ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: CÔNG NGHỆ THÔNG TIN – NGÔN NGỮ NHẬT

ĐỀ TÀI

ỨNG DỤNG HỖ TRỢ KHÁM CHỮA BỆNH TẠI BỆNH VIỆN TÍCH HỢP AI

Người hướng dẫn: **PGS. TS. NGUYỄN THANH BÌNH**

Sinh viên thực hiện: **NGUYỄN VĂN DŨNG**

Số thẻ sinh viên: **102210356**

Lớp: **21TCLC_Nhat2**

Đà Nẵng, 6/2025

ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: CÔNG NGHỆ THÔNG TIN – NGÔN NGỮ NHẬT

ĐỀ TÀI

ỨNG DỤNG HỖ TRỢ KHÁM CHỮA BỆNH TẠI BỆNH BỆNH VIỆN TÍCH HỢP AI

Người hướng dẫn: **PGS. TS. NGUYỄN THANH BÌNH**

Sinh viên thực hiện: **NGUYỄN VĂN DŨNG**

Số thẻ sinh viên: **102210356**

Lớp: **21TCLC_Nhat2**

Đà Nẵng, 6/2025

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP

1. Thông tin chung:

1. Họ và tên sinh viên: Nguyễn Văn Dũng
2. Lớp: 21TCLC_NHAT2 Số thẻ SV: 102210356
3. Tên đề tài: Ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI
4. Người hướng dẫn: PGS. TS. Nguyễn Thanh Bình
   Học hàm/ học vị: Phó giáo sư, Thạc sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp:

1. Về tính cấp thiết, tính mới, khả năng ứng dụng của đề tài: (điểm tối đa là 2đ)

   ……..

   ……..

2. Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đồ án: (điểm tối đa là 4đ)

   ……..

   ……..

3. Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp: (điểm tối đa là 2đ)

   ……..

   ……..

4. Đề tài có giá trị khoa học/ có bài báo/ giải quyết vấn đề đặt ra của doanh nghiệp hoặc nhà trường: (điểm tối đa là 1đ)

   ……..

   ……..

5. Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:

   ……..

   ……..

III. Tinh thần, thái độ làm việc của sinh viên: (điểm tối đa 1đ)

……..

IV. Đánh giá:

1. Điểm đánh giá: ……../10 (lấy đến 1 số lẻ thập phân)
2. Đề nghị: ☐ Được bảo vệ đồ án ☐ Bổ sung để bảo vệ ☐ Không được bảo vệ

Đà Nẵng, ngày tháng năm 2025

Người hướng dẫn

BẢN TÓM TẮT BẰNG TIẾNG NHẬT

AI 統合病院診療支援アプリ

AI integrated hospital medical support app

名前：NGUYEN VAN DUNG（グエン・ヴァン・ズン）

学生番号：10221356

指導教官：NGUYEN THANH BINH

所属：ダナン大学・工科大学・情報学部

Abstract: This AI-integrated hospital support app digitally stores patients' medical records, replacing traditional paper booklets prone to loss. It enables online appointment booking and queue management to reduce waiting times. The app suggests medical departments based on symptoms and assists doctors with disease prediction to improve diagnosis efficiency. It also automates answers to common patient questions, easing hospital staff workload. Designed for patients, doctors, and hospitals, this application enhances service quality and hospital workflow.

1. はじめに

現代社会において、患者の診療情報を管理する効率的な方法が求められています。従来、診療記録は紙の診察券やカルテに保存されており、紛失や検索の手間が問題となっていました。

また、多くの患者が病院で長時間待つ必要があり、予約や診療順の管理が不十分なため、不便さを感じています。さらに、診療に関する問い合わせが多く、医療スタッフの負担も増大しています。

本プロジェクトでは、AI 技術を活用した病院向け診療支援アプリを開発します。このアプリは、患者の診療情報を安全に管理し、症状から受診すべき診療科を提案するとともに、オンライン予約や診療予定の通知機能を提供します。

また、よくある質問に対する自動応答機能により、医療スタッフの負担軽減も目指します。

本アプリを通じて、患者・医療機関・医師の双方にとって利便性の高い診療環境を実現することを目的としています。

2. システム分析、設計

2.1. 機能分析

患者:

- 症状入力：患者が症状を入力し、受診すべき診療科を提案。
- オンライン予約：提案された診療科の予約（順番取り）をオンラインで実施（主要機能）。
- 診療予定通知：予約した診療日時の通知を受け取る。
- 診療状況確認：診察室の受付番号の進行状況をリアルタイムで確認。
- 診療履歴閲覧：過去の診療履歴を閲覧可能。
- 処方箋閲覧：医師が入力した処方箋を確認。

- 質問投稿：病院への質問をチャットボットまたはフォーラムで投稿。

医師:

- 診療履歴確認：患者の過去の診療履歴を閲覧。
- 診察進行更新：現在の受付番号や診察状況を更新。
- 診断・処方：患者に対する診断結果と処方箋の作成。
- 患者からの質問対応：患者からの質問に回答。

管理者:

- 質問対応：準備された FAQ にない質問に対して回答。
- 診療予約管理：患者の診療予約情報を管理。
- 医師情報管理：病院の医師情報を管理。
- 患者情報管理：患者の基本情報と診療情報を管理。

2.2. ユースケース図

図 1　概要のユースケース図

- データベース: MongoDB
- ソースコード管理: Git、Github。
- 導入環境: Ubuntu 22.10 x64。

3.2. 結果

モバイルアプリ

2.3. データベース図

図 2　データベース図

3. 実験と結果

3.1. 実験環境

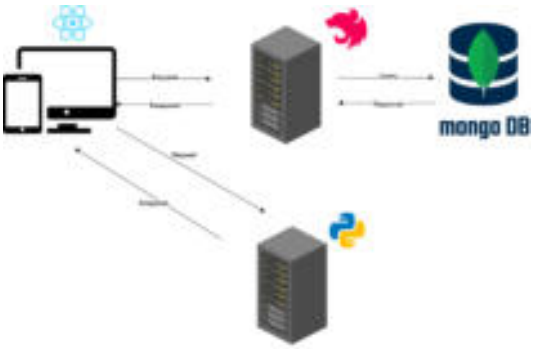


図 3　システム概要

- 開発ツール: Visual Studio Code、Postman。
- バックエンド開発言語: NestJS, TypeScript
- フロントエンド開発言語: ReactJS, JavaScript, React Native, TypeScript
- AI 開発言語: FlaskAPI, Python
- プログラムを実行する環境:
- Node.js: バージョン v20.19.2
- MongoDB: バージョン v6.0.17

- ようこそ画面

図 4 ようこそ画面

- ホーム画面

図 5 ホーム画面

- 予約作成画面

図 6 予約作成画面

- 専門分野詳細画面

図 7　専門部や詳細画面

ウェブサイト

ホーム視聴画面

図 8　ホーム視聴画面

医師クリニック画面

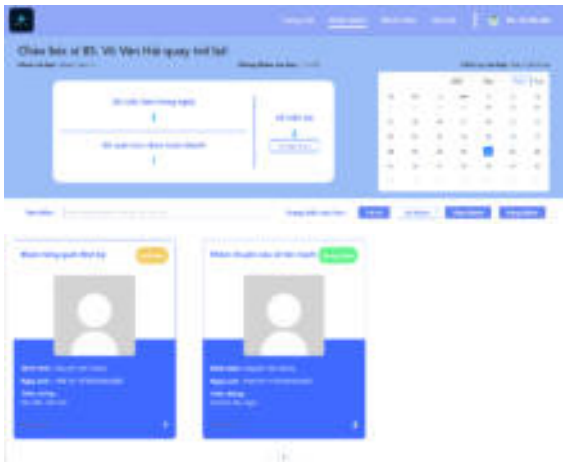


図 9　医師クリニック画面

- 患者一覧画面

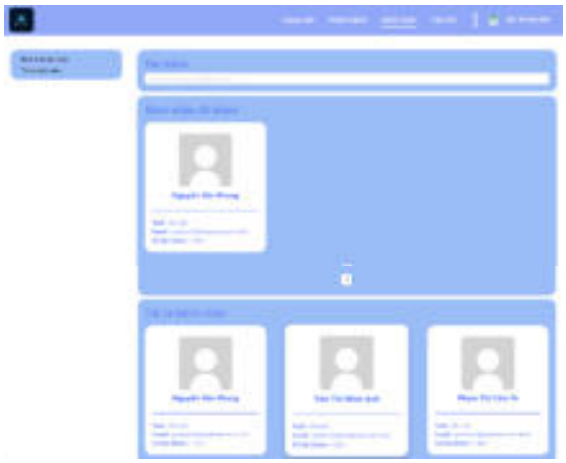


図 10　患者一覧画面

- 質問一覧画面

図 11　質問一覧画面

4. 結論と開発の方向性

4.1. 達成結果

理論研究とシステム構築の過程を通じて、本プロジェクトは当初設定された目標を概ね達成し、病院における診療活動に関するいくつかの現実的な課題の解決に貢献しました。

理論について:

- 病院の実際の運用プロセスに即したデータベース設計、ユーザー要件の特定、データモデリングに関する分析・設計スキルが向上しました。
- TypeScript、JavaScript、Python といったオブジェクト指向プログラミングの知識や、NestJS、Flask、ReactJS、React Native などのフレームワークを活用し、高いインタラクティブ性を持ち、患者・医師・病院スタッフのニーズに対応したシステムを構築しました。

申請に関して:

- システムは、患者が初期症状を入力することで適切な診療科を提案し、案内スタッフの負担軽減と時間短縮に貢献します。
- オンライン診療予約機能により、患者は事前に予約・受付番号を取得でき、診療時間が近づくと自動通知が届くため、病院での待ち時間が大幅に短縮されます。
- 医師は、患者が事前に入力した症状とシステムによる病気の可能性分析を参考にすることで、診断の精度とスピードを高めることができます。
- 診療情報（症状、診断結果、処方箋、再診予定など）が一元的に保存され、従来の紙の診察手帳のような紛失リスクを軽減し、いつでも参照可能です。

- チャットボットおよびFAQ管理システムを統合し、病院によく寄せられる質問に自動対応することで、カスタマーサポート業務の負担を軽減します。

4.2. 欠点

- 院内マップによる案内機能や服薬リマインダーなどの高度な機能は、開発期間の制約によりまだ十分に統合されていません。
- ユーザーインターフェースは機能面では満たしているものの、より直感的かつ親しみやすいデザインに最適化する必要があります。
- 現段階ではシステムの規模が小さく、性能も完全には最適化されていないため、同時アクセスが集中した場合に応答速度が低下する可能性があります。
- 診療予約管理や患者からの質問対応といった病院向けの一部管理機能は、まだ基本的なレベルにとどまっており、今後の開発段階でさらに充実させる必要があります。

4.3. 開発の方向性

- 患者の症状データや診療履歴をもとに、AIが診療科の案内や事前のリスク評価を行う機能を強化し、医師の診断をより迅速かつ的確にサポートします。
- 来院前に患者が現在の混雑状況や予想待ち時間を確認できるようにし、時間の有効活用と病院内の混雑緩和を実現します。
- スマートフォンから診療後の処方内容や再診日、医師からのアドバイスを確認できる個人用のヘルスダッシュボードを構築します。
- 問い合わせが多い項目に対応するだけでなく、患者の質問意図を理解して柔軟に返答できるAIチャットシステムを開発し、対応精度を高めます。

TÓM TẮT

Tên đề tài: Ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI

Sinh viên thực hiện: Nguyễn Văn Dũng

Số thẻ SV: 102210356 Lớp: 21TCLT_NHAT2

Tóm tắt đề tài:

Trong thời đại công nghệ số hiện nay, nhu cầu nâng cao chất lượng dịch vụ y tế và tối ưu hóa trải nghiệm khám chữa bệnh cho người dân ngày càng trở nên cấp thiết. Tuy nhiên, quy trình khám bệnh truyền thống vẫn còn tồn tại nhiều bất cập, như việc người dân phải chờ đợi xếp hàng dài để lấy số, lưu trữ thông tin khám chữa bệnh bằng sổ giấy dễ thất lạc, hay việc bệnh viện phải dành nhiều nhân lực để trả lời những câu hỏi lặp đi lặp lại. Trong bối cảnh đó, việc phát triển một ứng dụng hỗ trợ khám chữa bệnh tích hợp trí tuệ nhân tạo (AI) trở thành một giải pháp hữu ích và thực tiễn.

Ứng dụng này sẽ giúp người dùng dễ dàng nhập triệu chứng để được gợi ý chuyên khoa phù hợp, đồng thời có thể đặt lịch khám trước một cách nhanh chóng ngay trên điện thoại. Hệ thống sẽ tự động nhắc nhở người dùng khi đến gần thời gian khám, đồng thời cho phép họ theo dõi tiến trình gọi số tại bệnh viện trong thời gian thực, giúp tiết kiệm đáng kể thời gian chờ đợi.

Trong quá trình khám, bác sĩ có thể truy cập dữ liệu triệu chứng mà người bệnh đã khai báo trước đó, cùng với phân tích AI về các khả năng bệnh lý liên quan để đưa ra chẩn đoán chính xác hơn. Sau khi khám, toàn bộ thông tin như đơn thuốc, ngày tái khám, và lịch sử điều trị sẽ được lưu trữ trên hệ thống, giúp người bệnh dễ dàng tra cứu khi cần.

Ngoài ra, ứng dụng còn tích hợp chatbot thông minh để trả lời các câu hỏi thường gặp từ bệnh nhân, dựa trên bộ dữ liệu đã được bệnh viện chuẩn bị. Những câu hỏi chưa có sẵn sẽ được chuyển tiếp cho đội ngũ nhân viên y tế để phản hồi sau, góp phần giảm tải áp lực cho bệnh viện.

Bằng cách này, ứng dụng không chỉ là một công cụ hỗ trợ khám chữa bệnh thông minh mà còn là cầu nối hiệu quả giữa bệnh nhân, bác sĩ và bệnh viện, góp phần nâng cao chất lượng dịch vụ y tế, tiết kiệm thời gian và nguồn lực, đồng thời thúc đẩy quá trình chuyển đổi số trong lĩnh vực y tế một cách toàn diện.

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Nguyễn Văn Dũng Số thẻ sinh viên: 102210356

Lớp: 21TCLC_NHAT2 Khoa: Công nghệ thông tin Ngành: CNTT Việt-Nhật

1. *Tên đề tài đồ án:*
   Ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI
2. *Đề tài thuộc diện: ☐ Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện*
3. *Các số liệu và dữ liệu ban đầu:*
   Không có
4. *Nội dung các phần thuyết minh và tính toán:*
   Nội dung các phần thuyết minh bao gồm:
   - Tổng quan đề tài: Giới thiệu tổng quan, mục đích và ý nghĩa của đề tài, công nghệ, kỹ thuật, công cụ được sử dụng và những kết quả dự kiến đạt được.
   - Cơ sở lý thuyết: Trình bày những cơ sở lý thuyết được áp dụng trong đề tài.
   - Phân tích thiết kế hệ thống: Phân tích các yêu cầu về chức năng và phi chức năng của hệ thống và triển khai thiết kế hệ thống thông qua các biểu đồ thiết kế hệ thống.
   - Triển khai thực tế: Trình bày môi trường triên khai và kết quả thực tế đạt được.
   - Kết luận và hướng phát triển: Đánh giá kết quả đạt được, chưa đạt được và đưa ra những định hướng phát triển thêm trong tương lai.
   - Tài liệu tham khảo: Liệt kê các tài liệu tham khảo sử dụng trong đề tài.
5. *Các bản vẽ, đồ thị (ghi rõ các loại và kích thước bản vẽ):*
   Không có
6. *Họ tên người hướng dẫn:* PGS. TS. Nguyễn Thanh Bình
7. *Ngày giao nhiệm vụ đồ án:*
8. *Ngày hoàn thành đồ án:*

Đà Nẵng, ngày tháng năm 2025.

Trưởng Bộ môn …….. **Người hướng dẫn**

GVHD: PGS. TS. Nguyễn Thanh Bình

LỜI NÓI ĐẦU

Đề tài "Ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI" là sản phẩm của quá trình nghiên cứu và ứng dụng kiến thức chuyên ngành trong suốt thời gian học tập tại Khoa Công nghệ Thông tin – Trường Đại học Bách khoa – Đại học Đà Nẵng. Đây cũng là đồ án tốt nghiệp đánh dấu bước trưởng thành và sự hoàn thiện kỹ năng của em trong lĩnh vực công nghệ thông tin.

Trong bối cảnh y tế hiện đại ngày càng chú trọng đến việc nâng cao hiệu quả khám chữa bệnh và tối ưu hóa trải nghiệm người bệnh, việc ứng dụng trí tuệ nhân tạo và công nghệ số trong quản lý và hỗ trợ khám chữa bệnh là nhu cầu cấp thiết. Trên tinh thần đó, em đã lựa chọn phát triển một ứng dụng thông minh nhằm hỗ trợ người bệnh trong việc đặt lịch khám, lưu trữ hồ sơ y tế và nhận thông tin kịp thời, đồng thời giúp giảm tải công việc cho nhân viên y tế và bác sĩ.

Em xin gửi lời cảm ơn sâu sắc đến PGS. TS. Nguyễn Thanh Bình, người đã tận tâm hướng dẫn, góp ý và hỗ trợ em trong suốt quá trình thực hiện đề tài. Những chỉ dẫn và động viên quý báu của thầy là nguồn động lực lớn để em hoàn thành đồ án này.

Em cũng xin chân thành cảm ơn Ban giám hiệu cùng các thầy cô trong Khoa Công nghệ Thông tin đã truyền đạt kiến thức, tạo điều kiện học tập tốt nhất để em có thể phát triển bản thân và hoàn thành đồ án.

Mặc dù đã cố gắng hoàn thiện tốt nhất, em nhận thấy đề tài còn tồn tại nhiều hạn chế do kinh nghiệm và thời gian nghiên cứu có phần hạn hẹp. Em rất mong nhận được sự góp ý chân thành từ quý thầy cô và các bạn để có thể tiếp tục hoàn thiện và phát triển hơn trong tương lai.

Cuối cùng, em xin kính chúc quý thầy cô và các anh chị, các bạn luôn dồi dào sức khỏe, thành công trong công việc và cuộc sống. Em hy vọng Khoa Công nghệ Thông tin – Trường Đại học Bách khoa – Đại học Đà Nẵng sẽ không ngừng phát triển, tiếp tục là môi trường học tập và nghiên cứu lý tưởng cho các thế hệ sinh viên. Em xin trân trọng cảm ơn

Sinh viên thực hiện

Nguyễn Văn Dũng

LỜI CAM ĐOAN

Em xin cam đoan:

1. Nội dung trong đồ án này em thực hiện dưới sự hướng dẫn trực tiếp của PGS. TS. Nguyễn Thanh Bình. Các kết quả trong đồ án đều được thực hiện trong quá trình em đang học tập tại Khoa Công Nghệ Thông Tin, trường Đại học Bách Khoa Đà Nẵng.
2. Các tham khảo dùng trong đồ án đều được trích dẫn rõ ràng tên tác giả, tên công trình, thời gian, địa điểm công bố, được kèm với đường dẫn hợp lệ.
3. Nếu có những sao chép không hợp lệ, vi phạm quy chế đào tạo, em xin chịu hoàn toàn trách nhiệm.

Đà Nẵng, ngày……tháng……năm 2025

Sinh viên thực hiện

Nguyễn Văn Dũng

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Từ	Viết tắt của	Diễn giải
API	Application Programming Interface	Giao diện lập trình ứng dụng
CSS	Cascading Style Sheets	Ngôn ngữ định dạng phần tử trang
JS	Javascript	Ngôn ngữ lập trình kịch bản
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
EC2	Amazon Elastic Compute Cloud	Cở sở hạ tầng điện toán đám mây
JSX	JavaScript XML	JavaScript được mở rộng để hỗ trợ viết mã HTML tương tự XML
DOM	Document Object Model	Mô hình đối tượng tài liệu
DB	Database	Cơ sở dữ liệu

MỞ ĐẦU

1. Giới thiệu

Trong thời đại công nghệ phát triển nhanh chóng như hiện nay, việc chăm sóc sức khỏe ngày càng được mọi người quan tâm nhiều hơn. Tuy nhiên, không ít người vẫn gặp khó khăn khi đi khám chữa bệnh, từ việc phải xếp hàng chờ đợi lâu đến việc lưu giữ hồ sơ khám bệnh một cách rắc rối. Chính vì vậy, dự án Ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI ra đời với mong muốn giúp mọi người thuận tiện hơn trong quá trình khám chữa bệnh.

Ứng dụng này không chỉ giúp người bệnh dễ dàng nhập thông tin về triệu chứng, nhận gợi ý khoa khám phù hợp mà còn cho phép đặt lịch khám trước ngay trên điện thoại, tránh phải chờ đợi lâu ở bệnh viện. Bên cạnh đó, bác sĩ cũng sẽ nhận được sự hỗ trợ từ hệ thống AI giúp phân tích triệu chứng và lưu trữ thông tin khám chữa bệnh chính xác, từ đó nâng cao hiệu quả công việc và chất lượng chăm sóc bệnh nhân.

Đặc biệt, ứng dụng còn có chức năng chatbot trả lời những câu hỏi phổ biến của bệnh nhân, giúp giải đáp nhanh chóng và giảm tải áp lực cho đội ngũ y bác sĩ. Qua đó, người dùng sẽ cảm thấy yên tâm hơn khi đi khám, còn bệnh viện và bác sĩ sẽ làm việc hiệu quả, khoa học hơn.

Hy vọng rằng với sự hỗ trợ của công nghệ AI, dự án sẽ góp phần mang đến trải nghiệm khám chữa bệnh tiện lợi, nhanh chóng và thân thiện hơn cho mọi người, đồng thời góp phần hiện đại hóa hệ thống y tế trong tương lai.

2. Mục đích và ý nghĩa đề tài

2.1. Mục đích

Mục đích của ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI là xây dựng một nền tảng công nghệ tiên tiến nhằm tối ưu hóa quy trình khám chữa bệnh, đồng thời nâng cao chất lượng dịch vụ y tế, mang lại sự thuận tiện và trải nghiệm tốt nhất cho tất cả các bên liên quan: bệnh nhân, bác sĩ và bệnh viện.

- **Đối với bệnh nhân,** ứng dụng hướng tới việc giảm thiểu tối đa những phiền toái thường gặp khi đến bệnh viện như thời gian chờ đợi lâu, phải xếp hàng nhiều lần, hay phải đi lại nhiều lần để làm thủ tục. Thông qua việc nhập trước các triệu chứng và đặt lịch khám trực tuyến, bệnh nhân có thể chủ động sắp xếp thời gian phù hợp, đồng thời nhận được sự tư vấn nhanh chóng từ hệ thống

AI, giúp họ hiểu rõ hơn về tình trạng sức khỏe và có sự chuẩn bị tốt trước khi đến gặp bác sĩ. Điều này không chỉ giúp tiết kiệm thời gian mà còn giảm bớt áp lực tâm lý, nâng cao sự hài lòng khi sử dụng dịch vụ y tế.

- **Đối với bệnh viện,** ứng dụng giúp giảm tải áp lực vận hành bằng cách tự động xử lý và trả lời các câu hỏi thường gặp từ bệnh nhân qua chatbot AI. Hệ thống có khả năng học hỏi và cải thiện theo thời gian, từ đó giảm thiểu số lượng câu hỏi lặp đi lặp lại mà nhân viên y tế phải trả lời, giúp đội ngũ y bác sĩ và nhân viên bệnh viện tập trung hơn vào công tác khám chữa bệnh chuyên sâu và các hoạt động chăm sóc bệnh nhân khác. Ngoài ra, việc quản lý lịch khám và dữ liệu bệnh nhân cũng trở nên đồng bộ và hiệu quả hơn, góp phần nâng cao hiệu suất làm việc của toàn hệ thống.
- **Đối với bác sĩ,** ứng dụng cung cấp một công cụ hỗ trợ đắc lực giúp chuẩn bị thông tin về bệnh nhân trước khi tiến hành khám trực tiếp. Khi bệnh nhân nhập triệu chứng và các thông tin y tế cơ bản trước, hệ thống AI sẽ phân tích và đưa ra các nhận định ban đầu, giúp bác sĩ có cái nhìn tổng quan về tình trạng sức khỏe cũng như các dấu hiệu cần chú ý. Điều này giúp rút ngắn thời gian khám bệnh, tăng tính chính xác trong chẩn đoán và đưa ra phương án điều trị phù hợp hơn. Đồng thời, bác sĩ cũng có thể quản lý lịch khám và theo dõi tiến trình điều trị một cách thuận tiện thông qua ứng dụng.

Ngoài ra, ứng dụng còn hướng đến việc tạo ra một môi trường khám chữa bệnh thông minh và kết nối đa chiều, giúp các bên liên quan dễ dàng trao đổi, tương tác và hỗ trợ nhau một cách hiệu quả nhất. Việc tích hợp công nghệ AI không chỉ dừng lại ở khả năng tư vấn hay phân tích triệu chứng, mà còn mở ra nhiều cơ hội phát triển các tính năng mới trong tương lai như chẩn đoán hình ảnh, phân tích dữ liệu y tế lớn (Big Data), hỗ trợ ra quyết định y khoa... từ đó góp phần nâng cao chất lượng và hiệu quả chăm sóc sức khỏe cộng đồng.

Tóm lại, ứng dụng hỗ trợ khám chữa bệnh tại bệnh viện tích hợp AI không chỉ là một công cụ kỹ thuật mà còn là cầu nối thông minh, giúp rút ngắn khoảng cách giữa bệnh nhân và bác sĩ, giảm bớt gánh nặng cho hệ thống y tế, đồng thời nâng cao trải nghiệm khám chữa bệnh, góp phần phát triển nền y tế hiện đại, thân thiện và hiệu quả hơn.

2.2. Ý nghĩa

Ứng dụng hỗ trợ khám chữa bệnh tích hợp AI mang một ý nghĩa quan trọng không chỉ trong lĩnh vực y tế mà còn đối với sự phát triển của xã hội và công nghệ hiện đại.

Trong bối cảnh dân số ngày càng gia tăng cùng với nhu cầu chăm sóc sức khỏe ngày càng cao, việc ứng dụng công nghệ thông minh vào quy trình khám chữa bệnh là một bước tiến thiết yếu để nâng cao chất lượng dịch vụ y tế và đảm bảo sự công bằng trong tiếp cận chăm sóc sức khỏe.

Về mặt xã hội, ứng dụng giúp giảm thiểu tình trạng quá tải tại các bệnh viện, hạn chế tình trạng xếp hàng chờ đợi kéo dài, giúp bệnh nhân có trải nghiệm thân thiện, nhanh chóng và thuận tiện hơn khi đến khám. Qua đó, góp phần giảm bớt áp lực cho nhân viên y tế, giúp họ tập trung nhiều hơn vào việc chăm sóc và điều trị hiệu quả cho bệnh nhân.

Về mặt công nghệ, việc tích hợp AI trong ứng dụng khám chữa bệnh thể hiện xu hướng ứng dụng trí tuệ nhân tạo để hỗ trợ và nâng cao hiệu suất công việc trong ngành y tế. AI giúp phân tích dữ liệu triệu chứng bệnh một cách nhanh chóng và chính xác, hỗ trợ bác sĩ đưa ra chẩn đoán sớm và phù hợp hơn, góp phần hạn chế sai sót và tăng tính khách quan trong điều trị.

Bên cạnh đó, ứng dụng còn mở ra tiềm năng phát triển các giải pháp y tế thông minh khác trong tương lai như chăm sóc sức khỏe từ xa, dự đoán bệnh tật qua phân tích dữ liệu lớn, hay cá nhân hóa phác đồ điều trị dựa trên thông tin bệnh nhân. Điều này không chỉ giúp hiện đại hóa hệ thống y tế mà còn thúc đẩy sự phát triển bền vững của ngành y trong thời đại số.

Tóm lại, ý nghĩa của dự án không chỉ nằm ở việc ứng dụng công nghệ để giải quyết các vấn đề trước mắt mà còn hướng tới sự thay đổi toàn diện và lâu dài trong cách thức khám chữa bệnh, tạo tiền đề cho một nền y tế hiện đại, hiệu quả và nhân văn hơn.

3. Các bước triển khai

Giai đoạn 1: Thiết kế cơ sở dữ liệu và giao diện, bao gồm việc xây dựng cấu trúc database phù hợp với hệ thống và thiết kế giao diện người dùng cơ bản cho các nền tảng Backend, Frontend và Mobile.

Giai đoạn 2 : Khởi tạo dự án và chuẩn bị dữ liệu, bao gồm việc tạo cấu trúc thư mục cho toàn hệ thống và xây dựng dữ liệu mẫu để phục vụ quá trình phát triển chức năng.

Giai đoạn 3 : Xây dựng giao diện ứng dụng mobile cho người dùng với đầy đủ các chức năng như đăng ký, đăng nhập, đặt lịch hẹn, xem lịch sử khám bệnh, thông báo, chỉnh sửa thông tin cá nhân...

Giai đoạn 4 : Tạo và kết nối API cho các chức năng trên mobile app, đảm bảo hoạt động mượt mà giữa frontend và backend, bao gồm các chức năng như lịch hẹn, thông báo, chuyên khoa, chat, lịch sử khám...

Giai đoạn 5 : Xây dựng mô hình AI để nhận biết bệnh thông qua triệu chứng và tích hợp chatbot Gemini hỗ trợ người dùng đặt câu hỏi liên quan đến tình trạng sức khỏe.

Giai đoạn 6 : Xây dựng giao diện cho bác sĩ và admin, bao gồm các chức năng như xem lịch khám, quản lý bệnh nhân, bác sĩ, khoa, câu hỏi và thông tin cá nhân.

Giai đoạn 7 : Tạo và kết nối API cho giao diện bác sĩ và admin, đảm bảo các chức năng hoạt động đúng yêu cầu.

Giai đoạn 8 : Viết báo cáo và chạy thử hệ thống song song để kiểm tra, đánh giá hiệu quả và hoàn thiện sản phẩm.

4. Công nghệ và kỹ thuật sử dụng

Các công nghệ và kỹ thuật được sử dụng

- Front-end: ReactJs,SCSS.
- Mobile: ReactNative, Redux.
- Back-end: NestJS.
- Cloud : Cloudinary.
- Database: MongoDB.
- Quản lý mã nguồn: Git, Github.
- Quản lý server: Azura.
- Công cụ: Visual Studio Code, Postman.

5. Những kết quả đạt được dự kiến

5.1. Về mặt lý thuyết

- Củng cố kiến thức về quá trình phân tích, thiết kế và phát triển một hệ thống phần mềm phục vụ trong lĩnh vực y tế.
- Vận dụng linh hoạt các kiến thức đã học để xây dựng cơ sở dữ liệu, thiết kế giao diện người dùng và lập trình các chức năng cho từng nhóm đối tượng sử dụng như bệnh nhân, bác sĩ, quản trị viên.
- Nắm rõ cách tổ chức và triển khai dự án phần mềm theo từng giai đoạn cụ thể, từ lên kế hoạch, thiết kế, phát triển đến thử nghiệm và hoàn thiện sản phẩm.

5.2. Về mặt ứng dụng

- Xây dựng được một ứng dụng hỗ trợ khám chữa bệnh: Ứng dụng cung cấp đầy đủ các tính năng cần thiết như đăng ký tài khoản, đặt lịch khám, xem lịch sử khám bệnh, cập nhật thông tin cá nhân và thông tin chuyên khoa.
- Hỗ trợ tối ưu hóa quy trình hoạt động trong bệnh viện: Hệ thống giúp quản lý thông tin bệnh nhân, bác sĩ, lịch khám và khoa phòng một cách tập trung và hiệu quả.
- Cải thiện khả năng tiếp cận dịch vụ y tế cho người dùng: Ứng dụng cho phép người bệnh dễ dàng tra cứu thông tin, chủ động đặt lịch và cập nhật tình trạng sức khỏe mọi lúc, mọi nơi.
- Góp phần nâng cao trải nghiệm người dùng: Với giao diện thân thiện, trực quan, ứng dụng giúp người dùng thao tác dễ dàng và có cái nhìn tổng quan về quá trình chăm sóc sức khỏe cá nhân.

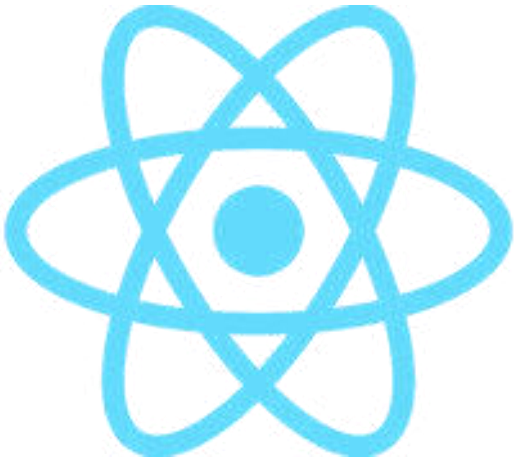
6. Nội dung đồ án

Đồ án bao gồm nội dung sau:

- ***Mở đầu*** *- Giới thiệu tổng quan, mục đích và ý nghĩa của đề tài, công nghệ, kỹ thuật, công cụ được sử dụng và những kết quả dự kiến đạt được.*
- ***Chương 1: Cơ sở lý thuyết*** *– Trình bày những cơ sở lý thuyết được áp dụng trong đề tài.*
- ***Chương 2: Phân tích thiết kế hệ thống*** *– Phân tích các yêu cầu về chức năng và phi chức năng của hệ thống và triển khai thiết kế hệ thống thông qua các biểu đồ thiết kế hệ thống.*
- ***Chương 3: Triển khai thực tế*** *- Trình bày môi trường triên kai và kết quả thực tế đạt được.*
- ***Kết luận và hướng phát triển*** *– Đánh giá kết quả đạt được, chưa đạt được và đưa ra những định hướng phát triển thêm trong tương lai.*
- ***Tài liệu tham khảo*** *– Liệt kê các tài liệu tham khảo sử dụng trong đề tài.*

CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về ReactJs và ReactNative



Hình 1.1 Logo của React

ReactJS là một thư viện JavaScript mã nguồn mở do Facebook phát triển, được sử dụng rộng rãi để xây dựng giao diện người dùng theo hướng component – tức là chia nhỏ giao diện thành các phần tử độc lập, có thể tái sử dụng. Với kiến trúc linh hoạt và hiệu quả, ReactJS giúp việc phát triển ứng dụng web trở nên dễ dàng, dễ mở rộng và bảo trì.

Trong quá trình làm đồ án, em sử dụng ReactJS kết hợp với JavaScript và công cụ Vite để tối ưu tốc độ khởi tạo và phát triển ứng dụng. Vite hỗ trợ cơ chế cập nhật module nóng (Hot Module Replacement), giúp cập nhật giao diện nhanh chóng mà không cần tải lại toàn bộ trang.

Các lợi ích và điểm mạnh của ReactJS gồm:

- **Phát triển giao diện theo hướng component:** Giúp tổ chức mã nguồn gọn gàng, dễ mở rộng và tái sử dụng.
- **Sử dụng Virtual DOM:** Cải thiện hiệu suất bằng cách cập nhật phần thay đổi mà không thao tác trực tiếp trên DOM thật.
- **JSX trực quan:** Kết hợp HTML và JavaScript trong một cú pháp giúp dễ viết, dễ đọc và thuận tiện trong quá trình phát triển.
- **Luồng dữ liệu một chiều (One-way data binding):** Dễ kiểm soát trạng thái và giảm khả năng phát sinh lỗi logic.
- **Tối ưu hiệu suất với Vite:** Thời gian build ngắn, tốc độ phản hồi nhanh và dễ cấu hình.
- **Quản lý trạng thái linh hoạt:** Dễ dàng tích hợp với Redux hoặc Context API để xử lý dữ liệu hiệu quả.

- **Hỗ trợ SEO tốt:** Khi kết hợp với React Helmet, có thể tùy chỉnh metadata để cải thiện khả năng hiển thị trên công cụ tìm kiếm.
- **Cộng đồng lớn:** Nhiều tài liệu, công cụ hỗ trợ và thư viện mở rộng giúp học và ứng dụng nhanh chóng.

React Native là một framework do Facebook phát triển, cho phép lập trình viên xây dựng ứng dụng di động cho cả Android và iOS chỉ từ một mã nguồn duy nhất. Dựa trên nền tảng tư duy component giống như ReactJS, React Native giúp tối ưu quá trình phát triển ứng dụng di động với hiệu suất gần như native.

Trong đồ án, em sử dụng React Native kết hợp với TypeScript – ngôn ngữ lập trình mở rộng từ JavaScript có hỗ trợ kiểm tra kiểu dữ liệu – cùng với Redux để quản lý trạng thái toàn cục một cách chặt chẽ và hiệu quả.

Các điểm mạnh của React Native gồm:

- **Phát triển đa nền tảng:** Một codebase có thể triển khai trên cả Android và iOS, tiết kiệm thời gian và công sức.
- **Tái sử dụng logic và UI component:** Có thể sử dụng lại nhiều phần từ ReactJS và chia sẻ logic giữa web và mobile.
- **Hiệu suất gần với ứng dụng native:** Ứng dụng được biên dịch sang mã native, mang lại trải nghiệm mượt mà.
- **Hỗ trợ TypeScript:** Giúp giảm lỗi, tăng độ an toàn và dễ bảo trì khi dự án lớn.
- **Quản lý trạng thái hiệu quả:** Kết hợp với Redux giúp xử lý dữ liệu phức tạp một cách logic và rõ ràng.
- **Hot Reloading:** Hỗ trợ cập nhật giao diện nhanh trong quá trình phát triển, không cần build lại ứng dụng.
- **Cộng đồng phát triển mạnh:** Có nhiều thư viện hỗ trợ, plugin mở rộng, tài liệu học tập phong phú.

1.2. Tổng quan về NestJS

Hình 1.2. Logo của NestJS

NestJS là một framework phát triển ứng dụng Node.js được xây dựng dựa trên TypeScript, sử dụng kiến trúc hướng đối tượng và chịu ảnh hưởng mạnh từ các nguyên lý của Angular. NestJS cung cấp cho lập trình viên một cách tiếp cận hiện đại và có tổ chức khi phát triển các ứng dụng web, đặc biệt là các ứng dụng theo kiến trúc RESTful API và microservices.

NestJS mang đến nhiều tính năng nổi bật, hỗ trợ quá trình phát triển hiệu quả và bền vững, bao gồm:

- **Kiến trúc rõ ràng, mô-đun hóa:** NestJS tổ chức mã nguồn dựa trên các mô-đun (modules), giúp chia nhỏ ứng dụng thành các phần độc lập và dễ quản lý. Mỗi module có thể chứa controller, service và provider riêng, góp phần nâng cao khả năng mở rộng và tái sử dụng mã.
- **Hỗ trợ TypeScript toàn diện:** NestJS được viết hoàn toàn bằng TypeScript, một ngôn ngữ mở rộng của JavaScript với tính năng kiểm tra kiểu tĩnh. Việc sử dụng TypeScript giúp phát hiện lỗi ngay từ lúc biên dịch, cải thiện độ an toàn và độ tin cậy của mã nguồn trong các dự án lớn.
- **Phát triển RESTful API hiệu quả:** NestJS cung cấp các công cụ và decorator mạnh mẽ như @Controller(), @Get(), @Post() giúp định nghĩa rõ ràng các endpoint và xử lý các yêu cầu HTTP một cách hiệu quả. Nhờ đó, việc xây dựng và quản lý RESTful API trở nên nhanh chóng và có tổ chức.
- **Hỗ trợ kiến trúc microservices:** NestJS cung cấp sẵn các module hỗ trợ phát triển ứng dụng theo kiến trúc microservices như giao tiếp qua message broker (Kafka, Redis, RabbitMQ, MQTT...), từ đó tăng khả năng mở rộng và tính linh hoạt trong hệ thống phân tán.
- **Hệ sinh thái tích hợp mạnh mẽ:** NestJS dễ dàng tích hợp với nhiều công nghệ phổ biến như TypeORM, Prisma (ORM), Mongoose (MongoDB), Swagger (tạo tài liệu API), và GraphQL. Điều này giúp rút ngắn thời gian tích hợp và tăng hiệu quả phát triển.

- **Quản lý phụ thuộc thông minh với Dependency Injection (DI):** NestJS áp dụng nguyên lý Dependency Injection giúp quản lý và cung cấp các thành phần trong ứng dụng một cách linh hoạt. Cách tiếp cận này không những làm cho mã dễ kiểm thử mà còn giảm sự phụ thuộc cứng trong các lớp.
- **Bảo mật tích hợp sẵn:** NestJS hỗ trợ các giải pháp bảo mật phổ biến như xác thực JWT, bảo vệ route với Guards, kiểm soát truy cập qua Role-Based Access Control (RBAC), từ đó tăng cường khả năng phòng ngừa rủi ro bảo mật cho ứng dụng web.
- **Dễ dàng kiểm thử:** Với sự hỗ trợ sẵn từ các công cụ như Jest, NestJS cho phép viết unit test và integration test một cách dễ dàng nhờ kiến trúc phân tách rõ ràng giữa các layer (Controller, Service, Repository...).
- **Tài liệu rõ ràng, cộng đồng đang phát triển:** NestJS có tài liệu chính thức chi tiết, dễ tra cứu, cùng với cộng đồng lập trình viên tích cực chia sẻ, hỗ trợ trên các diễn đàn, GitHub và Stack Overflow.
- **Tích hợp với công cụ phát triển hiện đại:** NestJS hoạt động tốt với các công cụ như npm, Yarn, Docker và CI/CD pipelines, giúp dễ dàng triển khai, quản lý và vận hành ứng dụng trong môi trường thực tế.
- **Triển khai linh hoạt:** NestJS hỗ trợ triển khai ứng dụng trên nhiều nền tảng như Heroku, Vercel, AWS, Google Cloud hoặc thông qua Docker, phù hợp với nhu cầu đa dạng của các doanh nghiệp.

Với những ưu điểm kể trên, NestJS đã nhanh chóng trở thành một trong những lựa chọn hàng đầu cho việc xây dựng backend của các ứng dụng web hiện đại, đặc biệt trong các dự án có yêu cầu cao về cấu trúc, khả năng mở rộng và bảo trì lâu dài.

1.3. Tổng quan về Flask

Hình 1.3 Logo của Flask

Flask là một micro framework phát triển ứng dụng web bằng ngôn ngữ Python. Flask nổi bật với thiết kế nhẹ, linh hoạt, cho phép lập trình viên xây dựng ứng dụng nhanh chóng và dễ dàng mà không bị ràng buộc về cấu trúc dự án.

Flask cung cấp các chức năng cơ bản như định tuyến URL (routing), xử lý yêu cầu và phản hồi (request/response), cùng hệ thống template Jinja2 giúp tạo giao diện động. Ngoài ra, Flask hỗ trợ mở rộng qua các plugin để tích hợp với cơ sở dữ liệu, xác thực người dùng, và các tính năng nâng cao khác.

Điểm mạnh của Flask bao gồm:

- **Khởi tạo và phát triển ứng dụng nhanh, đơn giản:** Flask có thiết kế tối giản, không yêu cầu nhiều cấu hình phức tạp. Điều này giúp lập trình viên dễ dàng bắt đầu và phát triển ứng dụng một cách nhanh chóng, phù hợp với các dự án nhỏ hoặc prototype.
- **Linh hoạt trong thiết kế và mở rộng:** Flask không áp đặt cấu trúc bắt buộc, cho phép người dùng tự do tổ chức mã nguồn theo cách phù hợp nhất với dự án. Đồng thời, Flask hỗ trợ mở rộng thông qua nhiều plugin và thư viện bên ngoài, giúp thêm các tính năng khi cần.
- **Phù hợp để xây dựng API RESTful và các ứng dụng web nhỏ đến vừa:** Với trọng lượng nhẹ và hiệu suất tốt, Flask rất thích hợp để phát triển các dịch vụ API RESTful hoặc những ứng dụng web không quá phức tạp, nơi tốc độ phát triển và đơn giản hóa là ưu tiên hàng đầu.
- **Cộng đồng hỗ trợ rộng lớn và tài liệu phong phú:** Flask có cộng đồng người dùng đông đảo và nhiều tài liệu, tutorial, ví dụ minh họa giúp người mới dễ tiếp cận và phát triển kỹ năng nhanh chóng.Flask thường được dùng để tạo các ứng dụng prototype hoặc sản phẩm yêu cầu phát triển nhanh, đồng thời vẫn đáp ứng được các nhu cầu mở rộng về sau.

Flask thường được dùng để tạo các ứng dụng prototype hoặc sản phẩm yêu cầu phát triển nhanh, đồng thời vẫn đáp ứng được các nhu cầu mở rộng về sau.

1.4. Tổng quan về mạng LSTM (Long Short-Term Memory)

Mạng Long Short-Term Memory (LSTM) là một loại mạng nơ-ron hồi tiếp (RNN) được thiết kế đặc biệt để xử lý và học từ dữ liệu chuỗi dài. Khác với các mạng RNN truyền thống, LSTM có khả năng ghi nhớ thông tin quan trọng trong một khoảng thời gian dài hơn nhờ cơ chế cổng (gate) giúp kiểm soát việc lưu giữ hoặc loại bỏ thông tin. Điều này giúp LSTM vượt qua hạn chế vanishing gradient thường gặp trong các mạng hồi tiếp truyền thống.

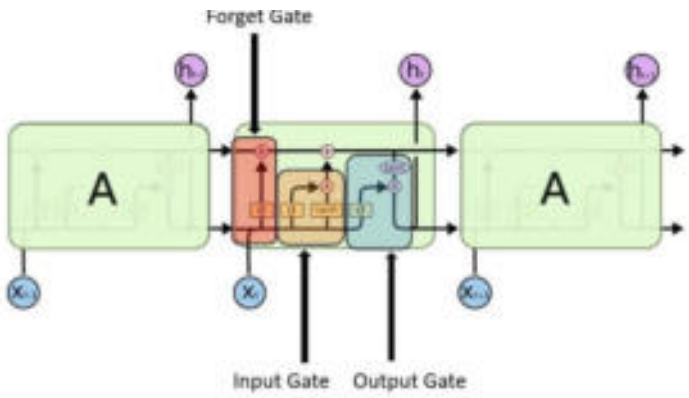
Trong bài toán dự đoán bệnh dựa trên văn bản người dùng nhập, dữ liệu đầu vào thường là các chuỗi ký tự hoặc từ có độ dài và cấu trúc phức tạp. LSTM được sử dụng để phân tích các chuỗi dữ liệu này, ghi nhớ ngữ cảnh và các đặc trưng quan trọng trong câu để từ đó đưa ra dự đoán chính xác về bệnh trạng.

Cơ chế hoạt động của LSTM bao gồm các thành phần cổng như:

- **Cổng quên (Forget Gate):** Quyết định thông tin nào không còn cần thiết và sẽ bị loại bỏ khỏi bộ nhớ.
- **Cổng vào (Input Gate):** Xác định thông tin mới quan trọng cần thêm vào bộ nhớ.
- **Cổng ra (Output Gate):** Điều khiển thông tin nào sẽ được sử dụng để tạo đầu ra tại bước thời gian hiện tại.

Nhờ cấu trúc này, LSTM có khả năng giữ lại những yếu tố quan trọng trong văn bản dài, giúp mô hình hiểu sâu sắc hơn ngữ cảnh và mối quan hệ giữa các từ hoặc câu. Do vậy, LSTM đặc biệt phù hợp với các bài toán xử lý ngôn ngữ tự nhiên như phân loại văn bản, nhận diện cảm xúc, và trong trường hợp này là dự đoán bệnh dựa trên nội dung mà người dùng nhập vào.

Việc ứng dụng LSTM giúp hệ thống không chỉ nhận dạng các từ khóa đơn lẻ mà còn hiểu được toàn bộ ý nghĩa của câu, từ đó hỗ trợ đưa ra dự đoán chính xác và đáng tin cậy hơn, góp phần nâng cao chất lượng dịch vụ khám chữa bệnh.



Hình 1.4 Mô hình mạng LTSM (Long Short-Term Memory)

1.5. Tổng quan về cơ sở dữ liệu MongoDB

Hình 1.5 Logo hệ quản trị cơ sở dữ liệu MongoDB

MongoDB là hệ quản trị cơ sở dữ liệu NoSQL mã nguồn mở, sử dụng mô hình document để lưu trữ dữ liệu. Thay vì sử dụng bảng và hàng như trong cơ sở dữ liệu quan hệ, MongoDB lưu trữ dữ liệu dưới dạng các document có cấu trúc linh hoạt (theo định

dạng BSON – Binary JSON). Các document này được nhóm thành các collection, cho phép lưu trữ thông tin phi cấu trúc hoặc bán cấu trúc một cách hiệu quả.

MongoDB đặc biệt phù hợp cho các ứng dụng có dữ liệu thay đổi thường xuyên, yêu cầu mở rộng linh hoạt hoặc không cần thiết lập quan hệ phức tạp giữa các bảng như trong SQL.

Ưu điểm:

- **Linh hoạt về cấu trúc dữ liệu:** Không yêu cầu schema cố định, dễ dàng thích nghi khi thay đổi yêu cầu dữ liệu.
- **Tốc độ xử lý nhanh:** Tối ưu cho các thao tác truy vấn và ghi dữ liệu lớn.
- **Khả năng mở rộng ngang (horizontal scaling):** Hỗ trợ sharding giúp phân phối dữ liệu trên nhiều máy chủ.
- **Tích hợp tốt với ứng dụng web:** Đặc biệt phù hợp với các ứng dụng hiện đại sử dụng Node.js, Express hoặc NestJS.

Nhược điểm:

- **Hạn chế trong xử lý quan hệ phức tạp:** Không hỗ trợ các join phức tạp như trong cơ sở dữ liệu quan hệ.
- **Quản lý dữ liệu không chặt chẽ:** Do không có schema bắt buộc, dễ phát sinh lỗi dữ liệu không đồng nhất.
- **Tiêu tốn tài nguyên lưu trữ:** Dữ liệu có thể bị trùng lặp giữa các document, gây tốn dung lượng.
- **Bảo mật cần cấu hình kỹ lưỡng:** Nếu không thiết lập đúng, có thể dẫn đến rủi ro bảo mật cao.

MongoDB hiện là một trong những hệ cơ sở dữ liệu NoSQL phổ biến nhất, được sử dụng rộng rãi trong các hệ thống yêu cầu tính mở rộng cao, xử lý dữ liệu lớn hoặc xây dựng các ứng dụng theo kiến trúc Microservices.

1.6. Tổng quan về điện toán đám mây Azura

1.6.1. Dịch vụ Azure Virtual Machines (VM)

Hình 1.6 Logo Azure Virtual Machines

Microsoft Azure là một nền tảng đám mây toàn cầu cung cấp nhiều dịch vụ đa dạng về lưu trữ, tính toán, phát triển và triển khai ứng dụng. Azure được sử dụng rộng rãi bởi các doanh nghiệp, tổ chức lớn và các nhà phát triển để xây dựng và vận hành các hệ thống trên nền điện toán đám mây.

Trong số các dịch vụ của Azure, Azure Virtual Machines (VM) là dịch vụ chủ lực cho phép người dùng tạo và quản lý các máy chủ ảo trên nền tảng điện toán đám mây của Azure. Với Azure VM, người dùng có thể thuê các tài nguyên máy chủ với cấu hình tùy chỉnh, bao gồm số lượng CPU, dung lượng RAM và ổ lưu trữ phù hợp với nhu cầu ứng dụng của mình.

Azure Virtual Machines sử dụng các công nghệ ảo hóa tiên tiến để chạy nhiều máy ảo độc lập trên cùng một tài nguyên phần cứng vật lý, giúp tối ưu hóa hiệu suất và sử dụng tài nguyên. Người dùng có thể lựa chọn hệ điều hành phổ biến như Windows Server, Ubuntu, hoặc các bản phân phối Linux khác để cài đặt trên các VM.

Ngoài việc cung cấp khả năng khởi tạo, dừng, khôi phục và xóa máy ảo theo nhu cầu, Azure VM còn tích hợp các tính năng bảo mật nâng cao như mã hóa dữ liệu ổ đĩa, quản lý truy cập và tường lửa, giúp bảo vệ dữ liệu và ứng dụng trên đám mây.

Một điểm mạnh của Azure VM là khả năng mở rộng linh hoạt, cho phép tự động tăng hoặc giảm số lượng máy ảo dựa trên tải công việc thực tế, giúp tối ưu chi phí và hiệu năng sử dụng.

Azure Virtual Machines cũng dễ dàng tích hợp với các dịch vụ khác trong hệ sinh thái Azure như Azure Storage, Azure SQL Database, Azure Kubernetes Service (AKS), từ đó tạo thành các giải pháp ứng dụng đa dạng và phức tạp trên nền tảng đám mây..

1.6.2. Dịch vụ Azure Blob Storage

Hình 1.7 Logo Azure Blob Storage

Azure Blob Storage là dịch vụ lưu trữ đối tượng của Microsoft Azure, được thiết kế để lưu trữ lượng lớn dữ liệu phi cấu trúc như hình ảnh, video, tài liệu, sao lưu, và dữ liệu ứng dụng. Dịch vụ này cung cấp khả năng lưu trữ với độ bền cao, tính sẵn sàng lớn và khả năng mở rộng linh hoạt trên toàn cầu.

Azure Blob Storage cho phép các nhà phát triển và quản trị hệ thống lưu trữ dữ liệu an toàn, truy xuất nhanh và dễ dàng mở rộng khi cần thiết. Dữ liệu được lưu trữ trong các container và có thể được quản lý theo nhiều cấp độ bảo mật khác nhau, bao gồm mã hóa dữ liệu khi lưu trữ và khi truyền tải.

Dịch vụ này hỗ trợ nhiều lớp lưu trữ khác nhau, từ lớp nóng (hot) dành cho dữ liệu truy cập thường xuyên, đến lớp lạnh (cool) và lớp lưu trữ dài hạn (archive) với chi phí thấp hơn cho dữ liệu ít truy cập.

Azure Blob Storage tích hợp chặt chẽ với các dịch vụ điện toán và phân tích của Azure, giúp xây dựng các ứng dụng lưu trữ và xử lý dữ liệu hiệu quả trên đám mây.

1.7. Kết chương 1

Chương 1 đã trình bày tổng quan các công nghệ và công cụ chính được sử dụng trong quá trình phát triển hệ thống, bao gồm ReactJS và React Native cho giao diện người dùng, NestJS cho xây dựng backend API, Flask phục vụ một số chức năng xử lý chuyên biệt, cũng như mạng LSTM trong bài toán phân tích và dự đoán dựa trên văn bản. Mỗi công nghệ đều mang lại những ưu điểm nổi bật, phù hợp với từng yêu cầu cụ thể trong hệ thống như hiệu suất, khả năng mở rộng, tính linh hoạt và dễ bảo trì.

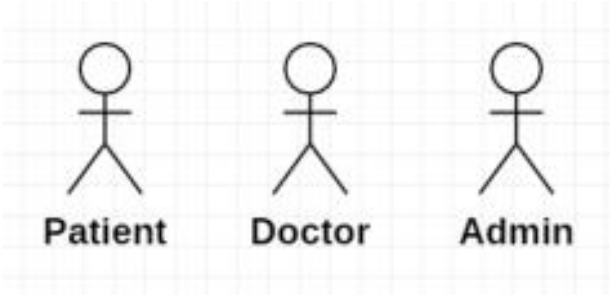
Việc lựa chọn và kết hợp các công nghệ này không chỉ giúp tối ưu hóa quy trình phát triển mà còn đảm bảo chất lượng, khả năng mở rộng cũng như hiệu quả hoạt động của sản phẩm. Những kiến thức nền tảng được trình bày trong chương này sẽ làm cơ sở vững chắc cho các chương tiếp theo, nơi sẽ đi sâu vào phân tích yêu cầu hệ thống, thiết kế kiến trúc, cũng như triển khai và đánh giá hiệu quả mô hình trong thực tế.

CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Các tác nhân chính của hệ thống

Trang web được xây dựng bao gồm 3 tác nhân chính hoạt động:

- Bệnh nhân (Patient)
- Bác sĩ (Doctor)
- Quản trị viên (Admin)



Hình 2.1 Các tác nhân trong hệ thống

2.1.1. Bệnh nhân

Quản lý tài khoản:

- **Đăng nhập:** Cho phép bệnh nhân đăng nhập vào hệ thống bằng cách cung cấp tên đăng nhập và mật khẩu đã đăng ký trước đó.
- **Cập nhật thông tin:** Bệnh nhân có thể cập nhật thông tin cá nhân như tên, email, hình ảnh đại diện, và các thông tin khác liên quan đến tài khoản của họ.
- **Đổi mật khẩu:** Bệnh nhân có thể thay đổi mật khẩu của mình bằng cách cung cấp mật khẩu cũ và nhập mật khẩu mới.
- **Đăng xuất:** Cho phép bệnh nhân đăng xuất khỏi hệ thống, kết thúc phiên làm việc hiện tại.

Sử dụng tài khoản:

- **Tạo lịch hẹn:** Cho phép bệnh nhân nhập vào các triệu chứng hiện tại (như ho, sốt, đau bụng, chóng mặt, v.v.). Hệ thống sử dụng dữ liệu y tế để phân tích và dự đoán loại bệnh có thể mắc, đồng thời gợi ý bệnh nhân nên khám ở chuyên khoa phù hợp. Sau đó, bệnh nhân có thể chọn ngày, giờ và bác sĩ để đặt lịch khám online.
- **Thanh toán lịch hẹn :** Sau khi đặt lịch khám thành công, bệnh nhân có thể tiến hành thanh toán trực tiếp trên ứng dụng thông qua các cổng thanh toán phổ biến

như MoMo hoặc ZaloPay. Hệ thống đảm bảo thông tin thanh toán an toàn và xác nhận lịch hẹn ngay sau khi giao dịch hoàn tất.

- **Xem danh sách các lịch hẹn theo từng ngày :** Giúp bệnh nhân dễ dàng quản lý và theo dõi tất cả lịch hẹn đã đặt, được hiển thị dưới dạng lịch theo ngày. Mỗi lịch hẹn đi kèm thông tin cụ thể như tên bác sĩ, phòng khám, thời gian khám, trạng thái xác nhận.
- **Theo dõi trạng thái phòng khám của bác sĩ :** Hiển thị số thứ tự hiện tại đang được khám tại phòng của bác sĩ mà bệnh nhân đã đặt lịch. Tính năng này giúp bệnh nhân chủ động di chuyển đến phòng khám đúng giờ, giảm thời gian chờ đợi tại bệnh viện.
- **Nhận thông báo :** Gửi các thông báo quan trọng cho bệnh nhân bao gồm: Nhắc lịch hẹn khám sắp tới, Trạng thái xác nhận thanh toán, Câu hỏi & trả lời từ bệnh viện hoặc bác sĩ, Cập nhật thay đổi lịch hoặc hủy lịch khám
- **Xem lịch sử các cuộc hẹn :** Cho phép bệnh nhân truy cập lại toàn bộ lịch sử các lần khám chữa bệnh trước đây, bao gồm: ngày khám, bác sĩ, chuyên khoa, chẩn đoán, đơn thuốc, và các ghi chú liên quan. Giúp bệnh nhân dễ dàng theo dõi tình trạng sức khỏe của mình theo thời gian.
- **Xem sổ khám bệnh :** Tính năng này giống như một quyển sổ khám bệnh điện tử, lưu lại đầy đủ chi tiết từng lần khám, kết quả xét nghiệm, đơn thuốc, lời dặn của bác sĩ và lịch tái khám. Bệnh nhân có thể truy cập mọi lúc mọi nơi mà không sợ mất như sổ giấy truyền thống.
- **Đặt câu hỏi cho bệnh viện :** Bệnh nhân có thể gửi các câu hỏi liên quan đến quy trình khám bệnh, giờ làm việc, chi phí,... đến hệ thống. Ứng dụng sẽ sử dụng chatbot thông minh để trả lời các câu hỏi thường gặp hoặc chuyển đến nhân viên bệnh viện nếu câu hỏi chưa có trong hệ thống.

2.1.2. Bác sĩ

Quản lý tài khoản

- **Đăng nhập:** Cho phép bác sĩ đăng nhập vào hệ thống bằng cách cung cấp tên đăng nhập và mật khẩu đã đăng ký trước đó.
- **Cập nhật thông tin:** Bác sĩ có thể cập nhật thông tin cá nhân như tên, email, hình ảnh đại diện, và các thông tin khác liên quan đến tài khoản của họ.
- **Đổi mật khẩu:** Bác sĩ có thể thay đổi mật khẩu của mình bằng cách cung cấp mật khẩu cũ và nhập mật khẩu mới.
- **Đăng xuất:** Cho phép bác sĩ đăng xuất khỏi hệ thống, kết thúc phiên làm việc hiện tại.

Sử dụng tài khoản

- **Cập nhập trạng thái khám bệnh trong phòng mình:** Cho phép bác sĩ chủ động cập nhật số thứ tự đang được khám hiện tại. Ví dụ: "Đã tới bệnh nhân số 05" để hệ thống hiển thị công khai ra cho bệnh nhân biết. Tính năng này giúp giảm tình trạng chen lấn chờ đợi, hỗ trợ quản lý thời gian tốt hơn.
- **Xem danh sách bệnh nhân có trong hệ thống:** Bác sĩ có thể truy cập và xem danh sách tất cả bệnh nhân đã đăng ký trong hệ thống, bao gồm thông tin cơ bản như họ tên, ngày sinh, số điện thoại, mã bệnh nhân, giúp hỗ trợ tra cứu nhanh khi cần.
- **Xem thông tin bệnh án và danh sách các cuộc hẹn của bệnh nhân:** Khi khám cho một bệnh nhân cụ thể, bác sĩ có thể truy cập hồ sơ bệnh án điện tử của người đó để xem tiền sử bệnh, các chẩn đoán trước đó, đơn thuốc, xét nghiệm đã thực hiện và lịch sử các lần hẹn khám. Giúp bác sĩ nắm rõ tình trạng bệnh lý để chẩn đoán chính xác hơn.
- **Tạo lịch tái khám và cập nhập chuẩn đoán bệnh cho bệnh nhân:** Sau khi khám, bác sĩ có thể gửi kết luận chẩn đoán lên hệ thống và nếu cần, tạo luôn lịch tái khám cụ thể cho bệnh nhân (chọn ngày, giờ, và chuyên khoa). Điều này giúp quá trình điều trị liên tục và hiệu quả hơn.
- **Xem danh sách câu hỏi được đặt đến bệnh viện:** Hiển thị danh sách các câu hỏi mà bệnh nhân đã gửi tới bệnh viện hoặc bác sĩ, được phân loại theo chủ đề, thời gian gửi, người gửi. Bác sĩ có thể chọn lọc để xử lý theo chuyên môn của mình.
- **Trả lời câu hỏi của bệnh nhân:** Bác sĩ có thể trả lời trực tiếp các thắc mắc của bệnh nhân trên hệ thống, chẳng hạn như: giải đáp triệu chứng, hướng dẫn sơ bộ, hoặc hướng bệnh nhân đến khám chuyên khoa phù hợp. Các phản hồi sẽ được gửi dưới dạng tin nhắn hoặc thông báo đến bệnh nhân.

2.1.3. Quản trị viên

Quản lý tài khoản

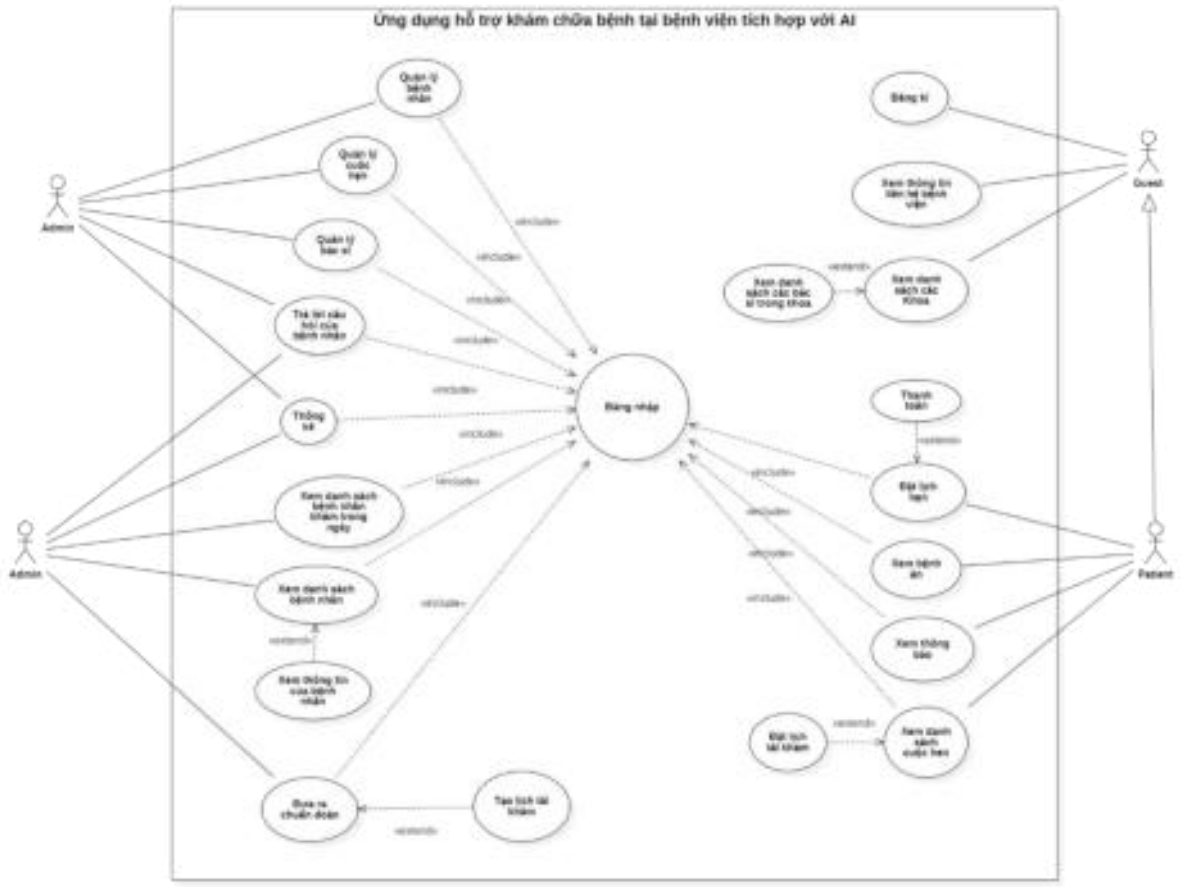
- **Đăng nhập:** Chức năng này cho phép người dùng đăng nhập vào hệ thống bằng cách cung cấp tên đăng nhập và mật khẩu đã đăng ký trước đó.
- **Đăng xuất:** Cho phép bác sĩ đăng xuất khỏi hệ thống, kết thúc phiên làm việc hiện tại.

Sử dụng tài khoản

- **Xem thống kê bệnh viện**: Quản trị viên có thể truy cập trang thống kê để theo dõi tổng quan hoạt động của bệnh viện, bao gồm số lượng bệnh nhân đang sử dụng hệ thống, số bác sĩ đang hoạt động, số lượng cuộc hẹn đã đặt gần đây, cùng với biểu đồ trực quan hiển thị số lượng cuộc hẹn theo từng tháng.
- **Quản lý tài khoản bệnh nhân:** Cho phép Quản trị viên theo dõi và chỉnh sửa thông tin tài khoản của tất cả bệnh nhân đang đăng ký trên hệ thống. Bao gồm các chức năng như cập nhật thông tin cá nhân, hoặc xóa tài khoản khi cần thiết. Điều này giúp đảm bảo dữ liệu người dùng luôn chính xác và có thể kiểm soát tốt hơn hoạt động trong hệ thống.
- **Quản lý tài khoản bác sĩ:** Quản trị viên có quyền tạo mới tài khoản cho bác sĩ, phân bổ họ vào các khoa tương ứng, cập nhật thông tin cá nhân hoặc chuyên môn, và thay đổi quyền truy cập khi cần. Việc quản lý chặt chẽ tài khoản bác sĩ giúp đảm bảo chất lượng chuyên môn và hoạt động phối hợp trong bệnh viện.
- **Quản lý thông tin các cuộc hẹn:** Cho phép xem danh sách toàn bộ các cuộc hẹn đã được bệnh nhân đặt, lọc theo ngày, khoa, bác sĩ, hoặc trạng thái (đã xác nhận, đã khám, bị hủy,...). Quản trị viên có thể cập nhật thông tin cuộc hẹn, thay đổi thời gian, hoặc hủy cuộc hẹn khi có yêu cầu từ bệnh viện hoặc bác sĩ.
- **Xem danh sách cậu hỏi của bệnh nhân :** Quản trị viên có thể xem tất cả các câu hỏi mà bệnh nhân gửi về hệ thống, bao gồm các câu hỏi thường gặp hoặc các thắc mắc chưa được trả lời. Tính năng này giúp theo dõi mối quan tâm của bệnh nhân để từ đó tối ưu quy trình hỗ trợ.
- **Trả lời câu hỏi của bệnh nhân :** Admin có thể trực tiếp phản hồi các câu hỏi đến từ bệnh nhân

2.2. Biểu đồ ca sử dụng

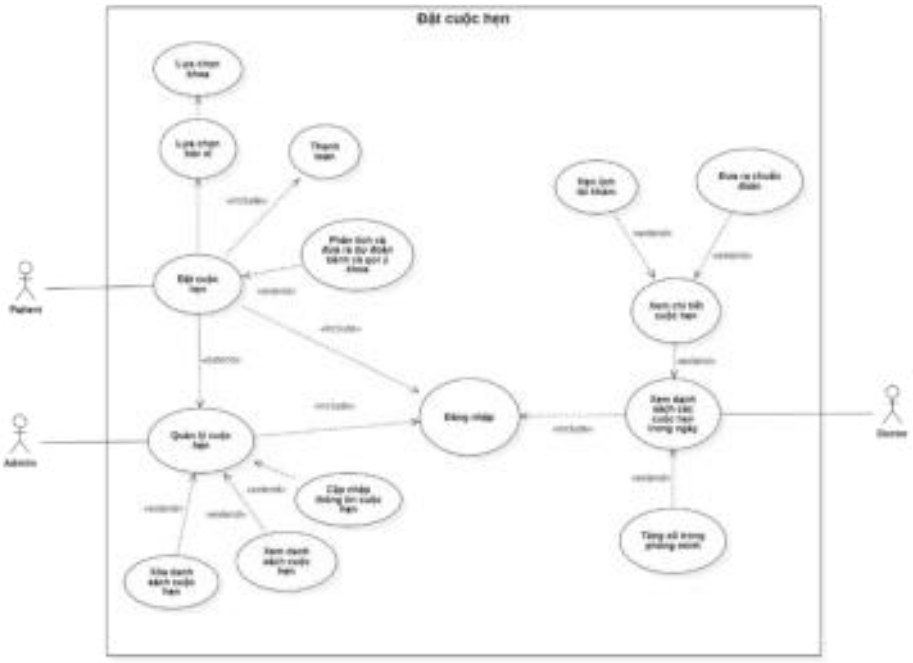
2.2.1. Biểu đồ ca sử dụng tổng quan



Hình 2.2 Biểu đồ ca sử dụng tổng quan

2.2.2. Biểu đồ ca sử dụng phân rã

Phân rã ca sử dụng chức năng "Đặt cuộc hẹn"

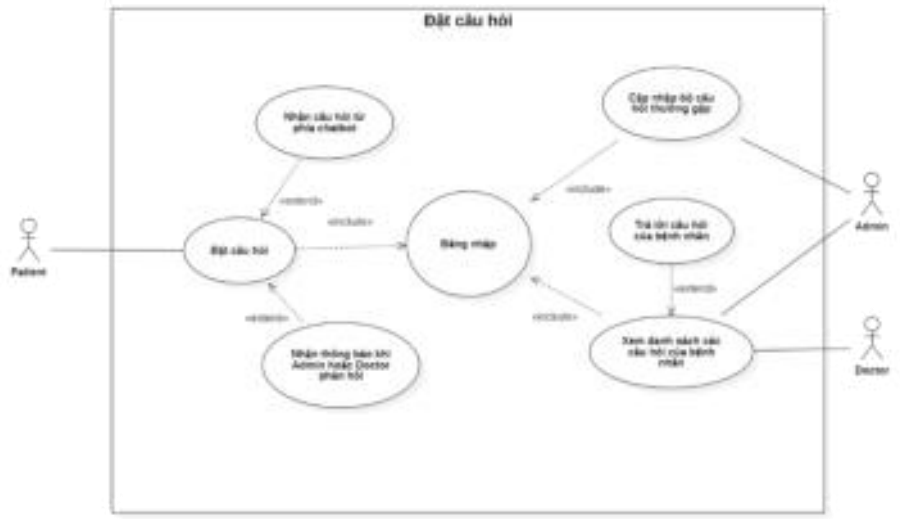


Hình 2.3 Phân rã ca sử dụng chức năng "Đặt cuộc hẹn"

Phân rã ca sử dụng chức năng "Quản lý thông sức khỏe cá nhân"

Hình 2.4 Phân rã ca sử dụng chức năng "Đặt câu hỏi"

Phân rã ca sử dụng chức năng "Đặt câu hỏi"



Hình 2.5 Phân rã ca sử dụng chức năng "Đặt câu hỏi"

2.3. Đặc tả biểu đồ ca sử dụng

2.3.1. Đặc tả chức năng "Quản lý tài khoản"

Bảng 2.1 Đặc tả ca sử dụng chức năng "Đăng nhập"

Mã ca sử dụng	UC01
Tên ca sử dụng	Đăng nhập
Mô tả	Cho phép người dùng đăng nhập vào ứng dụng, và phân quyền cho từng đối tượng
Tác nhân	Bệnh nhân, bác sĩ, quản trị viên
Điều kiện trước	Người sử dụng vào màn hình đăng nhập. Người dùng có tài khoản và mật khẩu đã được tạo
Điều kiện sau	Phiên đăng nhập được tạo ra và được lưu trữ lại (token)
Các bước thực hiện	1. Người dùng điền các thông tin đăng nhập: tên người dùng và mật khẩu 2. Bấm vào nút "Đăng nhập" 3. Hệ thống kiểm tra dữ liệu. 4. Đưa ra thông báo và dẫn đến màn đã đăng nhập với sự phân quyền
Kết quả	Đưa đến màn hình homepage ứng với vai trò của người đăng nhập
Trường hợp lỗi	1. Người dùng nhập các thông tin đăng nhập không chính xác 2. Thông báo lỗi ra màn hình. 3. Sai mật khẩu 4. Thiết bị chưa được kết nối mạng

Bảng 2.2 Đặc tả ca sử dụng chức năng "Đăng ký"

Mã ca sử dụng	UC02
Tên ca sử dụng	Đăng ký
Mô tả	Cho phép người dùng đăng ký tài khoản bệnh nhân để đăng nhập vào hệ thống
Tác nhân	Khách
Điều kiện trước	Người sử dụng vào màn hình đăng ký. Tài khoản email của người dùng chưa được tạo trước đây
Điều kiện sau	Tạo một tài khoản mới theo tác nhân là bệnh nhân trong hệ thống
Các bước thực hiện	5. Người dùng điền các thông tin ký trước: tên người dùng và email và mật khẩu và xác nhân mật khẩu 6. Bấm vào nút "Đăng ký" 7. Hệ thống kiểm tra dữ liệu. 8. Đưa ra thông báo đăng ký thành công và dẫn đến màn đã đăng nhập.
Kết quả	Hiển thị thông báo đăng ký thành công và đưa đến màn hình đăng nhập ứng dụng
Trường hợp lỗi	1. Người dùng nhập các thông tin đăng nhập không chính xác 2. Thông báo lỗi ra màn hình. 3. Mật khẩu và xác nhận mật khẩu không giống nhau 4. Thiết bị chưa được kết nối mạng

2.3.2. Đặc tả chức năng "Quản lý đặt lịch hẹn"

Bảng 2.3 Đặc tả ca sử dụng chức năng "Đặt lịch hẹn "

Mã ca sử dụng	UC03
Tên ca sử dụng	Đặt lịch hẹn
Mô tả	Bệnh nhân có thể đặt lịch hẹn với bác sĩ
Tác nhân	Bệnh nhân
Điều kiện trước	Người dung đã truy cập vào hệ thống
Điều kiện sau	Hệ thống lưu trữ cuộc hẹn của bệnh nhân vào cơ sở dữ liệu
Các bước thực hiện	1. Người dùng truy cập chức năng đặt lịch hẹn trên điện thoại 2. Người dùng lựa chọn và nhập các thông tin cần thiết vào 3. Người dùng nhập triệu chứng và bấm nút phân tích 4. Hệ thống sẽ trả về người dùng danh sách các bệnh dự đoán 5. Người dùng lựa chọn hình thức thanh toán 6. Người dùng bấm nút "Đặt lịch" 7. Hệ thống kiểm tra dữ liệu 8. Hệ thống đưa ra thông báo đặt thành công và chuyển qua màn thanh toán thành công

Kết quả	Trả về thông báo thành công và chuyển qua màn thanh toán thành công
Trường hợp lỗi	1. Người dùng chưa lựa chọn và nhập chưa đủ các trường 2. Thông báo lỗi ra màn hình 3. Lịch hẹn của bác sĩ trong ngày đã hết

Bảng 2.4 Đặc tả ca sử dụng chức năng "Thanh toán lịch hẹn "

Mã ca sử dụng	UC04
Tên ca sử dụng	Thanh toán lịch hẹn
Mô tả	Bệnh nhân lựa chọn thanh toán theo 2 phương thức Momo hoặc ZaloPay, sau khi đặt lịch hẹn thành công
Tác nhân	Bệnh nhân
Điều kiện trước	Người dùng lựa chọn hình thức thanh toán là Momo hoặc ZaloPay khi đặt lịch hẹn và hiển thị màn hình chờ thanh toán
Điều kiện sau	Hệ thống ghi nhận thanh toán thành công và tiến hành tạo lịch hẹn
Các bước thực hiện	1. Người dùng nhấn vào nút thanh toán ngay trên giao diện chờ thanh toán 2. Hệ thống tự động chuyển qua giao diện website của nhà cung cấp thanh toán (Momo hoặc ZaloPay) 3. Người dùng tiến hành thực hiện các hành động kiểm tra và tiến hành thanh toán trên giao diện website của nhà cung cấp thanh toán (Momo hoặc ZaloPay) 4. Người dùng quay về lại màn hình chờ của ứng dụng 5. Hệ thống ghi nhận thanh toán thành công và chuyển qua màn hình thanh toán thành công
Kết quả	Hiển thị màn hình thanh toán thành công
Trường hợp lỗi	1. Người dùng không tiến hành thanh toán trong thời gian tồn tại mã 2. Hệ thống thanh toán của nhà cung cấp gặp lỗi hệ thống

Bảng 2.5 Đặc tả ca sử dụng chức năng "Chỉnh sửa thông tin lịch hẹn"

Mã ca sử dụng	UC05
Tên ca sử dụng	Chỉnh sửa thông tin lịch hẹn
Mô tả	Quản trị viên tiến hành chỉnh sửa thông tin lịch hẹn khi nhận được thông báo nhầm lẫn hoặc lỗi từ phía bệnh nhân
Tác nhân	Quản trị viên
Điều kiện trước	Người dung đã đăng nhập vào hệ thống bằng tài khoản quản trị viên

Điều kiện sau	Hệ thống ghi nhận và chỉnh sửa lại thông tin của lịch hẹn vào cơ sở dữ liệu
Các bước thực hiện	1. Người dùng truy cập vào giao diện màn hình quản lý lịch hẹn 2. Lựa chọn icon chỉnh sửa tương ứng với lịch hẹn mà mình cần chỉnh sửa 3. Người dùng tiến hành chỉnh sửa lại các thông tin tương ứng 4. Hệ thống ghi nhận và tiến hành kiểm tra 5. Hệ thống chỉnh sửa lại thông tin của lịch hẹn và cơ sở dữ liệu 6. Hệ thống hiển thị ra thông báo chỉnh sửa thành công
Kết quả	Hệ thống hiển thị ra thông báo chỉnh sửa thành công
Trường hợp lỗi	1. Dữ liệu người dùng đưa vào không phù hợp theo định dạng của hệ thống 2. Lịch hẹn của bác sĩ tương ứng trong hệ thống đã đầy không thể đặt thêm

Bảng 2.6 Đặc tả ca sử dụng chức năng "Tạo lịch tái khám"

Mã ca sử dụng	UC06
Tên ca sử dụng	Tạo lịch tái khám
Mô tả	Bệnh nhân nhận được thông báo đã được tạo lịch tái khám từ bác sĩ, bệnh nhân tiến hành tạo lịch tái khám với bác sĩ
Tác nhân	Bệnh nhân
Điều kiện trước	Người dùng nhận được tạo lịch tái khám từ bác sĩ
Điều kiện sau	Hệ thống lưu trữ lại thông tin lịch tái khám vào cơ sở dữ liệu
Các bước thực hiện	1. Người dùng vào giao diện thông tin chi tiết của cuộc hẹn trước đó 2. Người dùng tiến hành bấm nút "Tái khám" tương ứng với lịch hẹn bác sĩ đã tạo 3. Hệ thống chuyển giao diện qua màn hình đặt lịch hẹn và tự động chọn thể loại "Tái khám" và lựa chọn bác sĩ tương ứng. 4. Người dùng tiến hành nhập các thông tin cần thiết còn lại 5. Người dùng bấm nút "Đặt lịch" 6. Hệ thống hiển thị thông báo đặt lịch thành công và chuyển qua mà hình thanh toán thành công
Kết quả	Hiển thị thông báo đặt lịch thành công và chuyển qua màn hình thanh toán thành công
Trường hợp lỗi	1. Người dùng chưa lựa chọn và nhập chưa đủ các trường 2. Thông báo lỗi ra màn hình 3. Lịch hẹn của bác sĩ trong ngày đã hết

Bảng 2.7 Đặc tả ca sử dụng chức năng "Theo dõi trạng thái phòng khám"

Mã ca sử dụng	UC07
Tên ca sử dụng	Theo dõi trạng thái phòng khám
Mô tả	Bệnh nhân tiến hành theo dõi số thứ tự đang khám và khung giờ tương ứng với số thứ tự của mình được hiển thị trên giao diện điện thoại
Tác nhân	Bệnh nhân
Điều kiện trước	Người dùng đã đặt lịch hẹn và đã có số
Điều kiện sau	Hệ thống truy vấn và gửi về danh sách các số và số hiện tại ở phòng khám của bác sĩ tương ứng
Các bước thực hiện	1. Người dùng vào giao diện homepage và lựa chọn khoa tương ứng 2. Hệ thống hiển thị ra danh sách các bác sĩ có trong khoa đó 3. Người dùng lựa chọn bác sĩ mình đã đặt lịch 4. Hệ thống hiển thị ra danh sách các số và số hiện tại đang khám ở phòng khám của bác sĩ
Kết quả	Hiển thị ra danh sách các số và số hiện tại đang khám ở phòng khám của bác sĩ

2.3.3. Đặc tả chức năng "Quản lý bệnh án"

Bảng 2.8 Đặc tả ca sử dụng chức năng "Xem sổ khám bệnh điện tử"

Mã ca sử dụng	UC08
Tên ca sử dụng	Xem sổ khám bệnh điện tử
Mô tả	Bệnh nhân có thể xem lại các chuẩn đoán và các đơn thuốc mà bác sĩ đã kê khai
Tác nhân	Bệnh nhân
Điều kiện trước	Người dùng đã được bác sĩ chuẩn đoán và kê khai đơn thuốc
Điều kiện sau	Hệ thống truy vấn và gửi về danh sách lịch sử các chuẩn đoán của các bác sĩ
Các bước thực hiện	1. Người dùng nhấn vào nút trang cá nhân trên điện thoại 2. Hệ thống hiển thị ra danh sách các các nút tương ứng 3. Người dùng bấm nút "Sổ khám bệnh" 4. Hệ thống hiển thị ra danh sách các chuẩn đoán mà các bác sĩ đã chuẩn đoán 5. Người dùng tiến hành chọn vào sổ tương ứng 6. Hệ thống hiển thị ra thông tin của bệnh và thuốc tương ứng với số khám bệnh đó
Kết quả	Hiển thị ra thông tin của bệnh và thuốc tương ứng với số khám bệnh đó

Bảng 2.9 Đặc tả ca sử dụng chức năng "Xem lịch sử cuộc hẹn"

Mã ca sử dụng	UC09
Tên ca sử dụng	Xem lịch sử cuộc hẹn
Mô tả	Bệnh nhân có thể xem lại thông tin các cuộc hẹn mà mình đã đặt
Tác nhân	Bệnh nhân
Điều kiện trước	Người dùng đã đặt lịch hẹn
Điều kiện sau	Hệ thống truy xuất và gửi về dữ liệu lịch hẹn của người dùng tương ứng
Các bước thực hiện	1. Người dùng nhấn vào nút cá nhân trên điện thoại 2. Hệ thống hiển thị ra danh sách các các nút người dùng có thể tương tác được 3. Người dùng bấm vào "Lịch sử cuộc hẹn" 4. Hệ thống hiển thị ra danh sách các cuộc hẹn mà người dùng đã đặt 5. Người dùng tiến hành chọn ra cuộc hẹn mà mình muốn xem thông tin 6. Hệ thống hiển thị ra thông tin của cuộc hẹn tương ứng
Kết quả	Hiển thị ra thông tin của cuộc hẹn tương ứng

Bảng 2.10 Đặc tả ca sử dụng chức năng "Tạo/cập nhập chuẩn đoán bệnh án"

Mã ca sử dụng	UC10
Tên ca sử dụng	Tạo/Cập nhập chuẩn đoán bệnh án
Mô tả	Bác sĩ có thể tạo/cập nhập chuẩn đoán bệnh án của bệnh nhân trong trong nội dung cuộc hẹn
Tác nhân	Bác sĩ
Điều kiện trước	Bệnh nhân có đặt lịch khám với bác sĩ
Điều kiện sau	Hệ thống Tạo/cập nhập lại chuẩn đoán bệnh án
Các bước thực hiện	1. Người dùng lựa chọn lịch khám 2. Hệ thống hiển thị thông tin của bệnh nhân, triệu chứng và chuẩn đoán (nếu đã chuẩn đoán) 3. Bác sĩ tiến hành chuẩn đoán bệnh và nhập thuốc dựa trên triệu chứng và bệnh dự đoán của hệ thống 4. Bác sĩ bấm nút lưu 5. Hệ thống hiển thị thông báo đã tạo/cập nhập chuẩn đoán bệnh án thành công
Kết quả	Hiển thị thông báo đã tạo/cập nhập chuẩn đoán bệnh án thành công

2.3.4. Đặc tả chức năng "Hỗ trợ và tương tác"

Bảng 2.11 Đặc tả ca sử dụng chức năng "Đặt câu hỏi"

Mã ca sử dụng	UC11
Tên ca sử dụng	Đặt câu hỏi
Mô tả	Bệnh nhân tiến hành đặt câu hỏi cho bác sĩ và quản trị viên trả lời
Tác nhân	Bệnh nhân
Điều kiện trước	Người dùng đăng nhập vào hệ thống bằng tài khoản bệnh nhân
Điều kiện sau	Hệ thống lưu trữ câu hỏi của bệnh nhân và chatbot sẽ tiến hành trả lời lại dựa trên bộ câu hỏi và câu trả lời mà bệnh viện đã chuẩn bị
Các bước thực hiện	1. Người dùng nhấn vào nút cá nhân trên điện thoại 2. Hệ thống hiển thị ra danh sách các các nút người dùng có thể tương tác được 3. Người dùng bấm vào "Liên hệ" 4. Người dùng tiến hành nhập câu hỏi vào 5. Người dùng nhấn nút gửi 6. Hệ thống sẽ hiển thị câu hỏi của người dùng và câu trả lời của chatbot
Kết quả	Hiển thị câu hỏi của người dùng và câu trả lời của chatbot

Bảng 2.12 Đặc tả ca sử dụng chức năng "Trả lời câu hỏi"

Mã ca sử dụng	UC12
Tên ca sử dụng	Trả lời câu hỏi
Mô tả	Sau khi nhận được câu hỏi của bệnh nhân, bác sĩ hoặc quản trị viên sẽ tiến hành trả lời các câu hỏi không có trong bộ câu hỏi đã chuẩn bị
Tác nhân	Bác sĩ, Quản trị viên
Điều kiện trước	Bệnh nhân có đặt câu hỏi
Điều kiện sau	Hệ thống lưu trữ câu trả lời của bác sĩ / quản trị viên
Các bước thực hiện	1. Người dùng đến giao diện giải đáp thắc mắc của bệnh nhân 2. Người dùng nhập nội dung câu hỏi vào ô tìm kiếm 3. Hệ thống sẽ trả về một danh sách các câu hỏi có nội dung giống với nội dung người dùng đã nhập vào 4. Người dùng tiến hành trả lời câu hỏi của bệnh nhân 5. Người dùng bấm nút gửi 6. Hệ thống hiển thị thông báo trả lời câu hỏi của bệnh nhân thành công và hiển thị câu trả lời ở dưới câu hỏi của bệnh nhân

Kết quả	Hiển thị thông báo trả lời câu hỏi của bệnh nhân thành công và hiển thị câu trả lời ở dưới câu hỏi của bệnh nhân

2.3.5. Đặc tả chức năng "Quản lý nội bộ"

Bảng 2.13 Đặc tả ca sử dụng chức năng "Cập nhật thông tin bác sĩ/bệnh nhân"

Mã ca sử dụng	UC13
Tên ca sử dụng	Cập nhật thông tin bác sĩ/ bệnh nhân
Mô tả	Quản trị viên có thể cập nhập lại thông tin của bác sĩ/bệnh nhân
Tác nhân	Quản trị viên
Điều kiện trước	Người dung đã đăng nhập vào hệ thống với vai trò quản trị viên
Điều kiện sau	Hệ thống lưu trữ lại thông tin thay đổi của bác sĩ/ bệnh nhân vào cơ sở dữ liệu
Các bước thực hiện	1. Người dùng truy cập vào giao diện quản lý bác sĩ/ bệnh nhân 2. Người dùng tiến hành nhập tên hoặc email của bác sĩ/ bệnh nhân vào khung tìm kiếm 3. Hệ thống hiển thị danh sách bác sĩ/ bệnh nhân tương ứng 4. Người dùng bấm vào biểu tượng chỉnh sửa 5. Hệ thống hiển thị giao diện chỉnh sửa 6. Người dùng chỉnh sửa thông tin 7. Người dùng bấm nút "Lưu" 8. Hệ thống hiển thị thông báo cập nhập thành công và thông tin trên giao diện đã được đổi lại
Kết quả	Hiển thị thông báo cập nhập thành công và thông tin trên giao diện đã được đổi lại

Bảng 2.14 Đặc tả ca sử dụng chức năng "Tạo tài khoản bác sĩ/ bệnh nhân"

Mã ca sử dụng	UC14
Tên ca sử dụng	Tạo tài khoản bác sĩ/ bệnh nhân
Mô tả	Quản trị viên có thể tạo tài khoản cho bác sĩ/ bệnh nhân
Tác nhân	Quản trị viên
Điều kiện trước	Địa chỉ email chưa được sử dụng để tạo tài khoản trước đây
Điều kiện sau	Hệ thống lưu trữ lại thông tin vừa tạo vào cơ sở dữ liệu

Các bước thực hiện	1. Người dùng viên truy cập vào giao diện quản lý bác sĩ / bệnh nhân 2. Người dùng bấm vào nút tạo tài khoản mời 3. Hệ thống hiển thị giao diện tạo tài khoản mới 4. Người dùng nhập vào thông tin người dùng mới vào 5. Người dùng bấm nút "Gửi" 6. Hệ thống hiển thị thông báo tạo tài khoản thành công và tài khoản vừa tạo hiển thị trên danh sách tài khoản
Kết quả	Hiển thị thông báo tạo tài khoản thành công và tài khoản vừa tạo hiển thị trên danh sách tài khoản
Trường hợp lỗi	1. Tài khoản email đã được sử dụng để tạo tài khoản trước đó 2. Thông tin tài khoản nhập vào chưa đúng (số điện thoại không đủ 10 số, địa chỉ để rỗng, ….) 3. Hiển thị thông báo lỗi

2.3.6. Đặc tả chức năng "Quản lý thống kê"

Bảng 2.15 Đặc tả ca sử dụng chức năng "Thống kê lịch hẹn trong tuần"

Mã ca sử dụng	UC15
Tên ca sử dụng	Thống kê lịch hẹn trong tuần
Mô tả	Bác sĩ có thể xem được thống kê lịch hẹn trong tuần hiện tại (số cuộc hẹn trong từng tuần, số cuộc hẹn đã hoàn thành – chưa hoàn thành, …)
Tác nhân	Bác sĩ
Điều kiện trước	Người dung đã đăng nhập vào hệ thống với tài khoản bác sĩ
Điều kiện sau	Hệ thống truy vấn và trả về thống kê lịch hẹn trong tuần của bác sĩ tương ứng
Các bước thực hiện	1. Người dùng truy cập vào giao diện trang chủ 2. Hệ thống trả về số lượng lịch hẹn trong tuần hiện tại 3. Hệ thống trả về danh sách lịch hẹn trong ngày 4. Hệ thống trả về danh sách câu hỏi của bệnh nhân trong ngày
Kết quả	Hiển thị biểu đồ và bảng thống kê lịch hẹn trong tuần Hiển thị danh sách lịch hẹn trong ngày Hiển thị danh sách câu hỏi của bệnh nhân trong ngày

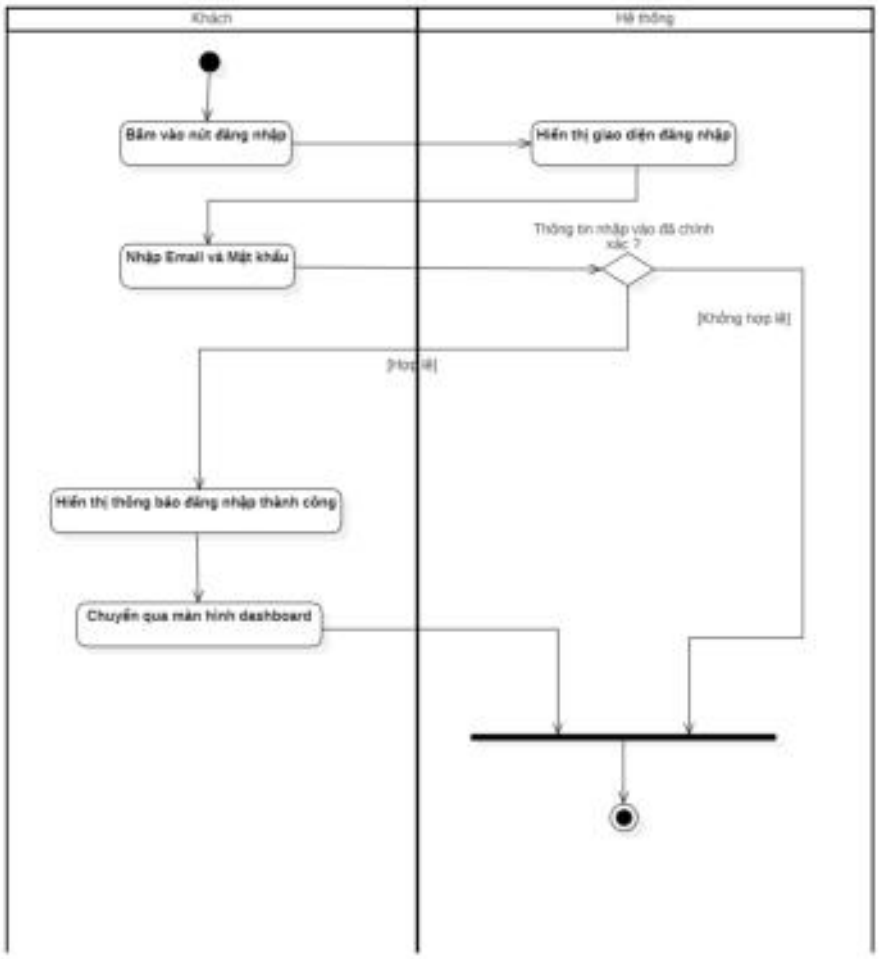
Bảng 2.16 Đặc tả ca sử dụng chức năng "Thống kê lịch hẹn trong tháng"

Mã ca sử dụng	UC16
Tên ca sử dụng	Thống kê lịch hẹn trong tháng
Mô tả	Quản trị viên có thể xem thống kê lịch hẹn của bệnh viện trong tháng hiện tại và danh sách các cuộc hẹn gần đây

Tác nhân	Quản trị viên
Điều kiện trước	Người dung đã đăng nhập vào hệ thống với tài khoản bác sĩ
Điều kiện sau	Hệ thống truy vấn và trả về dữ liệu thống kê lịch hẹn trong tháng
Các bước thực hiện	1. Người dùng truy cập vào giao diện trang chủ 2. Hệ thống trả về số lượng lịch hẹn trong tháng hiện tại 3. Hệ thống trả về danh sách lịch hẹn gần đây 4. Hệ thống trả về số lượng bác sĩ, bệnh nhân và cuộc hẹn đang có trong hệ thống
Kết quả	Hiển thị biểu đồ và bảng thống kê lịch hẹn trong tháng hiện tại Hiển thị danh sách lịch hẹn gần đây Hiển thị số lượng bác sĩ, bệnh nhân và cuộc hẹn đang có trong hệ thống

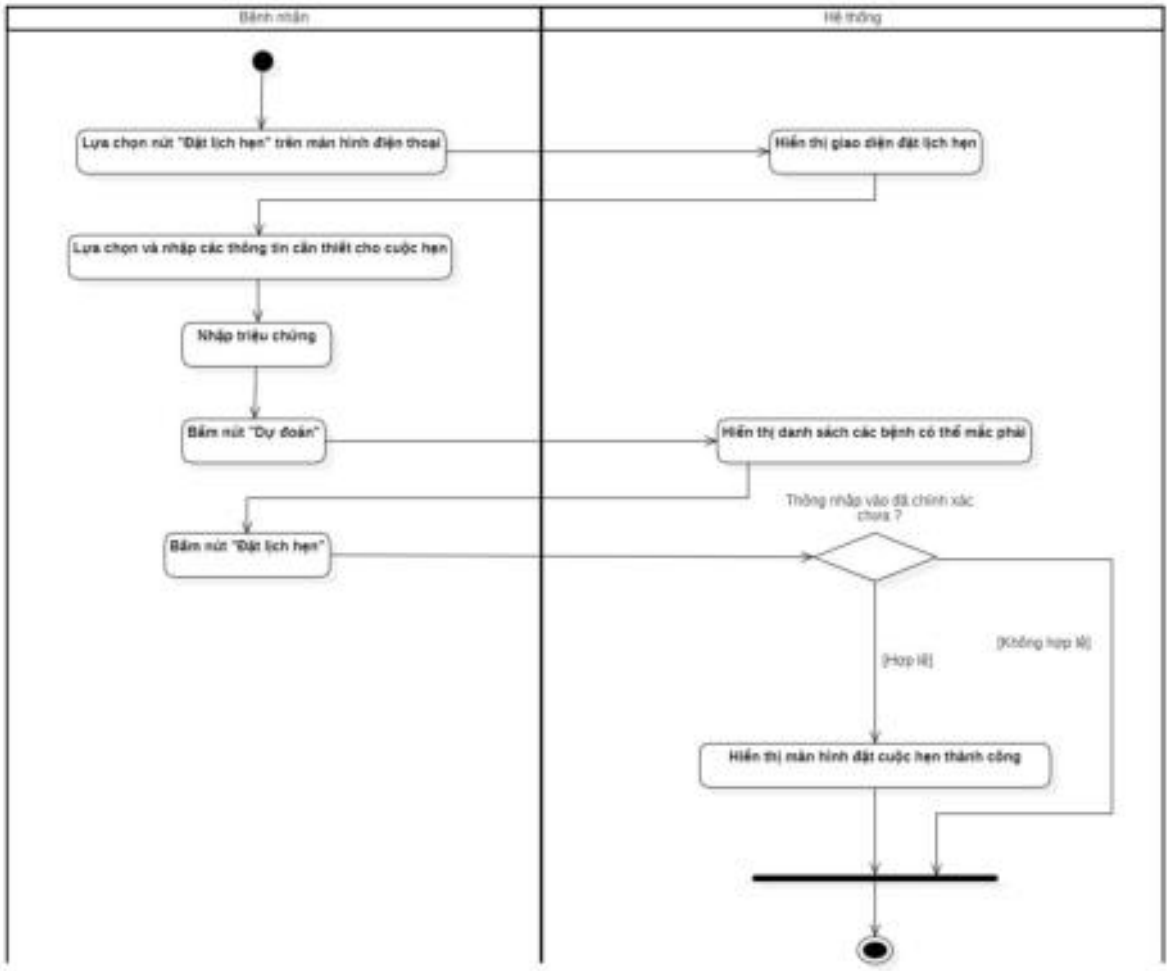
2.4. Biểu đồ hoạt động

2.4.1. Chức năng đăng nhập



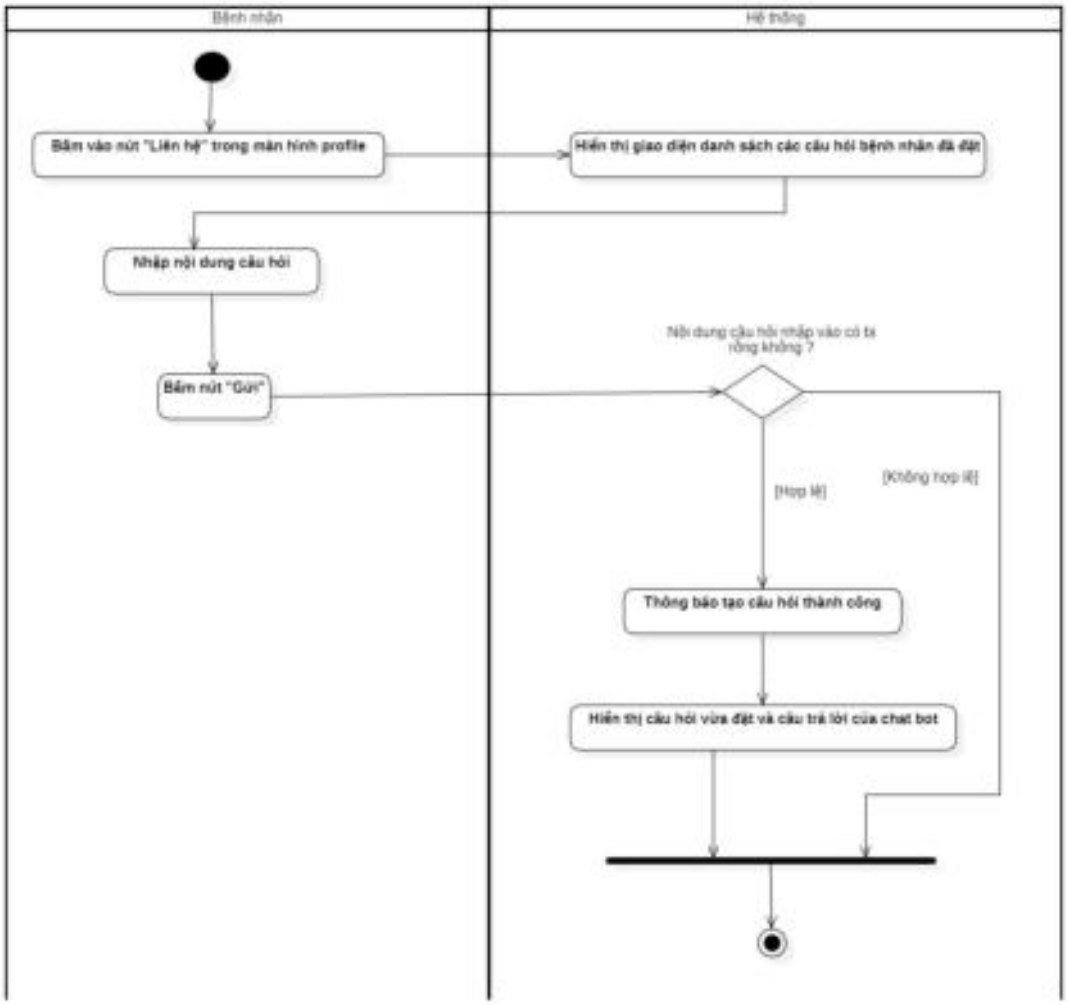
Hình 2.6 Sơ đồ hoạt động chức năng đăng nhập

2.4.2. Chức năng tạo lịch hẹn

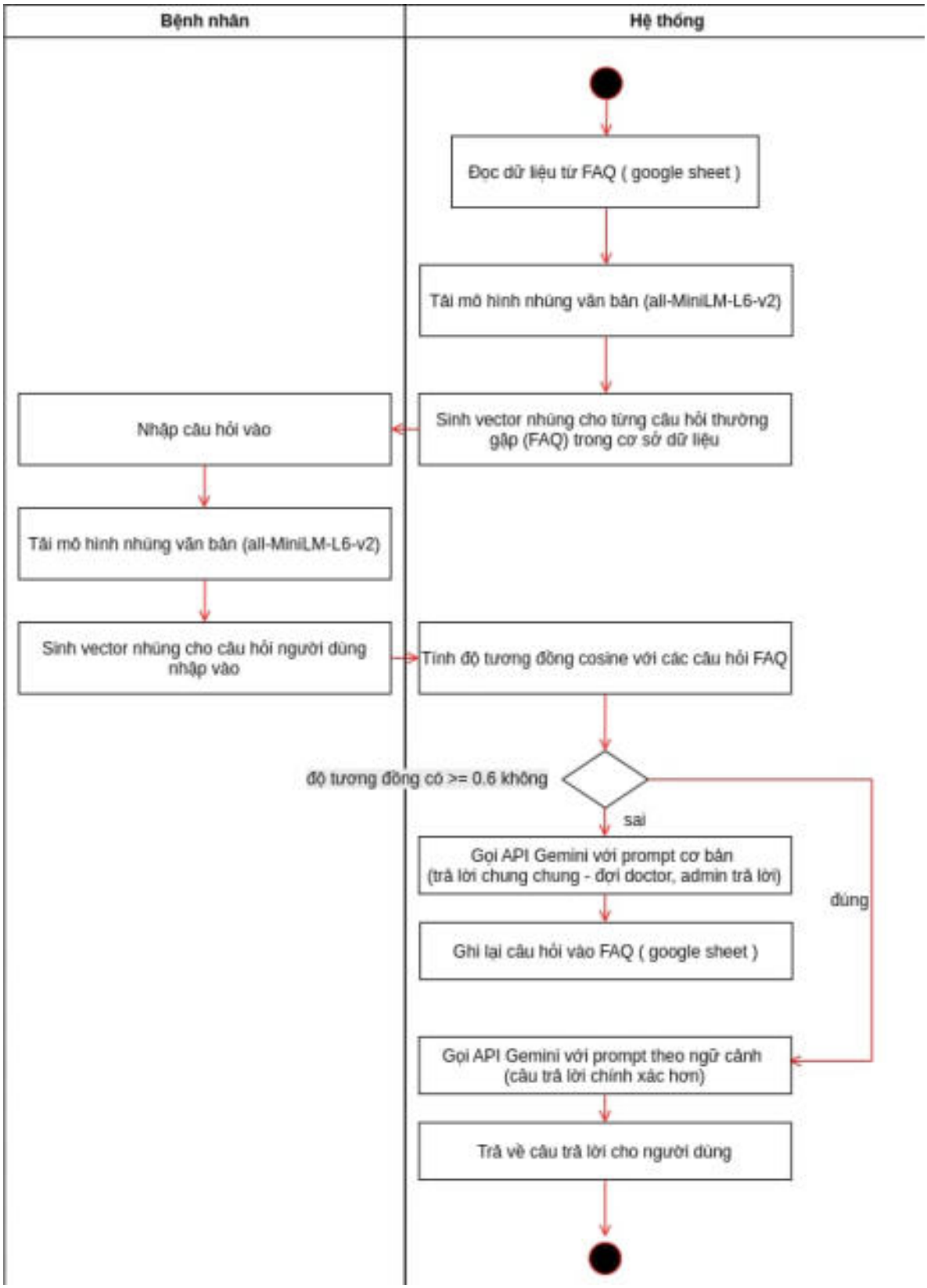


Hình 2.7 Sơ đồ hoạt động chức năng tạo lịch hẹn

2.4.3. Chức năng tạo câu hỏi

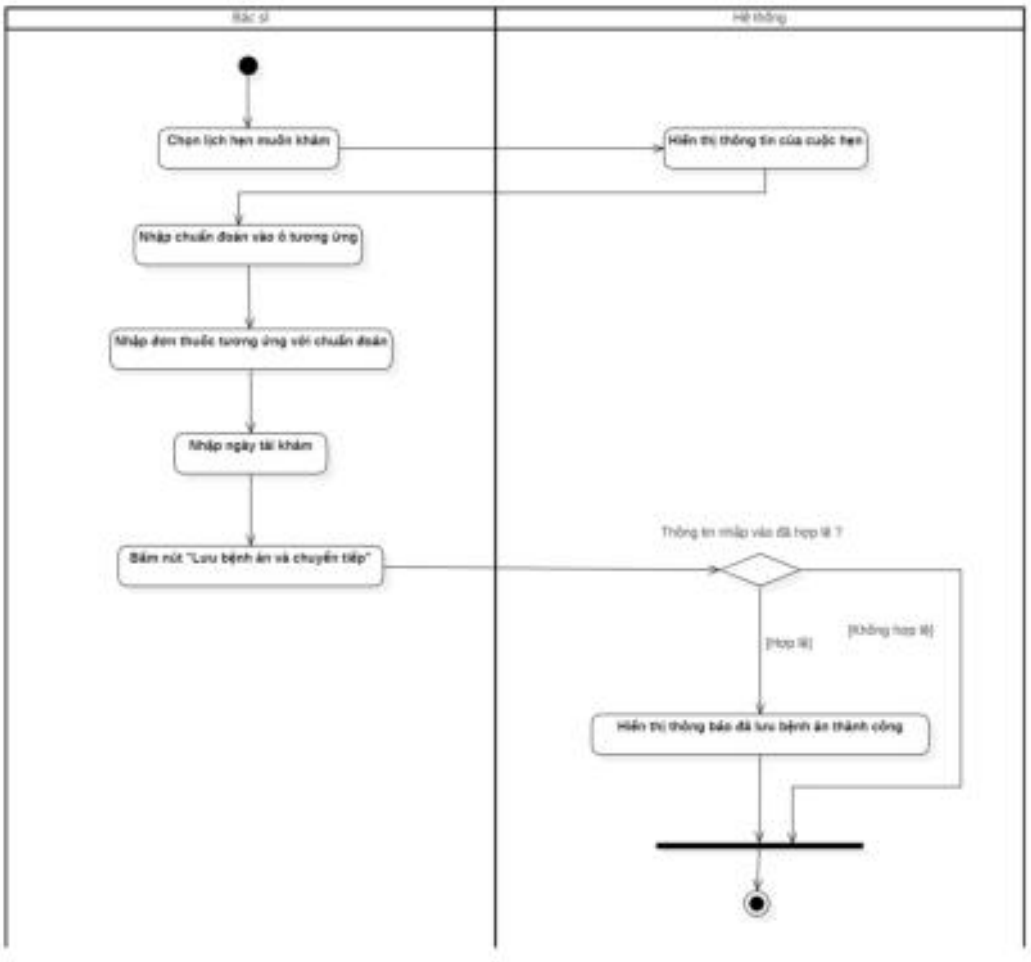


Hình 2.8 Sơ đồ hoạt động chức năng tạo câu hỏi



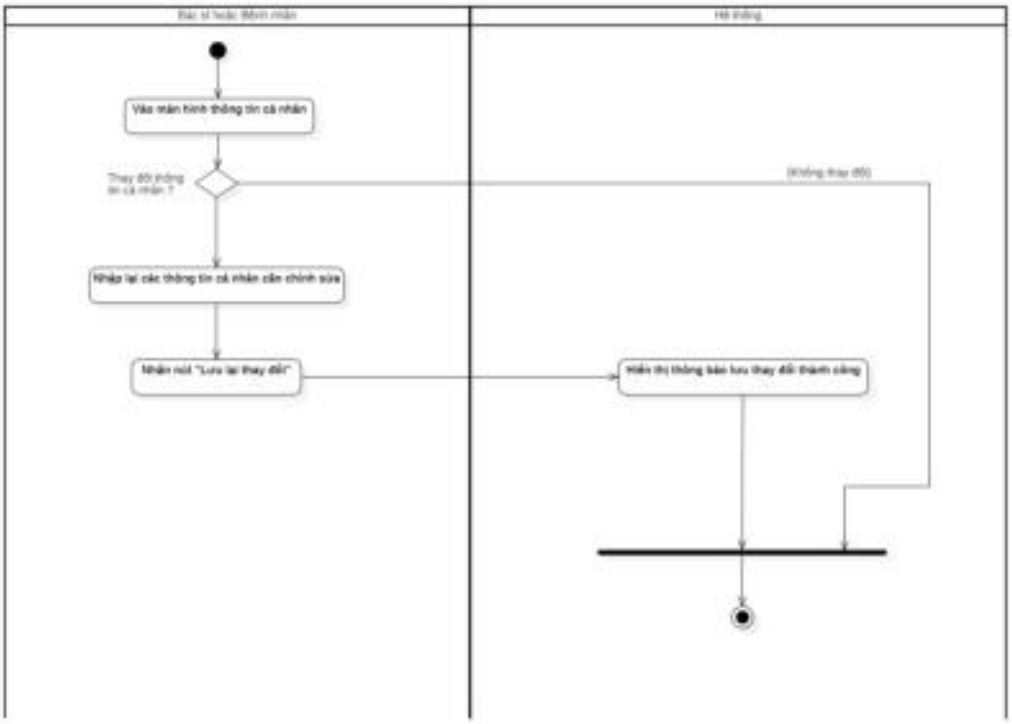
Hình 2 9 Sơ đồ hoạt động của hệ thống hỏi đáp dựa trên embedding và API Gemini

2.4.4. Chức năng khám chữa bệnh



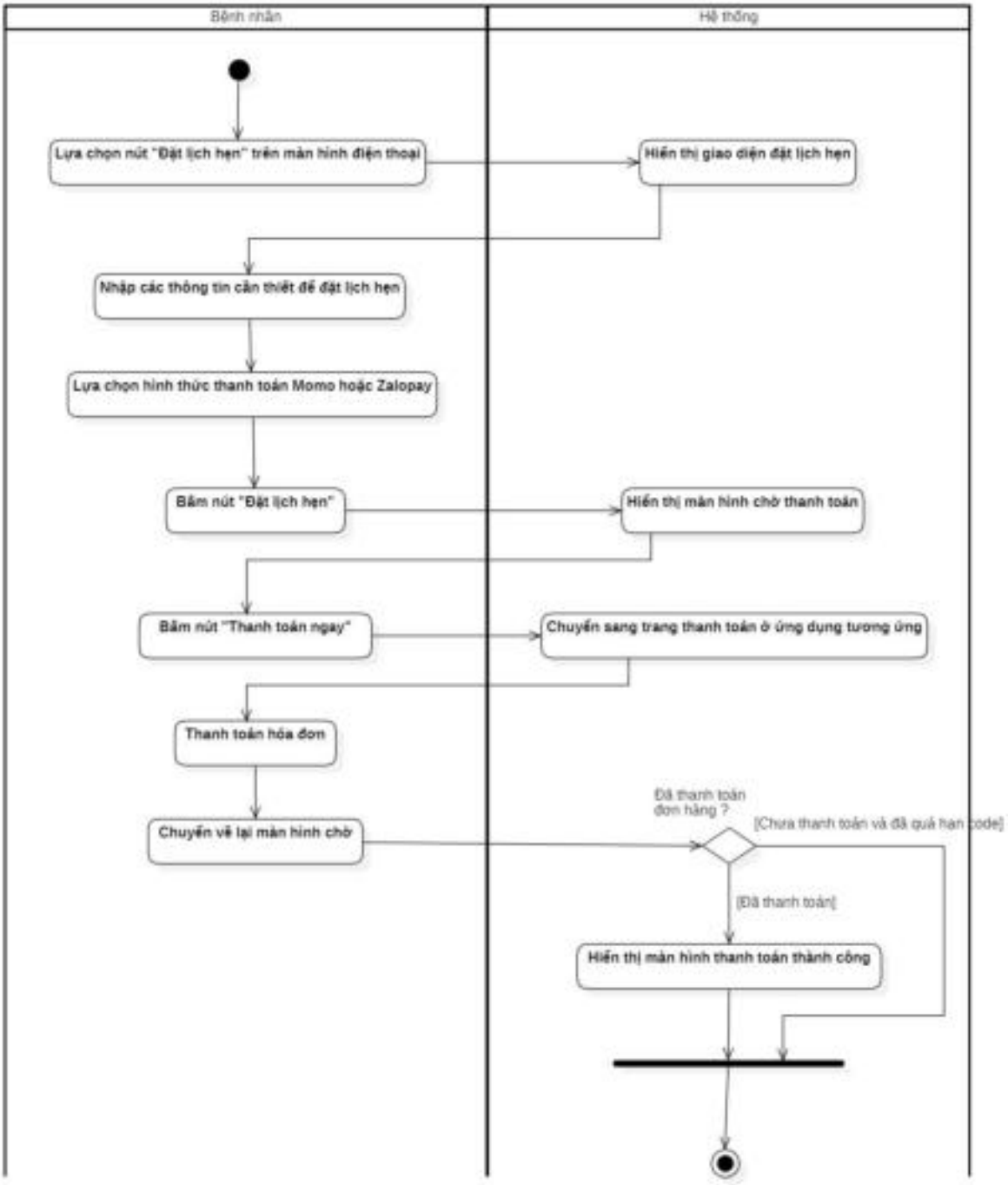
Hình 2 10 Sơ đồ hoạt động chức năng khám chữa bệnh

2.4.5. Chức năng cập nhật thông tin cá nhân



Hình 2.11 Sơ đồ hoạt động chức năng cập nhật thông tin cá nhân

2.4.6. Chức năng thanh toán



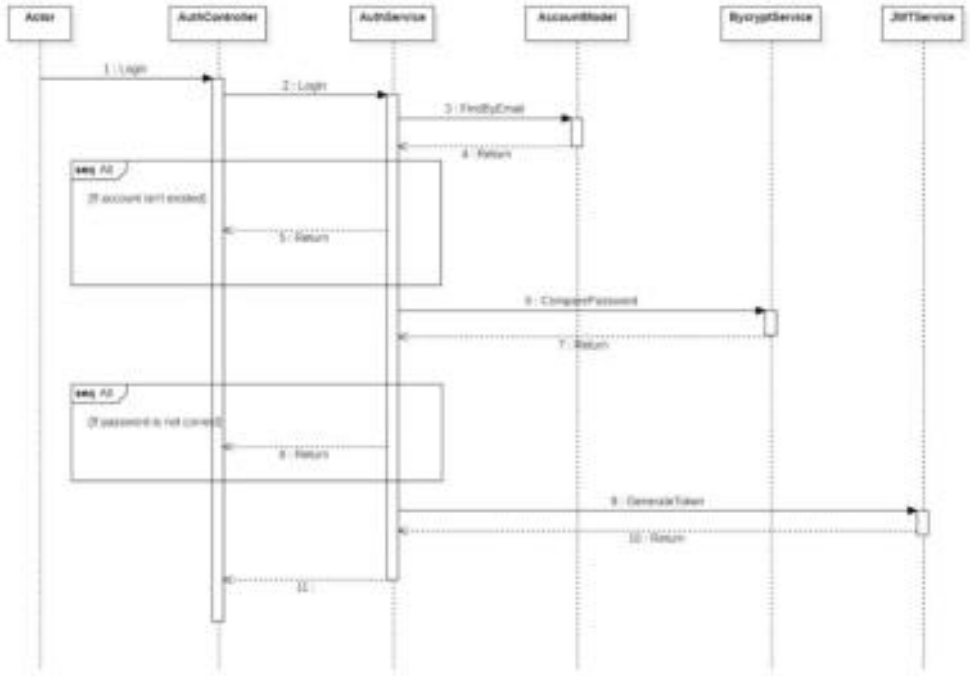
Hình 2.12 Sơ đồ hoạt động chức năng thanh toán

2.5. Biểu đồ lớp

Hình 2.13 Biểu đồ lớp

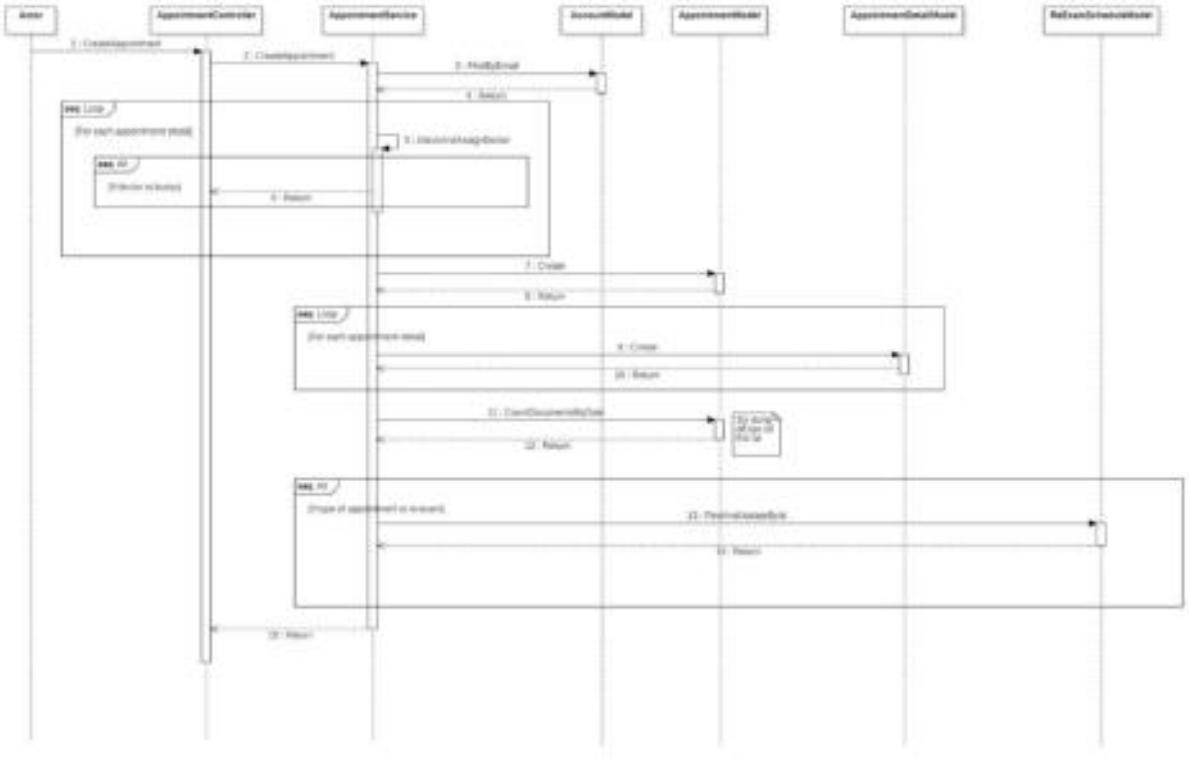
2.6. Biểu đồ tuần tự

2.6.1. Chức năng đăng nhập



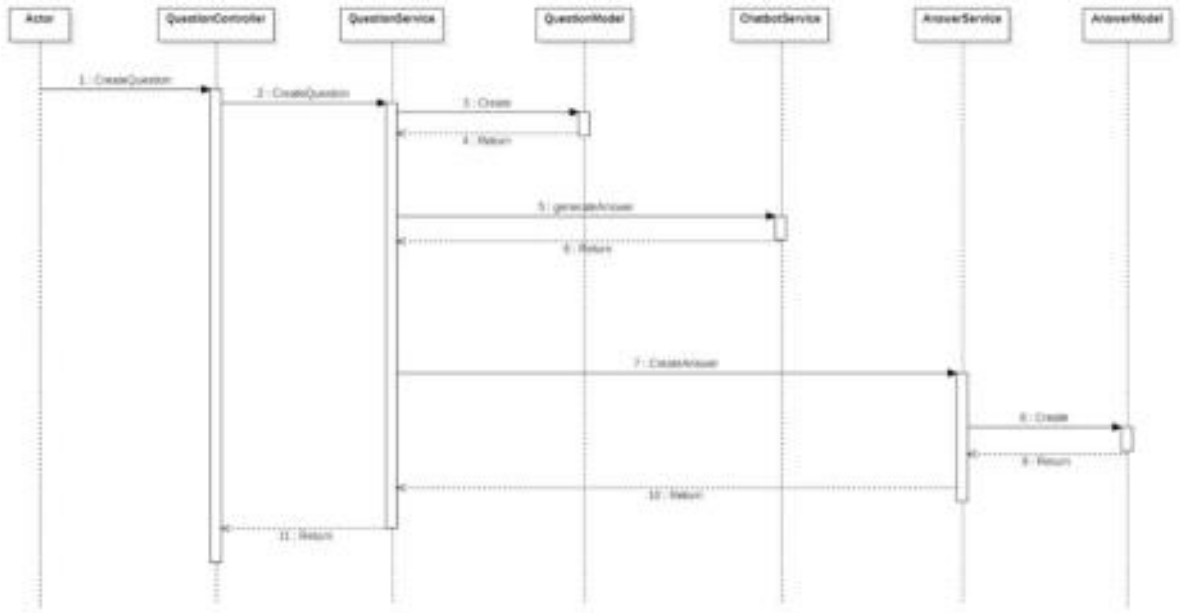
Hình 2.14 Biểu đồ tuần tự chức năng đăng nhập

2.6.2. Đặt lịch hẹn



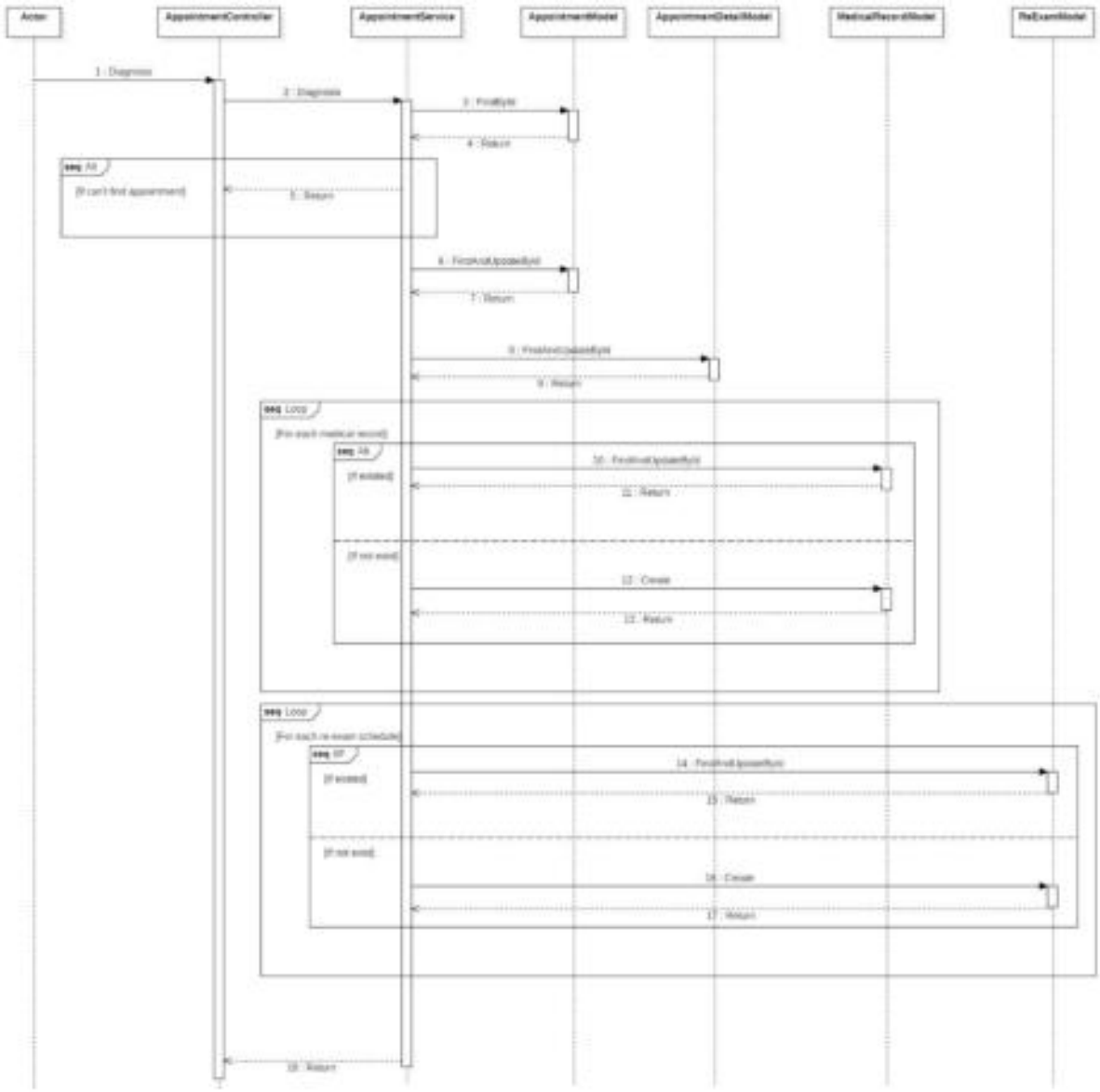
Hình 2.15 Biểu đồ tuần tự chức năng đặt lịch hẹn

2.6.3. Đặt câu hỏi



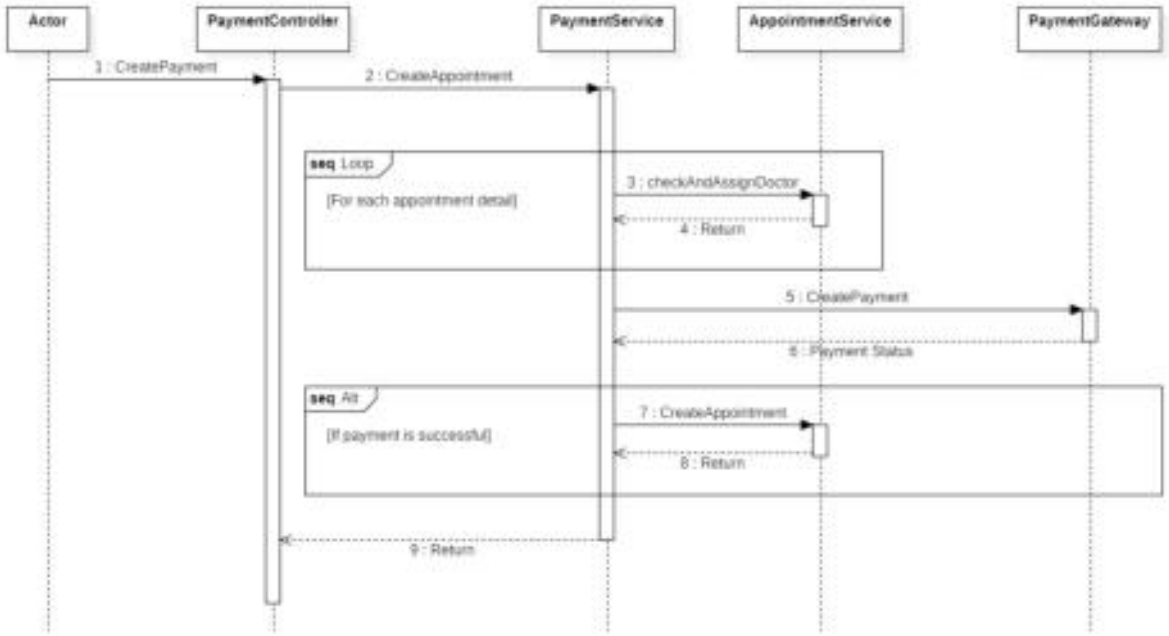
Hình 2.16 Biểu đồ tuần tự chức năng đặt câu hỏi

2.6.4. Khám bệnh



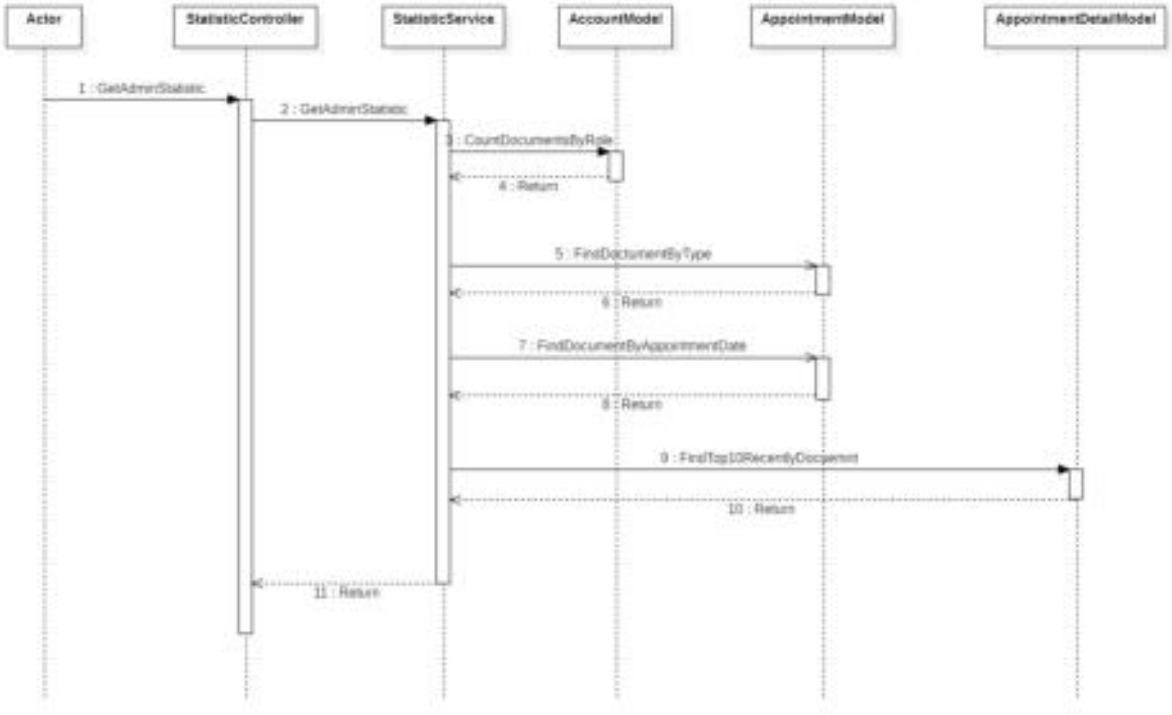
Hình 2.17 Biểu đồ tuần tự chức năng khám bệnh

2.6.5. Thanh toán



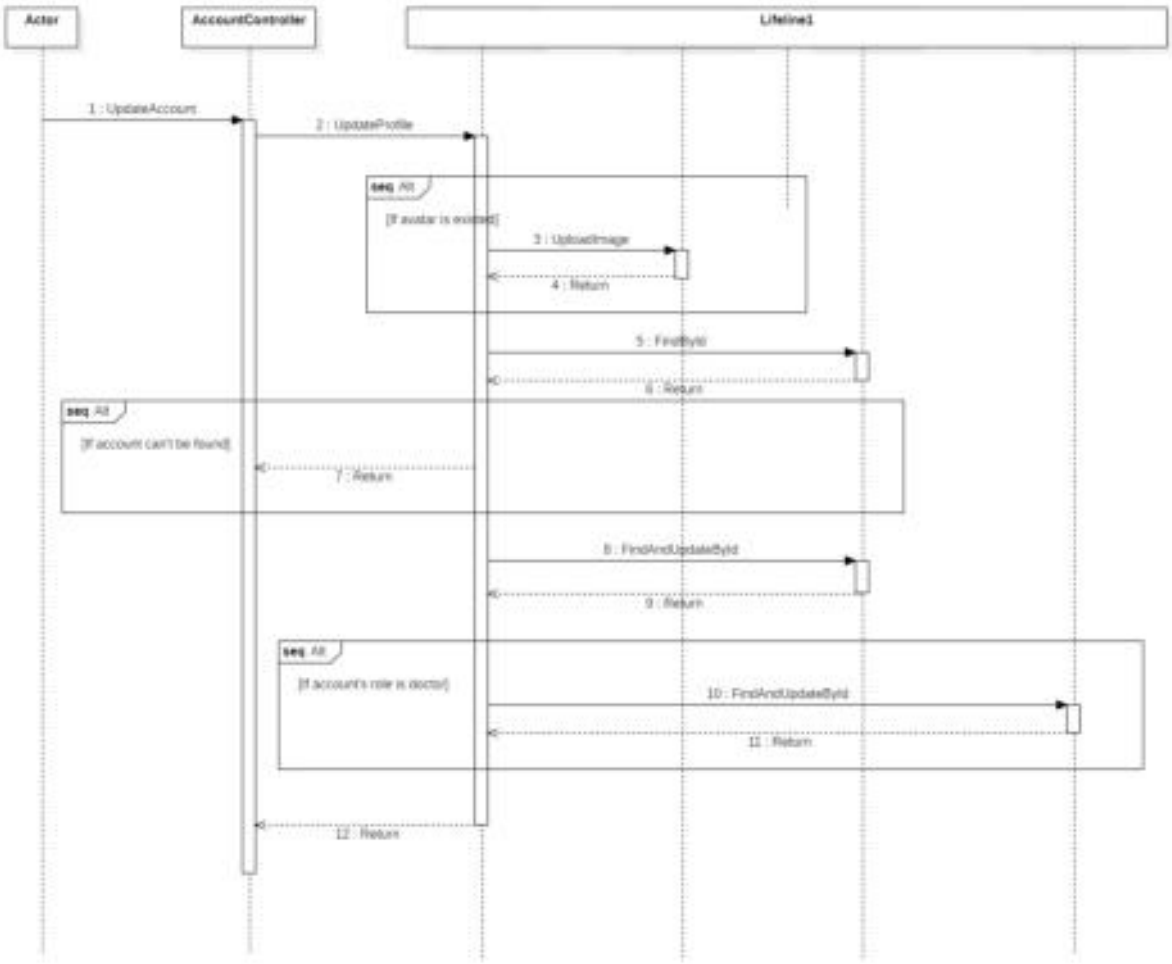
Hình 2.18 Biểu đồ tuần tự chức năng thay đổi mật khẩu

2.6.6. Thống kê



Hình 2.19 Biểu đồ tuần tự chức năng thống kê

2.6.7. Cập nhập thông tin cá nhân



Hình 2.20 Biểu đồ tuần tự chức năng cập nhập thông tin cá nhân

2.7. Cơ sở dữ liệu

Hình 2.21 Cơ sở dữ liệu

Cơ sở dữ liệu bao gồm các bảng:

Bảng Account: Lưu trữ thông tin tài khoản của người dùng trong hệ thống

Bảng 2.17 Mô tả bảng Account

Bảng Account		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
email	String	Địa chỉ email
password	String	Mật khẩu tài khoản
name	String	Tên người dùng
tel	String	Số điện thoại người dùng

address	String	Địa chỉ người dùng
avatar	String	Ảnh đại diện người dùng
gender	String	Giới tính
date_of_birth	Date	Ngày sinh người dùng
role	String	Vai trò người dùng (doctor, patient, admin)
createdAt	Date	Ngày tạo ra tài khoản
updatedAt	Date	Ngày cập nhập tài khoản

Bảng Doctor : Lưu trữ thông tin của bác sĩ trong hệ thống

Bảng 2.18 Bảng Doctor

Bảng Doctor		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
account_id	String	Khóa ngoại liên kết với bảng Account
specialization_id	String	Khóa ngoại liên kết với bảng Specialization
position	String	Chức vụ
base_time	Datetime	Khoảng thời gian bác sĩ khám trung bình
room	String	Phòng khám cố định của bác sĩ

Bảng Specialization: Lưu trữ thông tin các khoa trong hệ thống

Bảng 2.19 Mô tả bảng Specialization

Bảng Specialization:		
Tên cột	Kiểu dữ liệu	Mô tả

_id	String	Khóa chính
name	String	Tên khoa
image	String	Ảnh đại diện khoa
description	String	Mô tả khoa
base_price	Int32	Giá khám cơ bản của khoa

Bảng Appointment: Lưu trữ thông tin các cuộc hẹn trong hệ thống

Bảng 2.20 Mô tả bảng Appointment

Bảng Appointment		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
patient_id	String	Khóa ngoại liên kết tới bảng Patient
title	String	Tiêu đề cuộc hẹn
status	String	Trạng thái cuộc hẹn
appointment_date	Date	Ngày khám bệnh
symptoms	String	Triệu chứng của bệnh nhân
predicted_disease	Json	Bệnh dự đoán
number	Int32	Số thứ tự của cuộc hẹn
paymentMethod	String	Phương thức thanh toán
paymentCode	String	Mã thanh toán
reasonCancel	String	Lý do hủy cuộc hẹn

isPaid	Boolean	Trạng thái đã thanh toán hay chưa
price	Int32	Tổng tiền cuộc hẹn
type	String	Thể loại cuộc hẹn

Bảng AppointmentDetail: Lưu trữ thông tin các chi tiết cuộc hẹn trong hệ thống

Bảng 2.21 Mô tả bảng AppointmentDetail

Bảng AppointmentDetail		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
appointment_id	String	Khóa ngoại liên kết tới bảng Appointment
specialization_id	String	Khóa ngoại liên kết tới bảng Specialization
examStatus	String	Trạng thái khám bệnh
time	String	Thời gian khám bệnh
doctor_id	String	Khóa ngoại liên kết tới bảng Doctor
address	String	Phòng khám
price	Int32	Giá khám ở khoa và bác sĩ tương ứng
description	String	Mô tả chi tiết cuộc hẹn
createdAt	Date	Ngày tạo ra chi tiết cuộc hẹn
updatedAt	Date	Ngày chỉnh sửa chi tiết cuộc hẹn

Bảng MedicalRecord: Lưu trữ thông tin các bệnh án của bệnh nhân trong hệ thống

Bảng 2.22 Mô tả bảng MedicalRecord

Bảng MedicalRecord		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa ngoại
patient_id	String	Khóa ngoại liên kết tới bảng Patient
doctor_id	String	Khóa ngoại liên kết tới bảng Doctor
symptoms	String	Triệu chứng của bệnh
diagnosis	String	Chuẩn đoán của bác sĩ
prescription	Json	Đơn thuốc của bác sĩ
note	String	Ghi chú của bác sĩ
appointment_id	String	Khóa ngoại liên kết tới bảng Appointment
createdAt	Date	Ngày tạo ra bệnh án
updatedAt	Date	Ngày chỉnh sửa bệnh án

Bảng ReExamSchedule: Lưu trữ thông tin các ngày tái khám trong hệ thống

Bảng 2.23 Mô tả bảng ReExamSchedule

Bảng ReExamSchedule		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
medical_record_id	String	Khóa ngoại liên kết tới bảng MedicalRecord
re_exam_date	Date	Ngày tái khám
doctor_id	String	Khóa ngoại liên kết tới bảng Doctor

parent_appointment_id	String	Khóa ngoại cuộc hẹn gốc liên kết tới bảng Appointment
price	Int32	Giá tái khám
isPaid	Boolean	Trạng thái đã thanh toán tiền hay chưa
note	String	Ghi chú
appointment_id	String	Khóa ngoại cuộc hẹn được tạo dựa trên lịch tái khám liên kết tới bảng Appointment
createdAt	Date	Ngày tạo ra ngày tái khám
updatedAt	Date	Ngày chỉnh sửa ngày tái khám

Bảng Notification: Lưu trữ thông tin các thông báo trong hệ thống

Bảng 2.24 Mô tả bảng Notification

Bảng Notification		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
title	String	Tiêu đề thông báo
status	Boolean	Trạng thái đã đọc hay chưa
image	String	Hình ảnh thông báo
account_id	String	Khóa ngoại liên kết tới bảng Account
content	String	Nội dung
type	String	Kiểu thông báo

idTo	String	Khóa ngoại liên kết tới Item được nhắc đến trong thông báo (có thể là Appointment, ReExamSchedule)
createdAt	Date	Ngày tạo ra thông báo
updatedAt	Date	Ngày cập nhập thông báo

Bảng Question: Lưu trữ thông tin các câu hỏi của bệnh nhân trong hệ thống

Bảng 2.25 Mô tả bảng Question

Bảng Grammars		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
account_id	String	Khóa ngoại liên kết tới bảng Account
content	String	Nội dung câu hỏi
createdAt	Date	Ngày tạo ra câu hỏi
updatedAt	Date	Ngày cập nhập câu hỏi

Bảng Answer: Lưu trữ thông tin các câu trả lời của bác sĩ và quản trị viên trong hệ thống

Bảng 2.26 Mô tả bảng Answer

Bảng Answer		
Tên cột	Kiểu dữ liệu	Mô tả
_id	String	Khóa chính
question_id	String	Khóa ngoại liên kết với bảng Question
answer	String	Câu trả lời

account_id	String	Khóa ngoại liên kết tới bảng Account
createdAt	Date	Ngày tạo ra câu trả lời
updatedAt	Date	Ngày chỉnh sửa câu trả lời

2.8. Kết chương 2

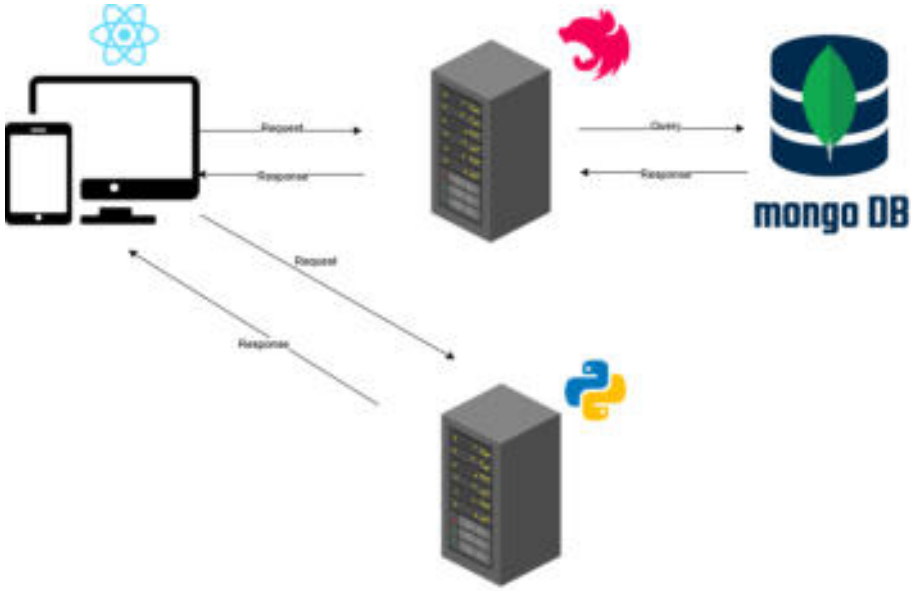
Chương 2 đã trình bày chi tiết về các thành phần chức năng và hoạt động chính của hệ thống thông qua việc phân tích các tác nhân liên quan, bao gồm Bệnh nhân, Bác sĩ và Quản trị viên. Mỗi tác nhân được mô tả rõ ràng về quyền hạn, chức năng sử dụng và vai trò trong toàn bộ hệ sinh thái ứng dụng.

Bên cạnh đó, các biểu đồ UML như biểu đồ ca sử dụng, biểu đồ tuần tự và biểu đồ hoạt động đã được sử dụng để trực quan hóa quy trình vận hành của hệ thống, giúp làm rõ cách thức tương tác giữa các thành phần cũng như luồng xử lý trong từng nghiệp vụ cụ thể. Phần thiết kế cơ sở dữ liệu cũng được giới thiệu nhằm đảm bảo khả năng lưu trữ, truy xuất và quản lý dữ liệu một cách hiệu quả, hỗ trợ hệ thống hoạt động ổn định và linh hoạt.

Tổng thể, chương này đóng vai trò nền tảng trong việc chuyển từ phân tích yêu cầu sang thiết kế hệ thống chi tiết, đồng thời cung cấp cơ sở để triển khai các chức năng trong chương tiếp theo một cách logic và khoa học.

CHƯƠNG 3. TRIỂN KHAI VÀ KẾT QUẢ

3.1. Cấu trúc hệ thống



Hình 3.1 Tổng quan về hệ thống

3.2. Môi trường phát triển

- Công cụ phát triển: Visual Studio Code, Postman
- Ngôn ngữ phát triển backend: TypeScript, NestJs
- Ngôn ngữ phát triển frontend: Javascript, ReactJS (Website), TypeScript, React Native (Mobile)
- Ngôn ngữ phát triển AI: Python, Flask
- Cơ sở dữ liệu: MongoDB Atlas
- Quản lý mã nguồn: Git, Github
- Môi trường để chạy chương trình:
  - Node.js: Phiên bản v20.19.2
  - Python: Phiên bản v3.11.4
  - MongoDB: Phiên bản v6.0.17

3.3. Môi trường triển khai

- Cơ sở dữ liệu: MongoDB Atlas
- Môi trường deploy: AWS EC2 Ubuntu 22.10 x64

3.4. Triển khai cấu trúc hệ thống

3.4.1. Tạo và cấu hình Azura Virtual Machine

- Đăng nhập vào Azura Portal.

- Truy cập mục Virtual Machines, chọn Create để tạo máy ảo mới.
- Đặt tên máy là datn, chọn hệ điều hành Linux (64-bit), và sử dụng VM generation V2 với kiến trúc x64.
- Tắt tính năng Hibernation.
- Chọn loại máy ảo: Standard B2s (2 vCPUs, 4 GiB RAM), phù hợp cho các ứng dụng web vừa và nhỏ.
- Thiết lập xác thực bằng SSH key hoặc mật khẩu tùy theo yêu cầu.
- Cấu hình Disk controller type là SCSI.
- Bỏ qua các cấu hình nâng cao như Host group, Host, Proximity placement group, và Capacity reservation group (để mặc định là N/A).
- Cấu hình Network Security Group (NSG) để mở các cổng cần thiết:
- SSH (22) – kết nối máy ảo qua terminal
- HTTP (80) và HTTPS (443) – nếu có triển khai web server
- MongoDB (27017) – nếu cần cho phép truy cập từ bên ngoài (khuyến khích giới hạn IP)
- Sau khi máy ảo được tạo, sử dụng SSH để kết nối và cài đặt các thành phần phần mềm cần thiết:
- Node.js – phục vụ backend ứng dụng
- MongoDB – hệ quản trị cơ sở dữ liệu NoSQL
- Nginx (nếu cần) – làm reverse proxy hoặc phục vụ frontend
- Các thư viện và môi trường khác tùy theo yêu cầu của dự án

3.4.2. Triển khai từng thành phần của hệ thống

a. Frontend (ReactJS)

- Cài đặt môi trường NodeJs version v20.19.2
- Cài đặt các dependencies
- Build ứng dụng cho môi trường production
- Triển khai lên server Nginx

b. Backend Server API (NestJS)

- Cài đặt môi trường NodeJs version v20.19.2
- Cài đặt các dependencies
- Build ứng dụng cho môi trường production

c. Backend Server AI (FlaskAPI)

Thu thập dữ liệu:

Dữ liệu bệnh học (medical_data.csv): Tập dữ liệu được thu thập từ các mô tả triệu chứng của bệnh nhân (gồm 600 dữ liệu với 212 bệnh khác nhau). Mỗi dòng bao gồm một câu mô tả triệu chứng bằng tiếng Anh, tương ứng với hai nhãn đầu ra:

Bệnh (Disease)

Chuyên khoa (Specialization)

Tập dữ liệu này đóng vai trò là nguồn thông tin đầu vào chính để huấn luyện mô hình dự đoán.

Patient_Problem	Disease	Specialization_English
Constant fatigue and muscle weakness, struggling to stay awake.	Chronic Fatigue Syndrome	General Internal Medicine, Neurology, Psychiatry
Frequent severe migraines, sensitivity to light and sound.	Migraine with Aura	Neurology
Sudden weight gain and feeling cold, especially in the hands and feet.	Hypothyroidism	Endocrinology
High fever, sore throat, and swollen lymph nodes, feeling very weak.	Mononucleosis	General Internal Medicine, Pediatrics
Excessive thirst and frequent urination, dry mouth persistently.	Diabetes Mellitus	Endocrinology, Nutrition
Sharp, stabbing chest pain that worsens when lying down or inhaling deeply.	Pericarditis	Cardiology
Unexplained weight loss and night sweats, feeling feverish.	Tuberculosis	Pulmonology, Radiology, Laboratory Medicine
Severe abdominal pain radiating to the back, nausea without vomiting.	Pancreatitis	Gastroenterology
Joint swelling and stiffness in the morning, especially in the fingers and wrists.	Rheumatoid Arthritis	General Internal Medicine, Orthopedics, Physical Medicine and Rehabilitation
Recurrent, painful blisters around the mouth and nose.	Herpes Simplex	Dermatology
Progressive difficulty swallowing, accompanied by heartburn.	Esophageal Reflux	Gastroenterology
Chronic diarrhea, abdominal cramps, and gas.	Irritable Bowel Syndrome	Gastroenterology
Intense itching, especially at night, with visible burrows on the skin.	Scabies	Dermatology
Yellowing of the skin and eyes, dark urine, and general fatigue.	Hepatitis	Gastroenterology, Laboratory Medicine
Sudden, severe headache with no known cause, blurred vision.	Subarachnoid Hemorrhage	Neurology, Intensive Care Unit (ICU)
Persistent cough with blood-tinged sputum, night sweats.	Lung Cancer	Oncology, Pulmonology, Radiology
Frequent nosebleeds, easy bruising, and prolonged bleeding from cuts.	Hemophilia	Laboratory Medicine, General Internal Medicine, Khoa Hồi sức cấp cứu
Severe itching and rash after consuming certain foods.	Food Allergy	Nutrition, General Internal Medicine, Khoa Hồi sức cấp cứu
Repeated urinary tract infections, burning sensation while urinating.	Urinary Tract Infection	Nephrology - Urology
Difficulty concentrating, memory lapses, and confusion.	Early Onset Dementia	Neurology, Psychiatry
I've been feeling extremely fatigued and find it hard to concentrate.	Chronic Fatigue Syndrome	General Internal Medicine, Neurology, Psychiatry
Experiencing frequent headaches, sensitivity to light and sound.	Migraine	Neurology
Noticed a persistent cough and occasional wheezing, especially at night.	Asthma	Pulmonology
Sudden episodes of rapid heartbeat, feeling faint.	Panic Attacks	Psychiatry, Neurology
Unintended weight loss and constant thirst.	Type 1 Diabetes	Endocrinology, Nutrition
Joint pain in hands and feet, stiffness in the morning.	Rheumatoid Arthritis	General Internal Medicine, Orthopedics, Physical Medicine and Rehabilitation

Hình 3.2 Dữ liệu thu thập

Tiền xử lý dữ liệu

Token hóa & padding: Văn bản được chuyển thành chuỗi số nguyên thông qua tokenizer, sau đó được padding đến độ dài cố định để phù hợp với mô hình LSTM.

Mã hóa nhãn: Các nhãn đầu ra (Disease, Specialization) được encode thành số hoặc one-hot vector.

Phân chia dữ liệu: Dữ liệu được chia thành hai tập: tập huấn luyện (train) và tập kiểm tra (test) theo tỷ lệ phổ biến (ví dụ: 80% - 20%).

Hình 3.3 Tiền xử lý dữ liệu

Lựa chọn mô hình học máy

LSTM (Long Short-Term Memory) : Mô hình sử dụng LSTM được thiết kế để dự đoán bệnh dựa trên mô tả triệu chứng đầu vào. Với khả năng ghi nhớ thông tin theo chuỗi, LSTM giúp mô hình hiểu ngữ cảnh trong văn bản và đưa ra dự đoán chính xác hơn.

Mô hình có 2 đầu ra:

- Dự đoán bệnh mà bệnh nhân có thể mắc.
- Xác định chuyên khoa phù hợp để điều trị.

```
input_layer = Input(shape=(max_length,))

# Tăng kích thước embedding và thêm Dropout
embedding = Embedding(input_dim=5000, output_dim=128)(input_layer)
embedding = Dropout(0.2)(embedding)

# Sử dụng Bidirectional LSTM với nhiều units hơn
lstm_layer = Bidirectional(LSTM(128, return_sequences=True))(embedding)
lstm_layer = Dropout(0.3)(lstm_layer)
lstm_layer = Bidirectional(LSTM(64))(lstm_layer)
lstm_layer = BatchNormalization()(lstm_layer)
lstm_layer = Dropout(0.3)(lstm_layer)

# Thêm lớp ẩn trước khi output
dense_layer = Dense(128, activation='relu')(lstm_layer)
dense_layer = Dropout(0.2)(dense_layer)

disease_output = Dense(len(label_encoder_disease.classes_), activation='softmax',
                       name='disease_output')(dense_layer)

specialization_output = Dense(len(label_encoder_specialization.classes_),
                              activation='softmax', name='specialization_output')(dense_layer)

model = Model(inputs=input_layer, outputs=[disease_output, specialization_output])
```

```
model = Model(inputs=input_layer, outputs=[disease_output, specialization_output])

model.compile(
    loss={'disease_output': 'categorical_crossentropy',
    'specialization_output': 'categorical_crossentropy'},
    optimizer='adam',
    metrics={'disease_output': ['accuracy'], 'specialization_output': ['accuracy']}
)

model.summary()
```

Hình 3.4 Lựa mô hình học máy

Huấn luyện mô hình

Thư viện sử dụng: TensorFlow/Keras.

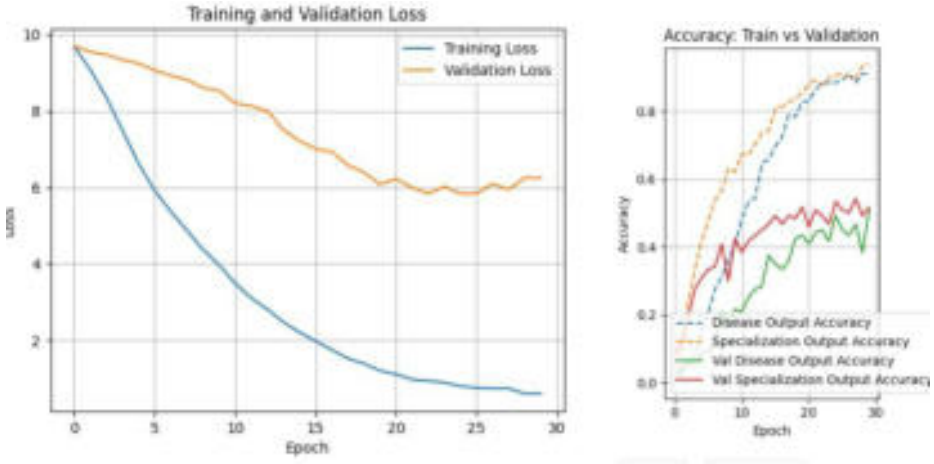
Cấu trúc mô hình: LSTM nhiều lớp kết hợp với các Dense layer cho từng nhãn đầu ra.

Tinh chỉnh siêu tham số: Learning rate, batch size, số epoch,… được thử nghiệm để tối ưu hiệu suất mô hình.

```
early_stop = EarlyStopping(
    monitor='val_loss',
    patience=5,              # Dừng nếu val_loss không giảm sau 5 epoch
    restore_best_weights=True
)

history = model.fit(
    X_train,
    {
        'disease_output': y_disease_train,
        'specialization_output': y_specialization_train
    },
    validation_data=(
        X_val,
        {
            'disease_output': y_disease_val,
            'specialization_output': y_specialization_val
        }
    ),
    epochs=50,
    batch_size=32,
    callbacks=[early_stop]
)
```

Hình 3.5 Huấn luyện mô hình



Hình 3.6 Mô hình huấn luyện

```
patient_input = "Itchy skin, red rash, sneezing"
make_prediction(patient_input)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 503ms/step
Predicted Disease: Psoriasis (79.4%)
Suggested Prescription: Topical corticosteroids; moisturizers. (41.0%)
Specialization: Khoa Da liễu (90.6%)
```

Hình 3.7 Kết quả huấn luyện

Triển khai Server AI (FlaskAPI)

Tích hợp mô hình học máy vào FlaskAPI: Định nghĩa các endpoint để nhận ảnh đầu vào từ client, xử lý và trả về kết quả nhận diện.

```
tokenizer = joblib.load(os.path.join(model_dir, "tokenizer.pkl"))
label_encoder_disease = joblib.load(os.path.join(model_dir, "label_encoder_disease.pkl"))
label_encoder_specialization = joblib.load(os.path.join(model_dir, "label_encoder_specialization.pkl"))
max_length = joblib.load(os.path.join(model_dir, "max_length.pkl"))
model = load_model(os.path.join(model_dir, "medical_model.h5"))
```

```
# --- Hàm dự đoán ---
def predict_disease(vietnamese_input, top_k=10):
    try:
        # Dịch tiếng Việt sang tiếng Anh
        english_input = GoogleTranslator(source='vi', target='en').translate(vietnamese_input)

        # Tiền xử lý văn bản
        sequence = tokenizer.texts_to_sequences([english_input])
        padded = pad_sequences(sequence, maxlen=max_length, padding='post')

        # Dự đoán
        predictions = model.predict(padded, verbose=0)
        disease_probs = predictions[0][0]
        specialization_probs = predictions[1][0]

        # Top-k bệnh
        top_indices = np.argsort(disease_probs)[::-1][:top_k]
        results = []

        # Chuyên khoa có xác suất cao nhất
        spec_index = np.argmax(specialization_probs)
        spec_en = label_encoder_specialization.inverse_transform([spec_index])[0]
        spec_vi = specialization_mapping.get(spec_en, spec_en)

        for i in top_indices:
            disease_en = label_encoder_disease.inverse_transform([i])[0]
            disease_vi = GoogleTranslator(source='en', target='vi').translate(disease_en)
            prob = float(round(disease_probs[i] * 100, 2))

            # Lấy chuỗi chuyên khoa từ mô hình (có thể là nhiều chuyên khoa ngăn cách bởi dấu phẩy)
            raw_specialization = label_encoder_specialization.inverse_transform([spec_index])[0]

            # Tách chuyên khoa tiếng Anh thành danh sách
            spec_list_en = [s.strip() for s in raw_specialization.split(',')]

            # Dịch từng chuyên khoa sang tiếng Việt
            spec_list_vi = [specialization_mapping.get(s, s) for s in spec_list_en]

            # Ghép lại thành chuỗi
            spec_vi = ", ".join(spec_list_vi)

            results.append({
                "name": disease_vi,
                "percent": prob,
                "specialization": spec_vi
            })

        return results

    except Exception as e:
        print(f"Prediction error: {str(e)}")
        raise

# --- Flask endpoint ---
@app.route('/predict', methods=['POST'])
def predict():
    data = request.json
    if 'Patient_Problem' not in data:
        return jsonify({"error": "Missing 'Patient_Problem' field"}), 400

    try:
        input_text = data['Patient_Problem']
        results = predict_disease(input_text)
        return jsonify({
            "original_input": input_text,
            "results": results
        })
    except Exception as e:
        return jsonify({"error": str(e)}), 500
```

Hình 3.8 Tích hợp mô hình học máy vào flaskAPI

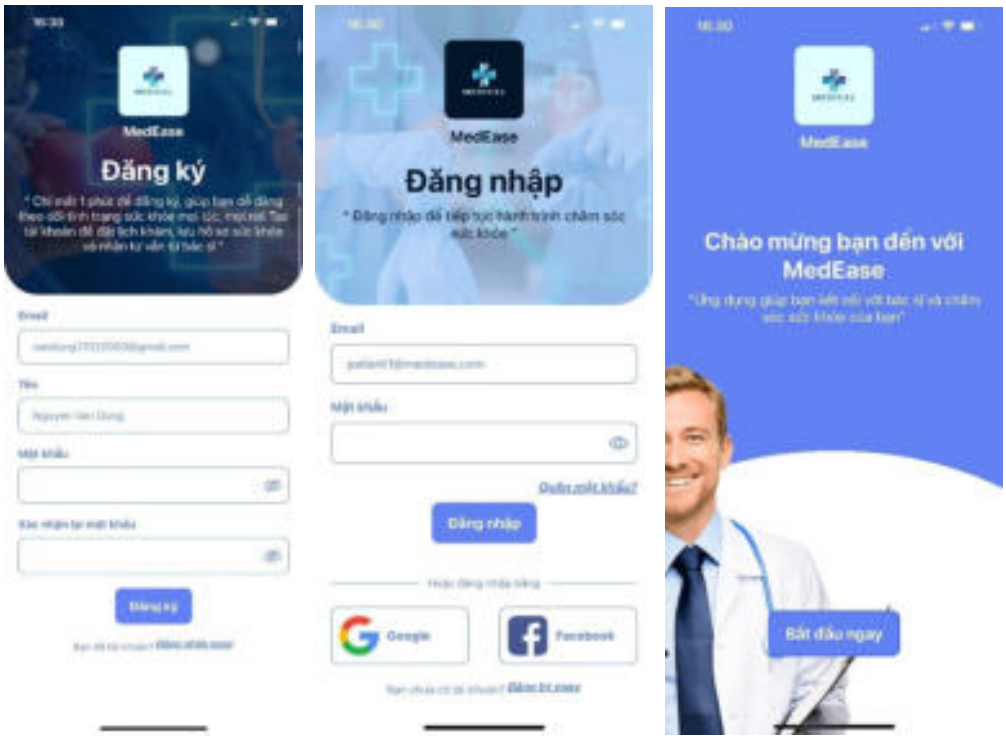
3.5. Kết quả thực tế

Hệ thống này bao gồm 3 view chính:

- View người dùng là bệnh nhân;
- View người dung là bác sĩ;
- View người dùng là quản trị viên.

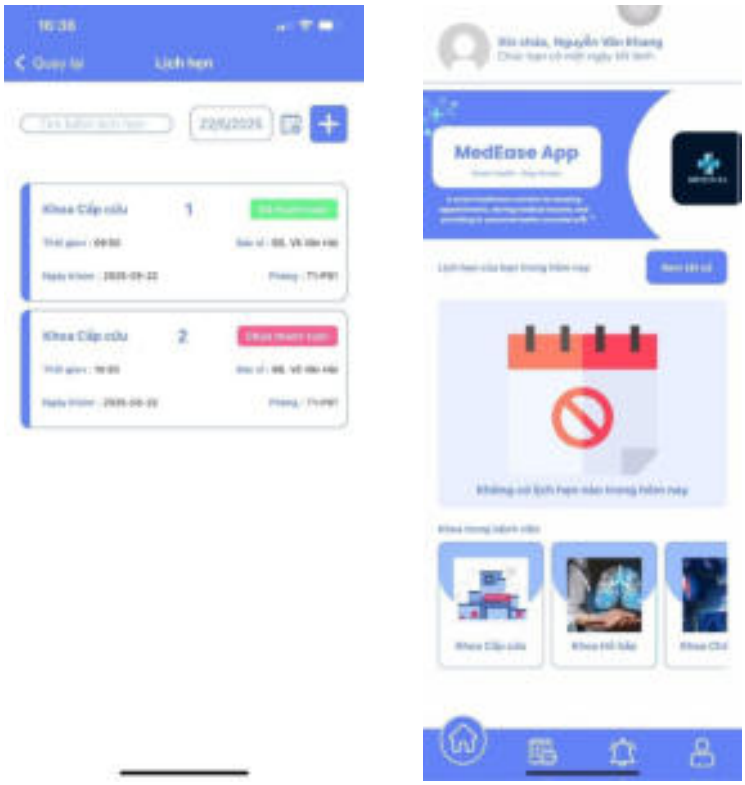
3.5.1. Giao diện người dùng là bệnh nhân

Giao diện màn hình chào mừng, đăng nhập và đăng kí: Giao diện được thiết kế tối giản với hai tông màu trắng và xanh nước biển, kết hợp các slogan khuyến khích bảo vệ sức khỏe, giúp người dùng dễ dàng truy cập và bắt đầu hành trình chăm sóc sức khỏe cùng AI.



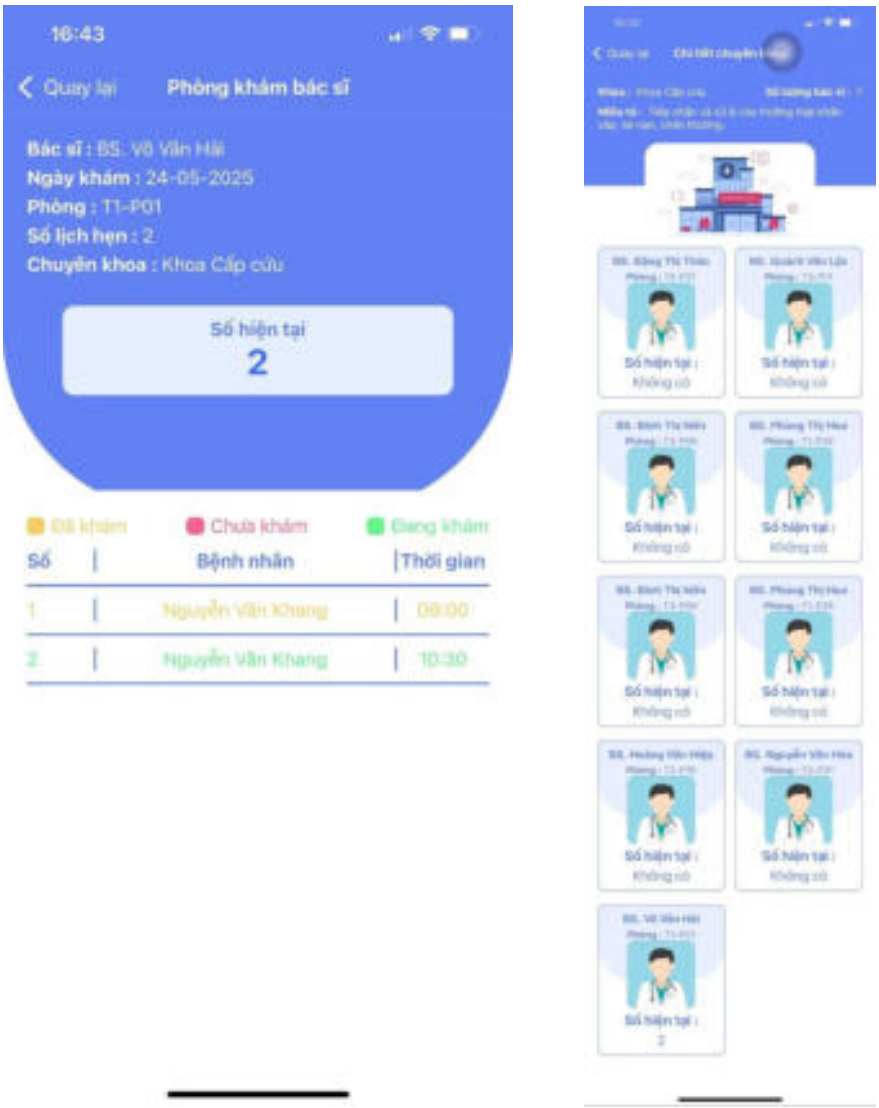
Hình 3.9 Giao diện màn hình chào mừng, đăng nhập và đăng kí

Giao diện màn hình chính và màn hình tìm kiếm lịch hẹn: Sau khi đăng nhập vào hệ thống bệnh nhân có thể xem tất cả lịch hẹn của mình trong hôm nay tại màn hình chính và có thể xem được danh sách các khoa hiện đang có trong bệnh viện. Bên cạnh đó bệnh nhân có thể xem được danh sách lịch hẹn của mình tại màn hình tìm kiếm lịch hẹn.



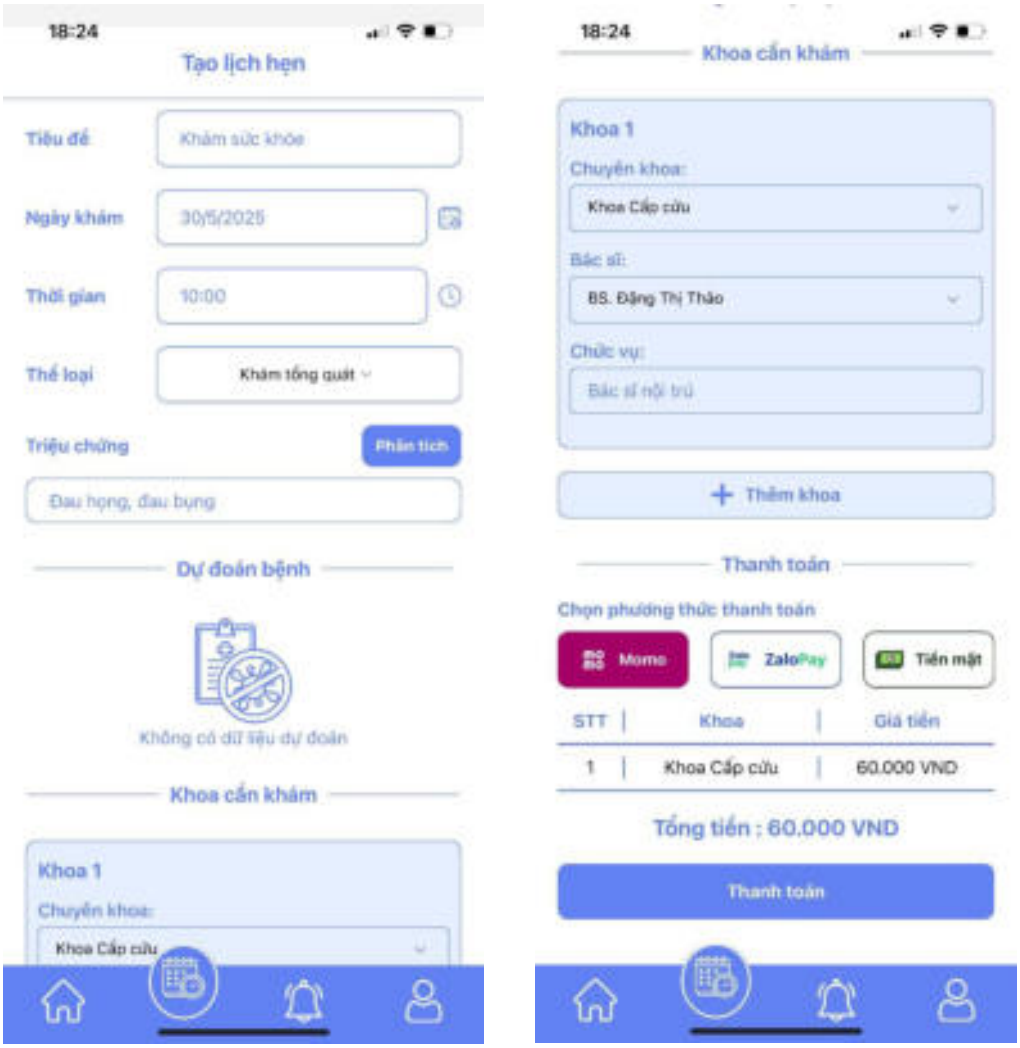
Hình 3.10 Giao diện màn hình chính và màn hình tìm kiếm lịch hẹn

Giao diện danh sách các bác sĩ đang có trong khoa và tình trạng trong phòng khám của bác sĩ : Tại màn hình chính bệnh nhân có thể chọn vào khoa tương ứng, màn hình sẽ hiển thị danh sách các bác sĩ đang có trong khoa đó. Khi bệnh nhân nhấn vào một bác sĩ bất kì sẽ hiển thị ra thông tin và tình trạng khám bệnh trong phòng khám bác sĩ tương ứng



Hình 3.11 Giao diện màn hình danh sách các bác sĩ đang có trong khoa và tình trạng trong phòng khám bác sĩ

Giao diện đặt lịch hẹn: Bệnh nhân có thể đặt lịch hẹn với bác ngay tại giao diện điện thoại và khi nhập triệu chứng vào bệnh nhân có thể nhận về lại danh sách các bệnh dự đoán bằng cách bấm vào nút "Phân tích". Bệnh nhân có thể lựa chọn hình thức thanh toán là zalopay và momo

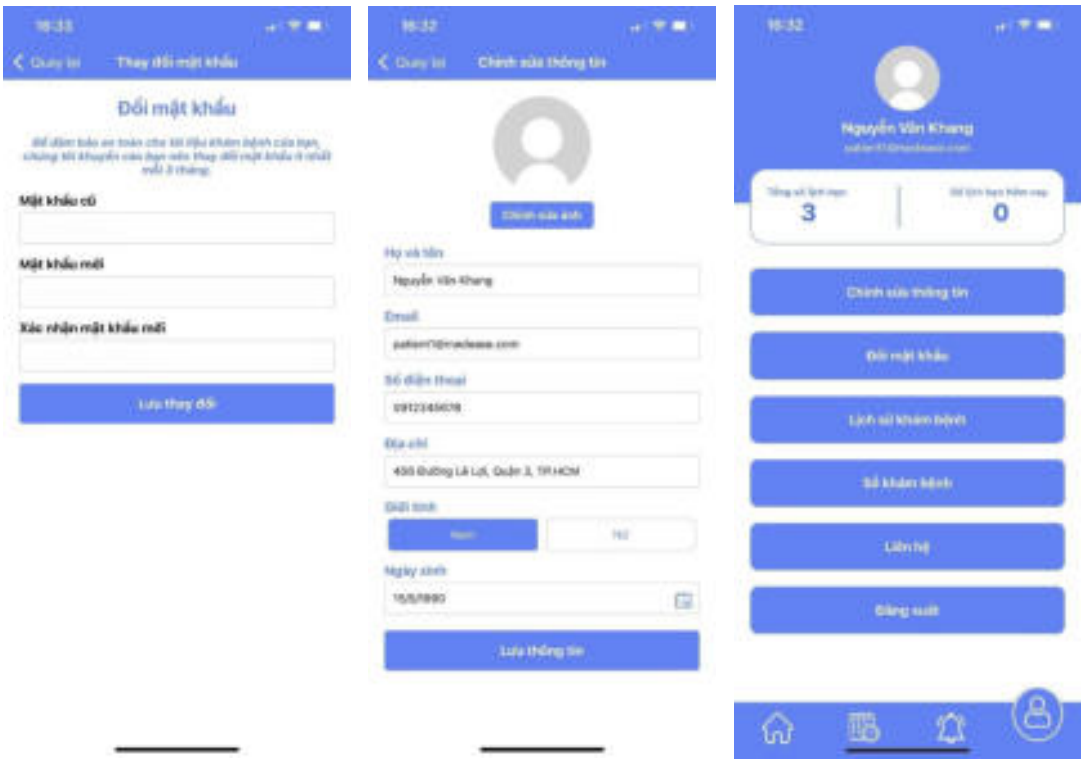


Hình 3.12 Giao diện đặt lịch hẹn

Giao diện màn hình thông báo: Bệnh nhân có thể nhận được thông báo về các thể loại liên quan đến cuộc hẹn, hỏi đáp, tái khám, … trong giao diện màn hình thông báo.

Hình 3.13 Giao diện màn hình thông báo

Giao diện màn hình trang cá nhân và chỉnh sửa thông tin cá nhân và mật khẩu: Bệnh nhân có thể xem các chức năng mình có thể sử dụng tại màn hình trang cá nhân. Bên cạnh đó bệnh nhân còn có thể chỉnh sửa lại thông tin của mình và mật khẩu tại 2 trang tương ứng

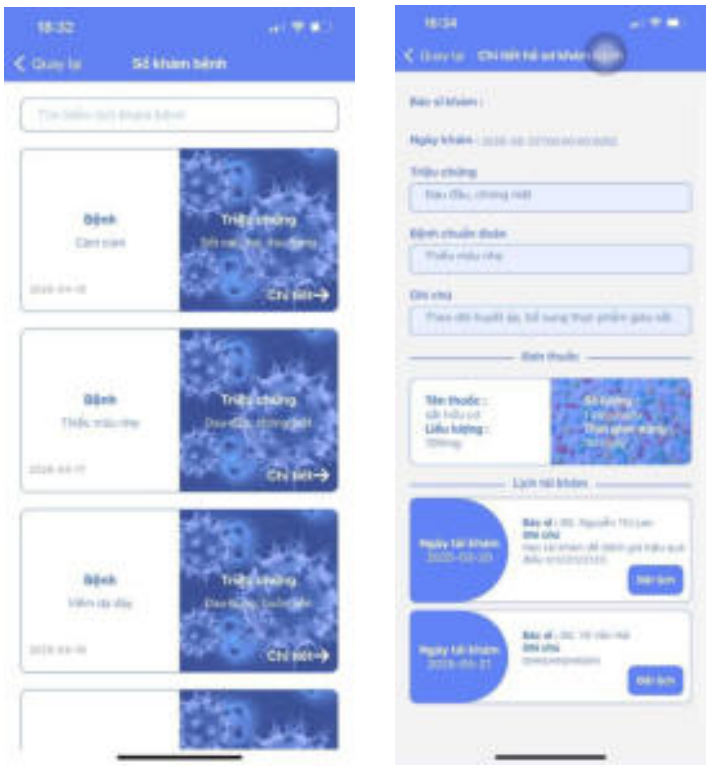


Hình 3.14 Giao diện màn hình trang cá nhân và chỉnh sửa thông tin cá nhân và mật khẩu

Giao diện màn hình lịch sử và chi tiết các cuộc hẹn: Bệnh nhân có thể xem lại danh sách và chi tiết các cuộc hẹn mà mình đã đặt trên hệ thống tại giao diện màn hình lịch sử và chi tiết các cuộc hẹn.

Hình 3.15 Giao diện màn hình lịch sử và chi tiết các cuộc hẹn

Giao diện màn hình danh sách và chi tiết sổ khám bệnh: Bệnh nhân có thể xem lại danh sách các chuẩn đoán mà bác sĩ đã khám cũng như danh sách thuốc tương ứng với từng bệnh tại danh sách và chi tiết sổ khám bệnh.



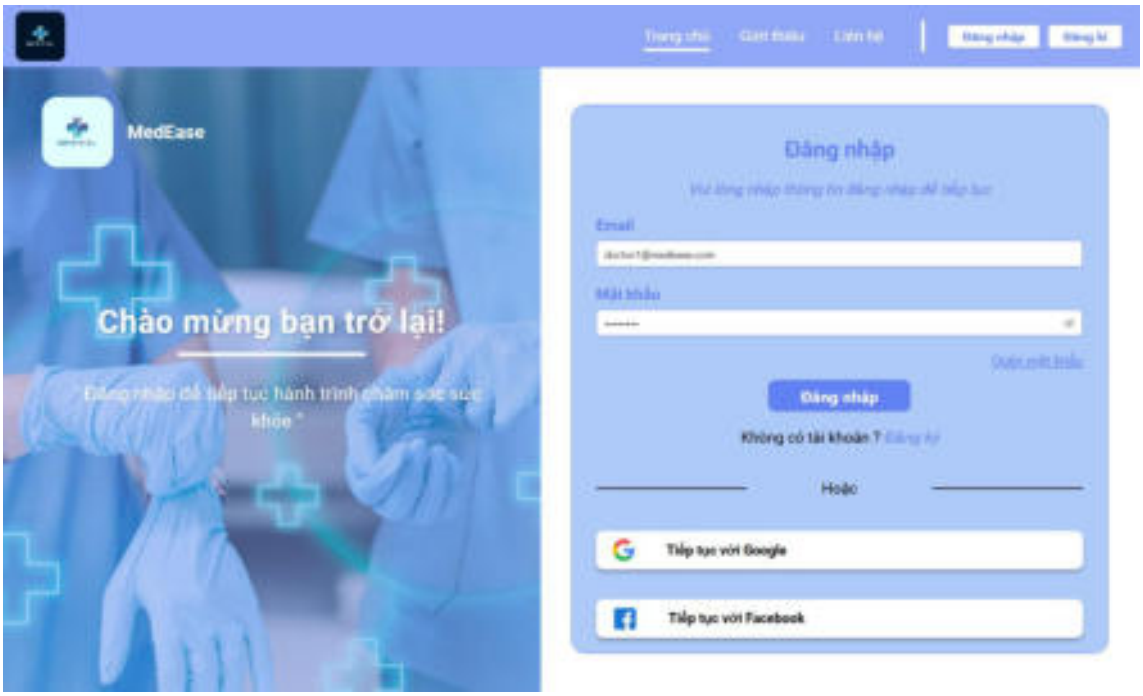
Hình 3.16 Giao diện màn hình danh sách và chi tiết sổ khám bệnh

3.5.2. Giao diện người dùng là bác sĩ

Giao diện trang chủ: Khách có thể xem slide, danh sách các bác sĩ và các khoa hiện đang có ở trong bệnh viên cũng như khách có thể xem phần giới thiệu về bệnh viện và phương thức liên lạc với bệnh viện tại giao diện trang chủ này.

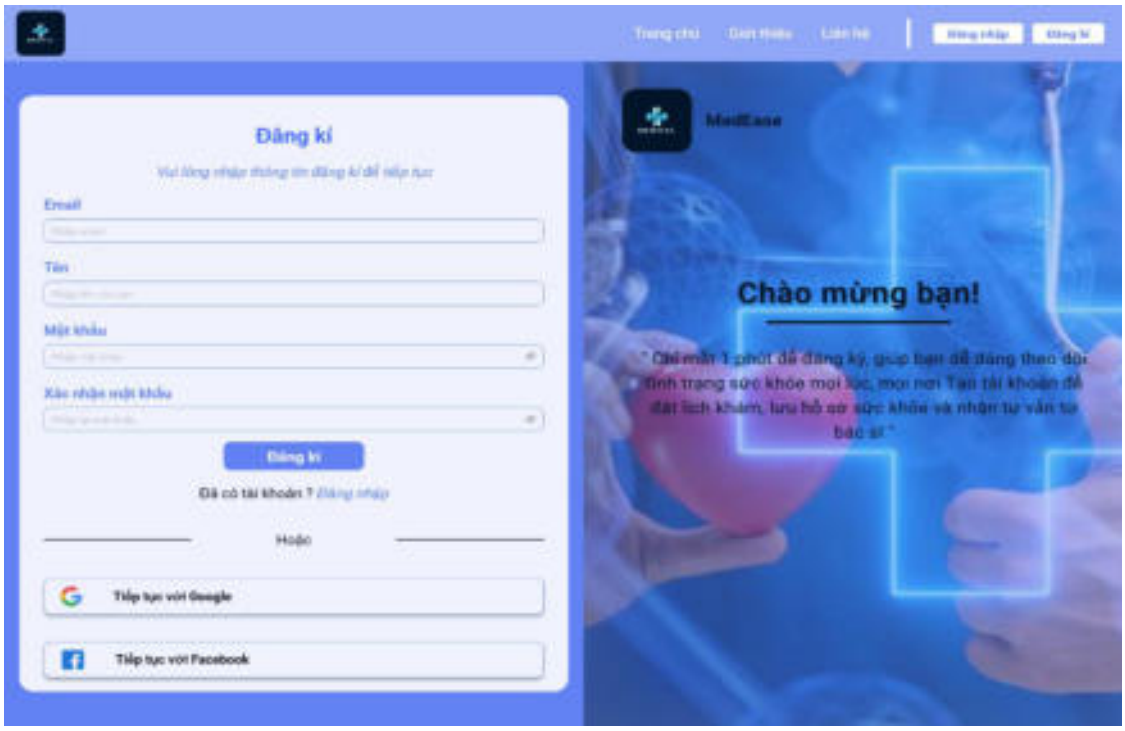
Hình 3.17 Giao diện trang chủ

Giao diện màn hình đăng nhập: Khách hàng có thể đăng nhập vào hệ thống dưới 2 vai trò là quản trị viên và bác sĩ của bệnh viện tại màn hình này.



Hình 3.18 Giao diện màn hình đăng nhập

Giao diện màn hình đăng kí: Khách hàng cũng có thể tạo tài khoản bệnh nhân của mình tại màn hình này, khách hàng chỉ có thể tạo tài khoản bệnh nhân tại màn hình này, đối với tài khoản bác sĩ chỉ do quản trị viên có thể tạo được.

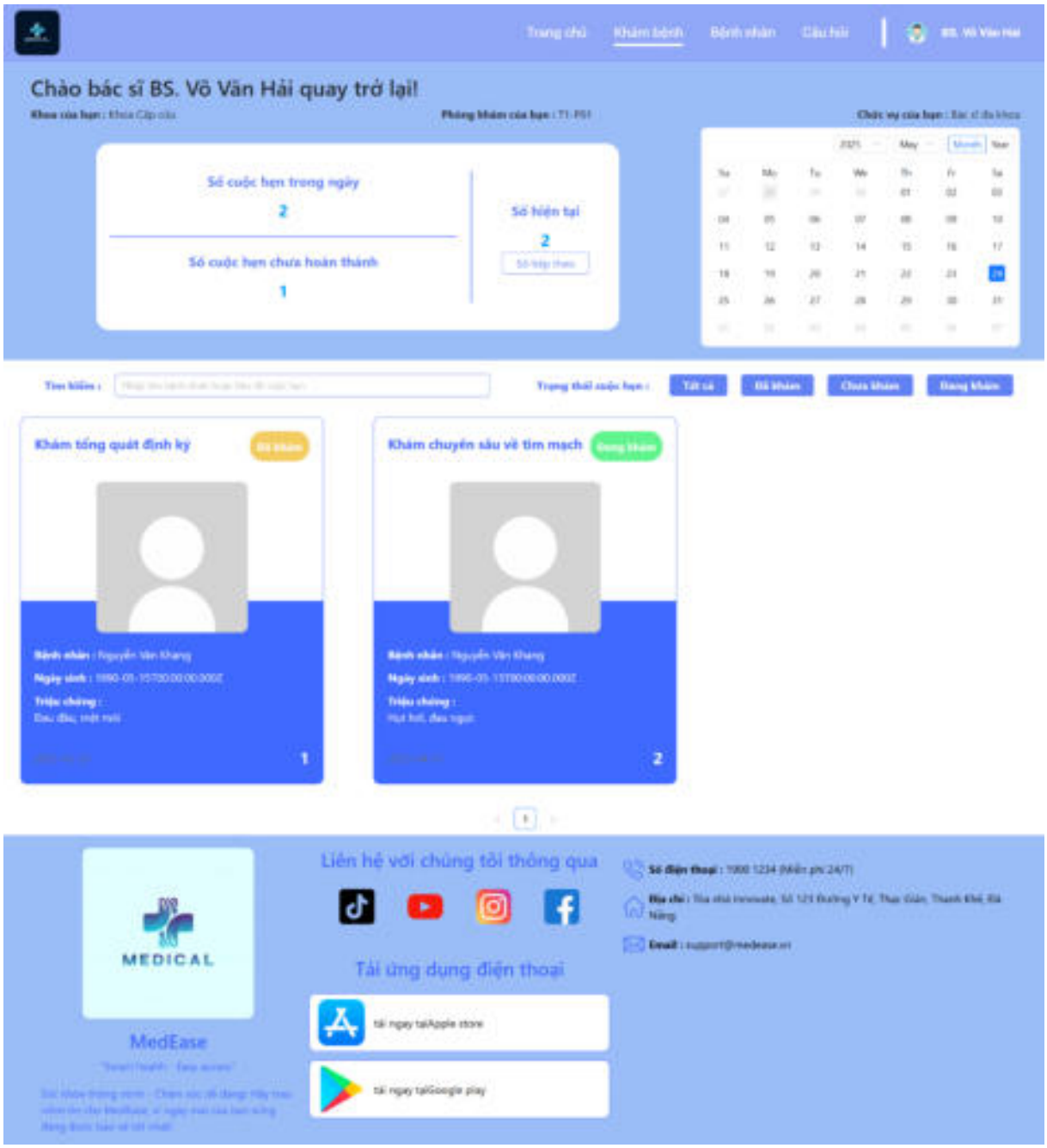


Hình 3.19 Giao diện màn hình đăng kí

Giao diện màn hình chính của bác sĩ: Sau khi đăng nhập vào hệ thống, bác sĩ xem thống kê số lịch hẹn trong tuần, danh sách các cuộc hẹn trong ngày và danh sách các câu hỏi của bệnh nhân tại màn hình này.

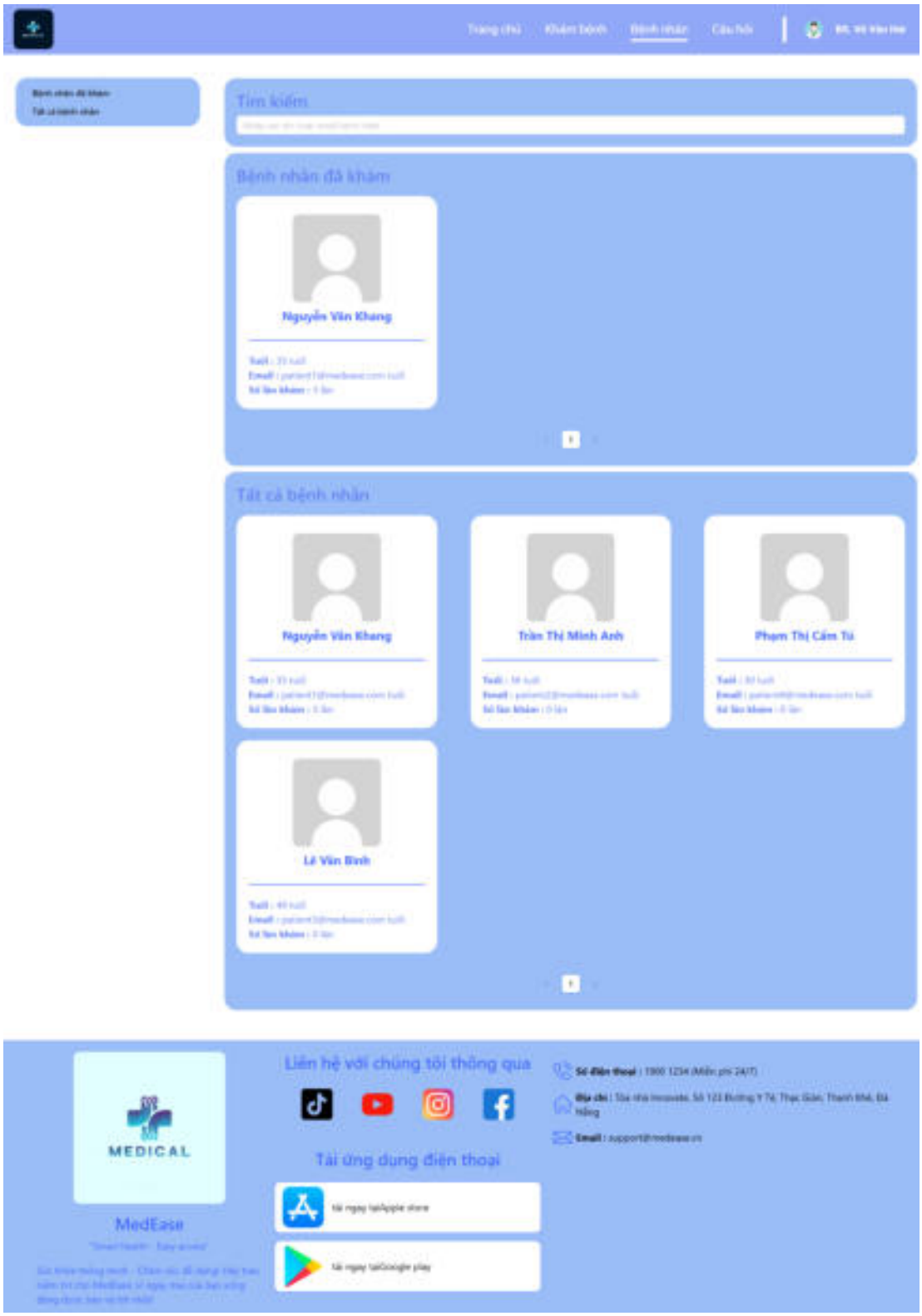
Hình 3.20 Giao diện màn hình đăng kí

Giao diện màn hình phòng khám: Bác sĩ có thể xem được số lượng lịch hẹn, danh sách các cuộc hẹn của mình theo từng ngày và tiến hành chuyển tiếp số tại màn hình phòng khám của mình



Hình 3.21 Giao diện màn hình phòng khám

Giao diện màn hình danh sách bệnh nhân: Bác sĩ có thể xem danh sách các bệnh nhân có trong hệ thống và các bệnh nhân mà mình đã khám trong màn hình này.

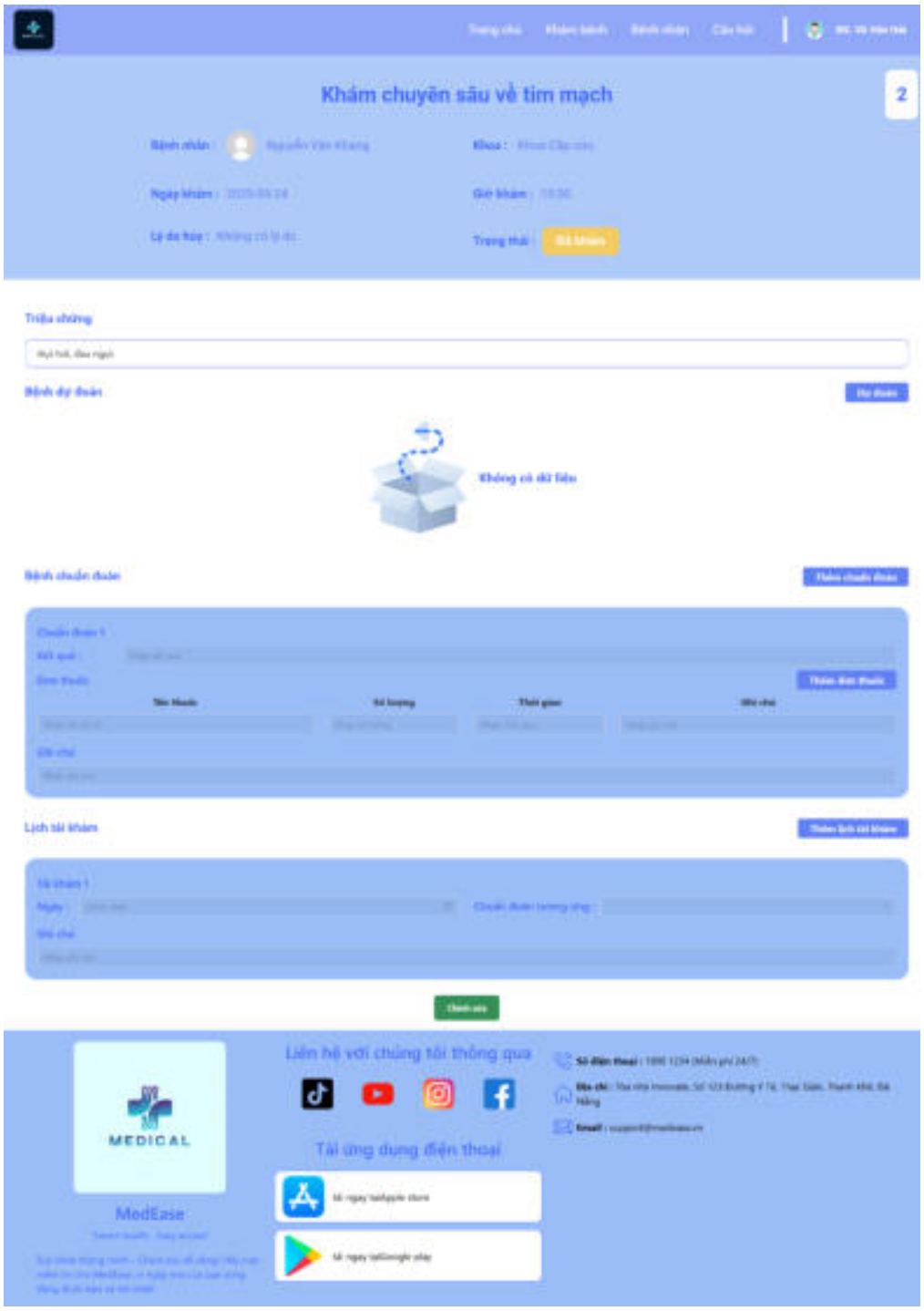


Hình 3.22 Giao diện màn hình danh sách bệnh nhân

Giao diện màn hình danh sách các câu hỏi: Bác sĩ có thể xem và trả lời danh sách các câu hỏi do bệnh nhân đặt ra tại màn hình này.

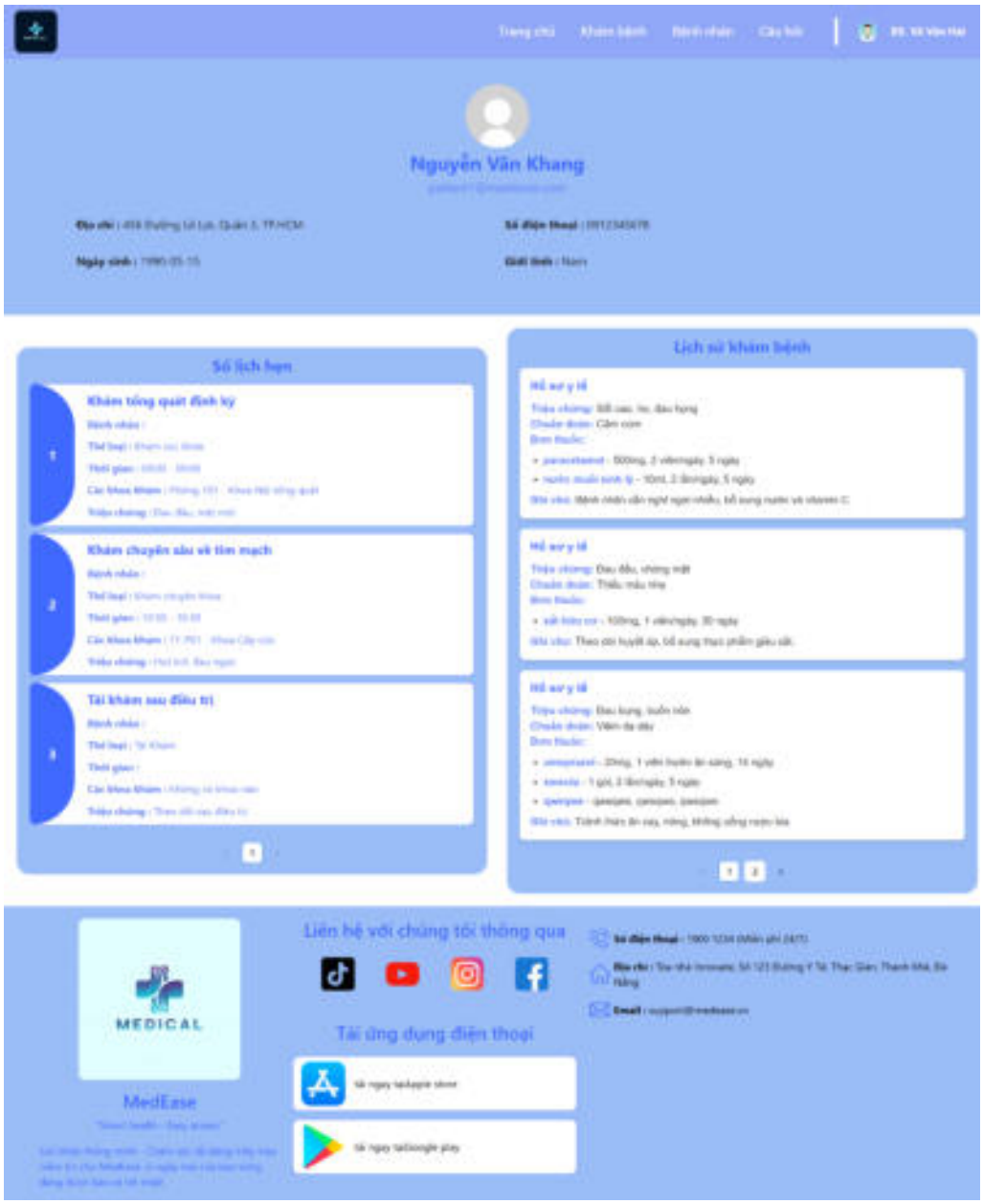
Hình 3.23 Giao diện màn hình danh sách các câu hỏi

Giao diện màn hình chi tiết cuộc hẹn: Bác sĩ có thể xem nội dung cuộc hẹn, triệu chứng của bệnh nhân. Sau đó bác sĩ có thể đưa ra chuẩn đoán của mình và thêm lịch tái khám cho bệnh nhân tại màn hình này. Nếu như bác sĩ đã đưa ra chuẩn đoán và lịch tái khám bác sĩ có thể chỉnh sửa lại nội dung của cuộc hẹn này.



Hình 3.24 Giao diện màn hình chi tiết cuộc hẹn

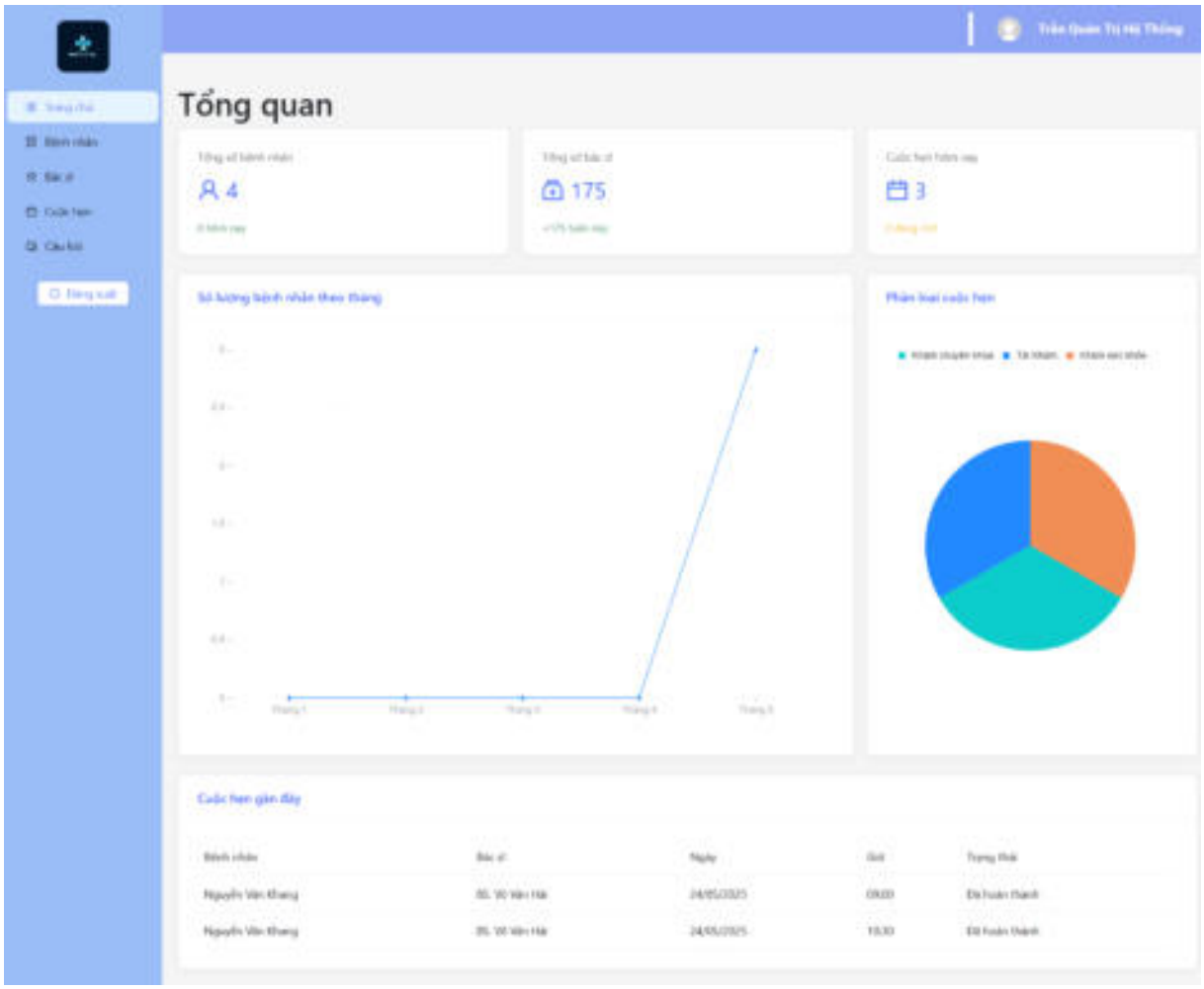
Giao diện màn hình thông tin cá nhân bệnh nhân: Bác sĩ có thể xem lại danh sách lịch hẹn và sổ khám bệnh của bệnh nhân tại màn hình thông tin cá nhân của bệnh nhân bằng cách chọn vào bệnh nhân tương ứng tại màn hình danh sách bệnh nhân.



Hình 3.25 Giao diện màn hình thông tin cá nhân bệnh nhân

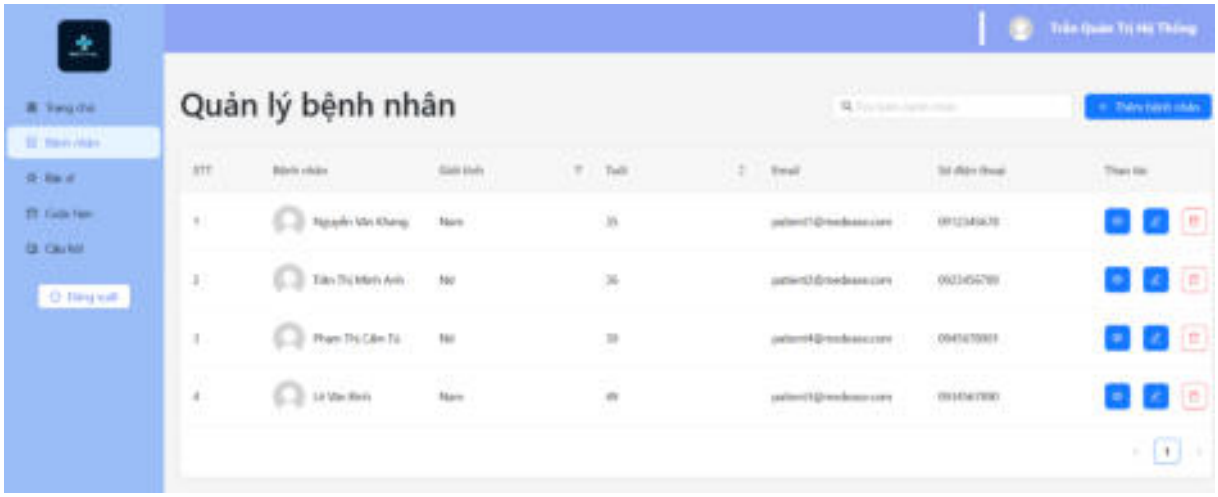
3.5.3. Giao diện quản trị viên

Giao diện thống kê hệ thống: Quản trị viên có thể xem số lượng bệnh nhân và bác sĩ cũng như số lượng cuộc hẹn hôm nay đang có trong hệ thống tại màn hình này. Ngoài ra quản trị viên có thể xem được thống kê cuộc hẹn theo tháng và theo từng thể loại tại màn hình thống kê này



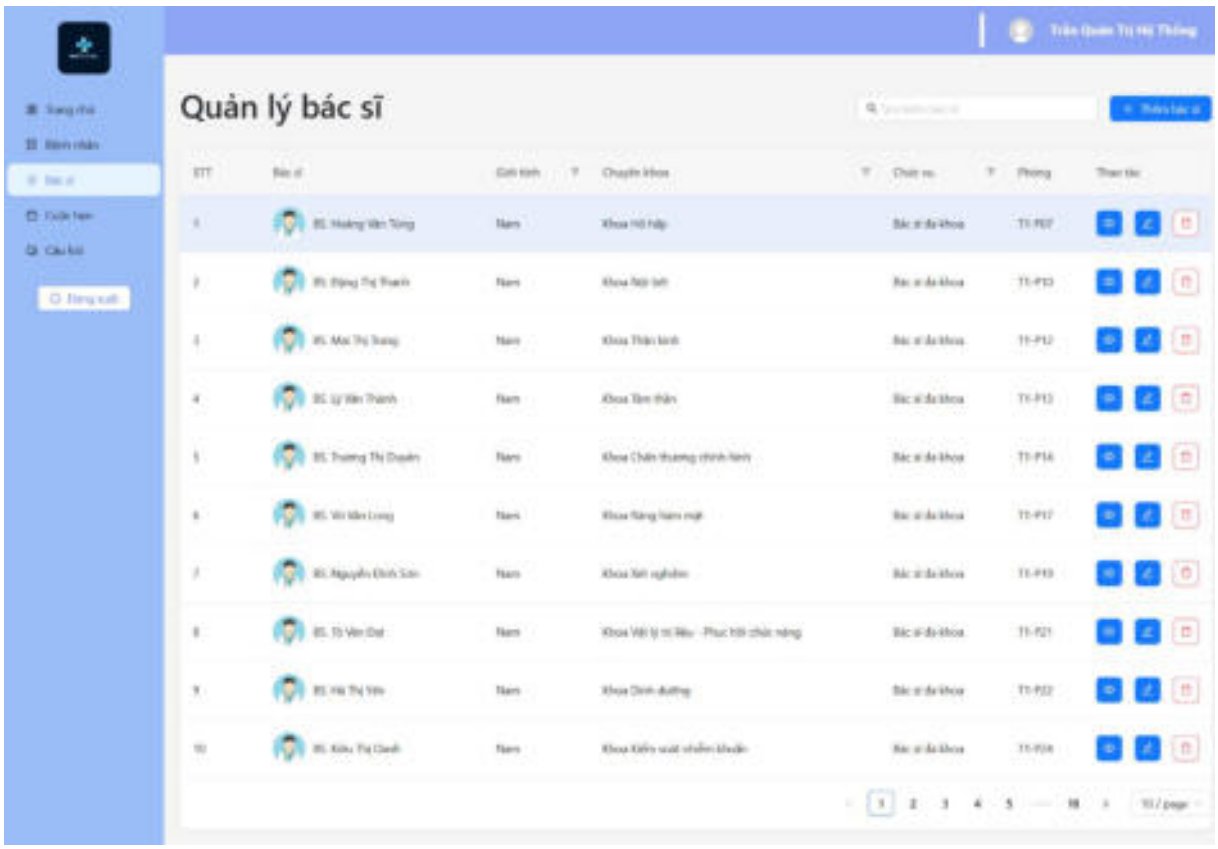
Hình 3.26 Giao diện thống kê hệ thống

Giao diện màn hình quản lý bệnh nhân: Quản trị viên có thể xem thông tin, chỉnh sửa, xóa và thêm bệnh nhân tại màn hình này



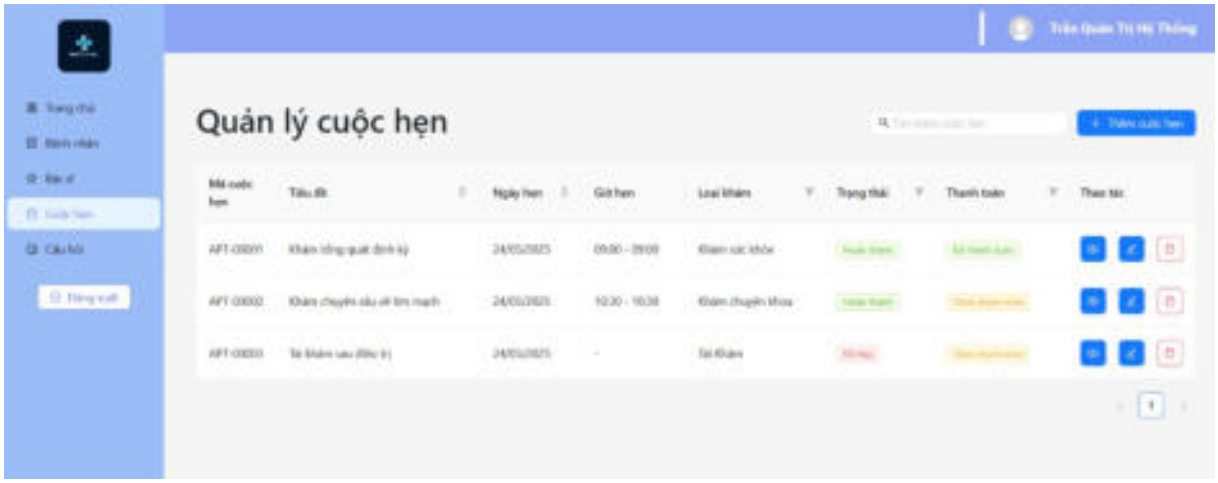
Hình 3.27 Giao diện quản lý bệnh nhân

Giao diện quản lý bác sĩ : Quản trị viên có thể xem thông tin, chỉnh sửa, xóa và thêm bác sĩ tại màn hình này



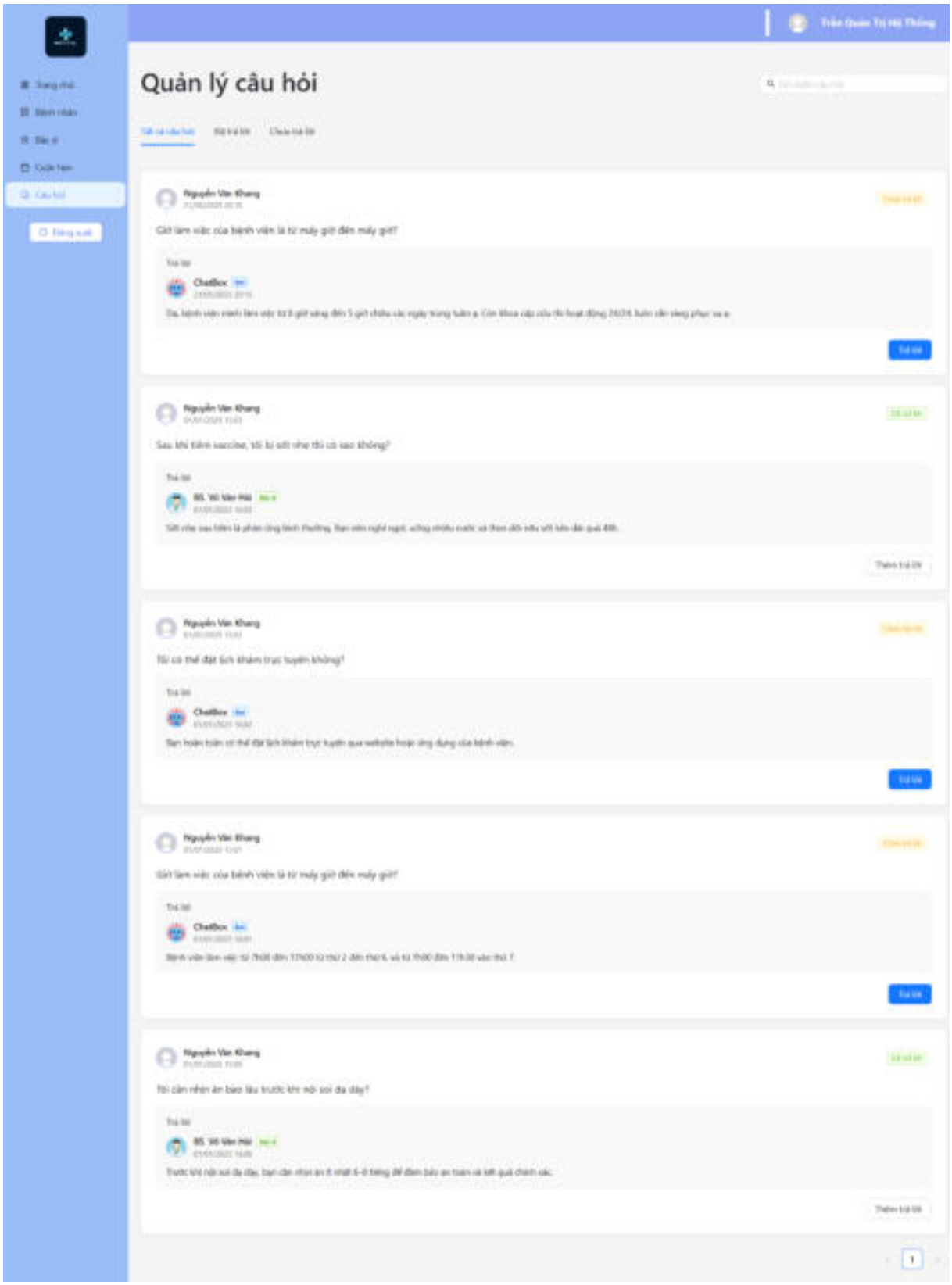
Hình 3.28 Giao diện quản lý bác sĩ

Giao diện quản lý cuộc hẹn: Quản trị viên có thể xem thông tin, chỉnh sửa, xóa và tạo cuộc hẹn tại màn hình này



Hình 3.29 Giao diện quản lý cuộc hẹn

Giao diện quản lý danh sách câu hỏi: Quản trị viên có thể xem và trả lời các câu hỏi của bệnh nhân đặt ra cho bệnh viện tại màn hình này.



Hình 3.30 Giao diện quản lý câu hỏi

3.6. Kết chương 3

Chương 3 đã trình bày toàn bộ quá trình triển khai hệ thống, bao gồm thiết kế kiến trúc tổng thể, xây dựng giao diện người dùng, thiết lập các chức năng chính cũng như tích hợp mô hình AI vào hệ thống thực tế. Hệ thống được phát triển với cấu trúc rõ ràng, dễ mở rộng và đáp ứng được các yêu cầu đặt ra về mặt kỹ thuật cũng như trải nghiệm người dùng.

Kết quả triển khai cho thấy hệ thống hoạt động ổn định, giao diện thân thiện và hỗ trợ hiệu quả quá trình chẩn đoán, tư vấn sức khỏe từ xa. Đây là bước hoàn thiện quan trọng, góp phần hiện thực hóa ý tưởng ứng dụng công nghệ AI trong lĩnh vực chăm sóc y tế, mở ra hướng tiếp cận mới trong việc hỗ trợ người dùng và bác sĩ trong công tác khám chữa bệnh.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết quả đạt được

Trong quá trình triển khai dự án, em đã nghiên cứu, phân tích và ứng dụng kiến thức chuyên môn vào thực tế, từ đó đạt được các mục tiêu trọng yếu đã đề ra.

Về mặt lý thuyết:

- Hệ thống hóa kiến thức về phân tích và thiết kế cơ sở dữ liệu, bao gồm việc xác định yêu cầu nghiệp vụ y tế, xây dựng mô hình dữ liệu phù hợp với hồ sơ sức khỏe điện tử, lịch sử điều trị và thông tin người dùng.
- Củng cố và vận dụng thành thạo các ngôn ngữ lập trình (TypeScript, JavaScript), cũng như một số framework hiện đại như NestJS, ReactJS, React Native để phát triển ứng dụng web tương tác.
- Nắm được nguyên lý hoạt động cơ bản của mô hình ứng dụng tích hợp trí tuệ nhân tạo trong phân tích dữ liệu và hỗ trợ chẩn đoán.

Về mặt thực tiễn ứng dụng:

- Ứng dụng đã đáp ứng được các chức năng cốt lõi như đăng ký khám bệnh, quản lý lịch hẹn, tra cứu hồ sơ bệnh án và theo dõi tiến trình điều trị. Giao diện người dùng được thiết kế tối giản, dễ hiểu, phù hợp với cả nhân viên y tế lẫn bệnh nhân trong quá trình sử dụng.
- Hệ thống tích hợp công nghệ trí tuệ nhân tạo (AI) để hỗ trợ việc thu thập triệu chứng, phân tích sơ bộ và dự đoán bệnh , từ đó hỗ trợ bác sĩ trong việc ra quyết định lâm sàng. Đặc biệt, việc tích hợp chatbot thông minh giúp giải đáp các thắc mắc phổ biến của người bệnh một cách tự động, giảm tải thời gian phản hồi cho đội ngũ y tế và nâng cao trải nghiệm người dùng.
- Thông qua đó, ứng dụng góp phần hiện đại hóa quy trình khám chữa bệnh, tối ưu hiệu suất vận hành và từng bước thay thế các thủ tục giấy tờ truyền thống bằng quy trình điện tử hóa linh hoạt và hiệu quả hơn.

2. Những vấn đề còn tồn tại

- Chức năng quản trị: Một số tính năng quản trị nâng cao (như thống kê chuyên sâu theo chuyên khoa, quản lý phân quyền đa cấp) chưa được phát triển đầy đủ do giới hạn thời gian thực hiện.
- Hiệu suất xử lý: Khi số lượng người dùng truy cập đồng thời tăng cao, hệ thống có thể phát sinh độ trễ trong phản hồi.

- Trải nghiệm người dùng: Thiết kế giao diện cần được điều chỉnh để phù hợp hơn với người dùng lớn tuổi hoặc những người không quen sử dụng công nghệ.
- Tính năng tương tác: Các công cụ tương tác thời gian thực giữa bác sĩ và bệnh nhân như tư vấn trực tuyến, chat, gửi đơn thuốc điện tử vẫn đang ở mức cơ bản.
- Dữ liệu học máy: Hệ thống AI hiện tại chỉ hoạt động ở mức tham khảo, chưa được huấn luyện với dữ liệu y tế thực tế đầy đủ để hỗ trợ chuyên sâu.

3. Hướng phát triển

Mặc dù hệ thống hiện tại đã đáp ứng các chức năng cơ bản trong quy trình khám chữa bệnh, tuy nhiên vẫn còn nhiều tiềm năng để mở rộng và nâng cao trải nghiệm người dùng trong tương lai. Một số định hướng phát triển có thể triển khai nếu có thêm thời gian và nguồn lực bao gồm:

- Tối ưu lộ trình khám bệnh: Đối với bệnh nhân cần thăm khám ở nhiều khoa, hệ thống sẽ tự động sắp xếp thứ tự khám hợp lý nhất, nhằm tiết kiệm thời gian di chuyển và hạn chế chờ đợi không cần thiết.
- Hướng dẫn di chuyển trong bệnh viện: Cung cấp bản đồ nội bộ của bệnh viện và chỉ dẫn đường đi đến từng phòng khám, hỗ trợ người bệnh di chuyển thuận tiện hơn, đặc biệt là tại các cơ sở có quy mô lớn.
- Mở rộng cơ sở dữ liệu y tế: Nâng cấp kho dữ liệu liên quan đến tiền sử bệnh, chỉ số sức khỏe và kết quả xét nghiệm nhằm hỗ trợ bác sĩ đưa ra chẩn đoán chính xác hơn.
- Tăng cường khả năng tương tác giữa bác sĩ và bệnh nhân: Triển khai hệ thống hỏi – đáp hoặc tư vấn từ xa để người bệnh có thể liên hệ, trao đổi với bác sĩ ngay cả khi chưa đến trực tiếp bệnh viện.

Thông qua các định hướng trên, hệ thống không chỉ hỗ trợ bệnh nhân hiệu quả hơn mà còn góp phần xây dựng một môi trường khám chữa bệnh thông minh, hiện đại và lấy người dùng làm trung tâm.

TÀI LIỆU THAM KHẢO

[1] NestJs: https://docs.nestjs.com

[2] React: https://react.dev

[3] MongoDB Atlas : https://www.mongodb.com/docs/atlas/getting-started

[4] Flask : https://flask.palletsprojects.com/en/stable

[5] Azure Virtual Machines – Microsoft: https://learn.microsoft.com/en-us/azure/virtual-machines

[6] Azure Blob Storage – Microsoft : https://learn.microsoft.com/en-us/azure/storage/blobs

[7] Azure App Service Deployment – Microsoft : https://learn.microsoft.com/en-us/azure/app-service

[8] LTSM - Understanding LSTM Networks : https://colah.github.io/posts/2015-08-Understanding-LSTMs

[9] Docker : https://docs.docker.com/get-started/get-docker/

[10] Bootstrap : https://getbootstrap.com/docs/4.1/getting-started/introduction/

[11] Redux : https://redux.js.org/introduction/getting-started