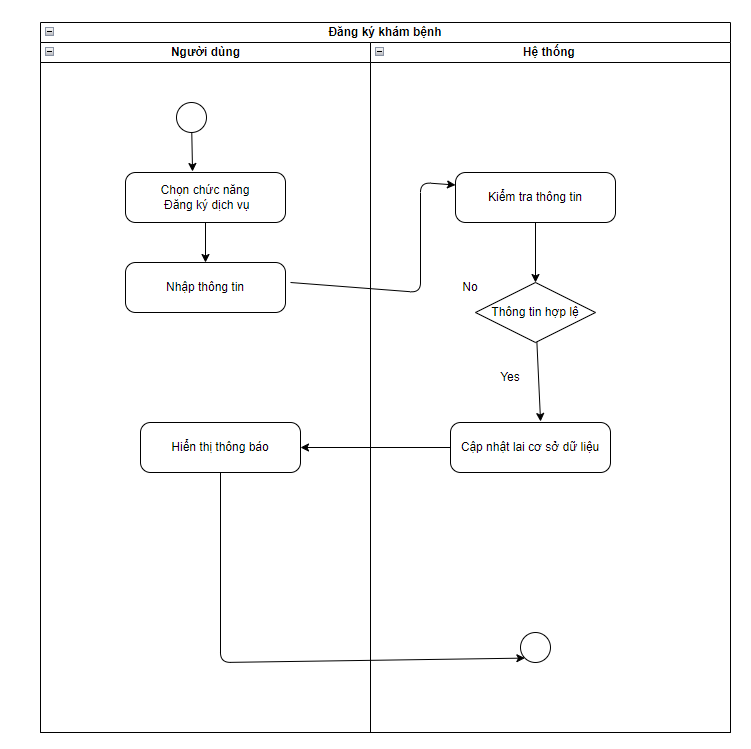
### Đăng ký khám bệnh

  
Hình 2.10 Use case Đăng ký khám bệnh

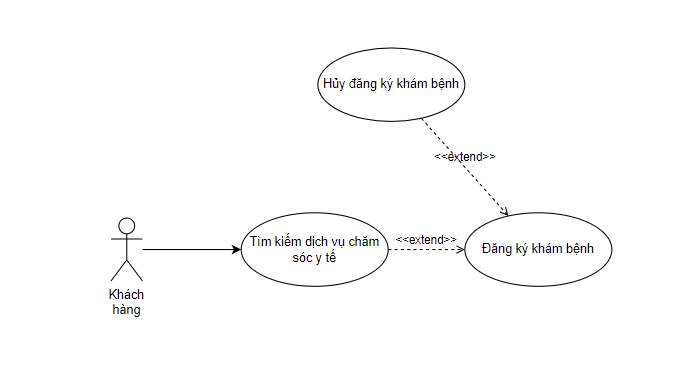
#### 2.3.5.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Đăng ký khám bệnh |
| Tóm tắt | Người dùng đăng ký sử dụng các dịch vụ chăm sóc y tế mà họ đã tìm thấy trong quá trình tìm kiếm. |
| Tác nhân | Khách hàng |
| Use Case liên quan | Đăng nhập, Tìm kiếm dịch vụ chăm sóc y tế |
| Dòng sự kiện chính | Người dùng mở ứng dụng -> Chọn dịch vụ chăm sóc y tế -> Hiển thị thông tin chi tiết về dịch vụ -> Nhập thông tin cá nhân và lịch hẹn -> Ấn Đăng ký |
| Dòng điều kiện phụ | Nếu thông tin cá nhân hoặc lịch hẹn không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt, người dùng đã đăng nhập và đã tìm thấy dịch vụ chăm sóc y tế mà họ muốn đăng ký |
| Hậu điều kiện | * Đăng ký thành công -> người dùng được chuyển đến màn hình chính của ứng dụng theo vai trò * Đăng ký không thành công -> người dùng được nguyên màng hình đăng ký và thông báo lỗi |

#### 2.3.5.2 Sơ đồ hoạt động

  
Hình 2.11 Sơ đồ hoạt động Use case đăng ký khám bệnh

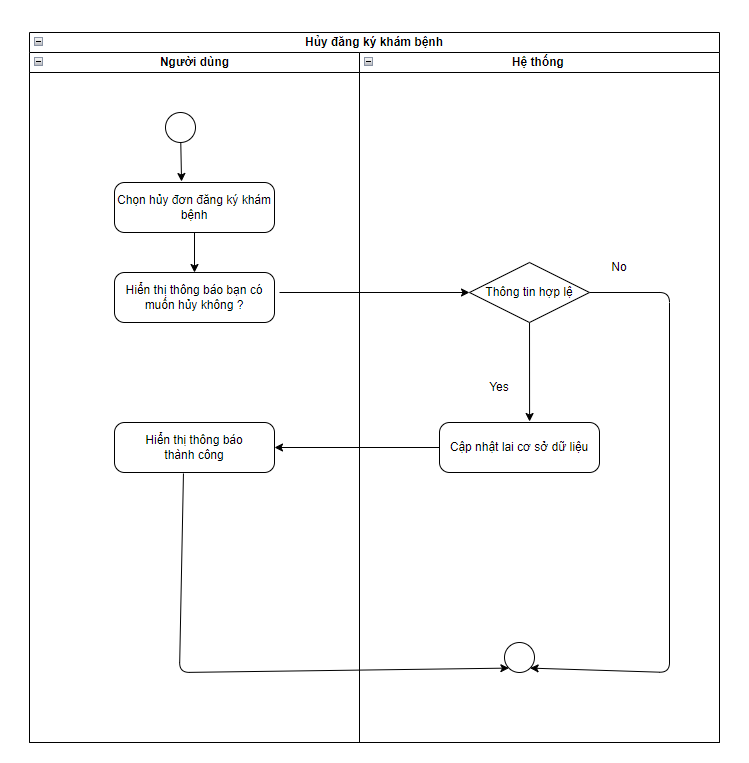
### Hủy đăng ký khám bệnh

  
Hình 2.12 Use case Hủy đăng ký khám bệnh

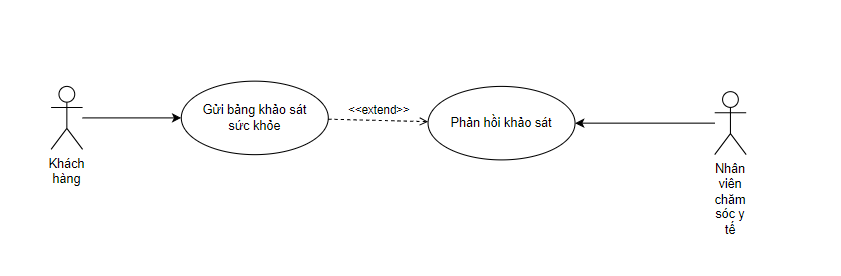
#### 2.3.6.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Hủy đăng ký khám bệnh |
| Tóm tắt | Người dùng hủy đăng ký khám bệnh tại dịch vụ chăm sóc y tế mà họ đã đăng ký trước đó. |
| Tác nhân | Khách hàng, nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập, Đăng ký khám bệnh |
| Dòng sự kiện chính | Người dùng mở ứng dụng -> Chọn dịch vụ chăm sóc y tế đã đăng ký -> Chọn Hủy đăng ký |
| Dòng điều kiện phụ | Không có |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt, người dùng đã đăng nhập và đã đăng ký khám bệnh |
| Hậu điều kiện | Hủy đăng ký thành công -> người dùng được chuyển đến màn hình chính của ứng dụng theo vai tròHủy đăng ký không thành công -> người dùng được nguyên màng hình đăng ký và thông báo lỗi |

#### 2.3.6.2 Sơ đồ hoạt động

  
Hình 2.13 Sơ đồ hoạt động Use case hủy đăng ký khám bệnh

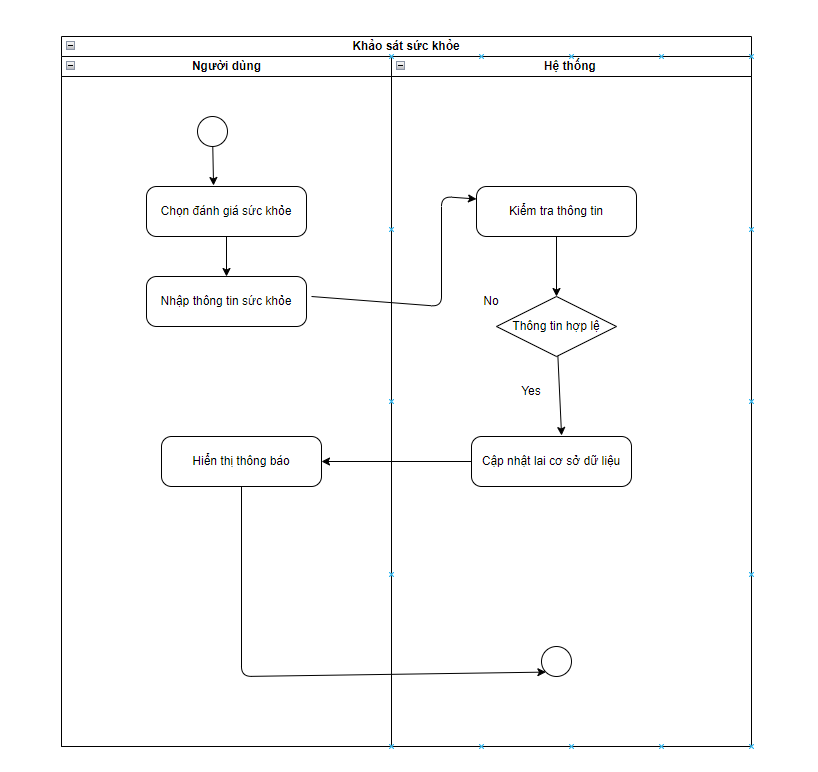
### Gửi đánh giá sức khỏe và phản hồi từ nhân viên chăm sóc y tế

  
Hình 2.14 Use case gửi bảng đánh giá sức khỏe

#### 2.3.7.1 Đặc tả Use case

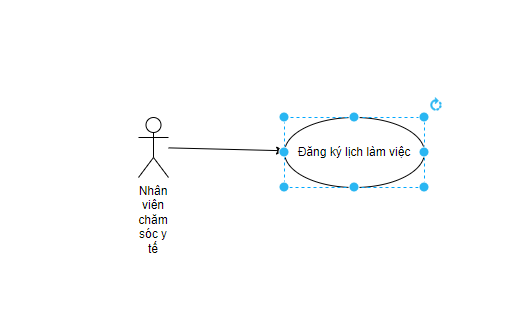
|  |  |
| --- | --- |
| Tên Use case | Hủy đăng ký khám bệnh |
| Tóm tắt | Cho phép khách hàng gửi bản đánh giá sức khỏe và nhân viên y tế đưa mô tả và các biện pháp phòng tránh. |
| Tác nhân | Khách hàng, nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập trên ứng dụng di động |
| Dòng sự kiện chính | Khách hàng mở ứng dụng -> Truy cập phần đánh giá sức khỏe -> Hoàn thành bản đánh giá ->Nhân viên y tế đăng nhập -> Truy cập phần đánh giá sức khỏe -> Đưa ra mô tả và biện pháp phòng tránh -> Cập nhật trạng thái đánh giá |
| Dòng điều kiện phụ | Nếu thông tin đánh giá không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt, khách hàng và nhân viên y tế đã đăng nhập |
| Hậu điều kiện | Đánh giá và phản hồi thành công -> khách hàng và nhân viên y tế được chuyển đến màn hình chính của ứng dụng theo vai trò,Đánh giá hoặc phản hồi không thành công -> khách hàng hoặc nhân viên y tế được nguyên màng hình đánh giá và thông báo lỗi |

#### 2.3.7.2 Sơ đồ hoạt động



Hình 2.15 Sơ đồ hoạt động Use case gửi bảng đánh giá sức khỏe

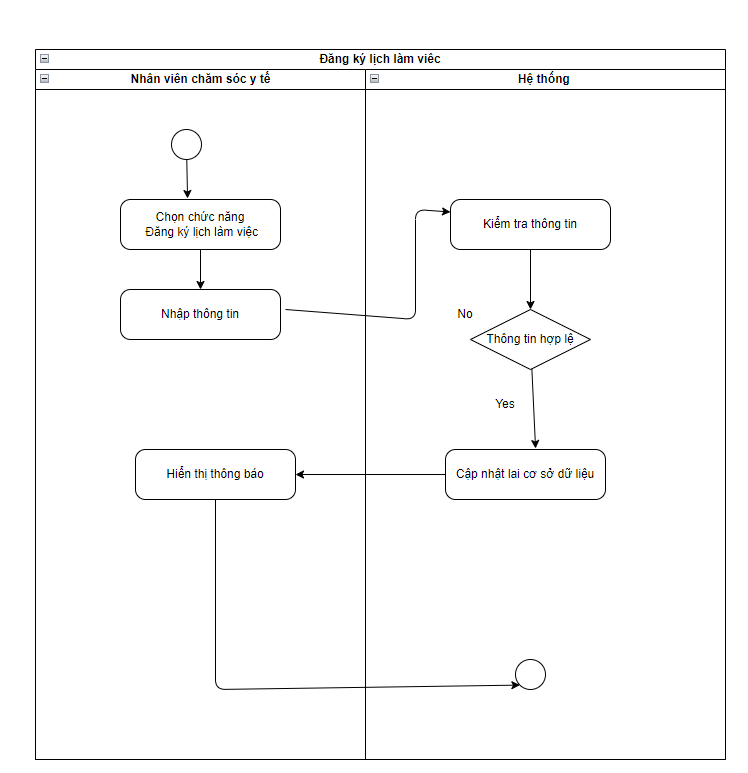
### Đăng ký lịch làm việc

  
Hình 2.16 Use case Đăng ký lịch làm việc

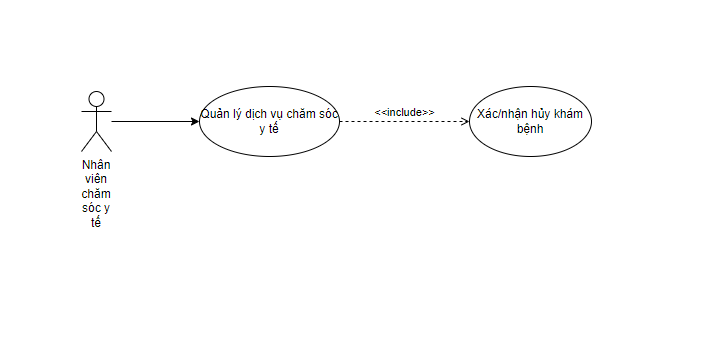
#### 2.3.8.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Đăng ký lịch làm việc |
| Tóm tắt | Cho phép nhân viên chăm sóc y tế đăng ký lịch làm việc của họ trên hệ thống. |
| Tác nhân | Nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập trên ứng dụng di động |
| Dòng sự kiện chính | Nhân viên chăm sóc y tế mở ứng dụng -> Truy cập phần quản lý lịch làm việc -> Nhập thông tin lịch làm việc -> Ấn Đăng ký |
| Dòng điều kiện phụ | Nếu thông tin lịch làm việc không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và nhân viên chăm sóc y tế đã đăng nhập |
| Hậu điều kiện | * Đăng ký thành công -> nhân viên chăm sóc y tế được chuyển đến màn hình chính của ứng dụng theo vai trò * Đăng ký không thành công -> nhân viên chăm sóc y tế được nguyên màng hình đăng ký và thông báo lỗi |

#### 2.3.8.2 Sơ đồ hoạt động

****Hình 2.17 Sơ đồ hoạt động Use case Đăng ký lịch làm việc

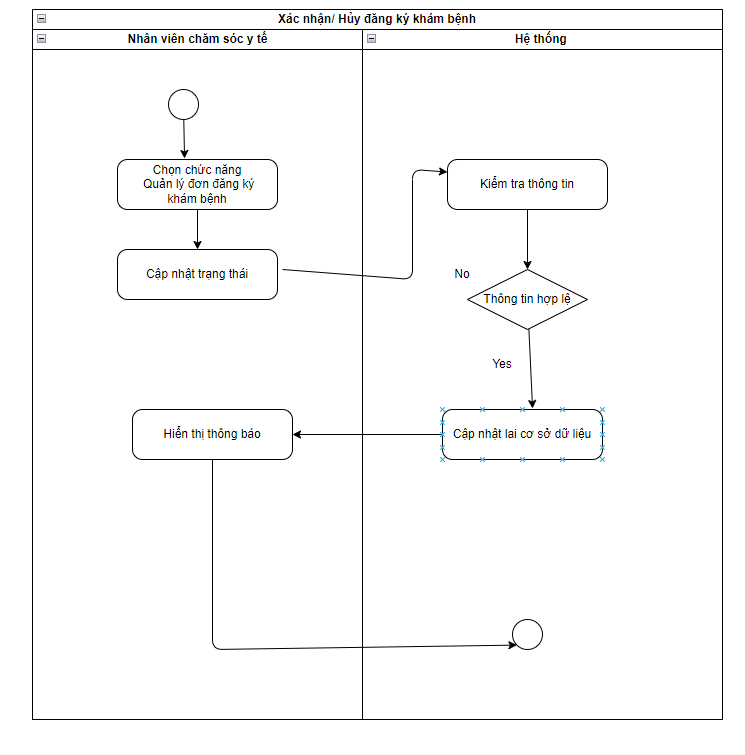
### Xác nhận/ hủy đăng ký khám bệnh

  
  
  
Hình 2.18 Use case Xác nhận/ Hủy đăng ký lịch làm việc

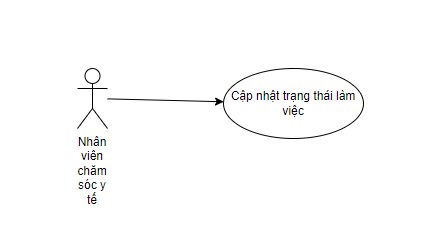
#### 2.3.9.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Nhận/hủy đăng ký khám bệnh tại dịch vụ |
| Tóm tắt | Cho phép nhân viên chăm sóc y tế nhận hoặc huỷ đăng ký dịch vụ của khách hàng. |
| Tác nhân | Khách hàng, nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập trên ứng dụng di động, Đăng ký sử dụng dịch vụ chăm sóc y tế |
| Dòng sự kiện chính | Nhân viên chăm sóc y tế mở ứng dụng -> Truy cập phần quản lý đăng ký dịch vụ -> Xem danh sách các đăng ký dịch vụ chờ xử lý -> Chọn nhận hoặc huỷ đăng ký dịch vụ |
| Dòng điều kiện phụ | Nếu không có đăng ký dịch vụ nào chờ xử lý, hiển thị thông báo “Không có đăng ký dịch vụ nào chờ xử lý” |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt, nhân viên chăm sóc y tế đã đăng nhập |
| Hậu điều kiện | Nhận/hủy đăng ký thành công -> nhân viên chăm sóc y tế được chuyển đến màn hình chính của ứng dụng theo vai tròNhận/hủy đăng ký không thành công -> nhân viên chăm sóc y tế được nguyên màng hình quản lý đăng ký và thông báo lỗi |

#### 2.3.9.2 Sơ đồ hoạt động

  
Hình 2.19 Sơ đồ hoạt động Use case Xác nhận/ Hủy đăng ký lịch làm việc

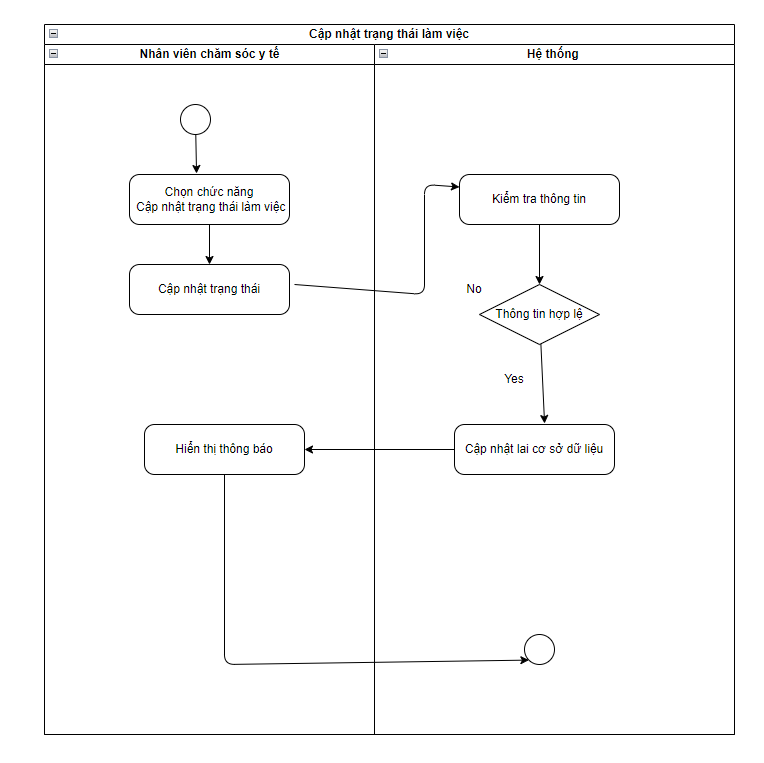
### Cập nhật trạng thái làm việc

  
Hình 2.20 Use case cập nhật trạng thái làm việc

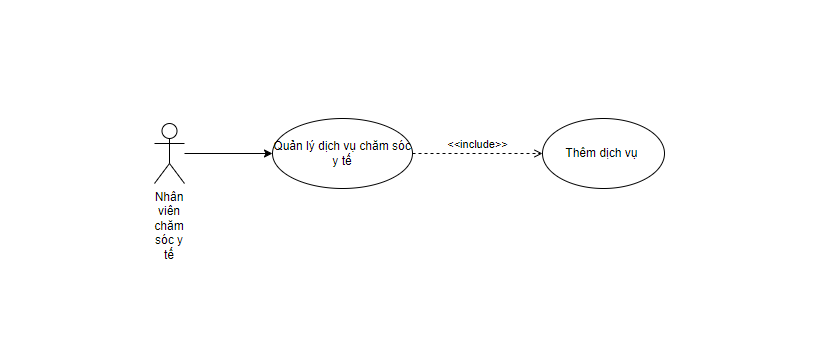
#### 2.3.10.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Cập nhật trạng thái làm việc |
| Tóm tắt | Cho phép nhân viên chăm sóc y tế cập nhật trạng thái làm việc của họ trên hệ thống. |
| Tác nhân | Nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập trên ứng dụng di động |
| Dòng sự kiện chính | Nhân viên chăm sóc y tế mở ứng dụng -> Truy cập phần quản lý trạng thái làm việc -> Chọn trạng thái làm việc -> Ấn Cập nhật |
| Dòng điều kiện phụ | Nếu trạng thái làm việc không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và nhân viên chăm sóc y tế đã đăng nhập |
| Hậu điều kiện | Cập nhật thành công -> nhân viên chăm sóc y tế được chuyển đến màn hình chính của ứng dụng theo vai tròCập nhật không thành công -> nhân viên chăm sóc y tế được nguyên màng hình quản lý trạng thái và thông báo lỗi |

#### 2.3.10.2 Sơ đồ hoạt động

  
Hình 2.21 Sơ đồ hoạt động Use case cập nhật trạng thái làm việc

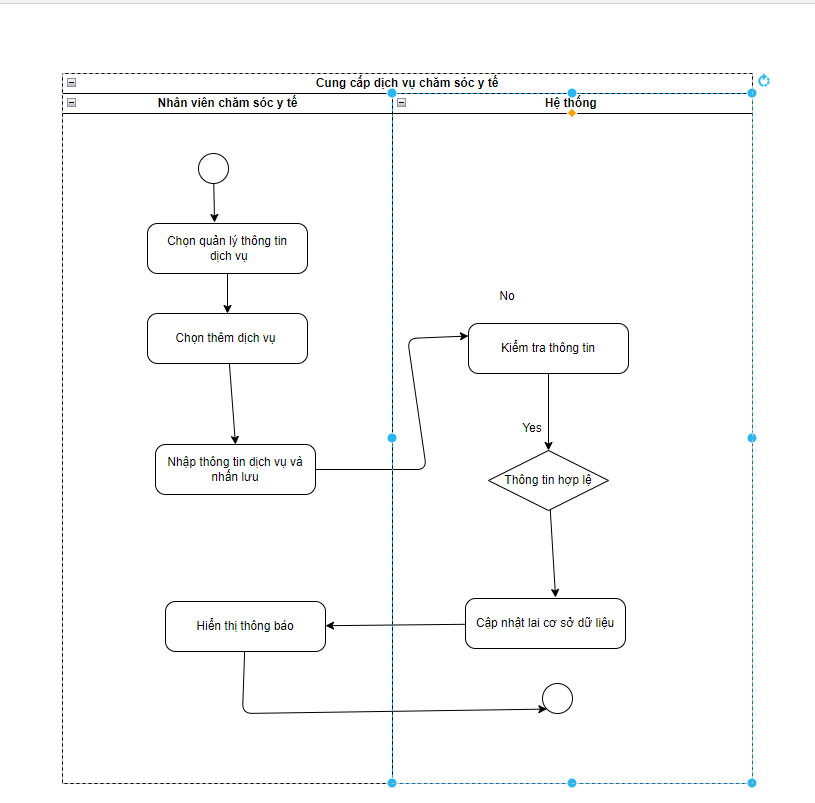
### Cung Cấp dịch vụ chăm sóc y tế

  
Hình 2.22 Use case cung cấp dịch vụ chăm sóc y tế

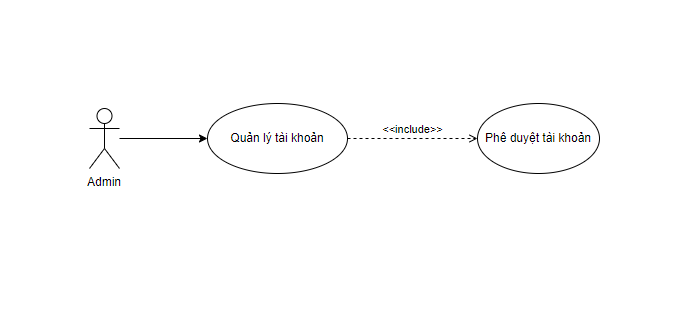
#### 2.3.11.1 Đặc tả Use case

|  |  |
| --- | --- |
| Tên Use case | Cung cấp dịch vụ chăm sóc y tế |
| Tóm tắt | Cho phép nhân viên chăm sóc y tế thêm dịch vụ chăm sóc y tế |
| Tác nhân | Nhân viên chăm sóc y tế |
| Use Case liên quan | Đăng nhập trên ứng dụng di động |
| Dòng sự kiện chính | Nhân viên chăm sóc y tế mở ứng dụng -> Truy cập phần quản lý dịch vụ chăm sóc y tế-> Chọn thêm dịch vụ -> Nhập thông tin dịch vụ -> Nhấn lưu |
| Dòng điều kiện phụ | Nếu thông tin nhập không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Ứng dụng đã được cài đặt và nhân viên chăm sóc y tế đã đăng nhập |
| Hậu điều kiện | Cập nhật thành công -> Hiển chị thông báo thêm thành côngCập nhật không thành công -> nhân viên chăm sóc y tế được nguyên màng hình và thông báo lỗi |

#### 2.3.11.2 Sơ đồ hoạt động

  
Hình 2.23 Sơ đồ hoạt động Use case cung cấp dịch vụ chăm sóc y tê

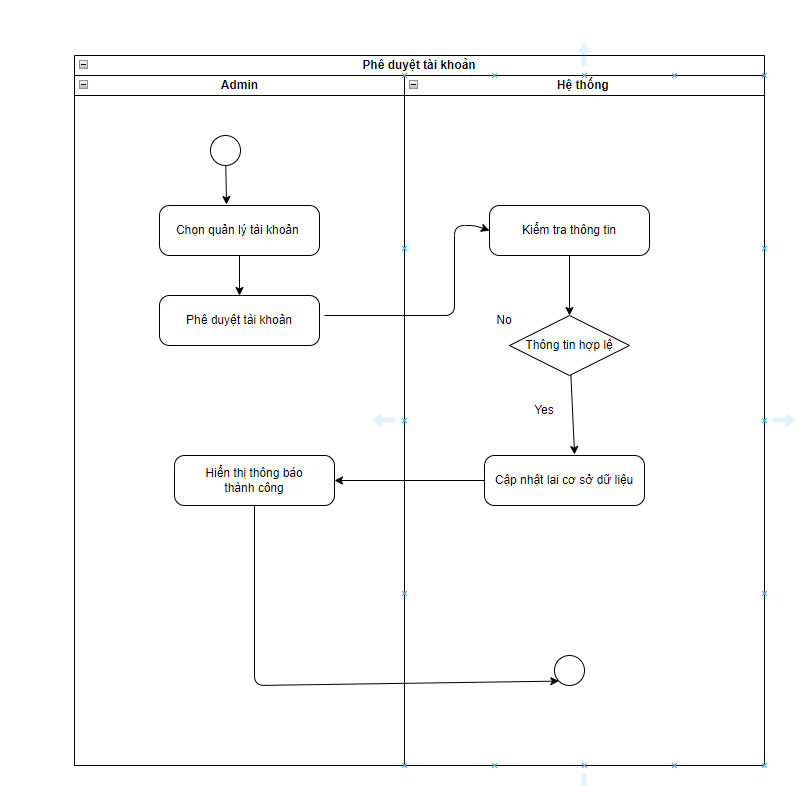
### Phê duyệt tài khoản

  
Hình 2.24 Use case phên duyệt tài khoản

#### 2.3.12.1 Đặc tả Use case

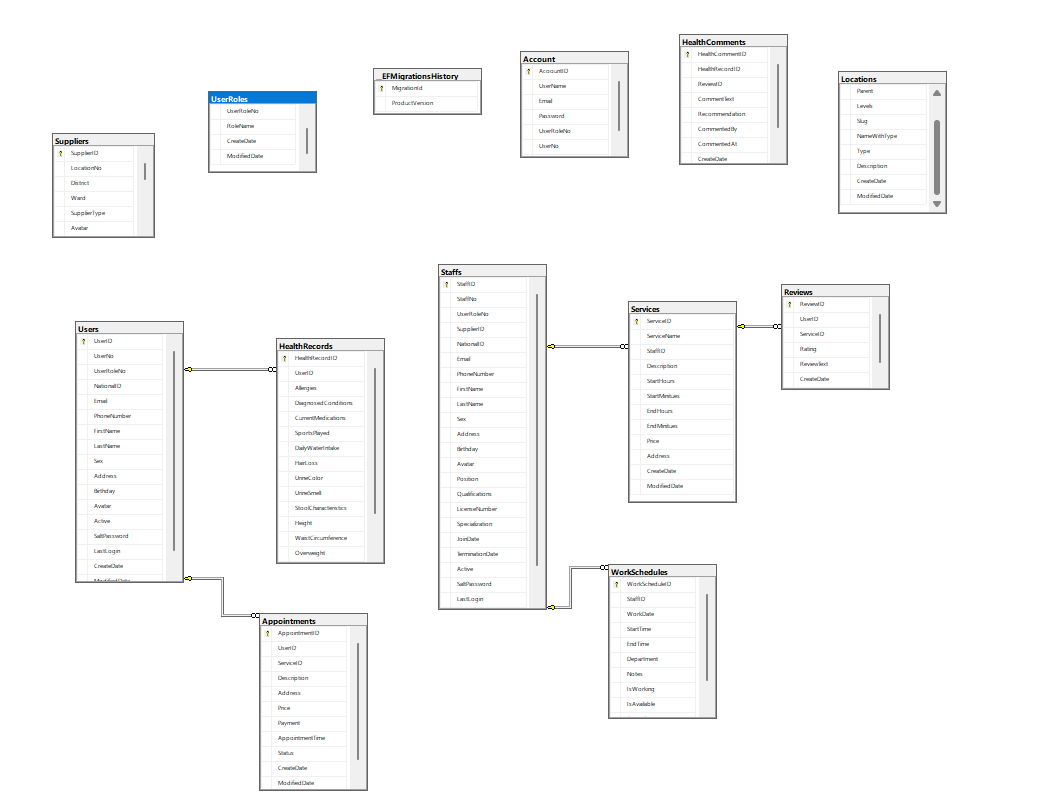
|  |  |
| --- | --- |
| Tên Use case | Phê duyệt tài khoản |
| Tóm tắt | Cho phép admin cấp quyền truy cập cho tài khoản nhân viên chăm sóc y tế |
| Tác nhân | Admin quản trị hệ thống |
| Use Case liên quan | Đăng nhập trên ứng dụng di động |
| Dòng sự kiện chính | Admin mở ứng dụng -> Truy cập phần quản lý tài khoản-> Chọn duyệt tài khoản |
| Dòng điều kiện phụ | Nếu tài khoản nhân viên chăm sóc y tế không hợp lệ, hiển thị thông báo lỗi |
| Điều kiện tiên quyết | Phải có tài khoản nhân viên chăm sóc tế |
| Hậu điều kiện | Cập nhật thành công -> tài khoản nhân viên chăm sóc y tế được truy cậpCập nhật không thành công -> tài khoản nhân viên chăm sóc y tế không được phép truy cập |

#### 2.3.12.2 Sơ đồ hoạt động

  
Hình 2.25 Sơ đồ hoạt động Use case phê duyệt tài khoản

## Thiết kế cơ sở dữ liệu

## Mô hình dữ liệu quan hệ

  
Hình 2.26 Cơ sở dữ liệu hệ thống

### Ý nghĩa của các bảng

#### 2.4.2.1 Bảng Appointments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | AppointmentID | int | PRIMARY KEY | Mã số lịch hẹn |
| 2 | ServiceNo | int | NULL | Mã số dịch vụ |
| 3 | UserNo | int | NULL | Mã số người dùng |
| 4 | LocationNo | int | NULL | Tỉnh/ thành |
| 5 | District | int | NULL | Quận/huyện |
| 6 | Ward | int | NULL | Phường/Xã |
| 7 | Address | nvarchar(255) | NULL | Địa chỉ |
| 8 | AppointmentTime | datetime | NULL | Thời gian lịch hẹn |
| 9 | Status | nvarchar(50) | NULL | Trạng thái |
| 10 | CreateDate | datetime | NULL | Ngày tạo lịch hẹn |
| 11 | ModifiedDate | datetime | NULL | Ngày sửa đổi lịch hẹn |

#### 2.4.2.2 Bảng HealthComments

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | HealthCommentID | int | PRIMARY KEY | Mã số bình luận |
| 2 | HealthRecordID | int | NULL | Mã số hồ sơ sức khỏe |
| 3 | ReviewID | int | NULL | Mã số đánh giá |
| 4 | CommentText | nvarchar(max) | NULL | Nội dung bình luận |
| 5 | Recommendation | nvarchar(max) | NULL | Đề xuất của người dùng |
| 6 | CommentedBy | int | NULL | Mã số người dùng |
| 7 | CommentedAt | datetime | NULL | Thời gian người dùng bình luận |
| 8 | CreateDate | datetime | NULL | Ngày tạo bình luận |
| 9 | ModifiedDate | datetime | NULL | Ngày sửa đổi bình luận |

#### 2.4.2.3 Bảng HealthyRecords

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | HealthRecordID | int | PRIMARY KEY | Mã số hồ sơ sức khỏe |
| 2 | Allergies | bit | NULL | Có dị ứng không |
| 3 | DiagnosedConditions | bit | NULL | Có chẩn đoán bệnh không |
| 4 | CurrentMedications | bit | NULL | Có đang dùng thuốc không |
| 5 | SportsPlayed | bit | NULL | Có chơi thể thao không |
| 6 | DailyWaterIntake | int | NULL | Lượng nước uống hàng ngày |
| 7 | HairLoss | bit | NULL | Có rụng tóc không |
| 8 | UrineColor | nvarchar(255) | NULL | Màu nước tiểu |
| 9 | UrineSmell | bit | NULL | Mùi nước tiểu |
| 10 | StoolCharacteristics | nvarchar(max) | NULL | Đặc điểm phân |
| 11 | Height | decimal(5, 2) | NULL | Chiều cao |
| 12 | WaistCircumference | decimal(5, 2) | NULL | Vòng eo |
| 13 | Overweight | bit | NULL | Có thừa cân không |
| 14 | CreateBy | int | NULL | Mã số người dùng |
| 15 | CreateDate | datetime | NULL | Ngày tạo hồ sơ sức khỏe |
| 16 | ModifiedDate | datetime | NULL | Ngày sửa đổi hồ sơ sức khỏe |

#### 2.4.2.4 Bảng Locations

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | LocationID | int | PRIMARY KEY | Mã số địa điểm |
| 2 | Name | nvarchar(50) | NULL | Tên địa điểm |
| 3 | Parent | int | NULL | Mã số địa điểm cha |
| 4 | Levels | int | NULL | Số cấp của địa điểm  (1: Tỉnh  /thành, 2: Quận/huyện. 3: Phường/xã) |
| 5 | Slug | nvarchar(100) | NULL | Viết tắt |
| 6 | NameWithType | nvarchar(100) | NULL | Tên địa điểm với loại |
| 7 | Type | int | NULL | Loại địa điểm |
| 8 | Description | nvarchar(max) | NULL | Mô tả địa điểm |
| 9 | CreateDate | datetime | NULL | Ngày tạo địa điểm |
| 10 | ModifiedDate | datetime | NULL | Ngày sửa đổi địa điểm |

#### 2.4.2.5 Bảng Reviews

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | ReviewID | int | PRIMARY KEY | Mã số đánh giá |
| 2 | UserID | int | NULL | Mã số người dùng, liên kết đến bảng người dùng |
| 3 | ServiceID | int | NULL | Mã số dịch vụ, liên kết đến bảng dịch vụ |
| 4 | Rating | int | NULL | Đánh giá của người dùng |
| 5 | ReviewText | nvarchar(max) | NULL | Nội dung đánh giá |
| 6 | CreateDate | datetime | NULL | Ngày tạo đánh giá |
| 7 | ModifiedDate | datetime | NULL | Ngày sửa đổi đánh giá |

#### 2.4.2.6 Bảng Services

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | ServiceID | int | PRIMARY KEY | Mã số dịch vụ |
| 2 | ServiceName | nvarchar(255) | NOT NULL | Tên dịch vụ |
| 3 | Description | nvarchar(max) | NULL | Mô tả dịch vụ |
| 4 | StartHours | decimal(10, 2) | NULL | Giờ mở cửa |
| 5 | Price | decimal(10, 2) | NULL | Giá dịch vụ |
| 6 | StartMinitues | decimal(10, 2) | NULL | Phút mở cửa |
| 7 | EndHours | decimal(10, 2) | NULL | Giờ đóng cửa |
| 8 | EndMinitues | decimal(10, 2) | NULL | Phút đóng cửa |
| 9 | Address | nvarchar(255) | NULL | Địa chỉ dịch vụ |
| 10 | CreateDate | datetime | NULL | Ngày tạo dịch vụ |
| 11 | ModifiedDate | datetime | NULL | Ngày sửa đổi dịch vụ |

#### 2.4.2.7 Bảng Staffs

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | StaffID | int | PRIMARY KEY | Mã số nhân viên |
| 2 | UserRoleNo | int | NULL | Mã số vai trò người dùng |
| 3 | SupplierID | int | NULL | Mã số nhà cung cấp |
| 4 | NationalID | nchar(255) | NULL | Chứng minh nhân dân |
| 5 | UserName | nvarchar(255) | NOT NULL | Tên người dùng |
| 6 | Password | nvarchar(255) | NOT NULL | Mật khẩu |
| 7 | Email | nvarchar(255) | NULL | Email |
| 8 | PhoneNumber | nvarchar(15) | NULL | Số điện thoại |
| 9 | FirstName | nvarchar(50) | NULL | Tên |
| 10 | LastName | nvarchar(50) | NULL | Họ |
| 11 | Sex | nvarchar(10) | NULL | Giới tính |
| 12 | Address | nvarchar(255) | NULL | Địa chỉ |
| 13 | Birthday | datetime | NULL | Ngày sinh |
| 14 | Avatar | nvarchar(max) | NULL | Ảnh đại diện |
| 15 | LocationNo | int | NULL | Tỉnh/thành |
| 16 | District | int | NULL | Quận/huyện |
| 17 | Ward | int | NULL | Phường/xã |
| 18 | Position | nvarchar(50) | NULL | Chức vụ |
| 19 | Qualifications | nvarchar(255) | NULL | Trình độ |
| 20 | LicenseNumber | nchar(255) | NULL | Số giấy phép |
| 21 | Specialization | nvarchar(100) | NULL | Chuyên môn |
| 22 | JoinDate | date | NULL | Ngày vào làm |
| 23 | TerminationDate | date | NULL | Ngày nghỉ việc |
| 24 | Active | bit | NULL | Trạng thái hoạt động |
| 25 | SaltPassword | nchar(255) | NULL | Salt mật khẩu |
| 26 | LastLogin | datetime | NULL | Thời gian đăng nhập gần nhất |
| 27 | CreateDate | datetime | NULL | Ngày tạo |
| 28 | ModifiedDate | datetime | NULL | Ngày sửa đổi |

#### 2.4.2.8 bảng WorkSchedules

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | WorkScheduleID | int | PRIMARY KEY | Mã số lịch làm việc |
| 2 | StaffID | int | NULL | Mã số nhân viên, liên kết đến bảng nhân viên |
| 3 | WorkDate | date | NULL | Ngày làm việc |
| 4 | StartTime | time(7) | NULL | Giờ bắt đầu làm việc |
| 5 | EndTime | time(7) | NULL | Giờ kết thúc làm việc |
| 6 | Department | nvarchar(100) | NULL | Phòng ban |
| 7 | Notes | nvarchar(max) | NULL | Ghi chú |
| 8 | IsWorking | bit | NULL | Có làm việc hay không |
| 9 | IsAvailable | bit | NULL | Có sẵn sàng làm việc hay không |
| 10 | CreateDate | datetime | NULL | Ngày tạo lịch làm việc |
| 11 | ModifiedDate | datetime | NULL | Ngày sửa đổi lịch làm việc |

#### 2.4.2.9 Bảng Suppliers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | SupplierID | int | PRIMARY KEY | Mã số nhà cung cấp |
| 2 | SupplierType | nvarchar(255) | NULL | Loại nhà cung cấp |
| 3 | Avatar | nvarchar(max) | NULL | Ảnh đại diện |
| 4 | TIN | nchar(255) | NULL | Mã số thuế |
| 5 | SupplierName | nvarchar(255) | NOT NULL | Tên nhà cung cấp |
| 6 | PhoneNumber | nvarchar(15) | NULL | Số điện thoại |
| 7 | Fax | nvarchar(15) | NULL | Số fax |
| 8 | Website | nvarchar(255) | NULL | Trang web |
| 9 | Address | nvarchar(255) | NULL | Địa chỉ |
| 10 | Phone | nvarchar(15) | NULL | Số điện thoại |
| 11 | Email | nvarchar(255) | NULL | Email |
| 12 | ContactPerson | nvarchar(100) | NULL | Người liên hệ |
| 13 | TaxID | nvarchar(20) | NULL | Mã số thuế |
| 14 | RegistrationDate | date | NULL | Ngày đăng ký |
| 15 | LicenseNumber | nvarchar(50) | NULL | Số giấy phép |
| 16 | ServicesProvided | nvarchar(max) | NULL | Dịch vụ cung cấp |
| 17 | Certification | nvarchar(max) | NULL | Chứng nhận |
| 18 | Notes | nvarchar(max) | NULL | Ghi chú |
| 19 | CreateDate | datetime | NULL | Ngày tạo |
| 20 | ModifiedDate | datetime | NULL | Ngày sửa đổi |

#### 2.4.2.10 Bảng UserRoles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | UserRoleID | int | PRIMARY KEY | Mã số vai trò người dùng |
| 2 | UserRoleNo | nchar(10) | NOT NULL | Mã số vai trò người dùng |
| 3 | RoleName | nvarchar(50) | NOT NULL | Tên vai trò người dùng |
| 4 | CreateDate | datetime | NULL | Ngày tạo vai trò người dùng |
| 5 | ModifiedDate | datetime | NULL | Ngày sửa đổi vai trò người dùng |

#### 2.4.2.14 Bảng Users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Khóa chính** | **Ý nghĩa** |
| 1 | UserID | int | PRIMARY KEY | Mã số người dùng |
| 2 | UserRoleNo | int | NULL | Mã số vai trò người dùng, liên kết đến bảng vai trò người dùng |
| 3 | NationalID | nchar(255) | NULL | Chứng minh nhân dân |
| 4 | UserName | nvarchar(255) | NOT NULL | Tên người dùng |
| 5 | Email | nvarchar(255) | NULL | Email |
| 6 | Password | nvarchar(255) | NOT NULL | Mật khẩu |
| 7 | PhoneNumber | nvarchar(15) | NULL | Số điện thoại |
| 8 | FirstName | nvarchar(50) | NULL | Tên |
| 9 | LastName | nvarchar(50) | NULL | Họ |
| 10 | Sex | nvarchar(10) | NULL | Giới tính |
| 11 | Address | nvarchar(255) | NULL | Địa chỉ |
| 12 | Birthday | datetime | NULL | Ngày sinh |
| 13 | Avatar | nvarchar(max) | NULL | Ảnh đại diện |
| 14 | Active | bit | NULL | Trạng thái hoạt động |
| 15 | SaltPassword | nchar(255) | NULL | Salt mật khẩu |
| 16 | LastLogin | datetime | NULL | Thời gian đăng nhập gần nhất |
| 17 | CreateDate | datetime | NULL | Ngày tạo |
| 18 | ModifiedDate | datetime | NULL | Ngày sửa đổi |

## Kết chương

Trong chương này, nhóm chúng em đã hoàn thành xây dựng hệ thống đăng ký dịch vụ chăm sóc y tế trực tuyến cho ứng di động. Hệ thống đã đáp ứng đầy đủ các chức năng đã đề ra, bao gồm:

* Đăng ký khám chữa bệnh
* Tìm kiếm thông tin về các cơ sở y tế
* Gửi bảnh đánh giá sức khỏe
* Đánh giá dịch vụ
* Xác nhận đăng ký khám bệnh
* Cung cấp dịch vụ chăm sóc y tế
* Phê duyệt tài khoản

Hệ thống đã được triển khai trên ứng dụng di động, giúp người dân có thể truy cập và sử dụng mọi lúc, mọi nơi.

Trong chương tiếp theo, nhóm chúng em sẽ tiếp tục nghiên cứu và phát triển các tính năng mới cho hệ thống, nhằm nâng cao chất lượng dịch vụ và đáp ứng nhu cầu ngày càng cao của người dân.

Nhóm chúng em tin rằng với những tính năng mới này, hệ thống sẽ trở thành một giải pháp chăm sóc sức khỏe hiệu quả và tiện lợi cho người dân.

# CỞ SỞ LÝ THUYẾT

## Giới thiệu về React Native

### Định nghĩa về React Native

  
  
Hình 3.1 Giới thiệu React Ntive

React Native (còn được gọi là RN) là một khung ứng dụng di động dựa trên JavaScript phổ biến cho phép tạo các ứng dụng di động được hiển thị nguyên bản cho iOS và Android.

Framework này cho phép tạo một ứng dụng cho các nền tảng khác nhau bằng cách sử dụng cùng một cơ sở mã (codebase).

React Native được Facebook phát hành lần đầu tiên dưới dạng một dự án mã nguồn mở vào năm 2015.

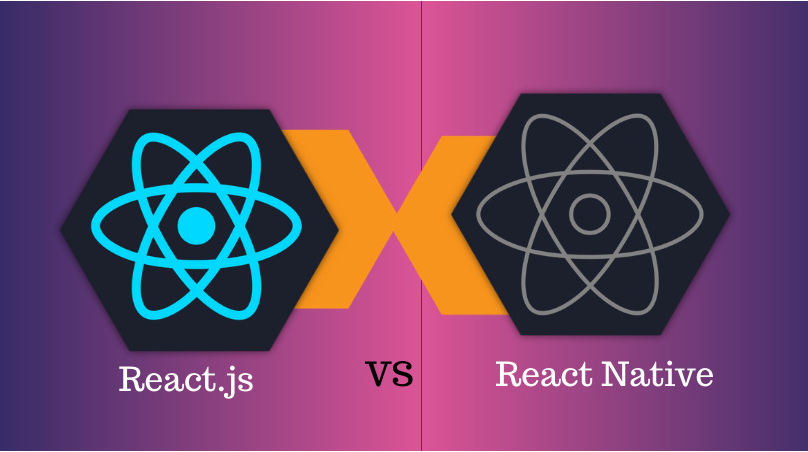
Chỉ trong vài năm, nó đã trở thành một trong những giải pháp hàng đầu được sử dụng cho mobile development. Việc phát triển React Native được sử dụng để cung cấp năng lượng cho một số ứng dụng di động hàng đầu thế giới, bao gồm Instagram, Facebook và Skype.

Có một số lý do đằng sau thành công toàn cầu của React Native.

* Thứ nhất, bằng cách sử dụng React Native, các công ty có thể tạo code chỉ một lần và sử dụng nó cho cả ứng dụng iOS và Android. Điều này nghĩa là tiết kiệm thời gian và tài nguyên rất lớn.
* Thứ hai, React Native được xây dựng dựa trên React – một thư viện JavaScript, vốn đã rất phổ biến khi framework di động được phát hành.
* Thứ ba, framework trao quyền cho frontend developer, những người trước đây chỉ có thể làm việc với các công nghệ dựa trên web, để tạo ra các ứng dụng mạnh mẽ, sẵn sàng sản xuất cho các nền tảng di động.

Thú vị là, cũng như nhiều phát minh mang tính cách mạng, React Native được phát triển như một phản ứng đối với … một sai lầm lớn về công nghệ

* React so với React Native

  
  
Hình 3.2 React.js và React Native

React Native không phải là một phiên bản React mới hơn, mặc dù React Native có sử dụng nó.

React (còn được gọi là ReactJS) là một thư viện JavaScript được sử dụng để xây dựng giao diện người dùng của một trang web. Tương tự như React Native, nó cũng được phát triển bởi nhóm kỹ sư của Facebook.

Trong khi đó**, React Native** – được cung cấp bởi React – cho phép các nhà phát triển sử dụng một tập hợp các thành phần giao diện người dùng (UI) để nhanh chóng biên dịch và khởi chạy các ứng dụng iOS và Android.

Cả React và React Native đều sử dụng hỗn hợp JavaScript và một ngôn ngữ đánh dấu đặc biệt, JSX. Tuy nhiên, cú pháp được sử dụng để hiển thị các phần tử trong các thành phần JSX khác nhau giữa React và React Native. Ngoài ra, React sử dụng một số HTML và CSS, trong khi React Native cho phép sử dụng các phần tử giao diện người dùng di động gốc.

Một ví dụ về mã từ cuộc thảo luận về Stack Overflow:

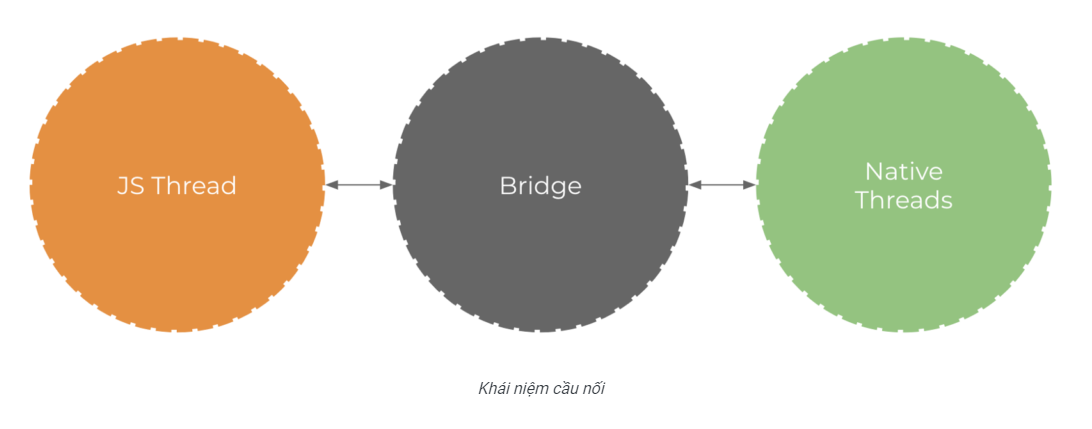
“React JSX hiển thị các thành phần giống HTML như <h1>, <p>, v.v. [Trong khi đó] react-native hiển thị các thành phần chế độ xem ứng dụng gốc như <View>, <Text>, <Image>, <ScrollView>, vì vậy không thể trực tiếp sử dụng lại mã thành phần UI trừ khi làm lại / thay thế tất cả các phần tử. “

Do đó, mặc dù hai framework có liên quan đến nhau, nhưng được sử dụng cho các mục đích khác nhau. Kiến thức về React sẽ không đủ để phát triển ứng dụng dành cho thiết bị iOS và Android.

### Lịch sử của React Native

* Khi Facebook lần đầu tiên quyết định cung cấp dịch vụ của mình trên thiết bị di động, thay vì xây dựng một ứng dụng gốc như nhiều người chơi công nghệ hàng đầu vào thời điểm đó, họ quyết định chạy với một trang web di động dựa trên HTML5.
* Tuy nhiên, giải pháp không chịu được thử thách của thời gian, để lại chỗ trống cho các cải tiến về giao diện người dùng và hiệu suất.
* Trên thực tế, vào năm 2012, Mark Zuckerberg thừa nhận rằng “sai lầm lớn nhất mà chúng tôi mắc phải với tư cách là một công ty là đặt cược quá nhiều vào HTML thay vì native”.
* Ngay sau đó, vào năm 2013, nhà phát triển của Facebook Jordan Walke đã có một khám phá đột phá – ông đã tìm ra phương pháp tạo các phần tử giao diện người dùng (UI element) cho các ứng dụng iOS bằng cách sử dụng JavaScript.
* Điều này đã châm ngòi một ngọn lửa và một cuộc thi Hackathon đặc biệt đã được tổ chức để khám phá thêm về mức độ phát triển thiết bị di động (cho đến nay, theo truyền thống dựa trên web) có thể được thực hiện bằng cách sử dụng các giải pháp JavaScript .
* Đó là cách React Native ra đời. Ban đầu chỉ được phát triển cho iOS, Facebook đã nhanh chóng theo dõi nó với sự hỗ trợ của Android, trước khi công khai framework này vào năm 2015.
* Chỉ ba năm sau, React Native là dự án lớn thứ hai trên GitHub, tính theo số lượng người đóng góp. Vào 2019, nó đứng vững ở vị trí thứ sáu, với hơn 9.100 người đóng góp.

### Cách thức hoạt động React Native

 Hình 3.3 Cách thức hoạt dộng React Native

Cách thức hoạt động của nó để ứng dụng React Native một cách tốt nhất. React Native hoạt động bằng cách tích hợp cho ứng dụng di động 2 thread là JS thread và Main thread. Main thread giữ vai trò cập nhật giao diện người dùng UI và xử lý tương tác người dùng ngay sau đó. Còn JS thread sẽ thực thi và xử lý các code javascript. Hai thread này hoạt động hoàn toàn độc lập với nhau. Vì thế phải sử dụng một cầu nối bridge để hai thread này có thể tương tác, giao tiếp được với nhau nhưng không phụ thuộc lẫn nhau. Nó chỉ là sự vận hành và chuyển đổi dữ liệu từ thread này sang thread khác.

### Lợi ích của việc sử dụng React Native

* **Khả năng tái sử dụng mã – phát triển đa nền tảng**

Khả năng tái sử dụng mã là lợi thế lớn nhất của React Native và nó chỉ ra rằng các ứng dụng có thể chạy hiệu quả trên nhiều nền tảng – đó là điều mà các CEO và [Product owner](https://gambaru.io/blog/product-owner-la-gi) thực sự đánh giá cao.

Họ có thể tích hợp 90% framework gốc để sử dụng lại mã cho cả hai hệ thống hoạt động.

Một điều tuyệt vời khác là có thể sử dụng mã ứng dụng web để phát triển ứng dụng dành cho thiết bị di động nếu cả hai đều đang sử dụng React Native.

Nó cũng tăng tốc thời gian phát triển vì nó bao gồm các thành phần được phát triển trước, được đưa vào thư viện mã nguồn mở.

* **Cộng đồng nhà phát triển lớn**

React Native là một nền tảng JavaScript mã nguồn mở cho phép các nhà phát triển đóng góp kiến ​​thức của họ vào sự phát triển của framework, tất cả mọi người đều có thể truy cập miễn phí.

Nếu bất kỳ nhà phát triển nào gặp sự cố khi phát triển ứng dụng, thì họ có thể nhờ cộng đồng hỗ trợ (tính đến giữa năm 2020, có gần 50.000 người đóng góp tích cực cho thẻ React Native trong Stack Overflow).

Sẽ luôn có ai đó có thể giúp họ giải quyết các vấn đề của họ – điều này cũng có tác động tích cực đến việc cải thiện kỹ năng viết mã.

* **Hiệu quả chi phí**

Một lợi thế khác của phát triển React Native là hiệu quả chi phí lớn hơn. Như đã đề cập trước đó, điều này là do các nhà phát triển có thể sử dụng cùng một mã để xây dựng các ứng dụng cho iOS và Android.

Điều đó nghĩa là không phải thuê hai nhóm nhà phát triển iOS và Android riêng biệt để hoàn thiện dự án của mình; một đội nhỏ là đủ để xây dựng nó.

Chi phí phát triển ứng dụng trong React Native thấp hơn nhiều so với các ứng dụng được tạo bằng ngôn ngữ không cho phép phát triển đa nền tảng.

* **Làm mới nhanh**

Làm mới nhanh cho phép các nhà phát triển chạy ứng dụng trong khi cập nhật lên phiên bản mới và sửa đổi giao diện người dùng.

Các thay đổi có thể nhìn thấy ngay lập tức và nhà phát triển không cần phải xây dựng lại toàn bộ ứng dụng.

Điều này dẫn đến hai lợi ích đáng kể: tiết kiệm thời gian – vì các lập trình viên tiết kiệm thời gian biên dịch và tăng năng suất – vì họ không mất bất kỳ trạng thái nào khi kết hợp các thay đổi vào ứng dụng.

* **Giao diện người dùng đơn giản**

Phát triển React Native sử dụng React JavaScript để xây dựng giao diện của ứng dụng, giúp ứng dụng phản hồi tốt hơn và nhanh hơn với thời gian tải giảm, dẫn đến trải nghiệm người dùng tổng thể tốt hơn.

Nhờ reactive UI và cách tiếp cận dựa trên thành phần, framework hoàn hảo để xây dựng các ứng dụng có cả thiết kế đơn giản và phức tạp.

* **Các ứng dụng nhanh**

Một số cho rằng mã React Native có thể có ảnh hưởng bất lợi đến hiệu suất của ứng dụng. Mặc dù JavaScript không chạy nhanh như mã gốc (native code), nhưng sự khác biệt này là không thể nhận thấy đối với mắt người.

* **Chứng minh tương lai**

Xem xét tốc độ mà framework tiếp nhận thị trường và cách tiếp cận đơn giản của nó để giải quyết các vấn đề phát triển, tương lai của React Native cho các ứng dụng đa nền tảng có vẻ tươi sáng.

Mặc dù nó có một vài nhược điểm, nhưng tốc độ và sự thuận tiện của việc phát triển đã bù đắp cho chúng.

React Native: Rủi ro và hạn chế

Dưới đây là bốn nhược điểm tiềm ẩn hàng đầu mà bạn cần biết trước khi quyết định phát triển một ứng dụng React Native.

* **Thiếu một số mô-đun tùy chỉnh**

Mặc dù React Native đã xuất hiện được vài năm, nhưng một số mô-đun tùy chỉnh vẫn còn chỗ để cải thiện hoặc hoàn toàn bị thiếu.

Điều này nghĩa là có thể cần chạy ba cơ sở mã riêng biệt (cho React Native, iOS và Android) thay vì chỉ một.

Điều này không thường xuyên xảy ra. Trừ khi bạn đang phát triển ứng dụng của mình từ đầu hoặc cố gắng hack một ứng dụng hiện có, bạn có thể sẽ không gặp phải những vấn đề này.

* **Các vấn đề về khả năng tương thích và gỡ lỗi**

Mặc dù điều này có thể gây ngạc nhiên – sau tất cả, React Native được sử dụng bởi những người chơi công nghệ hàng đầu – nó vẫn đang trong giai đoạn beta.

Các nhà phát triển của bạn có thể gặp nhiều vấn đề khác nhau với khả năng tương thích hoặc các công cụ gỡ lỗi.

Nếu các nhà phát triển của bạn không thành thạo React Native, điều này có thể ảnh hưởng tiêu cực đến sự phát triển của bạn khi họ dành thời gian cho việc khắc phục sự cố kéo dài.

* **Khả năng mở rộng**

Hầu hết thời gian, React Native sẽ hoạt động rất tốt cho bạn ngay cả khi ứng dụng của bạn cuối cùng phát triển thành một giải pháp phức tạp, tinh vi.

Rốt cuộc, các công ty như Facebook và Skype đã tìm thấy nhiều thành công với framework này và đã sử dụng nó một cách nhất quán trong nhiều năm.

Mặc dù vậy, một số công ty đã quyết định rút lui khỏi việc sử dụng React Native.

Ví dụ, Airbnb đã quyết định sử dụng framework cho ứng dụng di động của mình khi công ty chỉ là một công ty khởi nghiệp mới nổi.

Tuy nhiên, theo thời gian, React Native tỏ ra không phù hợp với kế hoạch tăng trưởng của công ty và Airbnb đã sử dụng đến việc phát triển hai ứng dụng gốc.

Với những tiến bộ hiện tại trong RN và với các lựa chọn kiến ​​trúc phần mềm phù hợp, các vấn đề về khả năng mở rộng có thể dễ dàng được ngăn chặn.

### Ứng dụng của React Native

1. **Facebook**

Facebook là một trong những ứng dụng React Native phổ biến nhất vì nó đã khai sinh ra ngôn ngữ lập trình này và là động lực chính đằng sau sự phát triển của nó.

  
  
Hình 3.4 Ứng dụng di động Facebook được phát biển bởi React Native

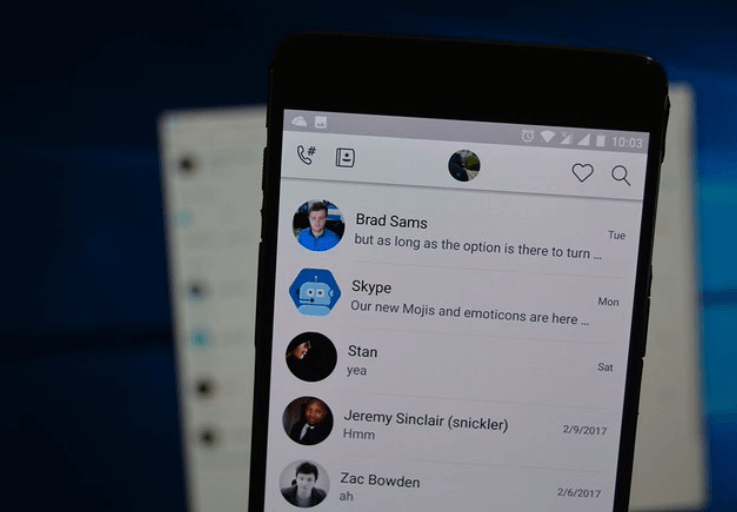
Facebook nhằm mục đích mang lại tất cả các lợi ích phát triển web cho thiết bị di động, như lặp lại nhanh chóng và có một nhóm phát triển sản phẩm duy nhất và đây là cách React Native ra đời.

Công ty đã sử dụng nó để phát triển ứng dụng Trình quản lý quảng cáo của riêng mình trong iOS và Android – cả hai phiên bản đều được tạo bởi cùng một nhóm phát triển.

1. **Skype**

Skype là một ví dụ điển hình khác về ứng dụng React Native dành cho thiết bị di động.

Vào năm 2017, Skype thông báo rằng họ đang xây dựng một ứng dụng hoàn toàn mới dựa trên React Native.

  
  
Hình 3.5 Ứng dụng di động Skype được phát biển bởi React Native

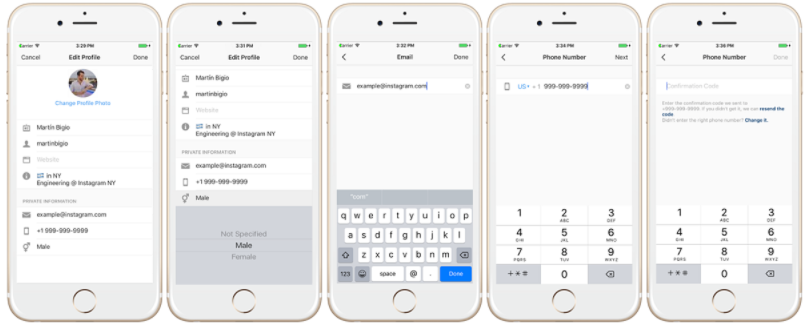
Điều này đã mang lại rất nhiều sự phấn khích từ người dùng, vì phiên bản cũ hơn gặp phải một số vấn đề.

Ứng dụng mới được thiết kế lại hoàn toàn, từ các biểu tượng đến giao diện nhắn tin mới, hiện có ba phần trò chuyện: tìm, trò chuyện và chụp.

Microsoft, công ty sở hữu Skype, đã quyết định sử dụng React Native không chỉ trong ứng dụng di động mà còn trong phiên bản desktop của nền tảng này.

1. **Instagram**

Instagram đã quyết định tích hợp React Native vào ứng dụng gốc hiện có của mình, bắt đầu với chế độ xem Thông báo đẩy có nguồn gốc là WebView.

   
  
Hình 3.6 Ứng dụng di động Instagram được phát biển bởi React Native

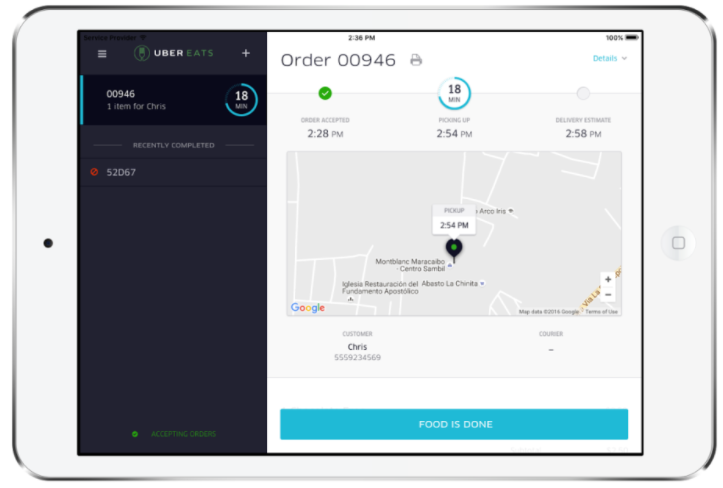
May mắn thay, không cần thiết phải xây dựng hạ tầng điều hướng, vì giao diện người dùng đủ đơn giản để đối phó mà không có.

Việc sử dụng React Native cho phép các nhóm sản phẩm cải thiện tốc độ của nhà phát triển lên 85-99%.

1. **UberEats**

UberEats là một ứng dụng khác dựa trên sự phát triển của React Native.

Nó khác với ứng dụng Uber ở chỗ nó bao gồm ba bên thay vì hai – nhà hàng, đối tác giao hàng và thực khách.

   
  
Hình 3.7 Ứng dụng di động Uber được phát biển bởi React Native

Điều này đòi hỏi phải xây dựng một dashboard đặc biệt cũng sẽ tính đến các nhà hàng.

Dashboard ban đầu được xây dựng riêng cho web đã hạn chế khả năng truyền đạt thông tin quan trọng tới các nhà hàng.

Nó cũng thiếu quyền truy cập vào các chức năng của thiết bị gốc như thông báo âm thanh, điều này ảnh hưởng tiêu cực đến trải nghiệm người dùng.

Nhóm đã có nhiều kinh nghiệm lập trình trong React, nhưng không đủ tiếp xúc với Android và iOS, vì vậy việc chọn React Native là một lựa chọn tự nhiên.

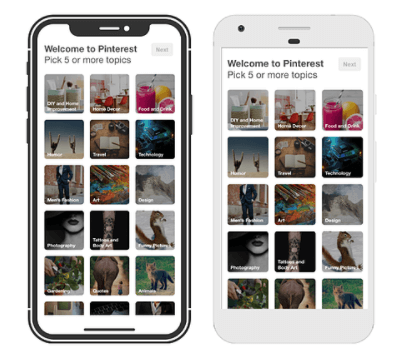
UberEats sử dụng một đống công nghệ lớn và React Native chỉ chiếm một phần nhỏ trong số đó.

Tuy nhiên, các nhà phát triển hài lòng với những gì nó có thể cung cấp và họ chắc chắn rằng nó có khả năng đáp ứng nhu cầu của họ khi thị trường phát triển.

1. **Pinterest**

Các kỹ sư của Pinterest thừa nhận rằng họ đã để mắt đến React Native và quan tâm đến việc khám phá các khả năng của nó kể từ khi ra mắt framework này vào năm 2015.

Vào thời điểm đó, họ đã cung cấp cho người dùng một ứng dụng web được cung cấp bởi Gestalt, bộ nguồn mở gồm các thành phần React UI của Pinterest.

  
  
Hình 3.8 Ứng dụng di động Facebook được phát biển bởi React Native

Vì cả Gestalt và React Native đều dựa trên React, các kỹ sư của Pinterest kỳ vọng rằng quá trình phát triển sẽ không phải là phức tạp và gian khổ.

Mặc dù họ chưa bao giờ có ý định thay thế hoàn toàn ứng dụng Gestalt của mình bằng React Native, nhưng họ muốn kiểm tra xem nó có thể được tích hợp trơn tru vào tech stack hiện tại của họ hay không.

Để kiểm tra các khả năng của React Native, họ đã quyết định xây dựng một nguyên mẫu một tính năng tích hợp quan trọng – Công cụ chọn chủ đề.

Quá trình thực hiện nguyên mẫu trong iOS mất mười ngày và đối với Android – thêm hai ngày.

Các kỹ sư ước tính rằng họ có thể tiết kiệm hơn một tuần thời gian thực hiện tiêu chuẩn. Ngoài tốc độ, họ cũng hài lòng với hiệu suất của iOS và Android.

Do đó, Pinterest đã quyết định đưa React Native vĩnh viễn vào tech stack của mình như một khung phát triển di động.

Kể từ khi được giới thiệu, nó đã được sử dụng để cung cấp không chỉ cho Bộ chọn chủ đề mà còn cho trình tự Đăng ký doanh nghiệp của Pinterest.

## Giới thiệu về .NET Core

### Định nghĩa về .NET Core

.NET Core là một nền tảng phát triển đa mục đích, mã nguồn mở được duy trì bởi Microsoft và cộng đồng .NET trên GitHub. Đó là nền tảng chéo (hỗ trợ Windows, macOS và Linux) và có thể được sử dụng để xây dựng các ứng dụng thiết bị, đám mây và IoT.

### Lịch sử của .NET Core

Ngày 12 tháng 11 năm 2014, Microsoft đã công bố .NET Core, trong nỗ lực hỗ trợ đa nền tảng cho .NET trên các hệ điều hành Linux và macOS. Khởi đầu .NET Core áp dụng mô hình phát triển mã nguồn mở thông thường dưới sự quản lý của .NET Foundation, được [Miguel de Icaza](https://vi.wikipedia.org/wiki/Miguel_de_Icaza), nhân viên Microsoft phụ trách phát triển phần mềm của dự án, mô tả .NET Core là "phiên bản .NET được thiết kế lại dựa trên phiên bản đơn giản hóa của các thư viện lớp".

Phiên bản .NET Core 1.0 phát hành vào ngày 27 tháng 6 năm 2016, cho phép phát triển theo bộ công cụ [Microsoft Visual Studio](https://vi.wikipedia.org/wiki/Microsoft_Visual_Studio) 2015 Update 3. Tiếp theo các phiên bản .NET Core 1.0.4 và .NET Core 1.1.1 được phát hành theo .NET Core Tools 1.0 và Visual Studio 2017 vào 7 tháng 3 năm 2017.

Phiên bản .NET Core 2.0 phát hành vào ngày 14 tháng 8 năm 2017, kèm với Visual Studio 2017 15.3, [ASP .NET Core](https://vi.wikipedia.org/w/index.php?title=ASP.NET_Core&action=edit&redlink=1) 2.0, và [Entity Framework](https://vi.wikipedia.org/wiki/Entity_Framework) Core 2.0. Phiên bản .NET Core 2.1 phát hành ngày 30 tháng 5 năm 2018 với bản 2.1.30 là phiên bản Hỗ trợ dài hạn LTS. Phiên bản .NET Core 2.2 phát hành ngày 4 tháng 12 năm 2018.

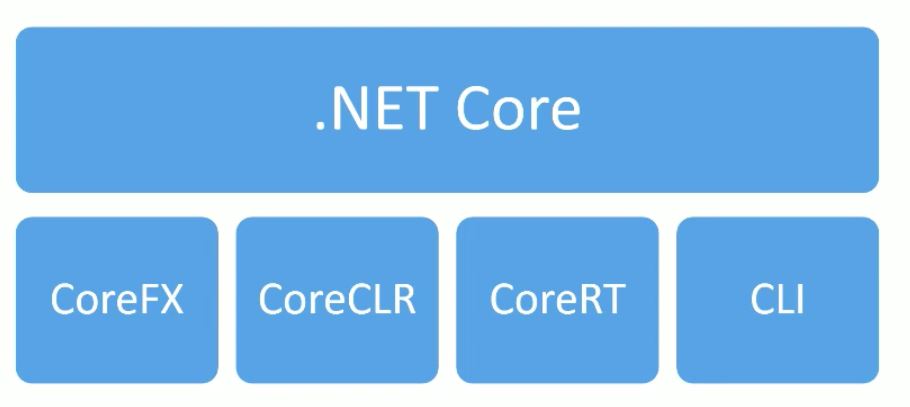
Lịch sử các phiên bản và kế hoạch phát triển .NET Core:

Hình 3.9 Lịch sử phát triển Net Core

### Thành phần của .NET Core

.NET Core bao gồm các nền tảng: .NET Compiler Roslyn, .NET Core framework CoreFX, .NET Core runtime CoreCLR, và ASP.NET Core.

Cấu trúc của .Net Core chi tiết:

* CoreFX:Nó được xem là nền tảng thư viện dành cho .NET Core.
* .NET Core runtime: cung cấp một hệ thống kiểu, tải lắp ráp, trình thu gom rác, interop gốc và các dịch vụ cơ bản khác. Các thư viện khung .NET Core cung cấp các kiểu dữ liệu nguyên thủy, các kiểu thành phần ứng dụng và các tiện ích cơ bản.
* ASP.NET Core runtime: cung cấp khung để xây dựng các ứng dụng kết nối internet , điện toán đám mây hiện đại, chẳng hạn như ứng dụng web, ứng dụng IoT và phụ trợ di động.
* .NET Core SDK và trình biên dịch ngôn ngữ (Roslyn và F #) cho phép trải nghiệm nhà phát triển .NET Core.
* Dotnet command, được sử dụng để khởi chạy các ứng dụng .NET Core và các lệnh CLI. Nó chọn thời gian chạy và lưu trữ thời gian chạy, cung cấp chính sách tải lắp ráp và khởi chạy các ứng dụng và công cụ.

Hình 3.10 Thành phần của .Net Core

### Đặc điểm .NET Core

* **Open-source Framework**: .NET Core là nền tảng nguồn mở được duy trì bởi Microsoft và có sẵn trên GitHub theo giấy phép MIT và Apache 2. Nó là một dự án .NET Foundation.
* **Cross-platform**: .NET Core chạy trên hệ điều hành Windows, macOS và Linux. Có thời gian chạy khác nhau cho mỗi hệ điều hành thực thi mã và tạo ra cùng một output.
* **Nhất quán giữa các kiến trúc**: Có thể thực thi mã giống nhau trong các kiến trúc tập lệnh khác nhau (bao gồm x64, x68 và RAM)
* **Hỗ trợ nhiều ngôn ngữ**: Có thể sử dụng ngôn ngữ lập trình C#, F# và Visual Basic để phát triển ứng dụng .NET Core.
* **Triển khai linh hoạt**: Các ứng dụng .NET Core có thể được triển khai song song (cài đặt toàn bộ người dùng hoặc toàn hệ thống). Nó cũng có thể triển khai với các Container Docker.
* **Khả năng tương thích**: .NET Core tương thích với .NET Framework và Mono API (thông qua .NET Standard).
* **Hỗ trợ bởi Microsoft**: Vì .NET Core được phát triển bởi Microsoft nên tài liệu cũng như các cập nhật, hỗ trợ được update thường xuyên. Từ đó, giúp người sử dụng có thể giải quyết vấn đề nhanh chóng.
* **Công cụ CLI**: .NET core bao gồm các công cụ CLI (Command – line –interface: Giao tiếp thông qua dòng lệnh) để phát triển và tích hợp liên tục.
* **Kiến trúc mô-đun**: .NET core hỗ trợ cách tiếp cận kiến trúc mô-đun bằng cách sử dụng các gói NuGet. Có các gói NuGet khác nhau cho các tính năng khác nhau có thể được thêm vào dự án .NET core nếu cần. Nhờ đó sẽ ít tiêu tốn dung lượng bộ nhớ, tăng hiệu suất và dễ bảo trì ứng dụng hơn.

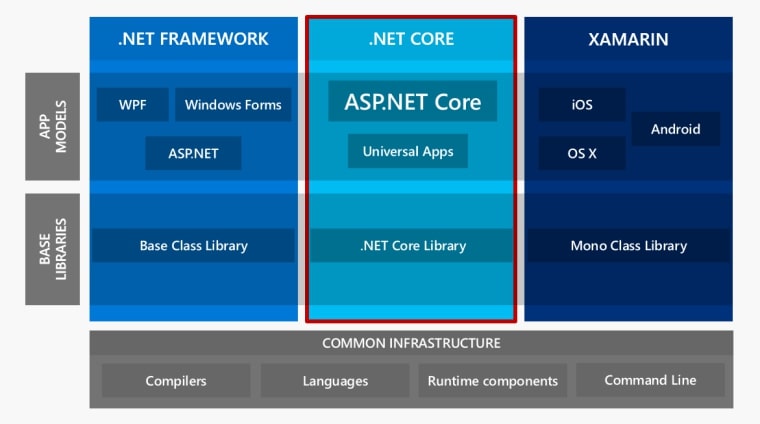
### Ưu điểm của .NET Core

* .NET core có thể được sử dụng cho nhiều nền tảng khác nhau như WIndows, Linux hay Mac OS. Khi ứng dụng nó với Visual Studio Code hoặc các công cụ khác sẽ giúp bạn phát triển ứng dụng trên các nền tảng khác nhau một cách dễ dàng.
* Với khả năng triển khai linh hoạt của mình, khi .NET core kết hợp với Container Docker hoặc Azure Kubernates Service sẽ giúp tiết kiệm tài nguyên đáng kể.
* Với kiến trúc Mô đun, .NET Core ít tiêu tốn dung lượng bộ nhớ, tăng hiệu suất và dễ bảo trì ứng dụng hơn.
* Tốc độ thực thi nhanh và khả năng mở rộng ứng dụng cũng là một trong những ưu điểm lớn của .NET core

### NET Framework và .NET Core

NET Framework được giới thiệu năm 2000 và trở thành nền tảng chủ yếu trong việc phát triển sản phẩm .NET của Microsoft. Hai nền tảng .NET Core và .NET Framework có một số điểm khác biệt:

* .NET Core không hỗ trợ tất cả các mô hình .NET Framework. Cụ thể, .NET Core không hỗ trợ ASP.NET Web Forms và ASP.NET MVC nhưng nó hỗ trợ ASP.NET Core MVC. Bắt đầu từ .NET Core 3.0 cũng hỗ trợ Window Forms và WPF chỉ trên nền tảng Windows.
* .NET Core chứa một tập lớn các tập con của Thư viện lớp cơ sở .NET Framework (.NET Framework Base Class Libraly) nhưng có những yếu tố khác biệt như tên các assembly, kiểu,…
* .NET Framework chỉ hỗ trợ Windows và Windows Server trong khi đó .NET Core hỗ trợ thêm macOS và Linux.

.NET Core là open source – điểm khác biệt cốt lõi với .NET Framework.  
  
Hình 3.11 Net core và Net framework

## Giới thiệu về ASP.NET Core

### Tổng quan về ASP.NET Core

ASP.NET Core là công nghệ phát triển ứng dụng web dựa trên hai nền tảng .NET Framework và .NET Core của Microsoft.

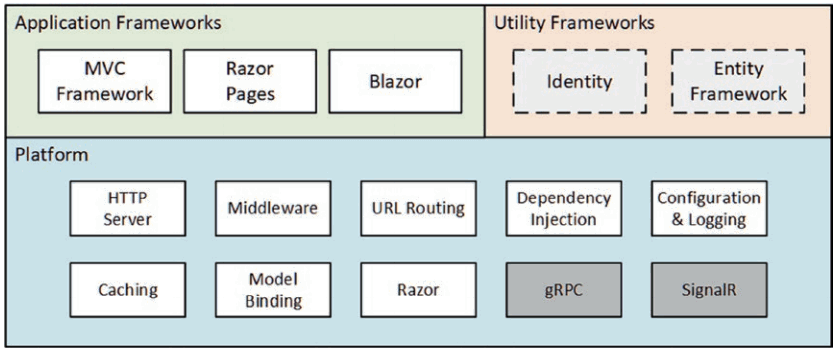
ASP.NET Core là phiên bản tiến hóa mới nhất của công nghệ ASP.NET, kế thừa những tính năng nổi trội từ ASP.NET cũ nhưng ASP.NET Core là nền tảng hoàn toàn mới, khác biệt so với ASP.NET trước đây.

ASP.NET xuất hiện lần đầu vào năm 2002 như là một phần của .NET Framework 1.0. Ứng dụng ASP.NET lúc này là ASP.NET Web Forms cho phép tạo các ứng dụng dựa trên giao diện đồ họa và mô hình hướng sự kiện. Theo thời gian, với việc các ứng dụng ASP.NET Web Forms trở nên cồng kềnh, khó bảo trì nên đến năm 2009, Microsoft cho ra đời phiên bản ASP.NET MVC nhằm tạo các ứng dụng web dựa trên mô hình MVC. Dù có nhiều ưu điểm nổi bật hơn ASP.NET Web Forms nhưng cả hai cùng dựa vào một nền tảng với tập tin trọng tâm  System.Web.dll. Sự phụ thuộc này kế thừa một số ưu điểm của .NET Framework như tính tin cậy cao và nhiều tính năng thuận tiện cho sự phát triển các ứng dụng web hiện đại trên nền tảng Windows, đồng thời cũng mang lại một số hạn chế. Sự hạn chế đầu tiên là sự thay đổi chậm chạp về nội dung trong tập tin System.Web.dll ảnh hưởng lớn đến khả năng mở rộng của ASP.NET, bên cạnh đó, các ứng dụng web hiện đại đòi hỏi thực thi đa nền (cross platform) tức là không chỉ thực thi trên Windows mà còn trên Linux hay MacOS.

ASP.NET Core tức công nghệ ASP.NET kế thừa nền tảng cũ .NET Framework và phát triển thêm những tính năng mới trên nền tảng .NET Core ra đời và Microsoft mang đến ASP.NET Core có những sự tiến hóa và hoàn hảo mới.

ASP.NET Core được biết đến với một số thay đổi mới mẻ lớn về kiến trúc, đó là kết quả từ việc học hỏi rất nhiều từ các Framework Module hóa khác. Hiện nay, ASP.NET Core không còn phụ thuộc cố định vào System.Web.dll nữa mà tập hợp các gói, các module hay còn được gọi là các Nuget Packages.

### Cấu trúc của ASP.NET Core

 Cấu trúc ASP.NET Core bao gồm 3 phần: Application Frameworks, Utility Frameworks, và Platform.

Hình 3.12 Cấu trúc của ASP.NET core

* **Application Frameworks:** Chứa những cái tên có lẽ tương đối quen thuộc như MVC Framework, Razor Pages hay Blazor. Đây là những framework giúp xây dựng các dạng khác nhau của ứng dụng web.
* **Utility Frameworks:** Chứa những thứ cảm tưởng như không liên quan đến Asp.net Core: Identity và Entity Framework. Khối này chứa những framework hỗ trợ cho ứng dụng, bao gồm bảo mật và cơ sở dữ liệu.
* **Platform:** Là những gì tạo nên nền tảng chung nhất mà mọi loại ứng dụng Asp.net Core đều cần sử dụng đến.

#### 3.2.2.1 Application Frameworks

Application Framework chứa các framework: MVC, Razor Pages, Blazor . Các framework này giúp giải quyết các loại vấn đề khác nhau, và giải quyết cùng một vấn đề theo những cách thức khác nhau.

Nhìn nhận theo một cách khác, đây là những mẫu kiến trúc khác nhau cho ứng dụng web mà Microsoft hỗ trợ. Có thể lựa chọn bất kỳ framework nào phù hợp với yêu cầu của dự án và sở thích cá nhân. Do vậy, chúng không loại trừ nhau mà còn tương trợ lẫn nhau. Trong quá trình phát triển ứng dụng, Có thể chuyển đổi từ framework này sang framework khác, hoặc kết hợp chúng.

* **MVC** là framework xây dựng theo nguyên lý phân chia nhiệm vụ (Separation of Concerns, SoC). MVC phân chia chức năng thành các phần của ứng dụng ra làm các nhóm độc lập
* **Razor Pages** là một framework đơn giản hướng tới xây dựng các trang web tự thân độc lập. Mô hình của Razor Pages đơn giản hơn so với MVC.
* **Blazor** là framework dành cho phát triển ứng dụng đơn trang (SPA) nhưng sử dụng ngôn ngữ C# thay cho JavaScript chạy trên trình duyệt. Đây là framework mới nhất và đang nhận được nhiều sự quan tâm. Blazor bao gồm hai mô hình chính: Blazor Server và Blazor WebAssembly

#### 3.2.2.2 Utility Frameworks

Utility Frameworks chứa hai framework (không bắt buộc) nhưng lại được sử dụng gần như trong mọi ứng dụng ASP.NET Core.

* **Entity Framework Core** là một ORM (Object-Relational Mapping) giúp ứng dụng tương tác với cơ sở dữ liệu. Entity Framework Core giúp ánh xạ (hai chiều) giữa các bảng cơ sở dữ liệu (ví dụ, SQL Server) với các domain class của ứng dụng. Entity Framework Core có thể xem là phiên bản hiện đại đa nền tảng của Entity Framework – ORM chính của .NET Framework cổ điển. Nếu đã biết Entity Framework có thể dễ dàng chuyển đổi sang Entity Framework Core.
* **Asp.net Core Identity** là framework dành cho xác thực (authentication) và xác minh quyền (authorization) người dùng trong ứng dụng.

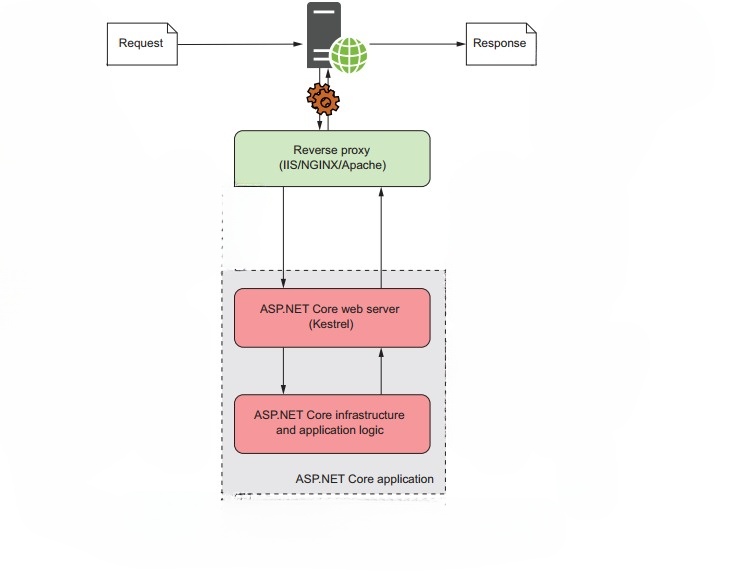
#### .2.2.3 Platform

Phần platform chứa nhiều thành phần cấp thấp cần thiết cho việc nhận và xử lý truy vấn HTTP, cũng như tạo ra các phản hồi phù hợp.

Mặc dù phần lớn thời gian trong quá trình phát triển ứng dụng làm việc chủ yếu với applicaton framework và bản thân các application framework cũng che đi rất nhiều phần của platform.

* **HTTP Server** còn gọi là built-in server với tên gọi Kestrel, có nhiệm vụ tiếp nhận truy vấn HTTP. Kestrel có thể hoạt động độc lập (tích hợp trong một ứng dụng khác) hoặc phối hợp với một web server thông thường (Apache, NGinx, IIS).
* **Middleware** là nhóm thành phần có nhiệm vụ xử lý truy vấn HTTP. Các middleware được xếp theo chuỗi. Khi truy vấn chạy qua mỗi khâu của chuỗi middle sẽ được xem xét xử lý.
* **URL Routing** là cơ chế ánh xạ chuỗi truy vấn HTTP sang thực thi một phương thức nào đó. Do vậy, mỗi URL (địa chỉ bạn gửi về server) sẽ tương ứng với thực thi một phương thức trên server.
* **Razor** là cơ chế sinh ra HTML từ dữ liệu và logic của chương trình. Razor được gọi là view engine trong Asp.net Core. Razor sử dụng loại cú pháp đặc biệt kết hợp giữa C# và HTML.
* **Model Binding** là cơ chế ánh xạ dữ liệu chứa trong truy vấn HTTP sang tham số của phương thức cần thực thi. Nhờ cơ chế Model Binding, việc xây dựng các phương thức trong Asp.net Core đơn giản như bất kỳ phương thức C# thông thường nào.
* **Dependency Injection** là cơ chế cho phép tự động sinh và chèn object vào một object khác. Asp.net Core xây dựng sẵn cơ chế này mà không cần đến một thư viện thứ ba (như Ninject, Unity).
* **Configuration & Logging** là cơ chế hỗ trợ cấu hình và lưu vết quá trình thực thi ứng dụng.
* **Caching** là cơ chế lưu tạm để tăng hiệu suất cho ứng dụng.
* **SignalR** là một framework riêng nhưng lại được sử dụng làm nền tảng cho ứng dụng Asp.net Core. SignalR cho phép tạo ra kênh truyền thông tốc độ cao (theo thời gian thực) và hai chiều giữa browser và web server. SignalR là nền tảng cho Blazor Server – loại ứng dụng đơn trang mà toàn bộ phần xử lý được đẩy về server và nhận lại kết quả (DOM) theo thời gian thực.
* **gRPC** là một chuẩn dành cho gọi hàm từ xa (Remote Procedure Call) đa nền tảng qua HTTP do Google (chữ g trong gRPC) phát triển. gRPC có thể hữu ích để các thành phần backend của ứng dụng (chạy trên server) trao đổi dữ liệu.

### Cách ASP.NET Core làm việc

 Người dùng từ trình duyệt web muốn gửi một yêu cầu đến ứng dụng ASP.NET Core cũng phải thông qua giao thức HTTP. Nhưng thay vì yêu cầu được gửi trực tiếp đến server (hay web server), nó phải thông qua một reverse-proxy server. Hiểu một cách đơn giản, một reverse-proxy server là một phần mềm có trách nhiệm nhận và chuyển tiếp yêu cầu từ người dùng đến server (hay web server). Các reverse-proxy server được công khai trên internet nhưng các web server thì chỉ có reverse-proxy server biết. Điều này làm tăng tính bảo mật và khả năng thực thi của các web server. Phần mềm reverse-proxy server trên Windows là IIS, trên Linux hay MacOS có thể là NGHINX hay Apache. Các bước thực thi được mô tả như hình sau đây:

Hình 3.13 Cơ chế hoạt động ASP.Net core

1. Yêu cầu được gửi đến reverse-proxy server trước khi đến web server thông qua HTTP Request

2. Yêu cầu được chuyển tiếp đến web server bởi reverse-proxy server

3. ASP.NET Core web server chuyển yêu cầu đến bộ phận xử lý

4. Yêu cầu được xử lý và phát sinh các nội dung hồi đáp đến trình duyệt

5. Kết quả xử lý được chuyển trở lại ASP.NET Core web server

6. ASP.NET Core web server chuyển kết quả đến reverse-proxy server

7. Kết quả hiển thị đến trình duyệt bởi HTTP Response

### Đặc điểm của ASP.NET Core

Có thể thấy, ASP.NET Framework tồn tại bền bỉ trên hệ thống Windows từ lâu thì ASP.NET Core mới bắt đầu ra đời để hỗ trợ các tính năng cho .NET Framework như Windows Forms, WPF. ASP.NET Core có thể hoạt động trên bất kỳ nền tảng nào mà đảm bảo có thể khắc phục được những hạn chế của .NET Framework – chaỵ duy nhất trên nền tảng Windows. ASP.NET Core được thiết kế để tối ưu và cung cấp tốt cho các Development Framework triển khai trên đám mây hoặc chạy on-promise.

Về cơ bản, ASP.NET Core được tạo thành từ một thư viện bản Net. .NET Standard 2.0 có thể chạy ở bất cơ nào mà nó có thể hỗ trợ được. Đồng thời, ASP.NET Core trở thành một mã nguồn mở. Đây là sự thay đổi lớn và vô cùng quan trọng mà trước đây khó có lập trình viên nào nghĩ đến. Chính vì thế, ASP.NET Core thu hút đông đảo các lập trình viên sử dụng để xây dựng các trang web.

ASP.NET Core là một phiên bản thiết kế của ASP.NET 4.x có sự thay đổi kiến trúc giúp Framework nhẹ nhàng hơn và kết hợp cả tính Module nhiều hơn nữa. Với một nền tảng có hiệu suất và tính tương thích hiệu quả, các lập trình viên vẫn trung thành xây dựng các web bằng ASP.NET Core.

### Ưu điểm của ASP.NET Core

* ASP.NET Core được thay đổi một số kiến trúc nên dẫn đến Modular Framework nhỏ hơn.
* ASP.NET Core không còn phụ thuộc vào nền tảng system.web.dll, Framework này dựa trên một tập hợp nhiều yếu tố của Nuget Packages.
* Có thể tối ưu ứng dụng của mình dễ dàng thông qua những Nuget Packages cần thiết.
* Những ứng dụng web được thiết kế trên nền tảng ASP.NET Core sẽ ít tiêu hao dung lượng bộ nhớ, bảo mật chặt chẽ, tốc độ thực thi nhanh, hiệu năng hoạt động tốt và ổn định.
* ASP.NET Core giúp giảm dịch vụ, dễ bảo trì ứng dụng và tiết kiệm chi phí.
* Các ứng dụng ASP.NET Core được xây dựng và khởi tạo ở đa dạng các nền tảng như Windows, Mac và Linux.
* Được phát triển từ .NET Core, ASP.NET Core hỗ trợ chất lượng App Vesioning.
* Mang đến những công cụ và hàng loạt tính năng hiện đại, đơn giản hóa quy trình phát triển web.
* Tương thích với hệ thống xây dựng web UI và web APIs.
* Có thể tích hợp những client – side Frameworks hiện đại và những luồng phát triển.
* Cấu hình dựa trên mô hình đám mây có sẵn.
* Dependency Injection (DI) được xây dựng sẵn.
* Đa dạng cấu hình, thân thiện với nhiều môi trường.
* Có tính năng host trên IIS hoặc self-host trong Process của riêng mình.
* Chuyển thực thể, các thành phần module, dùng chung toàn bộ Nuget Package.
* Mã nguồn mở và tập trung vào cộng đồng.

### ASP.NET và ASP.NET Core

ASP.NET và ASP.NET Core là 2 frameword hỗ trợ các nhà **phát triển các ứng dụng web** dựa trên nền tảng **.Net Framework** khác nhau

* **ASP.NET** là một framework do Microsoft phát triển miễn phí trên nền tảng Window giúp hỗ trợ các nhà phát triển có thể tạo ra các websites và web applications dựa trên các ngôn ngữ C#, Visual Basic và HTML, CSS, JavaScript. Sau khi ra **đời từ năm 2002**, ASP.NET đã trải qua rất nhiều phiên bản, nó hỗ trợ nhiều framwork con để tạo ra các ứng dụng web như : Web Forms, ASP.NET MVC, and ASP.NET Web Pages và nhiều công nghệ kèm theo để làm cho các ứng dụng viết trên nó trở lên đa dạng và linh động hơn.
* **ASP.NET Core** là **một framework mã nguồn mở**, được Microsoft và các nhà phát triển tạo ra nhằm cung cấp các ứng dụng web **hỗ trợ đa nền tảng** chứ không chỉ trên hề điều hành Microsoft Windows thông thường như ASP.NET. Được giới thiệu từ năm 2016, ASP.NET Core đang được kỳ vọng tạo ra những ứng dụng mới có hiệu xuất xử lý tốt hơn và ngày càng được hoàn thiện bởi sự chung sức của cộng đồng

|  |  |
| --- | --- |
| **ASP.NET** | **ASP.NET Core** |
| Được xây dựng chỉ để dành cho Windows. | Được xây dựng dành cho cả Windows, Mac và Linux. |
| Có hiệu suất tốt | Có hiệu suất cao hơn cả ASP.NET 4x. |
| Có thể chạy được trên .Net Framework hay được gọi là Full .Net Framework. | Có thể chạy trên .Net Core và Full .Net Framework. |
| Asp.Net hỗ trợ Web Forms, Asp.Net MVC và ASP.NET web Pages. | Asp.Net Core hỗ trợ cho các trang Web MVC, Web API và Asp.Net được thêm vào ban đầu trong .Net Core 2.0, không hỗ trợ cho Web Forms. |
| Chỉ sử dụng IIS phụ thuộc vào System.web.dll. | Asp.Net Core không phụ thuộc vào IIS và System.web.dll. |
| Sử dụng ngôn ngữ C#, VB, WCF, WPF và WF. | Chỉ hỗ trợ ngôn ngữ C#, F# và VB trong thời gian ngắn, không hỗ trợ WCF, WPF và WF. Tuy nhiên, Asp.Net Core lại có thể hỗ trợ cho các thư viện WCP có sẵn. |
| Asp.Net MVC có thêm các ứng dụng như Web.config, Global.asax, Application Start. | Asp.Net Core đang hỗ trợ Appsettings.json, không hỗ trợ tệp Web.config và Global.asax. |
| Hỗ trợ vùng chứa không được đánh giá quá cao. | Hỗ trợ vùng chứa phù hợp cho các triển khai như Docker. |
| Tất cả các phiên bản chính thức đều được hỗ trợ. | Hỗ trợ Core từ Visual Studio 2015 cập nhật lần thứ 3 |
| Người dùng cần biên dịch lại sau khi thay đổi mã. | Khi làm mới Core Browser sẽ tự động biên dịch và thực thi mã mà không cần phải dịch lại. |

### Lợi ích khi sử dụng ASP.NET Core

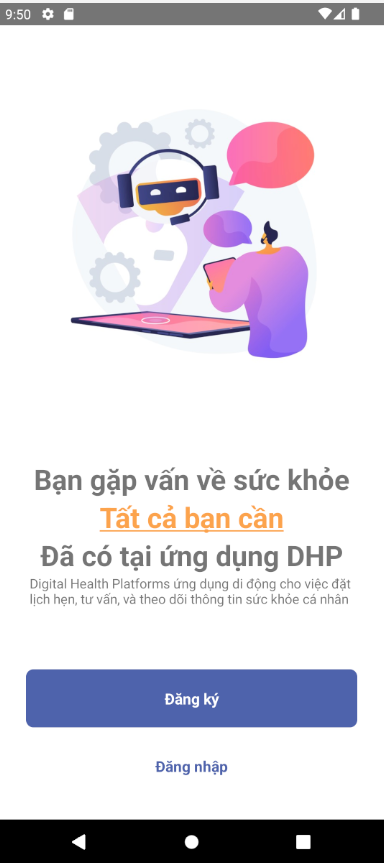
* **Hiệu suất nâng cao:** [Công ty phát triển ASP.NET](https://www.zenesys.com/asp-net-development-services) coi ứng dụng hoạt động tốt như thế nào là yếu tố chính trong khi lựa chọn khung để phát triển ứng dụng. Và trong trường hợp này, ASP.NET Core nhanh hơn nhiều so với ASP.NET MVC và đã cho thấy kết quả tuyệt vời so với các khung công tác khác. Một lý do cho hiệu suất nhanh chóng của khuôn khổ là thực tế là hệ thống tự động tối ưu hóa các mã của nó để cải thiện hiệu suất.
* **Hỗ trợ đa nền tảng:** Cần đảm bảo rằng ứng dụng sẽ hoạt động tốt trên mọi nền tảng khi tiến hành phát triển ứng dụng. ASP.NET Core là đa nền tảng và chạy trên Windows, Linux, Mac và tất cả các thiết bị khác. Do đó, hệ thống cho phép các nhà phát triển chọn bất kỳ hệ điều hành nào để thuận tiện cho họ vì nó cực kỳ linh hoạt.
* **Ít mã hơn:** Chất lượng của mã xác định chất lượng của một ứng dụng. ASP.NET Core cho phép các nhà phát triển viết ít câu lệnh hơn. Do đó, cấu trúc mã trở nên dễ dàng hơn và ít phải viết mã hơn. Điều này làm cho việc [phát triển ứng dụng dành cho thiết bị di động](https://www.zenesys.com/mobile-app-development) trở nên  hiệu quả về mặt chi phí đối với các tổ chức. Ngoài ra, nó cũng cung cấp nhiều quyền kiểm soát hơn cho các nhà phát triển khi quy trình có liên quan và đơn giản hóa việc gỡ lỗi.
* **Bảo trì dễ dàng:** ASP.NET yêu cầu ít mã hơn và ít mã hơn sẽ dễ bảo trì hơn. Các nhà phát triển có thể dễ dàng tối ưu hóa mã trong ASP.NET Core và tiết kiệm thời gian bảo trì ứng dụng.
* **Hỗ trợ phát triển ứng dụng web dựa trên đám mây:** Một lợi ích khác của ASP.NET Core là nó cung cấp các kiểu phát triển ứng dụng khác nhau và ứng dụng web dựa trên đám mây. Vì vậy, cách tiếp cận này là phù hợp nhất cho các doanh nghiệp và doanh nghiệp đã sẵn sàng để mở rộng. Phát triển dựa trên đám mây cung cấp cho các ứng dụng web tính linh hoạt, khả năng truy cập, tích hợp dễ dàng hơn, bảo vệ dữ liệu và hơn thế nữa.
* **Mã nguồn mở:** ASP.NET Core là mã nguồn mở, có nghĩa là bất kỳ chuyên gia ASP.NET Core nào cũng có quyền truy cập vào mã khung. Tất cả các nhà phát triển .NET Core đều có thể cải tiến công nghệ và sửa đổi nó theo nhu cầu phát triển ứng dụng của họ. Điều này giúp các nhà phát triển tạo ra các giải pháp web tốt nhất với ASP.NET Core.
* **Lưu trữ:** .NET Core cung cấp một máy chủ nội bộ cho mọi ứng dụng web ASP.NET Core theo mặc định. Nó cho phép chạy các ứng dụng ASP.NET Core trên Windows, Mac hoặc Linux. Hơn nữa, nó nhẹ và hỗ trợ lớp cổng bảo mật (SSL).
* **An ninh tốt hơn:** ASP.NET Core có một số tính năng tích hợp cho phép các nhà phát triển tạo các ứng dụng web an toàn hơn. Công nghệ này giúp duy trì việc thực thi, xác thực, ủy quyền và bảo vệ dữ liệu HTTPS dễ dàng hơn.
* **Phát triển nhanh chóng:** Phát triển nhanh có thể chứng minh là rất có lợi cho các dự án cần được quay vòng trong một thời gian ngắn, chẳng hạn như trong vòng hai đến ba tháng. Trong mô hình phát triển này, tập trung nhiều hơn vào các nhiệm vụ phát triển và tạo mẫu thay vì lập kế hoạch. ASP.NET Core linh hoạt và thích ứng với các thay đổi đồng thời giảm rủi ro tổng thể của dự án, mã hóa thủ công và lỗi đồng thời.
* **Khả năng di động tốt hơn:** ASP.NET Core có tính di động cao và tính di động có thể làm giảm đáng kể chi phí phát triển web. Khung cho phép di chuyển ứng dụng dễ dàng giữa các máy chủ. Ngoài ra, nhiều nhà phát triển có thể làm việc trên ứng dụng cùng một lúc.

## Kết chương

# XÂY DỰNG ỨNG DỤNG DI ĐỘNG

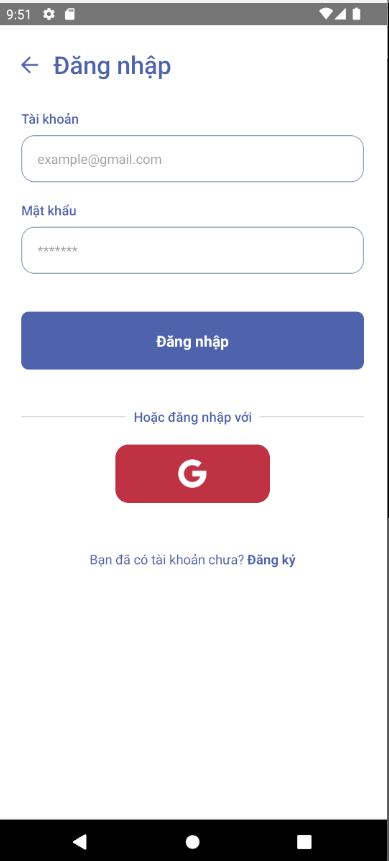
## Thiết kế giao diện

### Giao diện giới thiệu

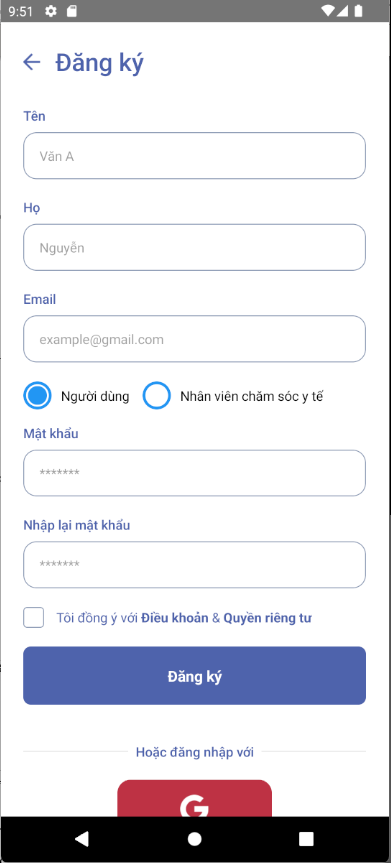


Hình 4.1 Giao diện giới thiệu ứng dụng

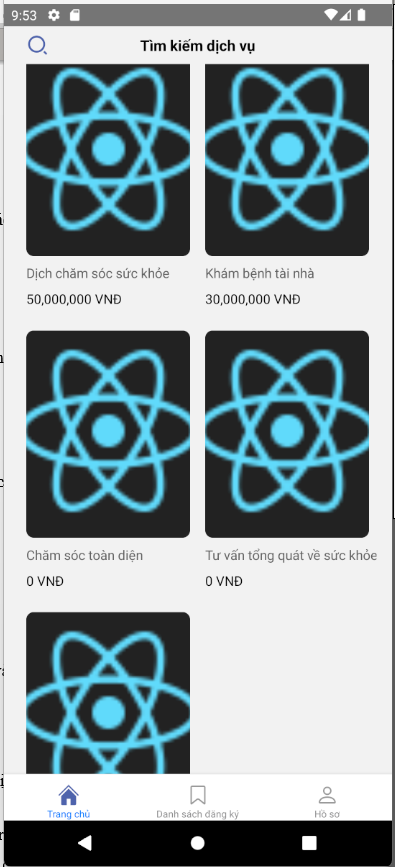
### Giao diện đăng nhập

  
  
Hình 4.2 Giao diện đăng nhập

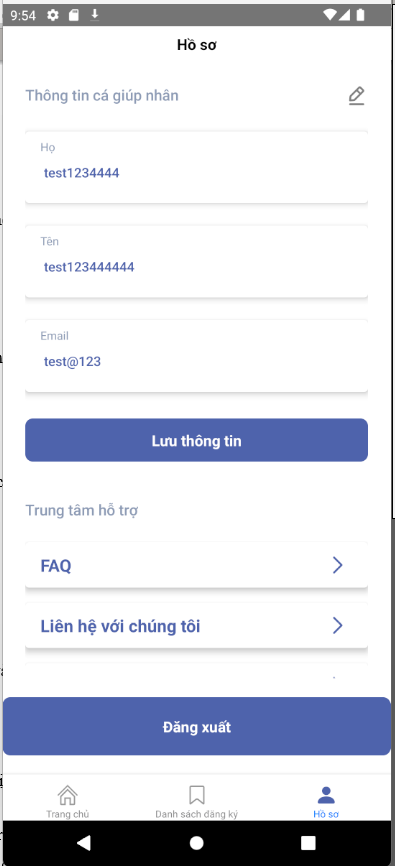
### Giao diện đăng ký

  
  
Hình 4.3 Giao diện đăng ký

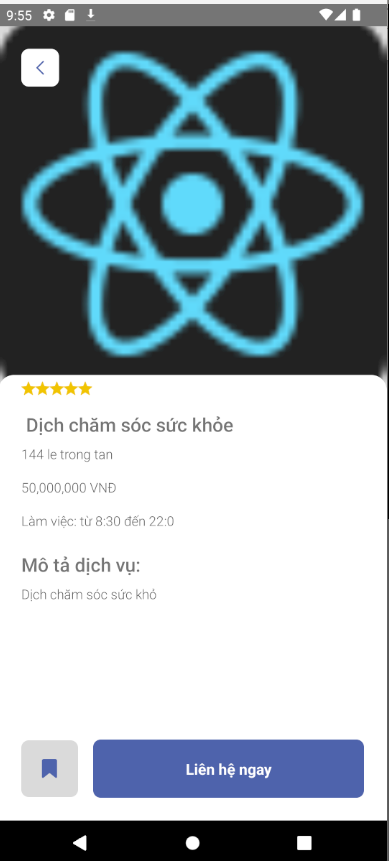
### Giao diện trang chủ người dùng

  
  
Hình 4.4 Giao diện trang chủ người dùng

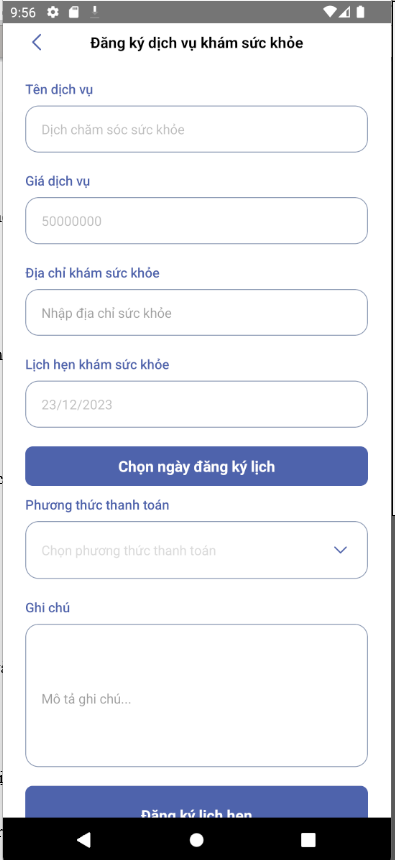
### Giao diện cập nhật thông tin tài khoản

  
  
Hình 4.5 Giao diện cập nhật thông tin tài khoản

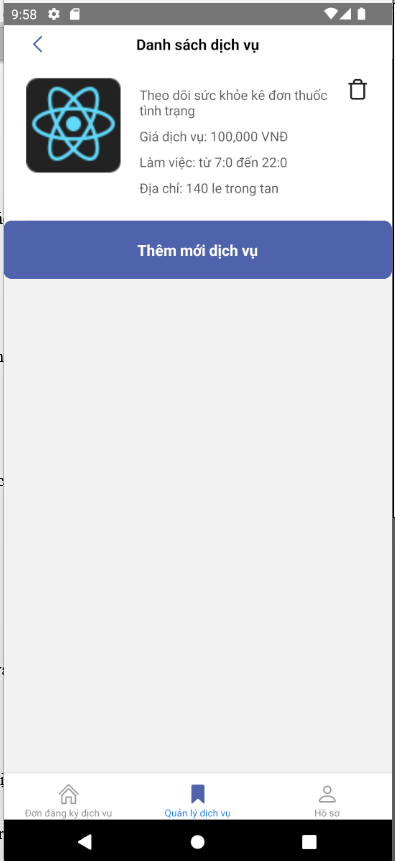
### Giao diện chi tiết dịch vụ

  
  
Hình 4.6 Giao diện chi tiết dịch vụ

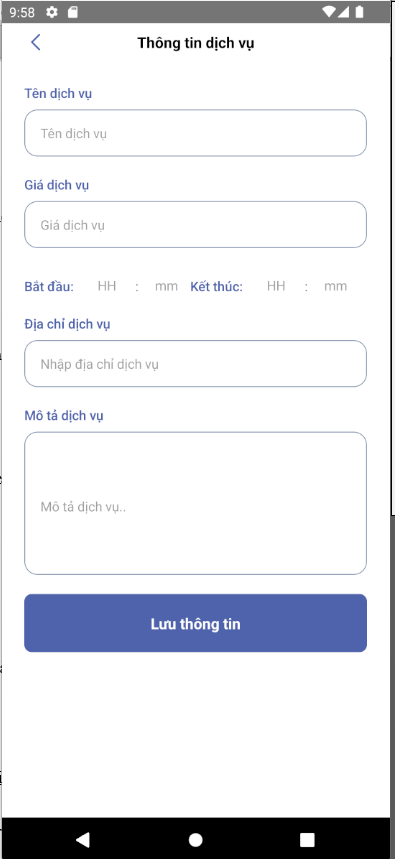
### Giao diện đăng ký khám bệnh

  
  
Hình 4.7 Giao diện đăng ký khám bệnh

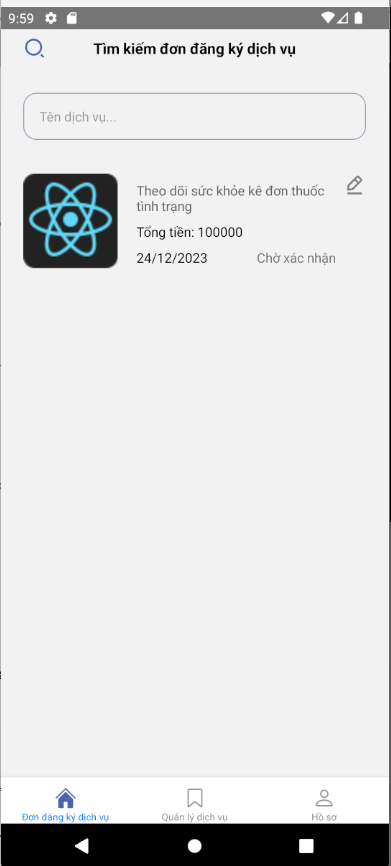
### Giao diện quản lý dịch vụ

  
  
Hình 4.8 Giao diện quản lý dịch vụ

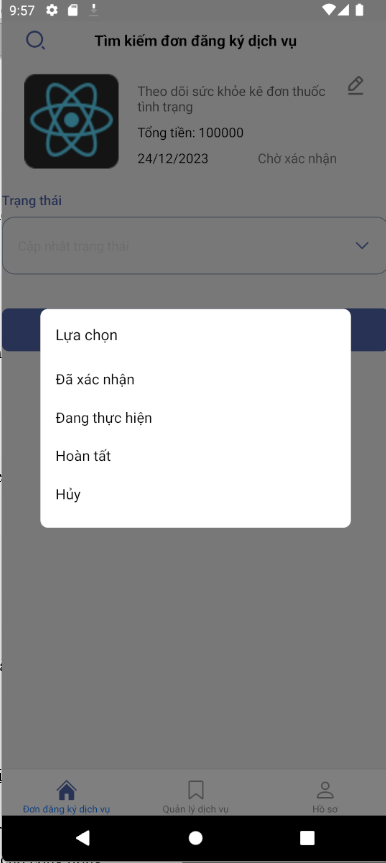
### Giao diện cung cấp thông tin dịch vụ

  
  
Hình 4.9 Giao diện cung cấp thông tin dịch vụ

### Giao diện quản lý danh sách đăng ký khám bệnh

  
  
Hình 4.10 Giao diện quản lý danh sách đăng ký khám bệnh

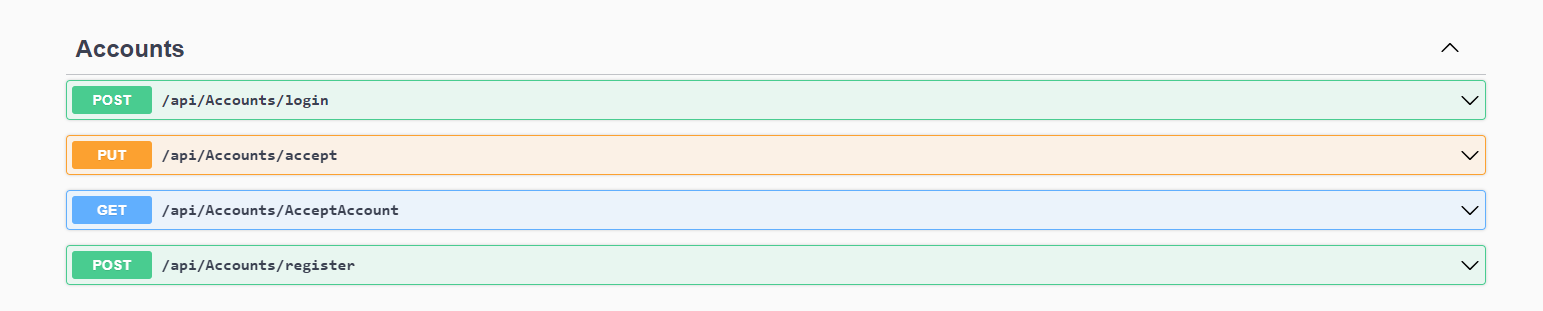
### Giao diện cập nhật trạng thái đăng ký khám bệnh



Hình 4.11 Giao diện cập nhật trạng thái đăng ký khám bệnh

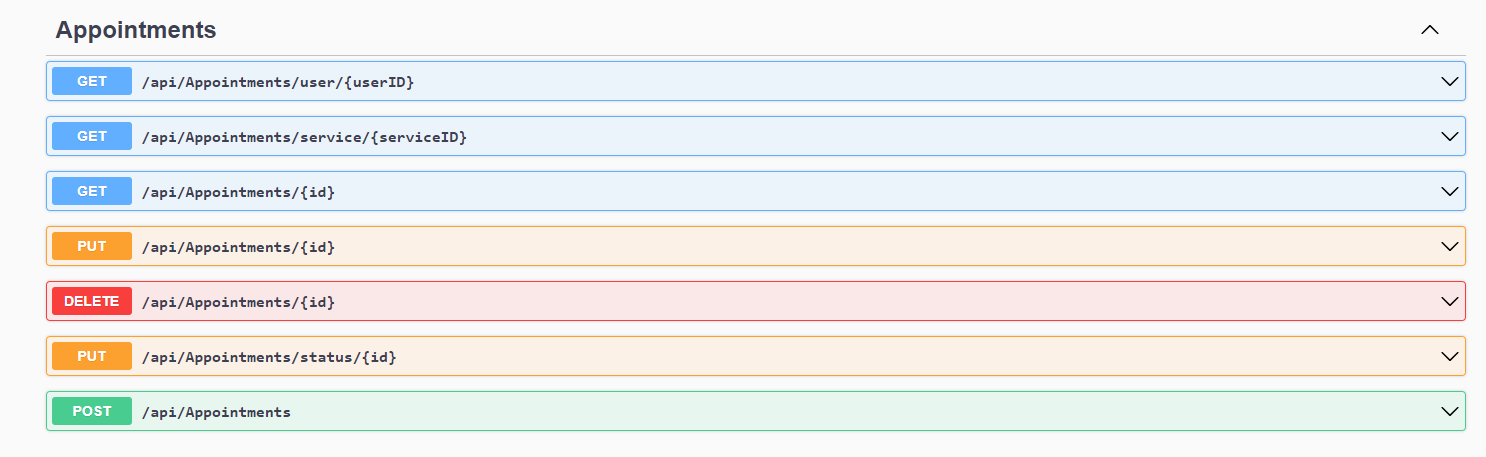
## Xây dựng API

### API Quản lý tài khoản



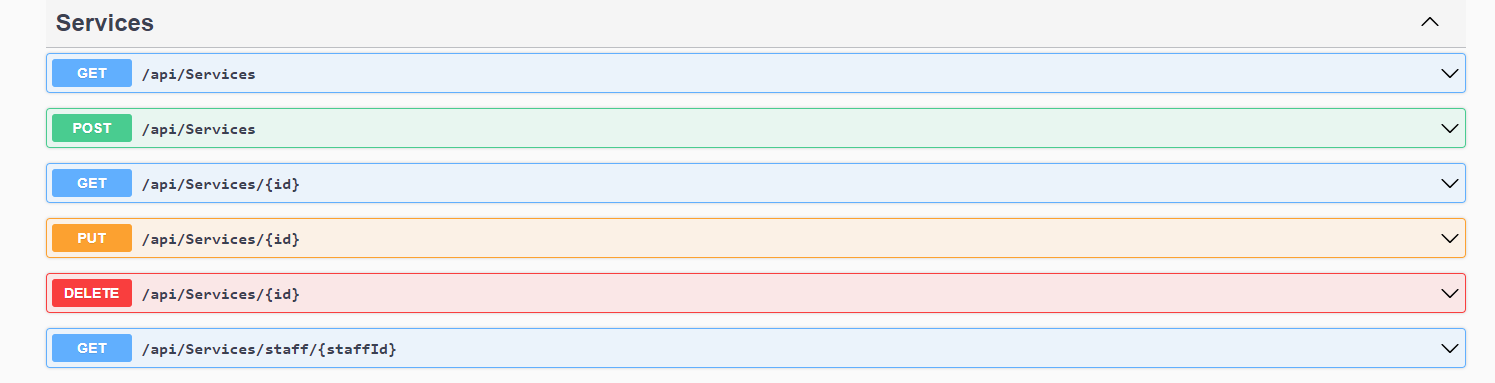
|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| Accounts | POST | /api/Accounts/login | Xác thực người dùng và bắt đầu đăng nhập |
| PUT | /api/Accounts/accept | Cấp quyền truy cập cho tài khoản |
| GET | /api/Accounts/AcceptAccount | Lấy danh sách các tài khoản chờ cấp quyền |
| POST | /api/Accounts/register | Đăng ký tài khoản mới |

### API Quản lý đăng ký dịch vụ



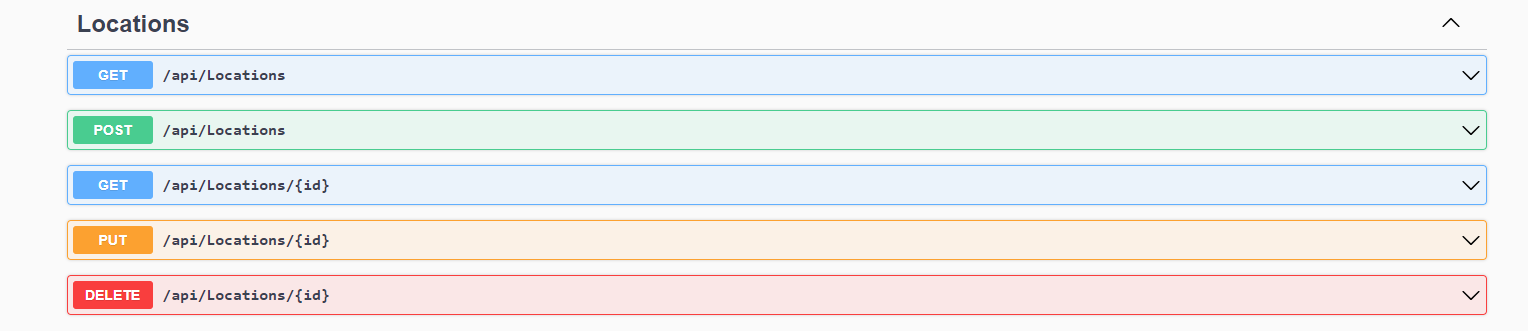
|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| Appointments | GET | /api/Appointments/user/{userID} | Lấy danh sách đăng ký lịch khám cho người dùng |
| GET | /api/Appointments/service/{serviceID} | Lấy danh sách đăng ký lịch khám theo dịch vụ |
| GET | /api/Appointments/{id} | Lấy thông tin chi tiết đăng ký lịch khám |
| PUT | /api/Appointments/{id} | Cập nhật thông tin chi tiết đăng ký lịch khám |
| DELETE | /api/Appointments/{id} | Xóa thông tin chi tiết đăng ký lịch khám |
| PUT | /api/Appointments/status/{id} | Cập nhật trạng thái của đăng ký lịch khám cụ thể (ví dụ: "Đã xác nhận", "Đã hoàn thành", v.v.). |
| POST | /api/Appointments | Tạo mới đăng ký khám |

### API Quản lý dịch vụ



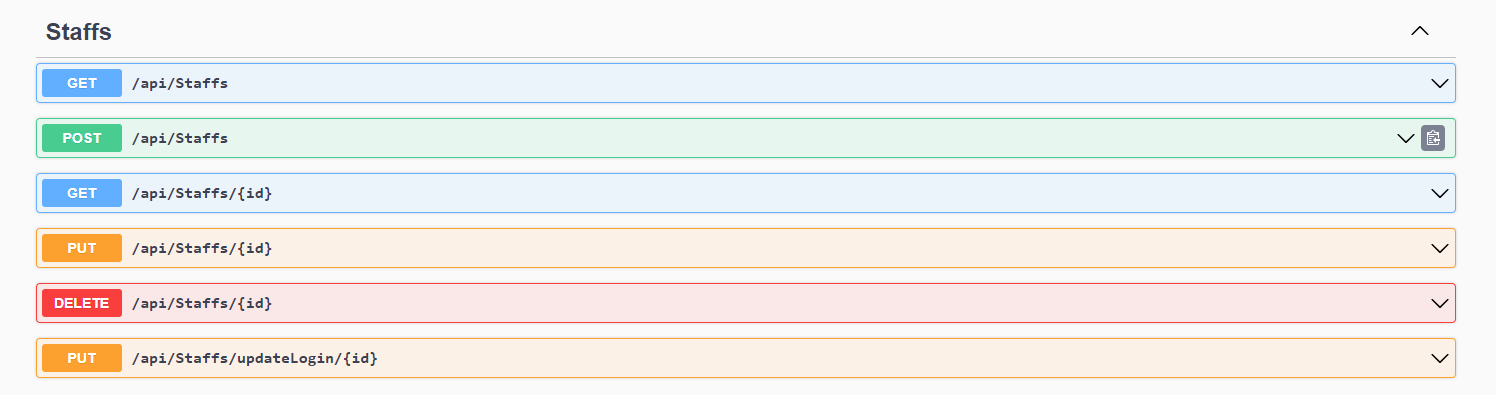
|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| Service | GET | /api/Services | Lấy danh sách tất cả các dịch vụ hiện có |
| POST | /api/Services | Tạo một dịch vụ mới |
| GET | /api/Services/{id} | Lấy thông tin chi tiết về một dịch vụ cụ thể |
| PUT | /api/Services/{id} | Cập nhật thông tin của một dịch vụ cụ thể |
| DELETE | /api/Services/{id} | Xóa một dịch vụ cụ thể |
| GET | /api/Services/staff/{staffId} | Lấy danh sách dịch vụ tho nhân viên chăm sóc y tế |

### API Quản lý địa điểm



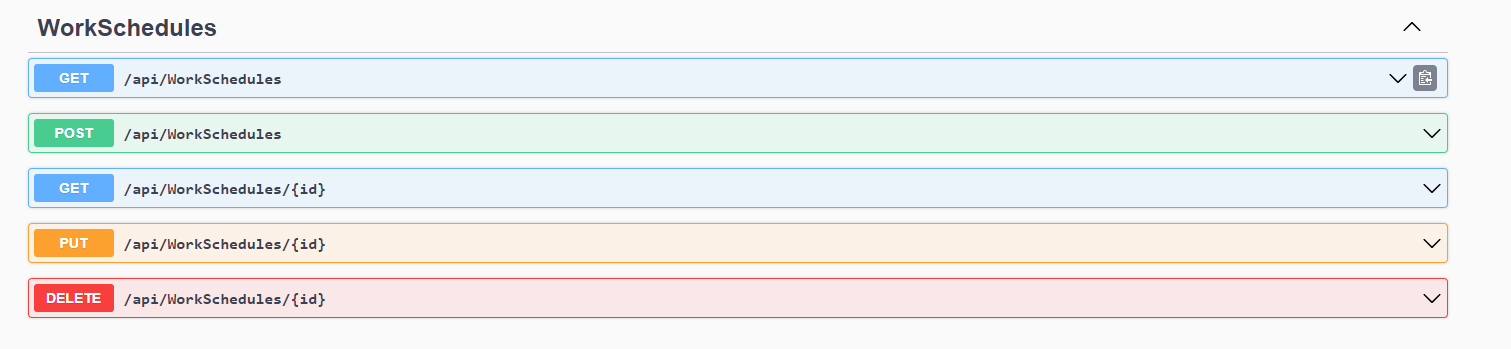
|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| Locations | GET | /api/Locations | Lấy danh sách tất cả các địa điểm hiện có |
| POST | /api/Locations | Tạo một địa điểm mới |
| GET | /api/Locations/{id} | Lấy thông tin chi tiết về một địa điểm cụ thể |
| PUT | /api/Locations/{id} | Cập nhật thông tin của một địa điểm cụ thể |
| DELETE | /api/Locations/{id} | Xóa một địa điểm cụ thể |

### API Quản lý nhân viên



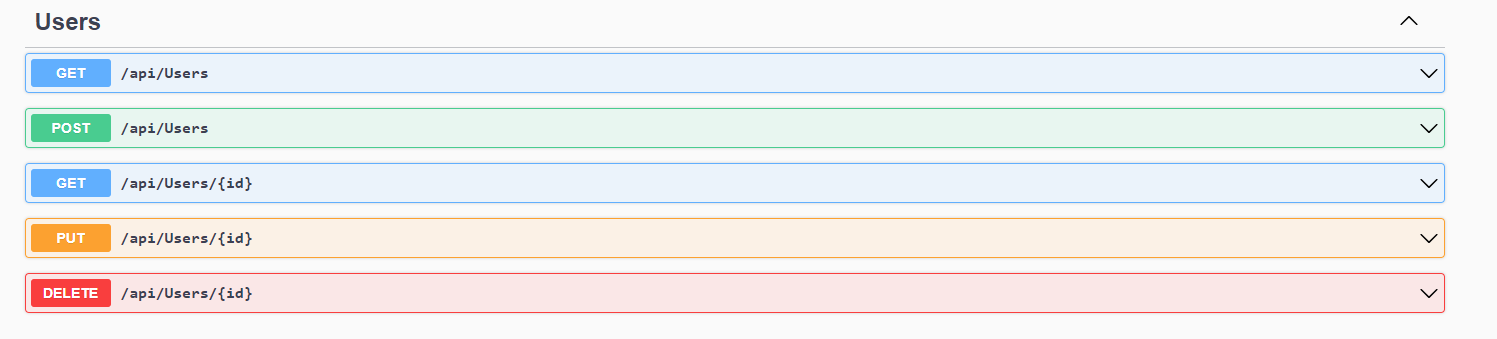
|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| Staffs | GET | /api/Staffs | Lấy danh sách tất cả các nhân viên chăm sóc y tế hiện có |
| POST | /api/Staffs | Tạo một nhân viên chăm sóc y tế mới |
| GET | /api/Staffs/{id} | Lấy thông tin chi tiết về một nhân viên chăm sóc y tế cụ thể |
| PUT | /api/Staffs/{id} | Cập nhật thông tin của một nhân viên chăm sóc y tế cụ thể |
| DELETE | /api/Staffs/{id} | Xóa một nhân viên chăm sóc y tế cụ thể |
| PUT | /api/Staffs/updateLogin/{id} | Cập nhật trạng thái đăng nhập cho tài khoản nhân viên chăm sóc y tế |

### API quản lý lịch làm việc



|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| WorkSchedules | GET | /api/WorkSchedules | Lấy danh sách tất cả các lịch làm việc hiện có |
| POST | /api/WorkSchedules | Tạo một lịch làm việc mới |
| GET | /api/WorkSchedules | Lấy thông tin chi tiết về một lịch làm việc cụ thể |
| PUT | /api/WorkSchedules/{id} | Cập nhật thông tin của một lịch làm việc cụ thể |
| DELETE | /api/WorkSchedules/{id} | Xóa một lịch làm việc cụ thể |

### API Quản lý thông tin người dùng



|  |  |  |  |
| --- | --- | --- | --- |
| **Controller** | **Method** | **Đường dẫn** | **Mô tả** |
| Users | GET | /api/Users | Lấy danh sách tất cả các người dùng hiện có |
| POST | /api/Users | Tạo một người dùng mới |
| GET | /api/Users/{id} | Lấy thông tin chi tiết về một người dùng cụ thể |
| PUT | /api/Users/{id} | Cập nhật thông tin của một người dùng cụ thể |
| DELETE | /api/Users/{id} | Xóa một người dùng cụ thể |

## Kết chương

Trong chương này, nhóm em đã tìm hiểu về hai khía cạnh quan trọng của xây dựng ứng dụng di động, đó là thiết kế giao diện và xây dựng API.

Thiết kế giao diện là quá trình tạo ra giao diện người dùng cho ứng dụng. Giao diện người dùng là thứ mà người dùng nhìn thấy và tương tác khi sử dụng ứng dụng. Nó cần phải dễ sử dụng, trực quan và phù hợp với mục tiêu của ứng dụng.

Xây dựng API là quá trình tạo ra giao diện lập trình ứng dụng cho ứng dụng. API cho phép các ứng dụng khác giao tiếp với ứng dụng của chúng ta. Nó cần phải rõ ràng, dễ sử dụng và có đầy đủ các chức năng cần thiết.

# KẾT LUẬN

## Kết luận

Việc xây dựng hệ thống đăng dịch vụ chăm sóc y tế trực tuyến là một giải pháp hiệu quả để giải quyết các khó khăn, bất cập trong việc đăng ký dịch vụ chăm sóc y tế hiện nay. Hệ thống này mang lại nhiều lợi ích cho cả khách hàng và nhân viên chăm sóc y tế, cụ thể như sau:

* Với khách hàng:
  + Tiện lợi, nhanh chóng, dễ dàng tìm kiếm và đăng ký dịch vụ chăm sóc y tế.
  + Không cần phải đến trực tiếp cơ sở y tế để đăng ký dịch vụ.
  + Có thể so sánh giá cả và chất lượng dịch vụ của các nhân viên chăm sóc y tế khác nhau.
  + Nhận được thông báo về lịch hẹn và kết quả chăm sóc y tế.
* Với nhân viên chăm sóc y tế:
  + Tăng khả năng tiếp cận khách hàng.
  + Quảng bá dịch vụ của mình đến nhiều người hơn.
  + Tiết kiệm thời gian và công sức trong việc tiếp nhận và xử lý các yêu cầu đăng ký dịch vụ.

## Hướng phát triển tương lai

Mặc dù hệ thống đã được hoàn thành nhưng vẫn còn một số hạn chế cần được khắc phục và phát triển thêm trong tương lai, cụ thể như sau:

* Nâng cấp mở rộng hệ thống:
  + Thêm các chức năng mới, đáp ứng nhu cầu ngày càng đa dạng của khách hàng và nhân viên chăm sóc y tế.
  + Tăng cường khả năng bảo mật và an toàn thông tin.
  + Tối ưu hóa hiệu suất và khả năng mở rộng của hệ thống.
* Ứng dụng công nghệ mới:
  + Tích hợp trí tuệ nhân tạo và chatbot để tự động hóa các tác vụ và hỗ trợ khách hàng 24/7.
  + Sử dụng công nghệ blockchain để lưu trữ và bảo mật thông tin.
  + Áp dụng các công nghệ tiên tiến khác để nâng cao chất lượng dịch vụ.

Với những hướng phát triển trên, hệ thống đăng dịch vụ chăm sóc y tế trực tuyến sẽ ngày càng hoàn thiện và trở thành một giải pháp chăm sóc sức khỏe hiệu quả cho cộng đồng.

Một số ý kiến cụ thể cho hướng phát triển tương lai của hệ thống đăng dịch vụ chăm sóc y tế trực tuyến:

* Nâng cấp mở rộng hệ thống:
  + Thêm chức năng đặt lịch hẹn trực tuyến cho khách hàng.
  + Thêm chức năng thanh toán trực tuyến cho dịch vụ chăm sóc y tế.
  + Thêm chức năng đánh giá chất lượng dịch vụ của khách hàng.
  + Thêm chức năng quản lý lịch sử chăm sóc y tế của khách hàng.
* Ứng dụng công nghệ mới:
  + Tích hợp trí tuệ nhân tạo để tự động nhận diện các triệu chứng bệnh của khách hàng.
  + Sử dụng công nghệ blockchain để lưu trữ thông tin sức khỏe của khách hàng một cách an toàn và bảo mật.
  + Áp dụng các công nghệ thực tế ảo (VR) và thực tế tăng cường (AR) để cung cấp các dịch vụ chăm sóc y tế từ xa.

# TÀI LIỆU THAM KHẢO

**Tiếng Việt**

[1] Jeremy Osborn, nhóm AGI Creative, HTML5 và CSS3 Thiết kế trang web thích ứng giàu tính năng, 2019

[2] Phạm Nguyễn Cương, Nguyễn Trần Minh, Hồ Bảo Quốc, Giáo Trình Phân Tích Thiết Kế Hệ Thống Thông Tin Theo Hướng Đối Tượng, Nhà xuất bản khoa học và kỹ thuật, 2016.

**Tiếng Anh**

[3] [Robin Wieruch](https://www.google.com/search?sxsrf=AJOqlzV17Zn1bXyDw8Qm74soPJ_5WZf4PQ:1675953164030&q=Robin+Wieruch&stick=H4sIAAAAAAAAAONgVuLVT9c3NEwzKagoNMkrecRowS3w8sc9YSn9SWtOXmPU5OIKzsgvd80rySypFJLmYoOyBKX4uVB18ixi5Q3KT8rMUwjPTC0qTc4AADQp2L5cAAAA&sa=X&ved=2ahUKEwit6YGI1Ij9AhUi5XMBHRSrALAQzIcDKAB6BAgmEAE), The Road to React: Your journey to master plain yet pragmatic React.js, 2022.

**Website**

[4] <https://www.hostinger.vn/huong-dan/react-la-gi-va-no-hoat-dong-nhu-the-nao>

[5] <https://reactjs.org/>

[6] <https://learn.microsoft.com/en-us/aspnet/core/release-notes/aspnetcore-6.0?view=aspnetcore-6.0>

[7] <https://fullstack.edu.vn/>