**TRƯỜNG ĐẠI HỌC PHENIKAA**

**BÁO CÁO BÀI TẬP LỚN**

**XÂY DỰNG ỨNG DỤNG ĐÁNH GIÁ XA SÚT TRÍ TUỆ CỦA NGƯỜI GIÀ BẰNG HỆ THỐNG CHUẨN ĐOÁN TỪ XA**

**20010943 Phan Thành Đạt 20010943**@st.phenikaa-uni.edu.vn

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | TS. TRẦN ĐỨC MINH |
| **Khoa:** | Công nghệ thông tin |
| **HÀ NỘI, 08/2024** | |

# Lời cam kết

Họ và tên nhóm sinh viên:

* Phan Thành Đạt

Điện thoại liên lạc: . . . .0342611368. . . . Email: . . . Phandat2002h@gmail.com

Lớp: . . Công nghệ thông tin Việt Nhật. . . . . . . Hệ đào tạo: . . Chính quy. .

Tôi cam kết Bài tập lớn (BTL) là công trình nghiên cứu của bản thân tôi. Các kết quả nêu trong BTL là trung thực, là thành quả của riêng tôi, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong BTL – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Tôi xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày 08 tháng 08 năm 2024*  Tác giả BTL  Phan Thành Đạt |

*Mẫu QT.ĐT.19.M08*

|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC PHENIKAA** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập - Tự do - Hạnh phúc** |

**NHẬN XÉT ĐỒ ÁN/KHÓA LUẬN TỐT NGHIỆP**

**CỦA GIẢNG VIÊN HƯỚNG DẪN**

Giảng viên hướng dẫn: ................................................................. Bộ môn:...................

Tên đề tài: .......................................................................................................................

.........................................................................................................................................

Sinh viên thực hiện: ..................................................................... Lớp:..........................

NỘI DUNG NHẬN XÉT

I. Nhận xét ĐAKLTN:

- Nhận xét về hình thức: ………………………………………………………………...

…………………………………………………………………………………………...

- Tính cấp thiết của đề tài: ………………………………………………………………

…………………………………………………………………………………………...

- Mục tiêu của đề tài: ……………………………………………………………………

…………………………………………………………………………………………...

- Nội dung của đề tài: …………………………………………………………………...

…………………………………………………………………………………………...

- Tài liệu tham khảo: ……………………………………………………………………

…………………………………………………………………………………………...

- Phương pháp nghiên cứu: ……………………………………………………………..

…………………………………………………………………………………………...

- Tính sáng tạo và ứng dụng: ……………………………………………………………

…………………………………………………………………………………………...

II. Nhận xét tinh thần và thái độ làm việc của sinh viên:

…………………………………………………………………………………………..

…………………………………………………………………………………………..

…………………………………………………………………………………………..

…………………………………………………………………………………………..

**III. Kết quả đạt được:**

…………………………………………………………………………………………..

…………………………………………………………………………………………..

…………………………………………………………………………………………..

…………………………………………………………………………………………...

**IV. Kết luận:** Đồng ý cho bảo vệ: Không đồng ý cho bảo vệ:

*Hà Nội, ngày … tháng … năm 20…*

**GIẢNG VIÊN HƯỚNG DẪN**

*(Ký, ghi rõ họ tên)*

*Mẫu QT.ĐT.19.M08*

|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO**  **TRƯỜNG ĐẠI HỌC PHENIKAA** | **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập - Tự do - Hạnh phúc** |

**NHẬN XÉT ĐỒ ÁN/KHÓA LUẬN TỐT NGHIỆP**

**CỦA GIẢNG VIÊN PHẢN BIỆN**

Giảng viên hướng dẫn: ................................................................. Bộ môn:...................

Tên đề tài: .......................................................................................................................

.........................................................................................................................................

Sinh viên thực hiện: ..................................................................... Lớp:..........................

Giảng viên hướng dẫn:....................................................................................................

NỘI DUNG NHẬN XÉT

I. Nhận xét ĐAKLTN:

- Bố cục, hình thức trình bày:.………………………………………………………...

…………………………………………………………………………………………...

- Đảm bảo tính cấp thiết, hiện đại, không trùng lặp:..………………………………

…………………………………………………………………………………………...

- Nội dung: ……………………………………………………………………..............

…………………………………………………………………………………………...

- Mức độ thực hiện:………………………………………………………………….....

…………………………………………………………………………………………....

**II. Kết quả đạt được:** :………………………………………………………………….

…………………………………………………………………………………………....

**III. Ưu như­ợc điểm:** :…………………………………………………………………...

…………………………………………………………………………………………....

**IV. Kết luận:** Đồng ý cho bảo vệ: Không đồng ý cho bảo vệ:

*Hà Nội, ngày … tháng … năm 20…*

**GIẢNG VIÊN PHẢN BIỆN**

*(Ký, ghi rõ họ tên)*

# Tóm tắt

Trong thời đại khoa học công nghệ và y học phát triển, tuổi thọ của con người ngày càng được kéo dài, dẫn đến số lượng người cao tuổi ngày càng tăng. Tuy nhiên, tuổi tác cao thường kéo theo sự suy giảm của một số chức năng trên cơ thể, đặc biệt là não bộ. Một trong những vấn đề phổ biến nhất mà người cao tuổi phải đối mặt là chứng suy giảm trí nhớ. Hiện tại, việc kiểm tra và đánh giá tình trạng này cần sự can thiệp của các chuyên gia chuyên nghiệp, đòi hỏi người cao tuổi phải đến trực tiếp các cơ sở y tế. Để giải quyết vấn đề này, tôi đã lựa chọn phát triển một hệ thống chuẩn đoán sơ bộ từ xa. Hệ thống này mang lại nhiều lợi ích, chẳng hạn như tiết kiệm chi phí và thời gian cho người dùng. Hệ thống chuẩn đoán từ xa hoạt động bằng cách sử dụng các công nghệ hiện đại để thu thập và phân tích dữ liệu về tình trạng trí nhớ của người dùng thông qua các bài kiểm tra trực tuyến. Sau đó, kết quả sẽ được gửi đến các chuyên gia y tế để đánh giá và đưa ra khuyến nghị. BTL này áp dụng những kiến thức đã học để cải thiện chất lượng sống của mọi người, đặc biệt là người cao tuổi. Kết quả cuối cùng của dự án là xây dựng được một ứng dụng đánh giá sa sút trí tuệ từ xa, giúp người cao tuổi dễ dàng tiếp cận dịch vụ y tế và nâng cao chất lượng cuộc sống.

# Mục lục

[Lời cam kết ii](#_Toc174743070)

[Tóm tắt vii](#_Toc174743071)

[Mục lục viii](#_Toc174743072)

[Danh mục hình vẽ xi](#_Toc174743073)

[Danh mục bảng xiii](#_Toc174743074)

[Danh mục các từ viết tắt xiv](#_Toc174743075)

[Danh mục thuật ngữ xv](#_Toc174743076)

[Chương 1 Giới thiệu đề tài 1](#_Toc174743077)

[1.1 Đặt vấn đề 1](#_Toc174743078)

[1.2 Mục tiêu và phạm vi đề tài 1](#_Toc174743079)

[1.3 Định hướng giải pháp 2](#_Toc174743080)

[1.4 Bố cục bài tập lớn 2](#_Toc174743081)

[Chương 2 Khảo sát và phân tích yêu cầu 4](#_Toc174743082)

[2.1 Khảo sát hiện trạng 4](#_Toc174743083)

[2.2 Tổng quan chức năng 6](#_Toc174743084)

[2.2.1 Biểu đồ use case tổng quan 6](#_Toc174743085)

[2.2.2 Biểu đồ use case phân 7](#_Toc174743086)

[2.3 Đặc tả chức năng 9](#_Toc174743087)

[2.3.1 Đặc tả use case login 9](#_Toc174743088)

[2.3.2 Đặc tả use case chỉnh sửa/tìm kiếm thông tin 12](#_Toc174743089)

[2.3.3 Đặc tả use case đánh giá kết quả của bệnh nhân. 18](#_Toc174743090)

[2.3.4 Đặc tả use case nhận bài kiểm tra của bệnh nhân 21](#_Toc174743091)

[2.3.5 Đặc tả use case kết nối tương tác giữa bệnh nhân và bác sĩ 23](#_Toc174743092)

[kết nối tương tác giữa bệnh nhân và bác sĩ 23](#_Toc174743093)

[2.4 Yêu cầu phi chức năng 25](#_Toc174743094)

[Chương 3 Công nghệ sử dụng 26](#_Toc174743095)

[**3.1 Ngôn ngữ java** 26](#_Toc174743096)

[**3.2 Famework angular** 29](#_Toc174743097)

[3.3 Ngôn ngữ Javascript 31](#_Toc174743098)

[3.4 Công nghệ peerJs 32](#_Toc174743099)

[3.5 Web RTC 33](#_Toc174743100)

[3.6 WebSocket 34](#_Toc174743101)

[3.7 công nghệ MySql 34](#_Toc174743102)

[Chương 4 Phát triển và triển khai ứng dụng 36](#_Toc174743103)

[4.1 Thiết kế kiến trúc 36](#_Toc174743104)

[4.1.1 Kiến trúc Mô hình MVVM( Model-view-view/model). 36](#_Toc174743105)

[4.1.2.BACK-END MICROSERVICE – SPRINGBOOT 37](#_Toc174743106)

[4.1.2 Các câu hỏi trong bào đánh giá 40](#_Toc174743107)

[4.2 Thiết kế chi tiết 40](#_Toc174743108)

[4.2.1 Thiết kế giao diện 40](#_Toc174743109)

[4.2.2 Thiết kế cơ sở dữ liệu 51](#_Toc174743110)

[4.3 Xây dựng ứng dụng 53](#_Toc174743111)

[4.3.1 Thư viện và công cụ sử dụng 53](#_Toc174743112)

[4.3.2 Kết quả đạt được 53](#_Toc174743113)

[4.4 Kiểm thử 54](#_Toc174743114)

[4.4.1 kiểm thử màn hình đăng nhập. 54](#_Toc174743115)

[4.4.2 kiểm thử màn hình quản lý. 55](#_Toc174743116)

[4.4.3 kiểm thử màn hình đánh giá bệnh nhân. 56](#_Toc174743117)

[4.4.4 kiểm thử chức năng thay đổi thông tin cá nhân của bản thân. 57](#_Toc174743118)

[4.4.5 kiểm thử màn hình kiểm tra của bệnh nhân. 57](#_Toc174743119)

[Chương 5 Các giải pháp và đóng góp nổi bật 59](#_Toc174743120)

[Chương 6 Kết luận và hướng phát triển 60](#_Toc174743121)

[6.1 Kết luận 60](#_Toc174743122)

[6.2 Hướng phát triển 60](#_Toc174743123)

[Tài liệu tham khảo 62](#_Toc174743124)

# Danh mục hình vẽ

[***Hình 1:Use case tổng quan của đề tài 6***](#_Toc174744419)

[***Hình 2:Use case đặc tả chức năng login 9***](#_Toc174744420)

[***Hình 3:Biểu đồ tuần tự chức năng đăng nhập 10***](#_Toc174744421)

[***Hình 4:Đặc tả chức năng tìm kiếm/ chỉnh sửa thông tin. 12***](#_Toc174744422)

[***Hình 5: Biểu đồ tuần tự chức năng chỉnh sửa và tìm kiếm thông tin. 12***](#_Toc174744423)

[***Hình 6: Đặc tả chức năng đánh giá kết quả của bệnh nhân 18***](#_Toc174744424)

[***Hình 7: Biểu đồ tuần tự chức năng đánh giá kết quả của bệnh nhân 19***](#_Toc174744425)

[***Hình 8: Biểu đồ tuần tự chức năng nhận bài kiểm tra của bệnh nhân 21***](#_Toc174744426)

[***Hình 9: Biểu đồ tuần tự chức năng kết nối tương tác giữa bệnh nhân và bác sĩ 23***](#_Toc174744427)

[***Hình 10: Sơ đồ hoạt động của fameWork Angular. 29***](#_Toc174744428)

[***Hình 11: Mô tả kiến trúc back-end. 38***](#_Toc174744429)

[***Hình 12: Sơ đồ nghiệp vụ tại back-end 39***](#_Toc174744430)

[***Hình 13:Minh họa chức năng đăng nhâp. 41***](#_Toc174744431)

[***Hình 14:Minh họa trang quản lý thông tin bệnh nhân. 42***](#_Toc174744432)

[***Hình 15:Minh họa trang quản lý thông tin người dùng. 42***](#_Toc174744433)

[***Hình 16:Minh họa chức năng thêm mới bệnh nhân. 43***](#_Toc174744434)

[***Hình 17:Minh họa chức năng thêm mới người dùng. 43***](#_Toc174744435)

[***Hình 18 :Minh họa thêm mới thành công người dùng/bệnh nhân. 44***](#_Toc174744436)

[***Hình 19 :Minh họa chức năng cập nhật thông tin bệnh nhân. 44***](#_Toc174744437)

[***Hình 20:Minh họa chức năng cập nhật thông tin người dùng. 45***](#_Toc174744438)

[***Hình 21: Minh họa cập nhật thông tin thành công 45***](#_Toc174744439)

[***Hình 22:Minh họa xem thông tin lịch sử đánh giá của bác sĩ 46***](#_Toc174744440)

[***Hình 23:Minh họa cảnh báo xóa người dùng/bệnh nhân. 46***](#_Toc174744441)

[***Hình 24: Minh họa xóa người dùng/bệnh nhân thành công 47***](#_Toc174744442)

[***Hình 25: Minh họa chọn các bài đánh giá tương ứng ở phía bác sĩ phụ trách. 47***](#_Toc174744443)

[***Hình 26:Minh họa chọn các bài đánh giá tương ứng ở phía bệnh nhân. 47***](#_Toc174744444)

[***Hình 27: Minh họa kiểm tra bài đánh giá thứ nhất ở màn hình bác sĩ. 48***](#_Toc174744445)

[***Hình 28: Minh họa kiểm tra bài đánh giá thứ nhất ở màn hình bệnh nhân. 48***](#_Toc174744446)

[***Hình 29: Minh họa cập nhật thông tin cá nhân của màn hình bác sĩ. 49***](#_Toc174744447)

[***Hình 30:Minh họa kiểm tra kết quả các bài đánh giá ở màn hình bệnh nhân. 49***](#_Toc174744448)

[***Hình 31: Minh họa kiểm tra bài đánh giá thứ hai ở màn hình bác sĩ. 50***](#_Toc174744449)

[***Hình 32: Minh họa kiểm tra bài đánh giá thứ hai ở màn hình bệnh nhân. 50***](#_Toc174744450)

[***Hình 33: Minh họa ghi chú đánh giá khi đã hoàn thành kiểm tra của bác sĩ. 51***](#_Toc174744451)

[***Hình 34:Minh họa biểu đồ E-R Diagram của dự án 51***](#_Toc174744452)

# Danh mục bảng

[bảng 1: Mô tả use case đăng nhập 10](#_Toc174745136)

[bảng 2: Luồng use case đăng nhập 11](#_Toc174745137)

[bảng 3: Mô tả use case chỉnh sửa và tìm kiếm thông tin. 13](#_Toc174745138)

[bảng 4:Luồng tìm kiếm thông tin 14](#_Toc174745139)

[bảng 5: Luồng thêm mới bệnh nhân 15](#_Toc174745140)

[bảng 6: Luồng chỉnh sửa thông tin bệnh nhân 16](#_Toc174745141)

[bảng 7: Luồng xóa người dùng 18](#_Toc174745142)

[bảng 8:Mô tả use case đánh giá kết quả của bệnh nhân 19](#_Toc174745143)

[bảng 9: Luồng đánh giá kết quả kiểm tra của bệnh nhân. 21](#_Toc174745144)

[bảng 10: Mô tả use case nhận bài kiểm tra của bệnh nhân 22](#_Toc174745145)

[bảng 11: Luồng nhận bài kiểm tra của bệnh nhân. 23](#_Toc174745146)

[bảng 12: Mô tả use case kết nối tương tác giữa bác sĩ và bệnh nhân 25](#_Toc174745147)

[bảng 13: So sánh giữa WebSocket và HTTP 34](#_Toc174745148)

[bảng 14:Danh sách thư viện và công cụ sử dụng 53](#_Toc174745149)

[bảng 15: Mô tả kiểm thử màn hình đăng nhập 55](#_Toc174745150)

[bảng 16:Mô tả kiểm thử màn hình quản lý 57](#_Toc174745151)

[bảng 17: Mô tả kiểm thử màn hình đánh giá bệnh nhân. 57](#_Toc174745152)

[bảng 18: Mô tả kiểm thử chức năng thay đổi thông tin cá nhân của bản thân. 57](#_Toc174745153)

[bảng 19: Mô tả kiểm thử màn hình kiểm tra của bệnh nhân. 58](#_Toc174745154)

# Danh mục các từ viết tắt

|  |  |
| --- | --- |
| API | Application Programming Interface  Giao diện lập trình ứng dụng |
| **DOM** | Document Object Model |
| **EUD** | End-User Development  Phát triển ứng dụng người dùng cuối |
| **GWT** | Google Web Toolkit  Công cụ lập trình Javascript bằng Java của Google |
| **HTML** | HyperText Markup Language  Ngôn ngữ đánh dấu siêu văn bản |
| **CNTT** | Công nghệ thông tin |
| **BTL** | Bài tập lớn |
| **SV** | Sinh viên |
| **UI/UX** | User Interface/ User Experience |

# Danh mục thuật ngữ

|  |  |
| --- | --- |
| Browser | Trình duyệt |
| **Cache memory** | Bộ nhớ đệm |
| **E-commerce** | Thương mại điện tử |
| **Bloatware** | Ứng dụng nhà sản xuất tích hợp vào thiết bị |
| **Interpreter** | Trình thông dịch |
| **Compiler** | Trình biên dịch |

# Giới thiệu đề tài

Chương 1 có độ dài từ 3 đến 6 trang với các nội dung sau đây

## Đặt vấn đề

Hiện nay, khi khoa học công nghệ và y học ngày càng phát triển, tuổi thọ của con người đã được kéo dài đáng kể. Điều này dẫn đến sự gia tăng đáng kể số lượng người cao tuổi trong xã hội. Tuy nhiên, cùng với sự gia tăng này là sự xuất hiện phổ biến của các vấn đề về sức khỏe, đặc biệt là suy giảm trí nhớ và sa sút trí tuệ. Đây là một vấn đề nghiêm trọng, ảnh hưởng đến chất lượng cuộc sống của người cao tuổi, gây khó khăn trong sinh hoạt hàng ngày và tạo gánh nặng cho gia đình và xã hội. Nếu vấn đề này được giải quyết, nó sẽ mang lại lợi ích to lớn, không chỉ cải thiện chất lượng cuộc sống cho người cao tuổi mà còn giảm bớt áp lực cho gia đình và hệ thống y tế. Hơn nữa, giải pháp cho vấn đề này có thể được áp dụng rộng rãi trong nhiều lĩnh vực khác, như chăm sóc sức khỏe từ xa, hỗ trợ tâm lý và tư vấn y tế, góp phần nâng cao chất lượng dịch vụ y tế và phát triển các ứng dụng công nghệ trong y học.

## Mục tiêu và phạm vi đề tài

Ở Việt Nam, hiện chưa có nhiều sản phẩm và dịch vụ hỗ trợ đánh giá tình trạng sa sút trí tuệ ở người cao tuổi. Vậy nên mọi người thường đến trực tiếp các cơ sở y tế để thực hiện các bài kiểm tra và đánh giá bởi các chuyên gia. Nhu cầu của người dùng hiện nay là có một hệ thống tiện lợi, tiết kiệm thời gian và chi phí, đồng thời vẫn đảm bảo được độ tương đối chính xác trong việc đánh giá tình trạng trí tuệ.

Qua việc so sánh và đánh giá tình trạng thực tế, tôi nhận thấy rằng cần phải có một hệ thống có thể đáp ứng được đầy đủ nhu cầu của người dùng. Hệ thống phải đáp ứng được một số hạn chế chính bao gồm: yêu cầu người dùng phải đến trực tiếp các cơ sở y tế, chi phí cao, thời gian chờ đợi lâu, và sự phức tạp trong việc sử dụng công nghệ.

Dựa trên những phân tích và đánh giá này, đề tài hướng tới việc giải quyết cụ thể các hạn chế hiện tại bằng cách phát triển một phần mềm đánh giá tình trạng sa sút trí tuệ từ xa cho người cao tuổi. Phần mềm này sẽ có các chức năng chính như: Trực tiếp được đánh giá từ các chuyên gia, thực hiện các bài kiểm tra trí nhớ trực tuyến, thu thập và phân tích dữ liệu tự động, và cung cấp kết quả đánh giá sơ bộ nhanh chóng. Mục tiêu của đề tài là tạo ra một đột phá trong việc sử dụng công nghệ để nâng cao chất lượng sống của người cao tuổi, giúp họ dễ dàng tiếp cận dịch vụ y tế mà không cần phải di chuyển đến các cơ sở y tế.

## Định hướng giải pháp

Để giải quyết vấn đề đã xác định, tôi sẽ áp dụng công nghệ web để phát triển một hệ thống đánh giá tình trạng sa sút trí tuệ từ xa. Công nghệ cho phép hệ thống thực hiện các bài kiểm tra trí nhớ trực tuyến và phân tích dữ liệu thu thập được một cách tự động và chính xác, từ đó đưa ra các đánh giá sơ bộ nhanh chóng. Hệ thống này cũng sẽ tích hợp với các chuyên gia y tế, cho phép họ trực tiếp đánh giá và theo dõi tình trạng sức khỏe của người cao tuổi dựa trên dữ liệu được thu thập.

Giải pháp của tôi là thiết kế và phát triển một ứng dụng có khả năng thực hiện các bài kiểm tra trí nhớ trực tuyến, thu thập dữ liệu, và cung cấp kết quả sơ bộ nhanh chóng, đồng thời tạo điều kiện cho các chuyên gia y tế tiếp cận dữ liệu này để đưa ra những đánh giá chuyên sâu hơn. Ứng dụng này sẽ giúp tiết kiệm thời gian, chi phí cho người dùng và nâng cao khả năng tiếp cận dịch vụ y tế cho người cao tuổi.

Với giải pháp được đưa ra tôi mong rằng mình có thể hoàn thiện một giải pháp toàn diện từ thiết kế đến triển khai một ứng dụng đánh giá tình trạng sa sút trí tuệ từ xa. Kết quả đạt được là xây dựng một ứng dụng hoạt động hiệu quả, đáp ứng đúng nhu cầu của người cao tuổi và giúp cải thiện chất lượng cuộc sống của họ.

## Bố cục bài tập lớn

Phần còn lại của báo cáo Bài tập lớn này được tổ chức như sau.

Về tổng quan chương 2 của bài tập lớn tập trung vào việc khảo sát và phân tích yêu cầu để xây dựng ứng dụng đánh giá sa sút trí tuệ từ xa cho người cao tuổi. Nội dung chương này bao gồm hai phần chính: (i) Khảo sát hiện trạng và (ii) Tổng quan chức năng. Trong phần này, tôi đã tiến hành phân tích, so sánh và đánh giá chi tiết ưu nhược điểm của các sản phẩm hiện có, qua đó xác định những tính năng quan trọng cần phát triển trong ứng dụng của mình. Ứng dụng sẽ có các chức năng chính như: thực hiện các bài kiểm tra trí nhớ trực tuyến, thu thập và phân tích dữ liệu tự động, cung cấp kết quả đánh giá sơ bộ, và cho phép các chuyên gia y tế trực tiếp tham gia đánh giá và theo dõi tình trạng trí nhớ của người cao tuổi.

Sau khi đã có thể xác định được hết những chức năng cần thiết thì bước tiếp theo cần làm là tìm hiểu và lựa chọn những công nghệ cần thiết để sử dụng trong dự án. Và đây cũng là nội dung chính mà chương 3 đề cập tới. Một số công nghệ được áp dụng vào đề tài ở Chương 3 có đề cập đến như back-end Springboot, font-end Angular, database MySql. Dựa vào nền tảng các công nghệ trên để có thể lựa chọn những kiến trúc hệ thống phù hợp để phát triển và triển khai ứng dụng ở chương số 4.

Với những công nghệ đã đề cập ở trên thì tôi đã lựa chọn được mô hình kiến trúc phù hợp cho đề tài của mình. Với Font-end thì tôi sử dụng MVVM( Model-view-view/model). Với springboot thì kiến trúc được lựa chọn là Microservices do ưu điểm rất dễ dàng nâng cấp và có thể hoạt động khi mà một service nào đó bị trục trặc mà không ảnh hưởng đến toàn bộ hệ thống. Sau khi chọn được kiến trúc thiết kế phù hợp sẽ tiến hành thiết kế giao diện và xây dựng chức năng của ứng dụng. Kết chương là quá trình kiểm thử được thực thi một cách chi tiết để ứng dụng có thể hoạt động mượt mà nhất có thể.

Sau khi hoàn thành quá trình xây dựng và phát triển ứng dụng, chương 5 sẽ đóng vai trò quan trọng trong việc trình bày những thành quả mà tôi đã đạt được và cảm thấy tâm đắc nhất trong toàn bộ dự án. Đặc biệt, chương này sẽ đi sâu vào việc mô tả cách thiết kế ứng dụng sao cho thân thiện với người cao tuổi, đảm bảo rằng giao diện và các chức năng của ứng dụng dễ sử dụng và phù hợp với đối tượng người dùng đặc biệt này. Ngoài ra, chương này cũng sẽ tập trung vào việc quản lý dữ liệu người dùng, nhấn mạnh tầm quan trọng của bảo mật thông tin cá nhân và các biện pháp bảo vệ dữ liệu mà tôi đã triển khai để đảm bảo sự an toàn và tin cậy cho người dùng. Chương 6 tiếp theo sẽ là phần tổng kết, hệ thống lại toàn bộ quá trình nghiên cứu và phát triển ứng dụng, đồng thời trình bày những ý tưởng mới và tiềm năng cho việc mở rộng và nâng cấp ứng dụng trong tương lai. Tôi sẽ đề xuất các hướng phát triển mới nhằm đáp ứng nhu cầu ngày càng đa dạng của xã hội, giúp ứng dụng trở nên hoàn thiện hơn và phục vụ tốt hơn cho cộng đồng.

# ­­Khảo sát và phân tích yêu cầu

## Khảo sát hiện trạng

Trong bối cảnh thực tế tại Việt Nam, việc khảo sát hiện trạng cho thấy rằng chưa có hệ thống chẩn đoán từ xa nào tương tự được phát triển và triển khai để đánh giá sa sút trí tuệ ở người cao tuổi. Điều này tạo ra một khoảng trống lớn trong việc cung cấp các dịch vụ chăm sóc sức khỏe cho nhóm đối tượng này, đặc biệt là những người sống ở vùng nông thôn hoặc vùng sâu vùng xa, nơi mà việc tiếp cận các cơ sở y tế chuyên sâu còn gặp nhiều khó khăn.

Khảo sát người dùng/khách hàng:

Người cao tuổi và gia đình họ ở Việt Nam thường gặp khó khăn trong việc tiếp cận các dịch vụ chăm sóc sức khỏe chuyên sâu, đặc biệt là trong việc đánh giá các vấn đề về trí nhớ. Một số người có thể không có điều kiện hoặc khả năng di chuyển đến các cơ sở y tế lớn ở thành phố để được kiểm tra. Bác sĩ tại các cơ sở y tế địa phương cũng gặp khó khăn trong việc chẩn đoán và theo dõi tình trạng của bệnh nhân một cách liên tục và chính xác do thiếu công cụ hỗ trợ và chuyên môn sâu.

Khảo sát các hệ thống đã có:

Hiện tại ở Việt Nam, chưa có hệ thống nào được thiết kế đặc biệt để hỗ trợ chẩn đoán từ xa về sa sút trí tuệ cho người cao tuổi. Một số cơ sở y tế lớn có triển khai các dịch vụ kiểm tra nhận thức, nhưng việc này vẫn yêu cầu người bệnh phải trực tiếp đến khám tại bệnh viện. Việc thiếu một hệ thống chẩn đoán từ xa làm giảm khả năng tiếp cận dịch vụ chăm sóc y tế cho nhiều người cao tuổi, đặc biệt là ở các vùng xa xôi.

Khảo sát các ứng dụng tương tự:

Trong khi các ứng dụng như Lumosity hay CogniFit đã có mặt trên thị trường quốc tế, tại Việt Nam, việc sử dụng các ứng dụng này còn rất hạn chế do rào cản ngôn ngữ, chi phí, và khả năng tiếp cận công nghệ. Hơn nữa, những ứng dụng này không được thiết kế đặc thù cho thị trường Việt Nam và không hỗ trợ chẩn đoán từ xa với sự tham gia của các chuyên gia y tế trong nước.

Yêu cầu phần mềm quan trọng cần phát triển:

Xây dựng hệ thống chẩn đoán từ xa: Phát triển một hệ thống có khả năng hỗ trợ người cao tuổi và gia đình của họ thực hiện các bài kiểm tra trí nhớ và nhận thức từ xa, với sự tham gia và đánh giá trực tiếp từ các chuyên gia y tế.

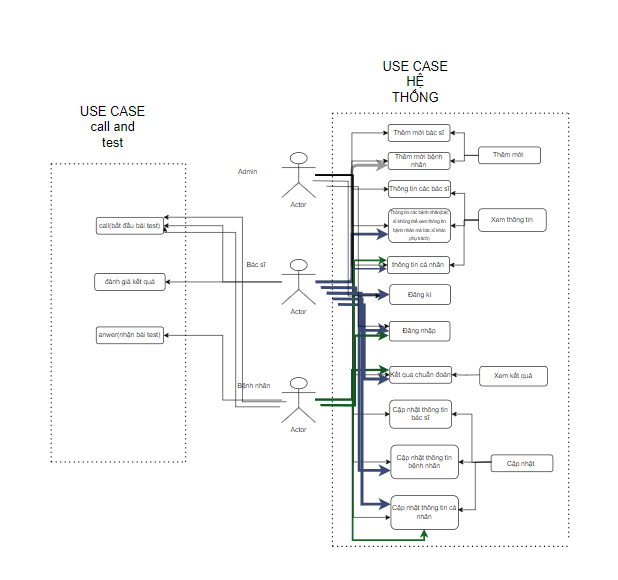
Tích hợp ngôn ngữ và văn hóa Việt Nam: Hệ thống cần được thiết kế phù hợp với ngôn ngữ, văn hóa và đặc điểm dân số của Việt Nam, đảm bảo dễ sử dụng và tiếp cận cho người cao tuổi.

Khả năng tiếp cận: Hệ thống cần hoạt động tốt trên các thiết bị phổ biến, dễ dàng truy cập qua internet và không đòi hỏi kỹ năng sử dụng công nghệ cao.

Phần khảo sát này cho thấy rõ nhu cầu cấp thiết về việc phát triển một hệ thống chẩn đoán từ xa tại Việt Nam, góp phần cải thiện chất lượng chăm sóc sức khỏe cho người cao tuổi trong bối cảnh hiện tại.

## Tổng quan chức năng

### Biểu đồ use case tổng quan



Hình 1:Use case tổng quan của đề tài

* Tác nhân tham gia:
* Admin: Là người quản lý hệ thống, có quyền cao nhết thực hiện các thao tác quản lý dữ liệu liên quan đến bác sĩ và bệnh nhân.
* Bác sĩ: Là người dùng chính của hệ thống, có quyền truy cập, xem thông tin bệnh nhân, thực hiện các bài test chẩn đoán, và đánh giá kết quả.
* Bệnh nhân: Là người nhận các bài test và tham gia vào quá trình chẩn đoán.
* Vai trò của từng tác nhân:
* Admin: Thực hiện các chức năng quản lý như thêm mới bác sĩ, thêm mới bệnh nhân, và cập nhật thông tin của họ trong hệ thống.
* Bác sĩ: Thêm mới bệnh nhân và thực hiện các chức năng liên quan đến việc chẩn đoán và đánh giá kết quả, bao gồm bắt đầu bài test và đánh giá kết quả chẩn đoán.
* Bệnh nhân: Tham gia vào các bài test do bác sĩ chỉ định và nhận kết quả chẩn đoán.
* Mô tả các use case chính:
* Thêm mới bác sĩ: Admin có thể thêm thông tin bác sĩ mới vào hệ thống.
* Thêm mới bệnh nhân: Admin có thể thêm thông tin bệnh nhân mới vào hệ thống.
* Thông tin các bác sĩ: Bác sĩ và admin có thể xem và quản lý thông tin bác sĩ.
* Cập nhật thông tin cá nhân: Admin, bác sĩ có thể cập nhật thông tin cá nhân của mình.
* Đăng ký và Đăng nhập: Bác sĩ và bệnh nhân có thể đăng ký và đăng nhập vào hệ thống.
* Bắt đầu bài test (Call): Bác sĩ có thể bắt đầu một bài test chẩn đoán cho bệnh nhân.
* Nhận bài test (Answer): Bệnh nhân nhận và thực hiện bài test.
* Đánh giá kết quả: Bác sĩ đánh giá kết quả của bài test để đưa ra chẩn đoán.

### Biểu đồ use case phân

2.2.2.1 Use case phân rã chức năng thêm mới bác sĩ(bệnh nhân)

1. Actor có quyền (admin, bác sĩ) đăng nhập vào hệ thống.
2. Trên màn hình quản lý người dùng chọn nút thêm mới để mở form đăng kí người dùng mới.
3. Actor nhập thông tin cá nhân cần thiết vào hệ thống rồi ấn nút thêm.
4. Khi hoàn thành hệ thống sẽ gửi yêu cầu thêm mới lên server.
5. Nếu thành công sẽ có thông báo thành công, nếu thất bại sẽ hiện thông báo thất bại từ server.

2.2.2.2 Use case phân rã chức năng cập nhật thông tin bệnh nhân

1. Actor có quyền (admin, bác sĩ) đăng nhập vào hệ thống.
2. Trên màn hình quản lý người dùng chọn nút chỉnh sửa trên thanh thao tác của mỗi bệnh nhân để mở form cập nhật người dùng.
3. Actor nhập thông tin cá nhân cần thiết vào hệ thống rồi ấn nút lưu.
4. Khi hoàn thành hệ thống sẽ gửi yêu cầu cập nhật lên server.
5. Nếu thành công sẽ có thông báo thành công, nếu thất bại sẽ hiện thông báo thất bại từ server.

2.2.2.3 Use case phân rã chức năng cập nhật thông tin cá nhân

1. Actor có quyền (admin, bác sĩ) đăng nhập vào hệ thống.
2. Trên màn hình menu chọn nút chỉnh thông tin cá nhân để mở form cập nhật thông tin.
3. Actor nhập thông tin cá nhân cần thiết vào hệ thống.
4. Khi hoàn thành hệ thống sẽ gửi yêu cầu cập nhật lên server.
5. Nếu thành công sẽ có thông báo thành công, nếu thất bại sẽ hiện thông báo thất bại từ server.

2.2.2.4 Use case phân rã chức năng Đăng nhập

1. Actor đăng nhập vào hệ thống.
2. Trên màn hình đăng nhâp actor nhập thông tin tên đăng nhập và mật khẩu.
3. Actor nhập thông tin cá nhân xong sẽ ấn vào nút đăng nhập vào hệ thống.
4. Khi hoàn thành hệ thống sẽ gửi yêu cầu đăng nhâp lên server.
5. Nếu thành công sẽ chuyển màn hình đến các trang tương ứng của hệ thống, nếu thất bại sẽ hiện thông báo thất bại từ server.

2.2.2.5 Use case phân rã chức năng bắt đầu bài kiểm tra(Call)

1. Actor có quyền (bác sĩ) đăng nhập vào hệ thống.
2. Trên màn hình quản lý người dùng chọn nút Call trên thanh thao tác của mỗi bệnh nhân đang online để mở một popup chọn bài kiểm tra tương ứng với 3 mức(dễ, trung bình, khó).
3. Bác sĩ chọn bài test tương ứng với bệnh nhân (từ dễ-trung bình-khó).
4. Khi hoàn thành bài test bác sĩ sẽ ấn nút kết thúc bài kiểm tra, hệ thống mở ra một hộp modle để bác sẽ có thể ghi lại tình trạng bệnh nhân. Khi xong chọn nút kết thúc để hoàn thành quá trình đánh giá.
5. Khi hoàn thành hệ thống sẽ gửi yêu cầu lên server.
6. Nếu thành công sẽ có thông báo thành công, nếu thất bại sẽ hiện thông báo thất bại từ server.

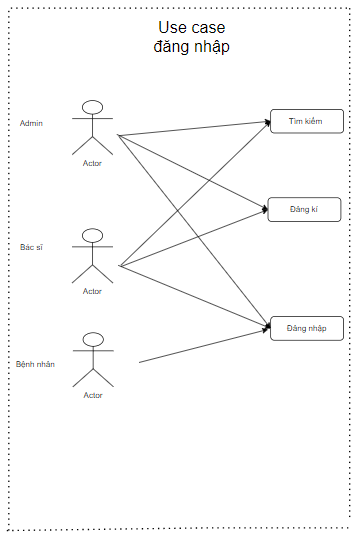
2.2.2.6 Use case phân rã chức năng nhận bài test (Answer)

1. Actor có quyền (bệnh nhân) đăng nhập vào hệ thống. Sau khi đăng nhập thành công sẽ được chuyển đến trang chọn bài kiểm tra với 3 mức độ tương ứng(dễ-trung bình-khó).
2. Bệnh nhân chọn bài kiểm tra tương ứng để làm theo chỉ định của bác sĩ.

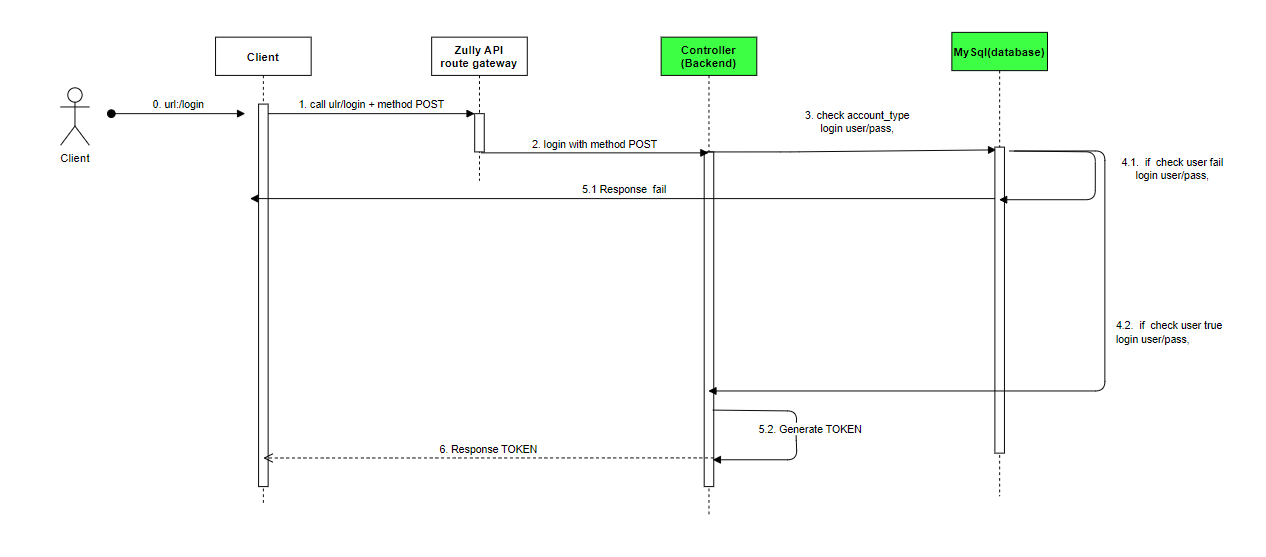
3. Sau khi kết thúc sẽ được chuyển về trang chọn bài kiểm tra

## Đặc tả chức năng

### Đặc tả use case login



Hình 2:Use case đặc tả chức năng login



Hình 3:Biểu đồ tuần tự chức năng đăng nhập

Mô tả:

|  |  |
| --- | --- |
| Tên use case | Login |
| Mô tả sơ bộ | Cho phép tác nhân đăng nhập vào hệ thống |
| Tác nhân | Admin, bác sĩ, bệnh nhân |
| Tiền điều kiện | Tác nhân chưa đăng nhập vào hệ thống |
| Hậu điều kiện | Tác nhân đã đăng nhập vào hệ thống |

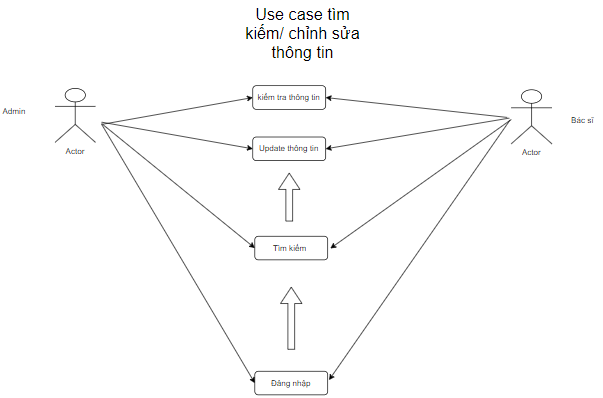
bảng 1: Mô tả use case đăng nhập

* ***Luồng sự kiện login***

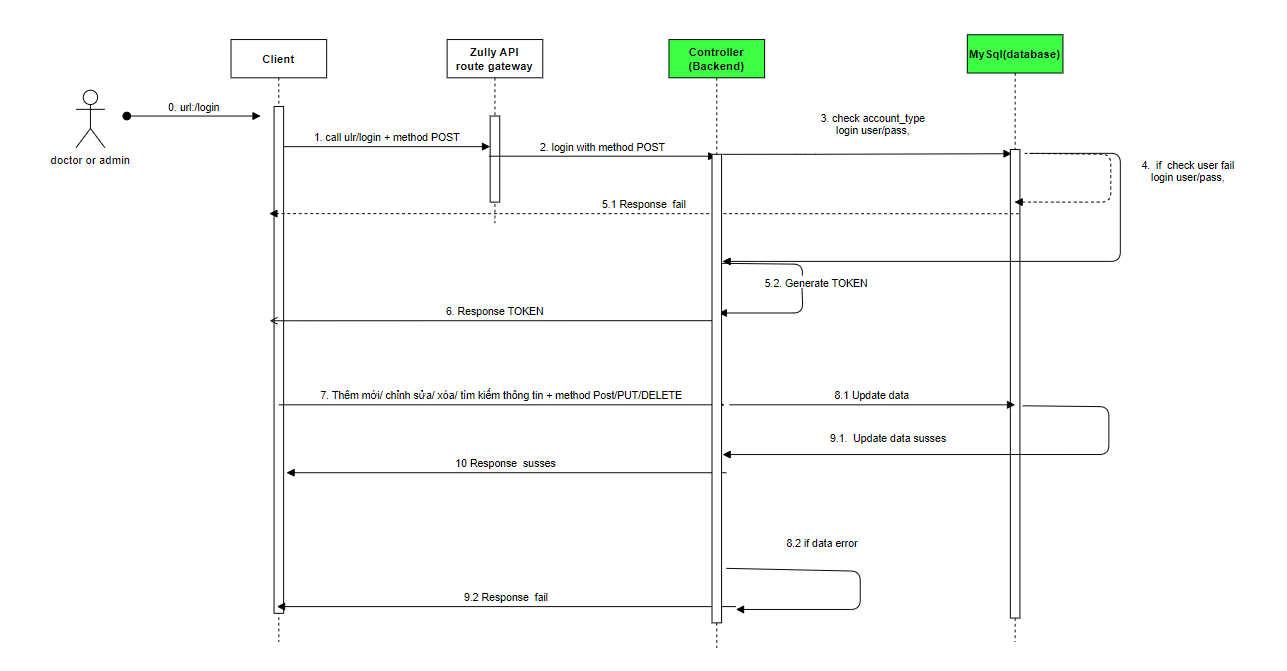
|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Nhập tài khoản và mật khẩu vào form đăng nhập |  |  |
| 2 |  | Hệ thống gửi yêu cầu đăng nhập kèm theo thông tin đến server |  |
| 3 |  |  | Kiểm tra dữ liệu được gửi lên. Nếu có lỗi sẽ dừng quá trình đăng nhập và gửi thông báo về front-end. |
| 4 |  | Hiển thị lỗi cho người dùng. Tiếp tục bước thứ nhất |  |
| 5 |  |  | Dữ liệu hợp lệ sẽ được gán cho token để lần request tiếp theo không cần đăng nhập lại. Sao đó gửi cho fornt-end. |
| 6 |  | Font-end hiển thị thành công và đưa người dùng đến trang tiếp theo. |  |
| 7 | Người dùng đăng nhập thành công vào được hệ thống |  |  |

bảng 2: Luồng use case đăng nhập

### Đặc tả use case chỉnh sửa/tìm kiếm thông tin



Hình 4:Đặc tả chức năng tìm kiếm/ chỉnh sửa thông tin.



Hình 5: Biểu đồ tuần tự chức năng chỉnh sửa và tìm kiếm thông tin.

|  |  |
| --- | --- |
| Tên use case | Chỉnh sửa và tìm kiếm thông tin. |
| Mô tả sơ bộ | Cho phép tác nhân tìm kiếm thông tin trong hệ thống, chính sửa, thêm, xóa các người dùng của hệ thống. |
| Tác nhân | Admin, bác sĩ. |
| Tiền điều kiện | Tác nhân đăng nhập vào hệ thống. |
| Hậu điều kiện | Tác nhân thấy được danh sách tìm kiếm và chỉnh sửa được thông tin. |

bảng 3: Mô tả use case chỉnh sửa và tìm kiếm thông tin.

* ***Luồng tìm kiếm thông tin***

|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Người dùng có quyền tìm kiếm tại trang quản lý thông tin người dùng chọn nhập các thông tin cần thiết cho quá trình tìm kiếm |  |  |
| 2 |  | Hệ thống sẽ kiểm tra định dạng của dữ liệu nếu có lỗi sẽ báo lỗi về cho người dùng. |  |
| 3 | Người dùng chọn nút tìm kiếm để bắt đầu tìm kiếm thông tin |  |  |
| 4 |  | Hệ thống sẽ gửi những thông tin đầu vào đến server để bắt đầu quá trình |  |
| 5 |  |  | Server nhận thông tin từ fornt-end và tìm kiếm trong cơ sở dữ liệu. Sau đó server sẽ trả kết quả về cho font-end |
| 6 |  | Font-end nhận dữ liệu từ API rồi hiển thị ra giao diện người dùng |  |
| 7 | Người dùng sẽ thấy kết quả tren giao diện hiện tại |  |  |

bảng 4:Luồng tìm kiếm thông tin

* ***Luồng thêm mới bệnh nhân***

|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Người dùng có quyền thêm mới tại trang quản lý thông tin người dùng chọn nhập các thông tin cần thiết cho quá trình tìm kiếm |  |  |
| 2 |  | Hệ thống sẽ kiểm tra định dạng của dữ liệu nếu có lỗi sẽ báo lỗi về cho người dùng. |  |
| 3 | Người dùng chọn nút thêm mới để bắt đầu thêm mới người dùng |  |  |
| 4 |  | Hệ thống sẽ gửi những thông tin đầu vào đến server để bắt đầu quá trình |  |
| 5 |  |  | Server nhận thông tin từ fornt-end và tìm kiếm trong cơ sở dữ liệu nếu người dùng chưa tồn tại hoặc đã xóa thì sẽ thêm mới vào cơ sở dữ liệu. Sau đó server sẽ trả kết quả về cho font-end |
| 6 |  | Font-end nhận dữ liệu từ API rồi hiển thị ra giao diện người dùng |  |
| 7 | Người dùng sẽ thấy kết quả trên giao diện hiện tại |  |  |

bảng 5: Luồng thêm mới bệnh nhân

* ***Luồng chỉnh sửa thông tin bệnh nhân***

|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Người dùng có quyền chỉnh sửa tại trang quản lý thông tin người dùng chọn nút chỉnh sửa tại thao tác của từng bệnh nhân |  |  |
| 2 |  | Hệ thống mở ra form chỉnh sửa để người dùng nhập thông tin cần chỉnh sửa |  |
| 3 | Nhập các thông tin cần chỉnh sửa vào form rồi chọn lưu |  |  |
| 4 |  | Hệ thống sẽ kiểm tra định dạng của dữ liệu nếu có lỗi sẽ báo lỗi về cho người dùng. |  |
| 5 | Người dùng chọn nút lưu để bắt đầu thay đổi thông tin |  |  |
| 6 |  | Hệ thống sẽ gửi những thông tin đầu vào đến server để bắt đầu quá trình |  |
| 7 |  |  | Server nhận thông tin từ fornt-end và tìm kiếm sau đó chỉnh sửa trong cơ sở dữ liệu. Sau đó server sẽ trả kết quả về cho font-end |
| 8 |  | Font-end nhận dữ liệu từ API rồi hiển thị ra giao diện người dùng |  |
| 9 | Người dùng sẽ thấy kết quả trên giao diện hiện tại |  |  |

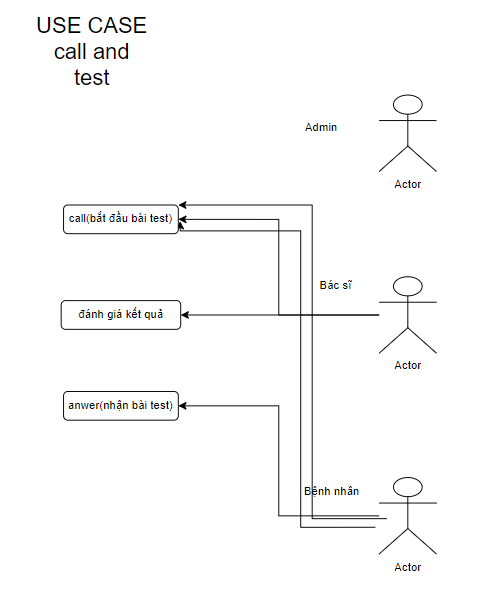
bảng 6: Luồng chỉnh sửa thông tin bệnh nhân

* ***Luồng xóa người dùng***

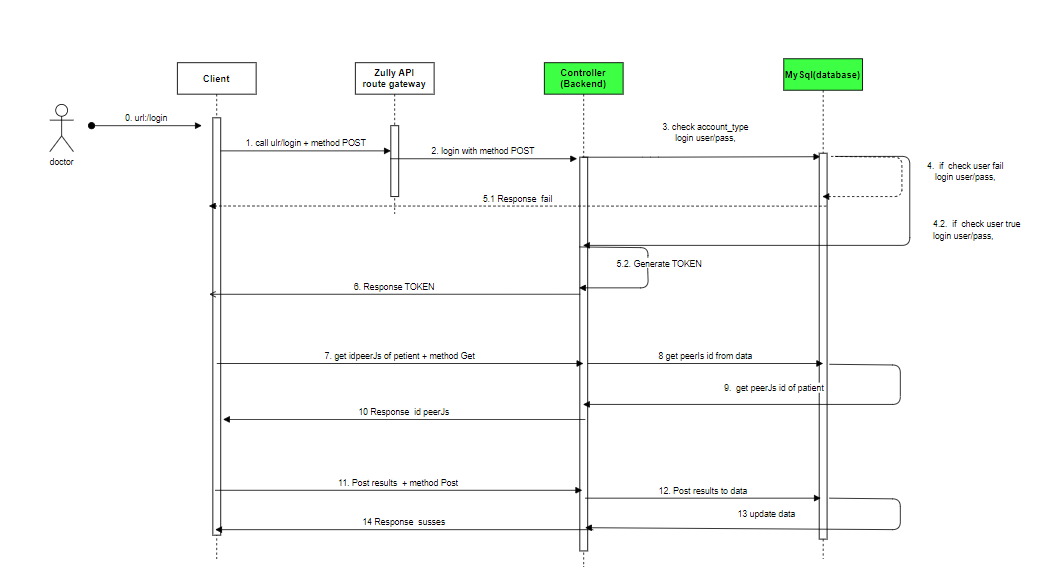
|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Người dùng có quyền tìm xóa bệnh nhân tại trang quản lý thông tin người dùng chọn thao tác xóa tại phần thao tác của muỗi người dùng |  |  |
| 2 |  | Hệ thống sẽ kiểm tra dữ liệu rồi hiển thị popup xác nhận để xác nhận có chắc chắn xóa hay không |  |
| 3 | Người dùng chọn nút đồng ý |  |  |
| 4 |  | Hệ thống sẽ gửi yêu cầu xóa đến server để bắt đầu quá trình xóa thông tin |  |
| 5 |  |  | Server nhận thông tin từ fornt-end và tìm kiếm trong cơ sở dữ liệu nếu có bản ghi sẽ tiến hành thay xóa dữ liệu. Sau đó server sẽ trả kết quả về cho font-end |
| 6 |  | Font-end nhận dữ liệu từ API rồi hiển thị ra giao diện người dùng |  |
| 7 | Người dùng sẽ thấy kết quả thành công trên giao diện hiện tại |  |  |

bảng 7: Luồng xóa người dùng

### Đặc tả use case đánh giá kết quả của bệnh nhân.



Hình 6: Đặc tả chức năng đánh giá kết quả của bệnh nhân



Hình 7: Biểu đồ tuần tự chức năng đánh giá kết quả của bệnh nhân

|  |  |
| --- | --- |
| Tên use case | Đánh giá kết quả của bệnh nhân |
| Mô tả sơ bộ | Cho phép tác nhân nhân tham gia vào quá trình đánh giá kết quả của bệnh nhân cụ thể |
| Tác nhân | bác sĩ |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống và bệnh nhân online |
| Hậu điều kiện | Tác nhân kết thúc quá trình đánh giá trở về trang quản lý bệnh nhân. |

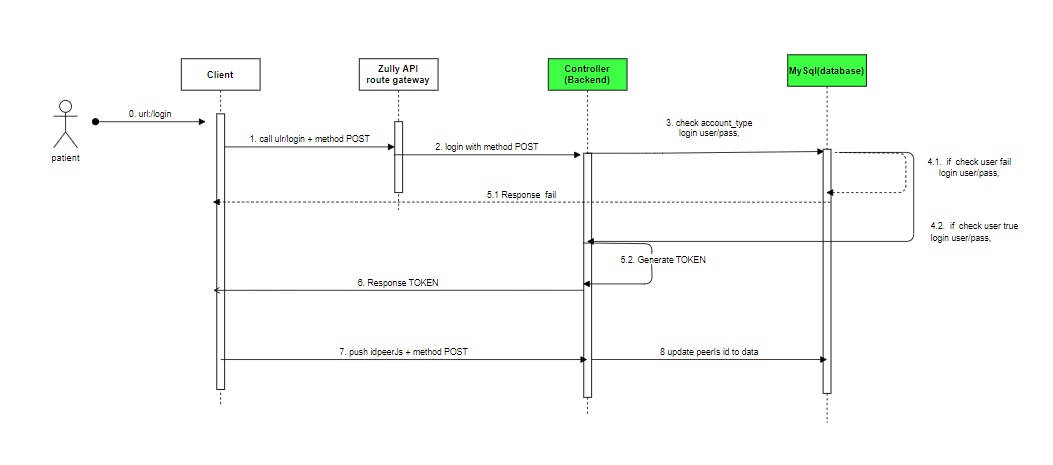
bảng 8:Mô tả use case đánh giá kết quả của bệnh nhân

* ***Luồng đánh giá kết quả kiểm tra của bệnh nhân.***

|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Bác sĩ sau khi đăng nhập thành công sẽ được hệ thống đưa đến trang quản lý bệnh nhân. Sau khi liên bệnh nhân đã online thì bác sĩ sẽ chọn nút call để bắt đầu quá trình đánh giá bệnh nhân tương ứng. |  |  |
| 2 |  | Hệ thống sẽ hiển thị ra 3 sự lựa chọn tương ứng với 3 bài kiểm tra. |  |
| 3 | Người dùng chọn bài kiểm tra tương ứng với bệnh nhân. |  |  |
| 4 |  | Hệ thống sẽ bắt đầu bài kiểm tra |  |
| 5 | Sau khi kết thúc bài kiểm tra bác sĩ sẽ chọn nút kết thúc |  |  |
| 6 |  | Font-end mở hộp thoại đánh giá để kết thúc quá trình đánh giá |  |
| 7 | Người dùng sẽ thấy kết quả mà hệ thống tự đánh giá rồi bác sĩ sẽ thực hiện đánh giá với bệnh nhân. Sau đó nhấn nút lưu để kết thúc |  |  |
| 8 |  | Hệ thống sẽ gửi dữ liệu đánh giá đến server để lưu trữ |  |
| 9 |  |  | Server nhận thông tin từ fornt-end và tìm kiếm trong cơ sở dữ liệu nếu có bản ghi sẽ tiến hành lưu dữ liệu. Sau đó server sẽ trả kết quả về cho font-end |
| 10 |  | Font-end nhận dữ liệu từ API rồi hiển thị ra giao diện người dùng |  |
| 11 | Người dùng sẽ thấy kết quả thành công trên giao diện hiện tại |  |  |

bảng 9: Luồng đánh giá kết quả kiểm tra của bệnh nhân.

### Đặc tả use case nhận bài kiểm tra của bệnh nhân



Hình 8: Biểu đồ tuần tự chức năng nhận bài kiểm tra của bệnh nhân

|  |  |
| --- | --- |
| Tên use case | nhận bài kiểm tra của bệnh nhân |
| Mô tả sơ bộ | Cho phép tác nhân làm bài kiểm tra đánh giá trên hệ thống |
| Tác nhân | bệnh nhân |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống |
| Hậu điều kiện | Tác nhân đã hoàn thành đánh giá và được định hướng về trang chọn bài kiểm tra. |

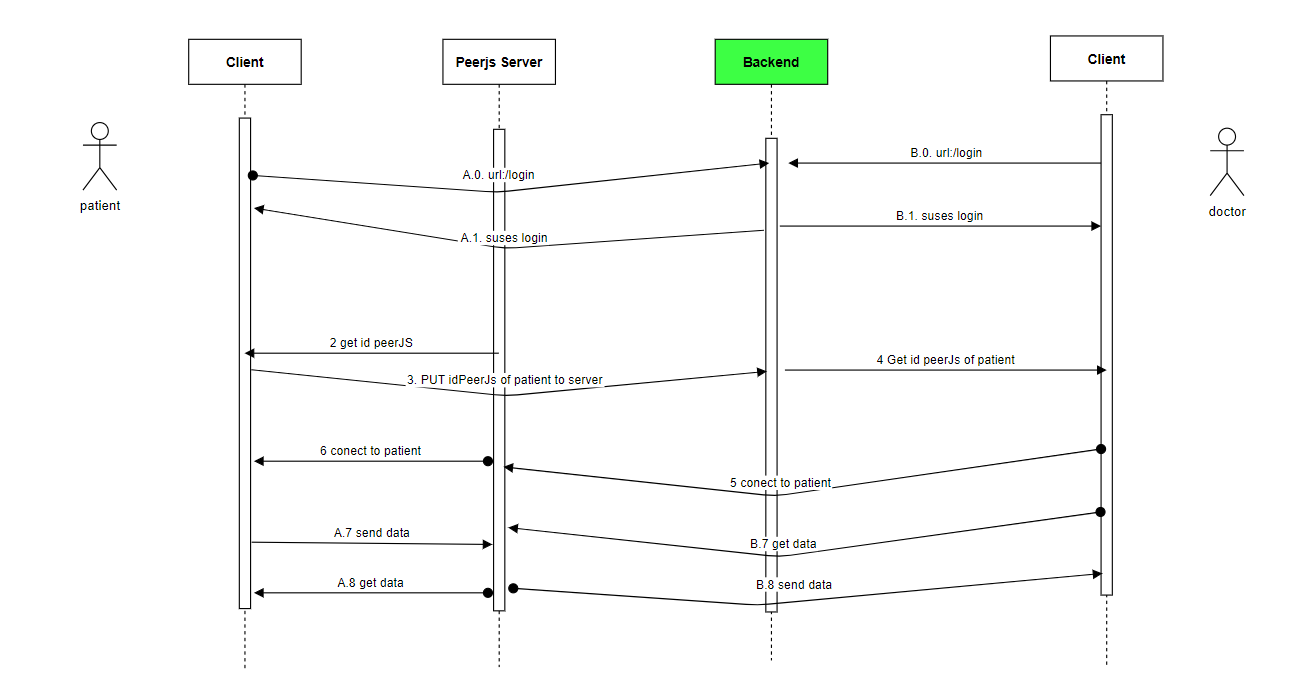
bảng 10: Mô tả use case nhận bài kiểm tra của bệnh nhân

* ***Luồng nhận bài kiểm tra của bệnh nhân.***

|  |  |  |  |
| --- | --- | --- | --- |
| Stt | Hành động tác nhân | Hệ thống thực hiện | |
| Font-end | Back-end |
| 1 | Bệnh nhân sau khi đăng nhập thành công sẽ được hệ thống đưa đến trang chọn bài kiểm tra. Sau khi liên bệnh nhân đã chọn bài kiểm tra tương ứng để bắt đầu quá trình đánh giá bệnh nhân tương ứng. |  |  |
| 2 |  | Hệ thống sẽ hiển thị ra phần kiểm tra tương ứng |  |
| 3 | Sẽ tiến hành đánh giá |  |  |
| 4 |  | Khi kết thúc quá trình đánh giá thì bệnh nhân sẽ được điều hướng tời trang chọn bài kiểm tra |  |

bảng 11: Luồng nhận bài kiểm tra của bệnh nhân.

### Đặc tả use case kết nối tương tác giữa bệnh nhân và bác sĩ



Hình 9: Biểu đồ tuần tự chức năng kết nối tương tác giữa bệnh nhân và bác sĩ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tên use case | | | kết nối tương tác giữa bệnh nhân và bác sĩ | |
| Mô tả sơ bộ | | | Cho phép tác nhân trao đổi thông tin với nhau nhằm phục vụ việc tương tác giữa bệnh nhân và bác sĩ | |
| Tác nhân | | | bác sĩ, bệnh nhân | |
| Tiền điều kiện | | | Tác nhân chuẩn bị làm bài đánh giá | |
| Hậu điều kiện | | | Tác nhân đã kết thúc bài đánh giá | |
| * Luồng kết nối tương tác giữa bệnh nhân và bác sĩ | | | | |
| Stt | Hành động tác nhân | Hệ thống thực hiện | | |
| Font-end | | Back-end |
| 1 | Bác sĩ và bệnh nhân đăng nhập vào hệ thống. |  | |  |
| 2 |  | Gửi yêu cầu đăng nhập tới server. | |  |
| 3 |  |  | | Kiểm tra thông tin và trả kết quả về font-end. |
| 4 |  | Kết quả thành công sẽ đưa bệnh nhân điều hướng đến trang chọn bài kiểm tra còn bác sĩ đến trang quản lý. | |  |
| 5 | Bệnh nhân khi bắt đầu bài kiểm |  | |  |
| 6 |  | Hệ thống nhận idPeerJs của bệnh nhân và gửi đến server | |  |
| 7 |  |  | | Server lưu dữ liệu idPeerJs của bênh nhân mới. |
| 8 | Bác sĩ bắt đầu quá trình đánh giá | Hệ thống lấy id Peerjs của bệnh nhân rồi kết nối tới | |  |
| 9 | Người bệnh thao tác trong quá trình kiểm tra | Hệ thống sẽ gửi thông tin tương ứng đến bác sĩ để đồng bộ hóa với thao tác của bệnh nhân. | |  |
| 10 | Bác sĩ thao tác trong quá trình kiểm tra | Hệ thống sẽ gửi thông tin tương ứng đến bệnh nhân để đồng bộ hóa với thao tác của bác sĩ. | |  |

bảng 12: Mô tả use case kết nối tương tác giữa bác sĩ và bệnh nhân

## Yêu cầu phi chức năng

* Yêu cầu phi chức năng login:

- Yêu cầu xác thực người gửi request, sử dụng JWT.

* Yêu cầu phi chức năng của hệ thống:

- Hiệu suất: Ứng dụng phải hoạt động nhanh chóng và mượt mà.

- Khả năng sử dụng: Ứng dụng phải dễ sử dụng và trực quan.

- Bảo mật: Ứng dụng phải bảo mật thông tin cá nhân của người dùng.

* Yêu cầu phi chức năng của backend:

- Khả năng mở rộng: Backend phải có khả năng mở rộng để đáp ứng nhu cầu

tăng trưởng của hệ thống.

- Hiệu suất: Backend phải xử lý các yêu cầu nhanh chóng và hiệu quả.

- Bảo mật: Backend phải bảo mật dữ liệu của hệ thống.

* Yêu cầu phi chức năng của cơ sở dữ liệu:

- Hiệu suất: Cơ sở dữ liệu phải truy xuất và thao tác dữ liệu nhanh chóng.

- Khả năng mở rộng: Cơ sở dữ liệu phải có khả năng mở rộng để lưu trữ

lượng dữ liệu lớn.

* Yêu cầu phi chức năng của giao diện quản trị:

- Dễ sử dụng: Giao diện quản trị phải dễ sử dụng và trực quan.

- Bảo mật: Giao diện quản trị phải bảo mật thông tin của hệ thống

# Công nghệ sử dụng

**3.1 Ngôn ngữ java**

**1. Java là gì ?**

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng

trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di

động.

Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem

năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sảnphẩm gia dụng, và có tên là Oak.Java được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun

MicroSystem.

Java được tạo ra với tiêu chí “Viết (code) một lần, thực thi khắp nơi” (Write Once,Run Anywhere – WORA). Chương trình phần mềm viết bằng Java có thể chạytrên mọi nền tảng (platform) khác nhau thông qua một môi trường thực thi vớiđiều kiện có môi trường thực thi thích hợp hỗ trợ nền tảng đó.

**2. Đặc điểm của ngôn ngữ lập trình Java.**

❖ Hướng đối tượng hoàn toàn

Trong quá trình tạo ra một ngôn ngữ mới phục vụ cho mục đích chạy được trên nhiều nền tảng, các kỹ sư của Sun MicroSystem muốn tạo ra một ngôn ngữ dễ học và quen thuộc với đa số người lập trình. Vì vậy họ đã sử dụng lại các cú pháp của C và C++.

Tuy nhiên, trong Java thao tác với con trỏ bị lược bỏ nhằm đảo bảo tính an toàn và dễ sử dụng hơn. Các thao tác overload, goto hay các cấu trúc như struct và union cũng được loại bỏ khỏi Java.

❖ Độc lập phần cứng và hệ điều hành

Một chương trình viết bằng ngôn ngữ Java có thể chạy tốt ở nhiều môi trường

khác nhau. Gọi là khả năng “cross-platform”. Khả năng độc lập phần cứng và hệ

điều hành được thể hiện ở 2 cấp độ là cấp độ mã nguồn và cấp độ nhị phân.

Ở cấp độ mã nguồn: Kiểu dữ liệu trong Java nhất quán cho tất cả các hệ điều hành và phần cứng khác nhau. Java có riêng một bộ thư viện để hỗ trợ vấn đề này.Chương trình viết bằng ngôn ngữ Java có thể biên dịch trên nhiều loại máy khác nhau mà không gặp lỗi.

Ở cấp độ nhị phân: Một mã biên dịch có thể chạy trên nhiều nền tảng khác nhau mà không cần dịch lại mã nguồn. Tuy nhiên cần có Java Virtual Machine để thông dịch đoạn mã này.

❖ Ngôn ngữ thông dịch

Ngôn ngữ lập trình thường được chia ra làm 2 loại (tùy theo các hiện thực hóa

ngôn ngữ đó) là ngôn ngữ thông dịch và ngôn ngữ biên dịch.

➢ Thông dịch (Interpreter) : Nó dịch từng lệnh rồi chạy từng lệnh, lần sau muốn chạy lại thì phải dịch lại.

➢Biên dịch (Compiler): Code sau khi được biên dịch sẽ tạo ra 1 file thường là .exe, và file .exe này có thể đem sử dụng lại không cần biên dịch nữa. Ngôn ngữ lập trình Java thuộc loại ngôn ngữ thông dịch. Chính xác hơn, Java là loại ngôn ngữ vừa biên dịch vừa thông dịch. Cụ thể như sau

Khi viết mã, hệ thống tạo ra một tệp .java. Khi biên dịch mã nguồn của chương

trình sẽ được biên dịch ra mã byte code. Máy ảo Java (Java Virtual Machine) sẽ thông dịch mã byte code này thành machine code (hay native code) khi nhận

được yêu cầu chạy chương trình.

Ưu điểm : Phương pháp này giúp các đoạn mã viết bằng Java có thể chạy được

trên nhiều nền tảng khác nhau. Với điều kiện là JVM có hỗ trợ chạy trên nền tảng này.

Nhược điểm : Cũng như các ngôn ngữ thông dịch khác, quá trình chạy các đoạn mã Java là chậm hơn các ngôn ngữ biên dịch khác (tuy nhiên vẫn ở trong một mức chấp nhận được).

❖ Cơ chế thu gom rác tự động

Khi tạo ra các đối tượng trong Java, JRE sẽ tự động cấp phát không gian bộ nhớ cho các đối tượng ở trên heap.

Với ngôn ngữ như C C++, bạn sẽ phải yêu cầu hủy vùng nhớ mà bạn đã cấp phát, để tránh việc thất thoát vùng nhớ. Tuy nhiên vì một lý do nào đó, bạn không hủy

một vài vùng nhớ, dẫn đến việc thất thoát và làm giảm hiệu năng chương trình.

Ngôn ngữ lập trình Java hỗ trợ cho bạn điều đó, nghĩa là bạn không phải tự gọi hủy các vùng nhớ. Bộ thu dọn rác của Java sẽ theo vết các tài nguyên đã được cấp. Khi không có tham chiếu nào đến vùng nhớ, bộ thu dọn rác sẽ tiến hành thu hồi vùng nhớ đã được cấp phát.

❖ Đa luồng

Java hỗ trợ lập trình đa tiến trình (multithread) để thực thi các công việc đồng

thời. Đồng thời cũng cung cấp giải pháp đồng bộ giữa các tiến trình (giải pháp sử

dụng priority…).

❖ Tính an toàn và bảo mật

➢ Tính an toàn

Ngôn ngữ lập trình Java yêu cầu chặt chẽ về kiểu dữ liệu.

Dữ liệu phải được khai báo tường minh.

Không sử dụng con trỏ và các phép toán với con trỏ.

Java kiểm soát chặt chẽ việc truy nhập đến mảng, chuỗi. Không cho phép sử dụng các kỹ thuật tràn. Do đó các truy nhập sẽ không vượt quá kích thước của mảng hoặc chuỗi. Quá trình cấp phát và giải phóng bộ nhớ được thực hiện tự động. Cơ chế xử lý lỗi giúp việc xử lý và phục hồi lỗi dễ dàng hơn.

➢ Bảo mật

Java cung cấp một môi trường quản lý chương trình với nhiều mức khác nhau.

Mức 1 : Chỉ có thể truy xuất dữ liệu cũng như phương phức thông qua giao diện mà lớp cung cấp

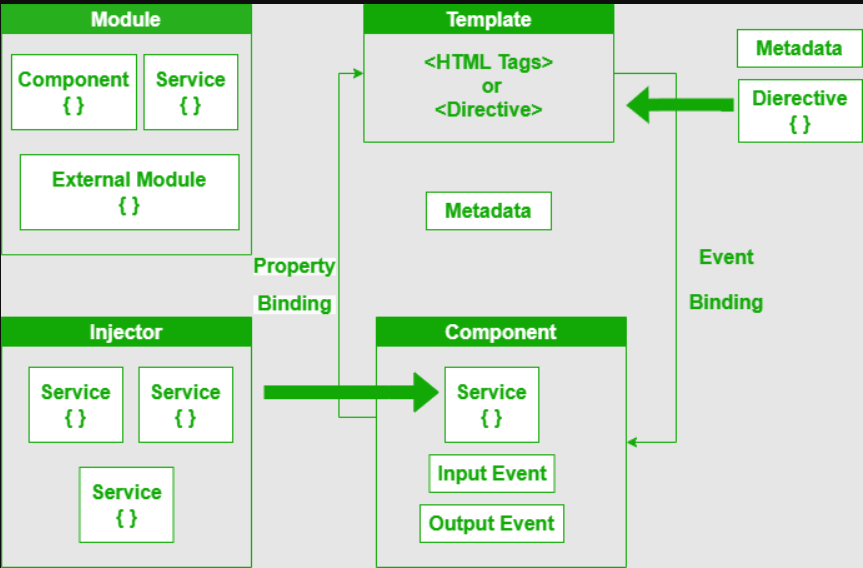
Mức 2 : Trình biên dịch kiểm soát các đoạn mã sao cho tuân thủ các quy tắc của ngôn ngữ lập trình Java trước khi thông dịch.

Mức 3 : Trình thông dịch sẽ kiểm tra mã byte code xem các đoạn mã này có đảmbảo được các quy định, quy tắc trước khi thực thi.

Mức 4: Java kiểm soát việc nạp các lớp vào bộ nhớ để giám sát việc vi phạm giớihạn truy xuất trước khi nạp vào hệ thống

**3.2 Famework angular**

Angular là một JavaScript frameworks được sử dụng để thiết kế giao diện web, được phát triển bởi Google và là một trong những Front-end frameworks mạnh mẽ nhất hiện nay.



Hình 10: Sơ đồ hoạt động của fameWork Angular.

Giải thích một số khái niệm mô tả mô hình hoạt động của fameWork Angular.

***1. Component***

Component: Là một lớp (class) trong Angular, chứa logic và dữ liệu của một phần của giao diện người dùng. Mỗi component có thể bao gồm các phần tử sau:

Template: Định nghĩa giao diện người dùng của component bằng HTML.

Metadata: Cung cấp thông tin về component cho Angular bằng cách sử dụng decorator @Component.

***2. Template***

Template: Là phần HTML của component, nơi định nghĩa cấu trúc và giao diện của component. Template có thể chứa các biểu thức Angular và các directive để hiển thị dữ liệu và xử lý sự kiện.

***3. Metadata***

Metadata: Là thông tin được cung cấp cho Angular về một class, thường sử dụng các decorator như @Component, @Directive, @Injectable, v.v. Metadata giúp Angular hiểu cách xử lý và tương tác với class đó.

***4. Directive***

Directive: Là các lớp để thêm hành vi vào các phần tử DOM. Có ba loại directive:

Component Directive: Các component thực sự là một loại directive với một template.

Structural Directive: Thay đổi cấu trúc của DOM (ví dụ: \*ngIf, \*ngFor).

Attribute Directive: Thay đổi cách mà một phần tử, component hoặc một directive khác xuất hiện hoặc hành xử (ví dụ: ngClass, ngStyle).

***5. Injector***

Injector: Là một phần của cơ chế Dependency Injection (DI) của Angular. Injector chịu trách nhiệm cung cấp các dependency cho component và service.

Service: Là các lớp chứa logic nghiệp vụ, có thể được chia sẻ giữa các component khác nhau thông qua DI.

***6. Property Binding***

• Property Binding: Là cách Angular cho phép bạn truyền dữ liệu từ component tới template. Nó sử dụng dấu ngoặc vuông [] để biểu thị rằng bạn đang liên kết giá trị của một thuộc tính từ component với một phần tử HTML trong template.

***7. Event Binding***

• Event Binding: Là cách Angular cho phép bạn lắng nghe và xử lý các sự kiện từ template. Nó sử dụng dấu ngoặc tròn () để biểu thị rằng bạn đang liên kết một sự kiện từ template với một phương thức trong component.

***8. Interaction between Template and Component***

• Interaction: Template và component tương tác với nhau thông qua property binding và event binding. Template hiển thị dữ liệu từ component thông qua property binding và gửi sự kiện về component thông qua event binding.

## 3.3 Ngôn ngữ Javascript

1. Javascript là gì ?

JavaScript là một ngôn ngữ lập trình website, được tích hợp và nhúng trong

HTML giúp website sống động hơn. JavaScript cho phép kiểm soát các hành vi

của trang web tốt hơn so với khi chỉ sử dụng mỗi HTML.

JavaScript là ngôn ngữ lập trình được hỗ trợ hầu như trên tất cả các trình duyệt

như Firefox, Chrome, … thậm chí các trình duyệt trên thiết bị di động.

2. Ưu và nhược điểm của Javascript.

❖ Ưu điểm

➢ Dễ học.

➢ Dễ phát hiện và sửa lỗi.

➢ Hoạt động trên nhiều trình duyệt, nền tảng.

➢ Giúp website tương tác dễ .

➢ Nhanh và nhẹ hơn các ngôn ngữ lập trình khác.

❖ **Nhược điểm**➢ Dễ bị khai thác.  
➢ Có thể dùng để thực thi mã độc trên máy tính của người dùng.  
➢ Có thể bị triển khai khác nhau tùy từng thiết bị dẫn đến việc không đồng  
nhất.

3. Cách hoạt động của Javascript.JavaScript thường được nhúng trực tiếp vào một trang web hoặc được tham chiếu  
qua file .js riêng. Nó là ngôn ngữ lập trình từ phía khách hàng, tức là script được  
tải về máy của người dùng đang truy cập và được xử lý tại đó, thay vì xử lý trên  
máy chủ rồi mới đưa kết quả tới khách hàng.

## 3.4 Công nghệ peerJs

PeerJS là một thư viện JavaScript mã nguồn mở giúp đơn giản hóa việc xây dựng các ứng dụng web sử dụng WebRTC để truyền tải dữ liệu và media (âm thanh, video) trực tiếp giữa các trình duyệt mà không cần thông qua máy chủ trung gian. WebRTC (Web Real-Time Communication) là một công nghệ cho phép trình duyệt và ứng dụng di động giao tiếp trực tiếp với nhau, mang lại khả năng gọi video, truyền file, và chia sẻ dữ liệu theo thời gian thực.

PeerJS cung cấp một API dễ sử dụng để tạo và quản lý các kết nối WebRTC. Dưới đây là một số tính năng chính của PeerJS:

1. **Đơn giản hóa kết nối**: PeerJS trừu tượng hóa sự phức tạp của việc thiết lập các kết nối WebRTC, cho phép nhà phát triển dễ dàng tạo ra các ứng dụng truyền tải thời gian thực.
2. **Khả năng tương tác**: Với PeerJS, bạn có thể tạo kết nối P2P giữa các trình duyệt khác nhau, hỗ trợ nhiều nền tảng và thiết bị.
3. **Dễ dàng mở rộng**: Thư viện này hỗ trợ nhiều loại kết nối, bao gồm kết nối dữ liệu và kết nối truyền tải media (âm thanh, video), giúp dễ dàng phát triển các ứng dụng từ đơn giản đến phức tạp.
4. **Máy chủ định danh**: PeerJS cung cấp một máy chủ định danh miễn phí để giúp các client tìm thấy nhau trên mạng, nhưng bạn cũng có thể thiết lập máy chủ riêng nếu cần kiểm soát nhiều hơn.
5. **Thân thiện với nhà phát triển**: PeerJS có tài liệu phong phú và nhiều ví dụ giúp các nhà phát triển nhanh chóng bắt đầu và xây dựng ứng dụng của mình.

PeerJS rất hữu ích trong việc phát triển các ứng dụng như hội nghị video, chia sẻ file trực tuyến, hay các trò chơi đa người chơi theo thời gian thực, giúp tận dụng tối đa khả năng của WebRTC mà không cần phải viết nhiều mã phức tạp.

PeerJS là một framework WebRTC được thiết kế để đơn giản hóa việc phát triển các ứng dụng truyền thông thời gian thực (RTC). Bằng cách trừu tượng hóa các logic phức tạp liên quan đến ICE (Interactive Connectivity Establishment) và quá trình signaling (tín hiệu hóa), PeerJS cho phép các nhà phát triển tập trung vào chức năng cốt lõi của ứng dụng mà không cần phải lo lắng về những chi tiết kỹ thuật phức tạp của WebRTC.

## 3.5 Web RTC

WebRTC là một API JavaScript cung cấp các tính năng truyền thông thời gian thực cho các ứng dụng web. Nó cho phép các ứng dụng web chia sẻ âm thanh, video và dữ liệu giữa các trình duyệt mà không cần cài đặt hoặc cấu hình phức tạp.

***Ưu điểm của WebRTC***

* Dễ triển khai: WebRTC tích hợp trực tiếp vào trình duyệt web, loại bỏ nhu cầu sử dụng plugin hoặc phần mềm riêng biệt.
* Giao tiếp thời gian thực: WebRTC hỗ trợ truyền âm thanh, video và dữ liệu tức thì giữa các trình duyệt.
* Bảo mật nâng cao: WebRTC sử dụng mã hóa SRTP để đảm bảo bảo mật giao tiếp.
* Giảm chi phí: Loại bỏ nhu cầu sử dụng dịch vụ bên thứ ba có thể dẫn đến tiết kiệm chi phí đáng kể.

***Danh sách các trình duyệt hỗ trợ WebRTC***

1. Google Chrome

2. Mozilla Firefox

3. Opera

4. Microsoft Edge

## 3.6 WebSocket

WebSocket là một công nghệ cung cấp kênh giao tiếp full-duplex qua một kết nối TCP duy nhất. Nó cho phép truyền dữ liệu hai chiều giữa máy chủ và trình duyệt mà không cần phải thiết lập lại kết nối.

***Ưu điểm của WebSocket***

* Giao tiếp hai chiều: WebSocket cho phép truyền dữ liệu cả hai chiều giữa máy chủ và trình duyệt, giúp tương tác trực tiếp và linh hoạt.
* Hiệu suất cao: So với việc sử dụng HTTP truyền thống, WebSocket giúp giảm độ trễ và tăng tốc độ truyền dữ liệu.
* Tiết kiệm băng thông: WebSocket giúp giảm lượng dữ liệu truyền qua mạng, tối ưu hóa hiệu suất mạng.

|  |  |  |
| --- | --- | --- |
| Đặc điểm | WebSocket | HTTP |
| Kiểu kết nối | Full-duplex | Half-duplex |
| Thiết lập kết nối | Một lần | Mỗi request/response |
| Truyền dữ liệu | Hai chiều | Một chiều |
| Độ trễ | Thấp | Cao |

bảng 13: So sánh giữa WebSocket và HTTP

## 3.7 công nghệ MySql

Hệ quản trị cơ sở dữ liệu quan hệ (RDBMS): MySQL lưu trữ dữ liệu trong các bảng và cho phép mối quan hệ giữa các bảng thông qua khóa chính và khóa ngoại.

SQL: Ngôn ngữ truy vấn cấu trúc được sử dụng để truy xuất và thao tác dữ liệu trong MySQL.

***Ưu điểm của MySQL:***

* Mã nguồn mở: MySQL là mã nguồn mở, cho phép người dùng truy cập và chỉnh sửa mã nguồn theo nhu cầu riêng của mình.
* Hiệu suất cao: MySQL có khả năng xử lý truy vấn rất nhanh và hiệu quả, đặc biệt với các cơ sở dữ liệu lớn và tải công việc cao.
* Khả năng mở rộng: MySQL có thể dễ dàng mở rộng để đáp ứng nhu cầu của các ứng dụng từ nhỏ đến lớn.
* Bảo mật tốt: MySQL cung cấp nhiều tính năng bảo mật mạnh mẽ như mã hóa dữ liệu và quyền truy cập người dùng.
* Hỗ trợ đa nền tảng: MySQL hoạt động trên nhiều hệ điều hành khác nhau, bao gồm Windows, Linux, và macOS.
* Cộng đồng lớn: Với một cộng đồng người dùng rộng lớn, MySQL có sẵn nhiều tài liệu, tài nguyên hỗ trợ, và các plugin bổ sung.

Nhược điểm của MySQL:

* Hạn chế về tính năng: So với một số hệ quản trị cơ sở dữ liệu thương mại khác như Oracle Database hay Microsoft SQL Server, MySQL có thể thiếu một số tính năng nâng cao.
* Khả năng mở rộng hạn chế: Trong một số trường hợp, MySQL có thể không mở rộng tốt như mong muốn khi dữ liệu và tải công việc tăng lên.
* Hạn chế về giao dịch: Mặc dù MySQL hỗ trợ giao dịch thông qua InnoDB, nhưng không mạnh mẽ như các hệ quản trị cơ sở dữ liệu tập trung vào giao dịch khác.
* Quản lý quyền truy cập: Quản lý quyền truy cập và người dùng trong MySQL có thể phức tạp và khó quản lý hơn đối với các hệ thống lớn.

# Phát triển và triển khai ứng dụng

## Thiết kế kiến trúc

### Kiến trúc Mô hình MVVM( Model-view-view/model).

1.View (Phần giao diện)

* View: Được định nghĩa trong một template bao gồm HTML cho một component. View có thể là toàn bộ layout hoặc một phần của layout đó.
* Template: Định nghĩa cấu trúc và giao diện của component bằng cách sử dụng HTML, Angular directives và bindings.
* HTML Template: Được sử dụng để hiển thị dữ liệu từ ViewModel (Component class).

2. Model

* Model: Được định nghĩa như các thuộc tính của Component class. Model là dữ liệu mà view sử dụng để hiển thị và thao tác.
* Properties in Component Class: Dữ liệu hoặc trạng thái của component. Các thuộc tính này được liên kết với view thông qua data binding.

3. ViewModel

* ViewModel: Là lớp quản lý cả view và model. ViewModel là phần code xử lý truy xuất dữ liệu, đồng thời thực thi các tương tác người dùng trên view.
* Component Class: Là ViewModel trong Angular. Nó chứa logic xử lý dữ liệu, quản lý trạng thái và xử lý sự kiện từ view.
* Methods in Component Class: Thực hiện các tương tác người dùng, truy xuất dữ liệu từ service và cập nhật model.

4. Angular sử dụng two-way binding , giúp bất kỳ thay đổi nào trên view, cũng sẽ được lưu lại trên model và ngược lại, các thay đổi từ model cũng được hiển thị ngay trên view.

* Angular hỗ trợ properties-binding, cho phép điều khiển DOM bằng cách ràng buộc các thuộc tính HTML với thuộc tính của component class.
* Angular hỗ trợ event-binding, cho phép gắn các event với các method được định nghĩa trong class, ví dụ như các HTML event.

5.Giảm kích thước component: Trong Angular, các component nhỏ gọn và cụ thể giúp tăng hiệu suất ứng dụng bằng cách:

Tối ưu hóa tải trang: Giảm thời gian tải trang ban đầu vì chỉ tải những phần cần thiết của giao diện người dùng.

* •Dễ bảo trì: Component nhỏ dễ quản lý và bảo trì hơn so với các component lớn và phức tạp.
* •Tái sử dụng: Component nhỏ và tái sử dụng được có thể dễ dàng được sử dụng lại trong nhiều phần khác nhau của ứng dụng, giúp giảm thiểu việc viết lại mã.

Angular Module (NgModule): Module là một cách để tổ chức và nhóm các component, directive, pipe và service có liên quan lại với nhau. Điều này giúp:

* •Quản lý dễ dàng: Nhóm các thành phần có liên quan vào một module giúp quản lý và duy trì ứng dụng dễ dàng hơn.
* •Tối ưu hóa tải trang: Khi các component có chung logic được nhóm lại, chúng có thể được tải cùng nhau, giảm thiểu việc tải không cần thiết.
* •Tối ưu hóa dependency: Module cũng giúp quản lý các dependency giữa các thành phần, đảm bảo rằng các thành phần cần thiết được tải và cung cấp đúng cách.

Lazy Loading: Đây là kỹ thuật tối ưu hóa hiệu suất tải trang bằng cách chỉ tải các module khi người dùng thực sự cần chúng. Điều này có nghĩa là:

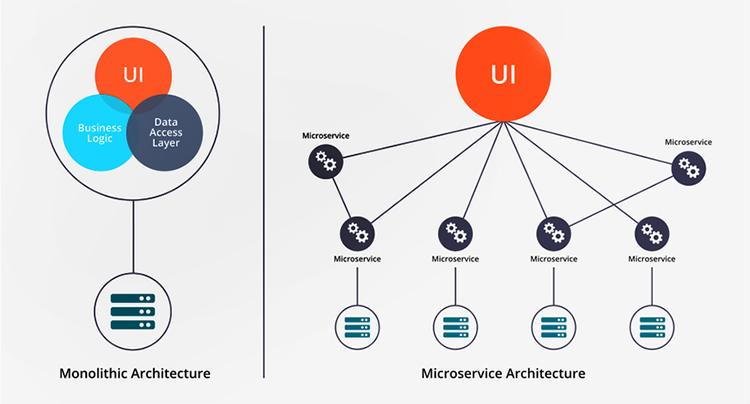
* •Tăng tốc độ tải trang ban đầu: Bằng cách chỉ tải những phần cần thiết, thời gian tải trang ban đầu sẽ nhanh hơn.
* •Tiết kiệm băng thông: Giảm lượng dữ liệu cần tải xuống, tiết kiệm băng thông và tài nguyên hệ thống.
* •Cải thiện trải nghiệm người dùng: Ứng dụng phản hồi nhanh hơn và trải nghiệm người dùng được cải thiện đáng kể.

### 4.1.2.BACK-END MICROSERVICE – SPRINGBOOT

Microservices là tên gọi cho kiến trúc chia service thành các thành phần nhỏ hơn( service), mỗi thành phần sẽ phục vụ một mục đích nghiệp vụ xác định, có liên quan.

Ở kiến trúc Monolithic, tất cả các request, các nghiệp vụ xử lý đều đẩy hết về một service, nên việc bảo trì, mở rộng trở nên khó khăn hơn, rủi ro hơn rất nhiều. Ví dụ như chúng ta muốn thay đổi, hoặc thêm mới một vài nghiệp vụ, chúng ta sẽ phải triển khai lại toàn bộ service, điều này, trong trường hợp chức năng mới bị lỗi, sẽ dễ gây ra ảnh hưởng tới các nghiệp vụ khác dù chúng không liên quan đến nhau.

Với kiến trúc Microservices, các nghiệp vụ được chia về các service khác nhau xử lý, nên khi cần nâng cấp, hoặc thêm mới ở nghiệp vụ nào, chúng ta chỉ cần triển khai lại service đó mà thôi. Hoặc nếu nâng cấp thêm nghiệp vụ mới, chúng ta hoàn toàn có thể triển khai trên một service mới.



Hình 11: Mô tả kiến trúc back-end.

Khi sử dụng microservices, chúng ta muốn nâng cấp công nghệ mới thì cũng rất dễ thực hiện do các services được triển khai riêng lẻ, trong khi điều đó là không thể với Monolithic.

Những điều cần tuân thủ khi triển khai kiến trúc Microservices

* Một service có phạm vi và chức năng giới hạn thì việc tập trung vào một nhiệm vụ sẽ giúp cho quá trình phát triển cũng như triển khai trở nên nhanh chóng hơn.
* Khi thiết kế, chúng ta nên xác định giới hạn cho các service dựa theo nghiệp vụ thực tế.
* Phải đảm bảo các service có thể phát triển cũng như triển khai một cách độc lập.
* Mục tiêu thiết kế là đưa ra phạm vi cho một microservices phục cụ cho một nghiệp vụ chứ không đơn giản là làm những dịch vụ nhỏ hơn. Khi đó, kích thước hợp lý của một services đó chính là kích thước đủ để đáp ứng cho các yêu cầu của một chức năng bên trong hệ thống.

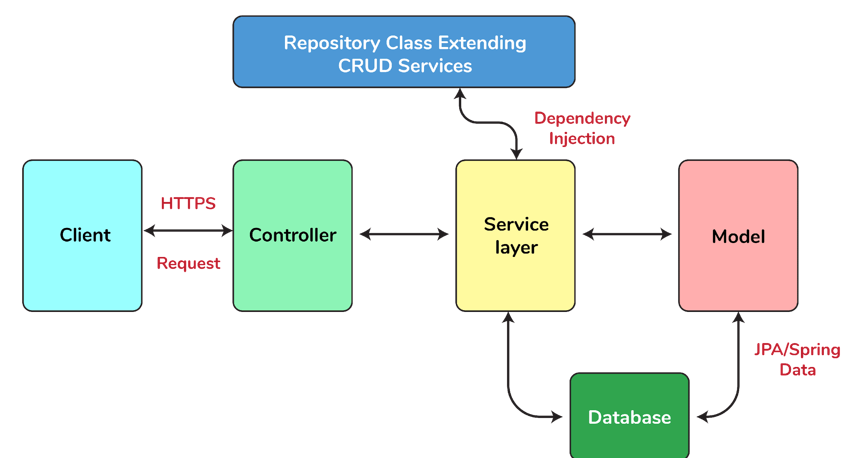
Eureka Service

Thành phần được phát triển bằng ngôn ngữ lập trình Java Spring Boot có sử dụng công nghệ Netflix Eureka Server. Thành phần trong kiến trúc Microservice hỗ trợ các Micorservice nhận biết nhau tại những địa chỉ, port riêng mà nó đang triển khai.

Zuul API Gateway( Authentication Gateway)

Thành phần được phát triển bằng ngôn ngữ lập trình Java Spring Boot có sử dụng công nghệ Netflix Zuul và Eureka Client. Thành phần này được đăng ký và quản lý bởi với Eureka Server. Thành phần này là cổng API cho phép định tuyến tất cả các http request từ Web UI Client (hoặc ứng dụng khác) đến được những Core Service mà nó mong muốn xử lý.

SPRINGBOOT và các nghiệp vụ tại core service



Hình 12: Sơ đồ nghiệp vụ tại back-end

* 1. Controller là nơi tiếp nhận các request được chuyển sang từ API Gateway và trả kết quả lại API Gateway sau khi request đã được xử lý trong service.
  2. Service là nơi xử lý nghiệp vụ, xử lý dữ liệu trước khi trả kết quả về Controller.
  3. Repository là nơi thực hiện kết nối, truy vấn dữ liệu từ Database và mapping các dữ liệu nhận được vào các Entity, hay các DTO.
  4. Entity mapping thuộc tính của class entity với các cột dữ liệu của bảng trong database, mỗi entity sẽ tương ứng với một bảng dữ liệu.
  5. DTO mapping các thông tin truy vấn từ database, nhưng DTO không như entity mapping theo từng bảng dữ liệu, mà sẽ bao gồm các thông tin người dùng mong muốn nhận về, có thể là một vài cột dữ liệu của bảng, hoặc các cột dữ liệu của các câu truy vấn lồng nhau phức tạp.

### Các câu hỏi trong bào đánh giá

* Bài đánh giá thứ nhất:
* Yêu cầu: chuẩn theo đánh giá của thang đo kiểm tra trạng thái tâm thần ngắn (MMSE). Thang đo kiểm tra trạng thái tâm thần ngắn (MMSE) là một test đánh giá trạng thái tâm thần một cách sơ bộ và đơn giản hiện nay đang được dùng phổ biến. Test này ban đầu được Folstein và công sự giới thiệu vào năng 1975, để phân biệt các bệnh nhân tâm thần chức năng và thực thể.
* Bài đánh giá thứ hai:
* Yêu cầu: dựa theo một bài kiểm tra cánh cửa với nội dung như sau:

Bài đánh giá sẽ có 2 phần tương tự câu hỏi như phần 1 nhưng phần 2 sẽ có hình ảnh dễ hơn phần 1. Phần 1 sẽ cho người già nhìn trước 12 cánh cửa mỗi 3s 1 cánh sau đó sẽ là 12 câu hỏi được đưa ra cũng 3s 1 câu hỏi. Khi người già có thể chọn đúng trên 5 cánh cửa của phần 1 sẽ được làm phần 2.

## Thiết kế chi tiết

### Thiết kế giao diện

**1. Yêu cầu**

❖ Thân thiện với người dùng: Đẹp, màu sắc hài hòa, dễ thao tác, các nút và

kết nối dễ nhận biết, font và kích cỡ chữ dễ đọc.

❖ Giao diện đơn giản nhưng vẫn bắt mắt người xem, cách sử dụng đơn giản

❖ Sắp xếp biểu tượng, vị trí các chức năng hợp lý, quen mắt với người dùng.

Không dùng những biểu tượng quá khó hiểu.

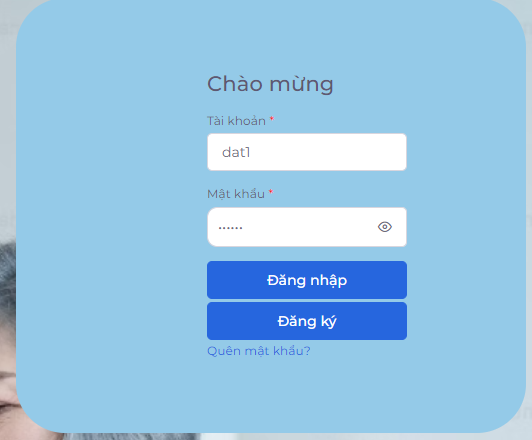
❖ Màu sắc trong giao diện hài hòa, tăng cảm giác dễ chịu cho người dùng.

Không dùng những tone màu quá nóng hoặc quá chói.

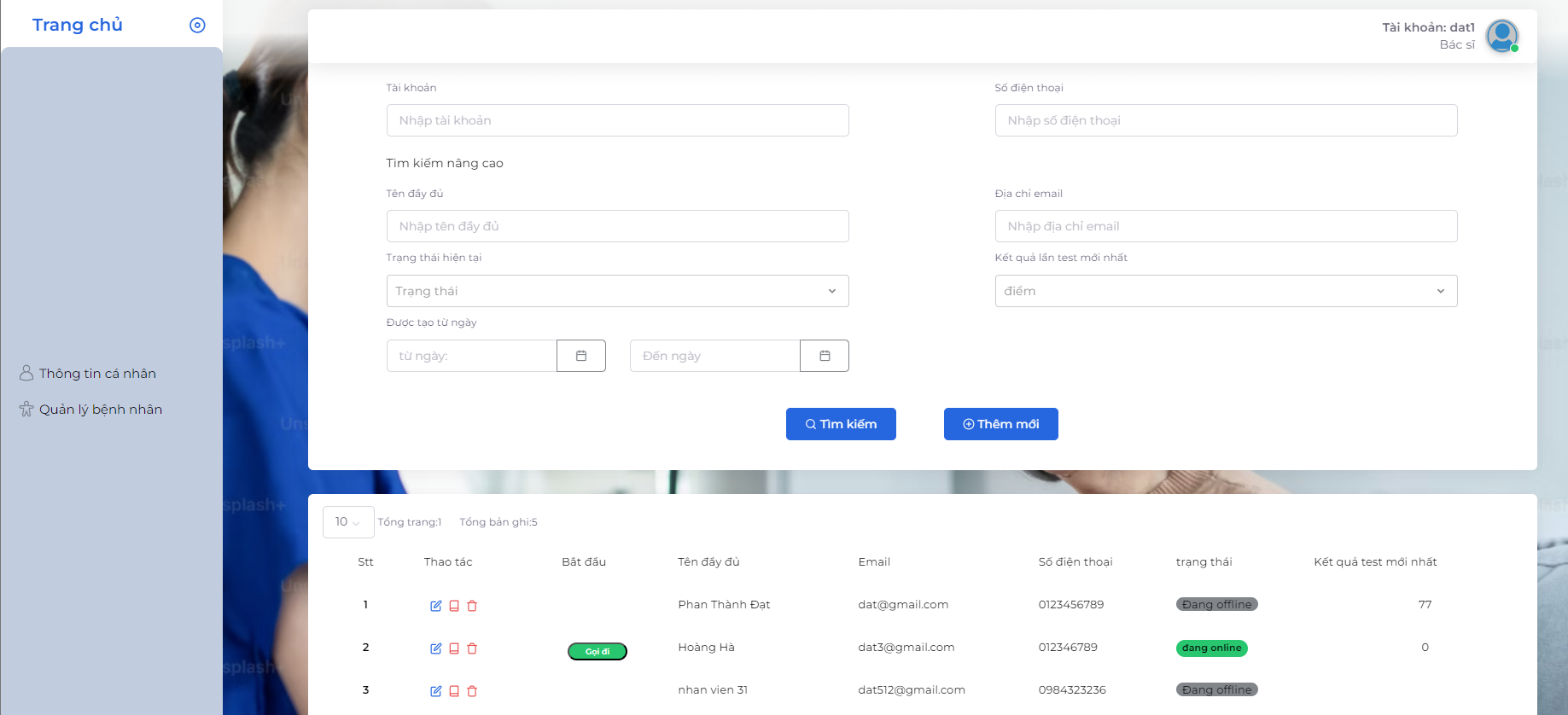
❖ Nội dung ngắn gọn, súc tích, từ ngữ chính xác, không gây hiểu nhầm.

**2. Hình ảnh minh họa**

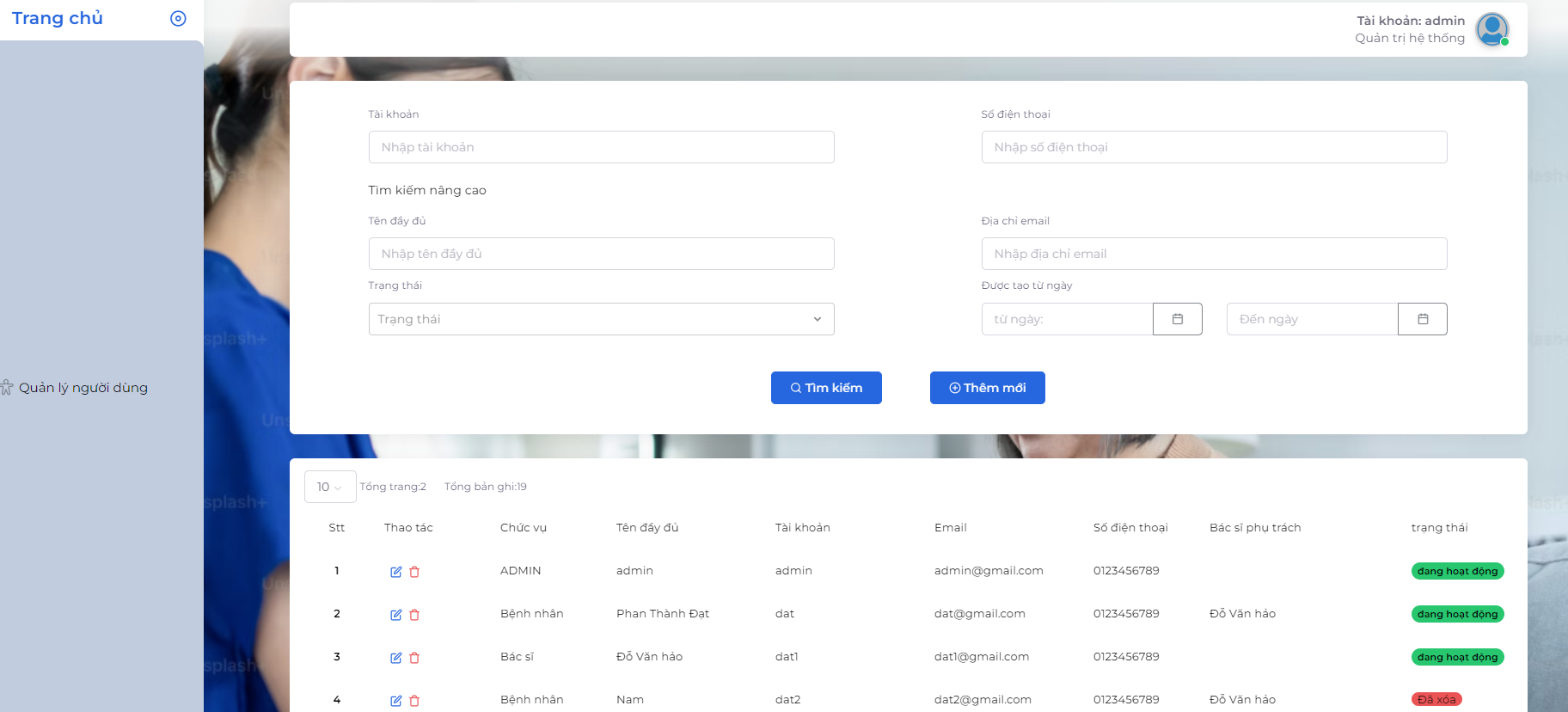
Dưới đây là một số hình ảnh minh họa cho một số chức năng quan trọng.



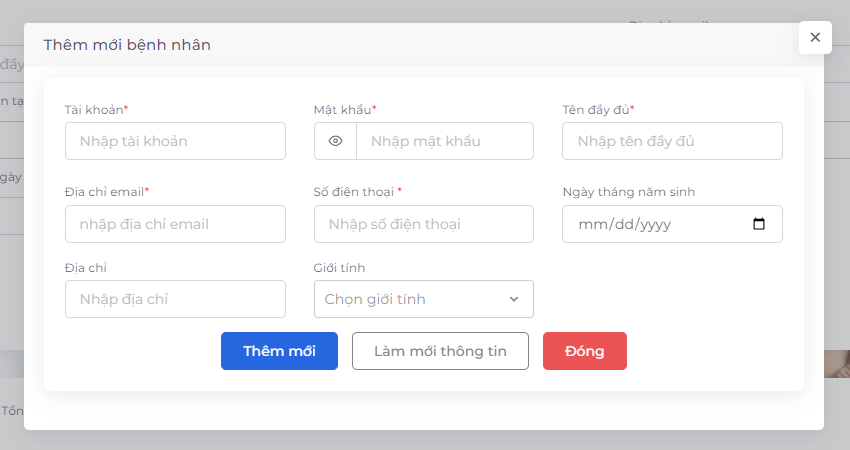
Hình 13:Minh họa chức năng đăng nhâp.



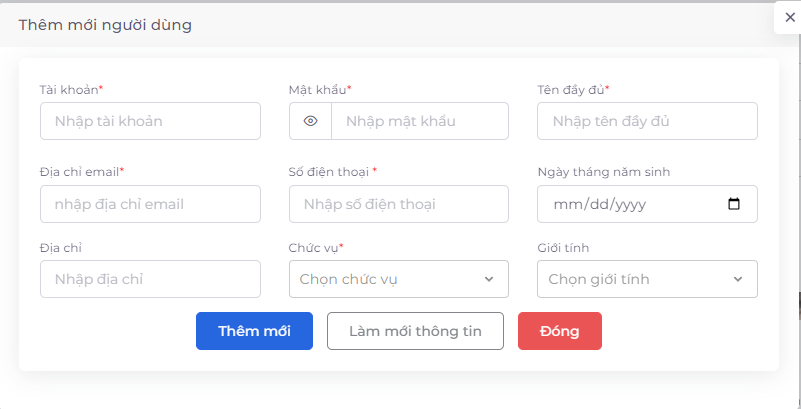
Hình 14:Minh họa trang quản lý thông tin bệnh nhân.



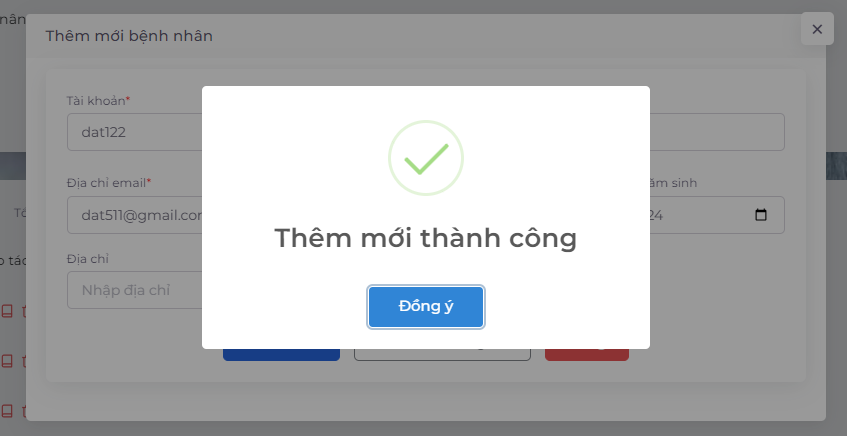
Hình 15:Minh họa trang quản lý thông tin người dùng.



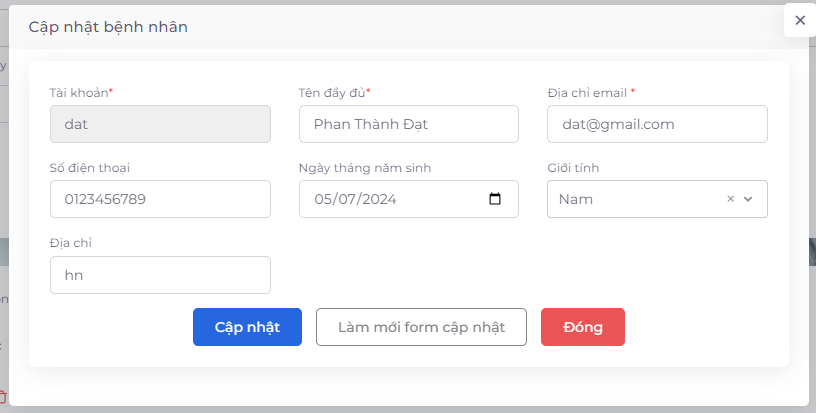
Hình 16:Minh họa chức năng thêm mới bệnh nhân.



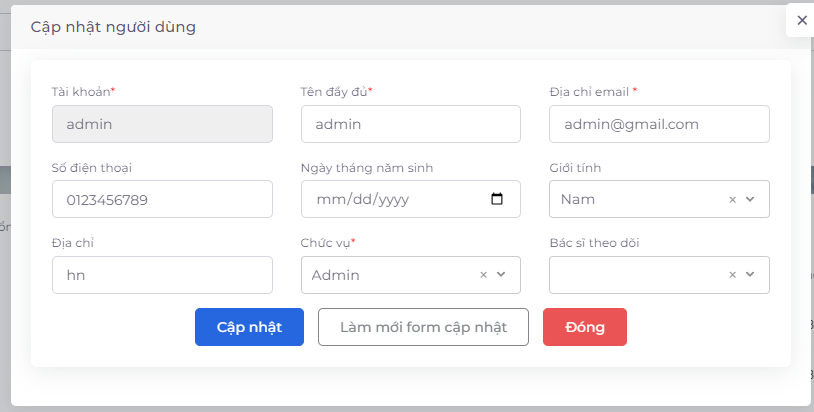
Hình 17:Minh họa chức năng thêm mới người dùng.



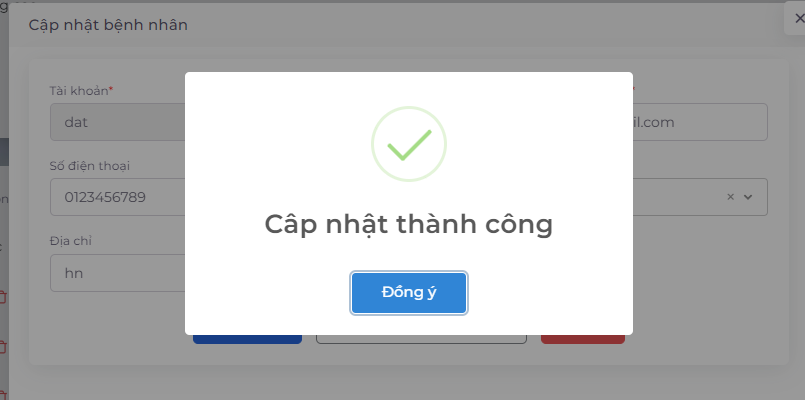
Hình 18 :Minh họa thêm mới thành công người dùng/bệnh nhân.



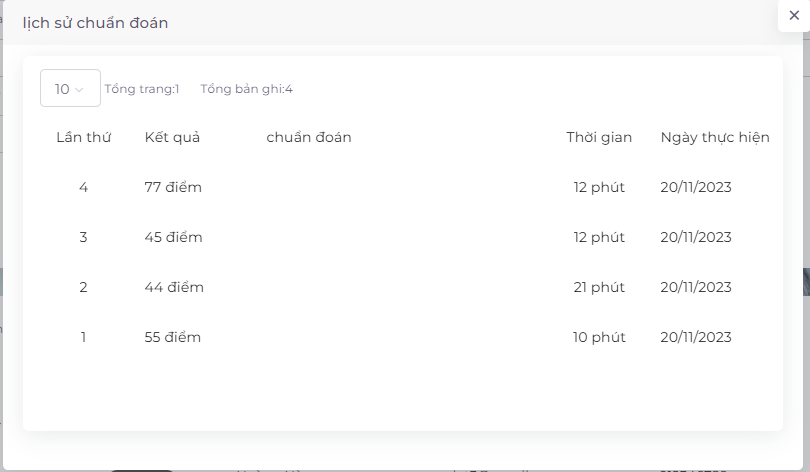
Hình 19 :Minh họa chức năng cập nhật thông tin bệnh nhân.



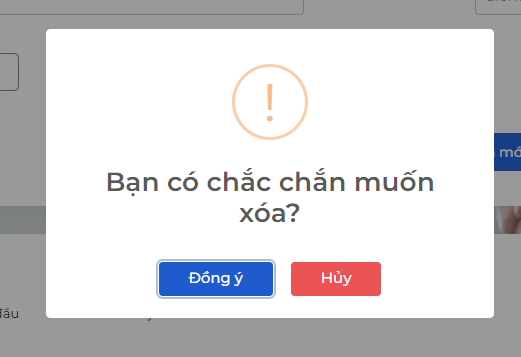
Hình 20:Minh họa chức năng cập nhật thông tin người dùng.



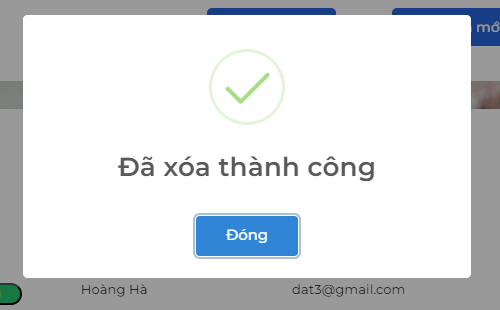
Hình 21: Minh họa cập nhật thông tin thành công



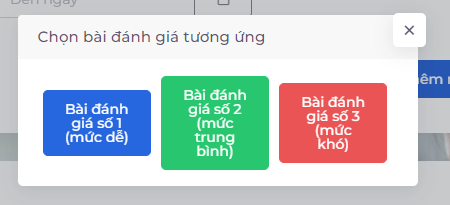
Hình 22:Minh họa xem thông tin lịch sử đánh giá của bác sĩ



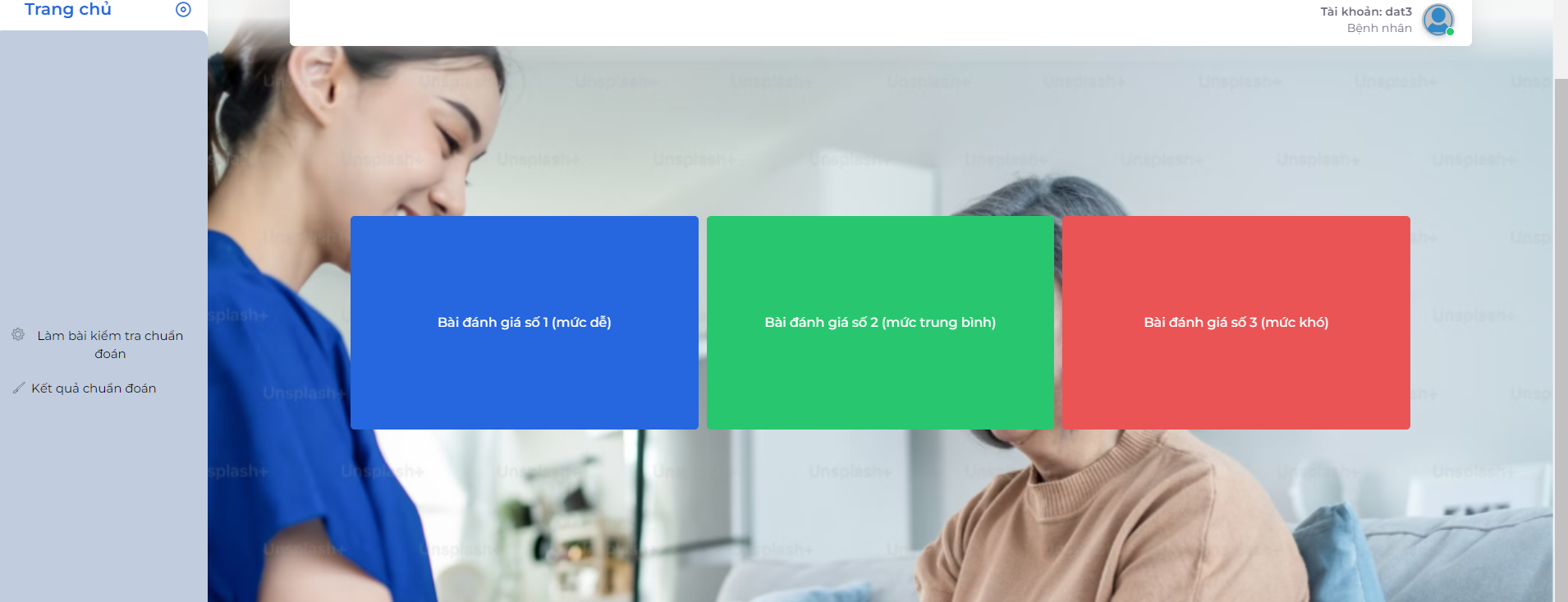
Hình 23:Minh họa cảnh báo xóa người dùng/bệnh nhân.



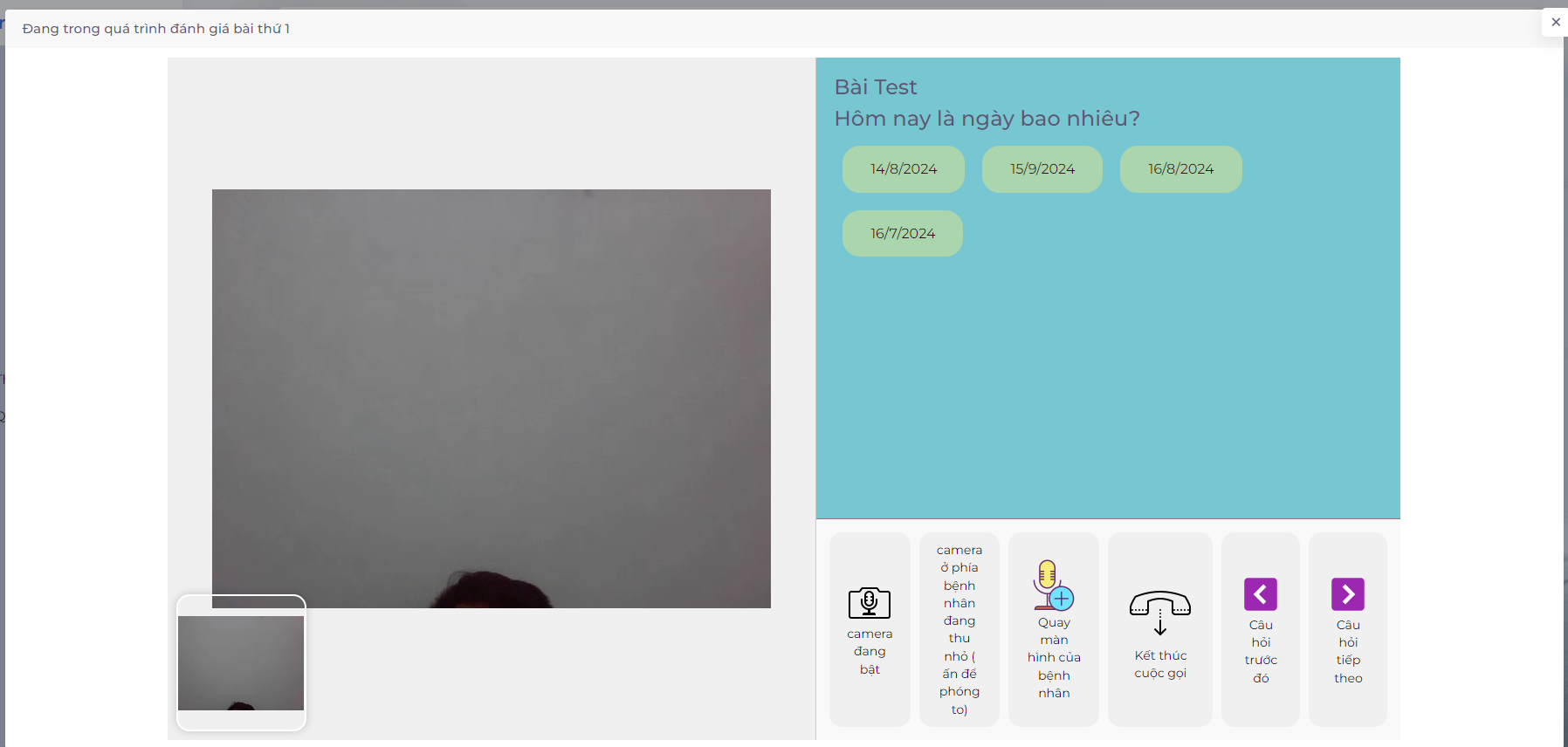
Hình 24: Minh họa xóa người dùng/bệnh nhân thành công



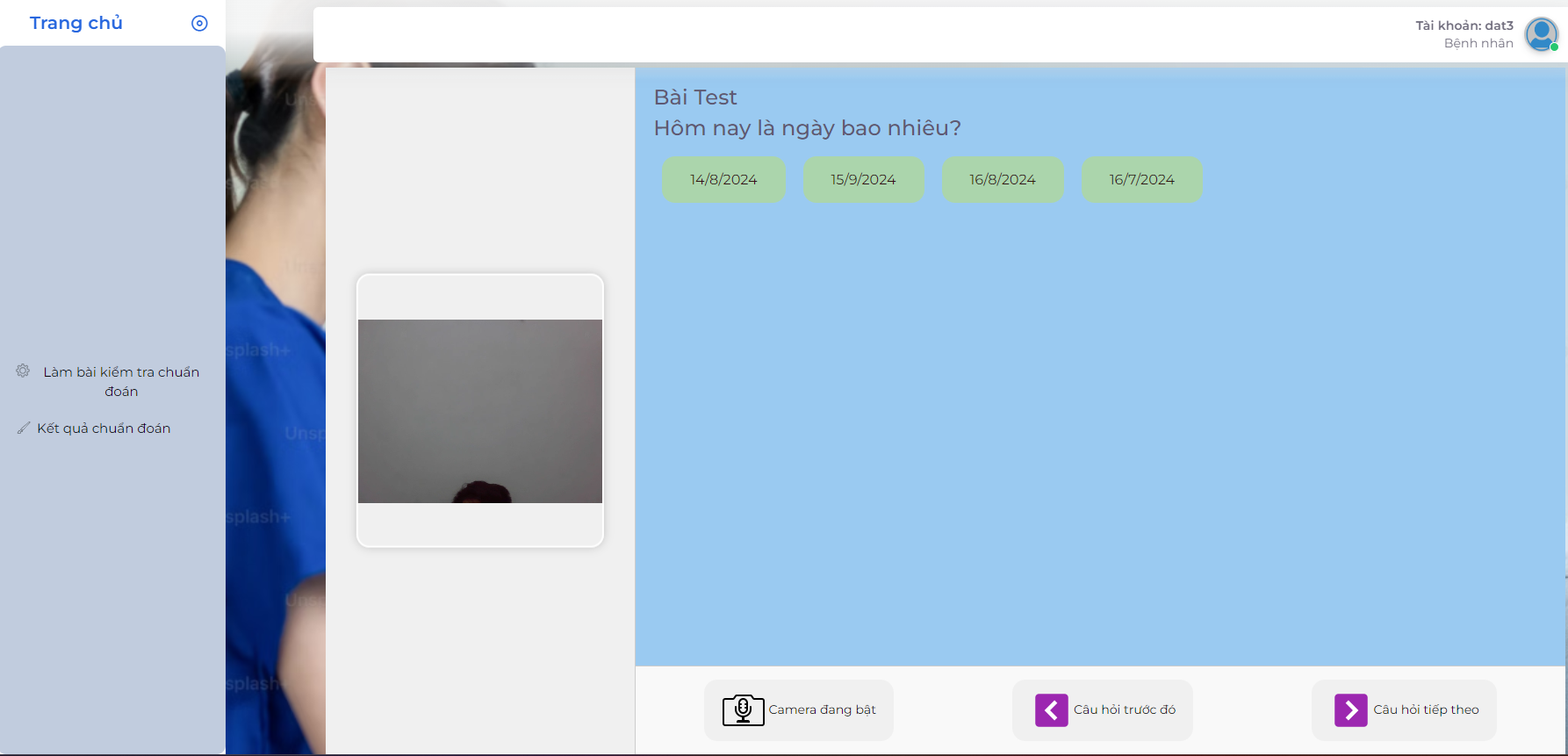
Hình 25: Minh họa chọn các bài đánh giá tương ứng ở phía bác sĩ phụ trách.



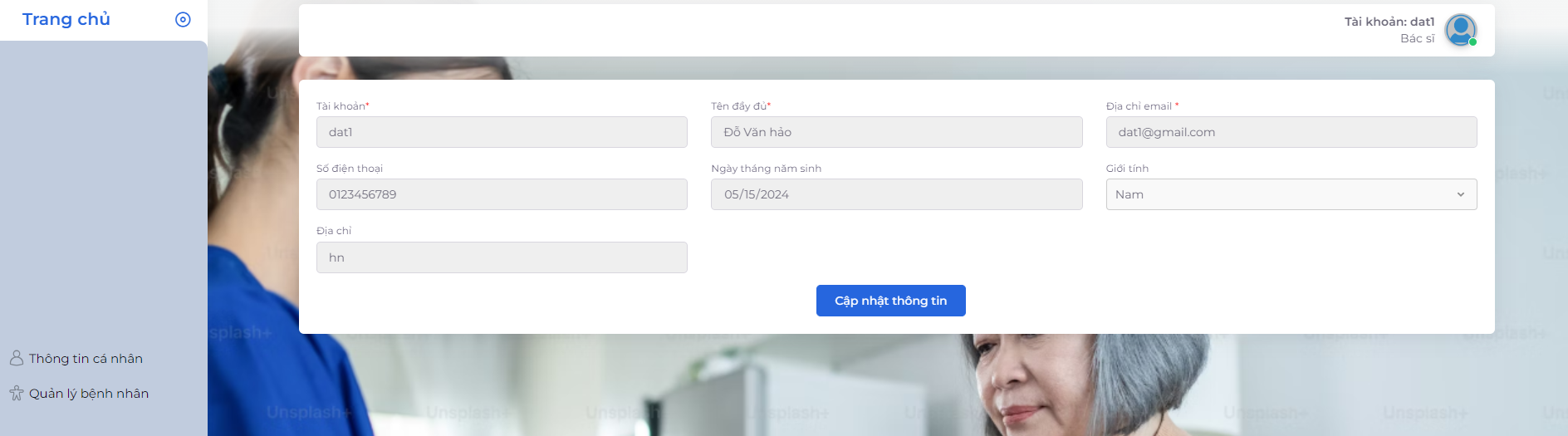
Hình 26:Minh họa chọn các bài đánh giá tương ứng ở phía bệnh nhân.



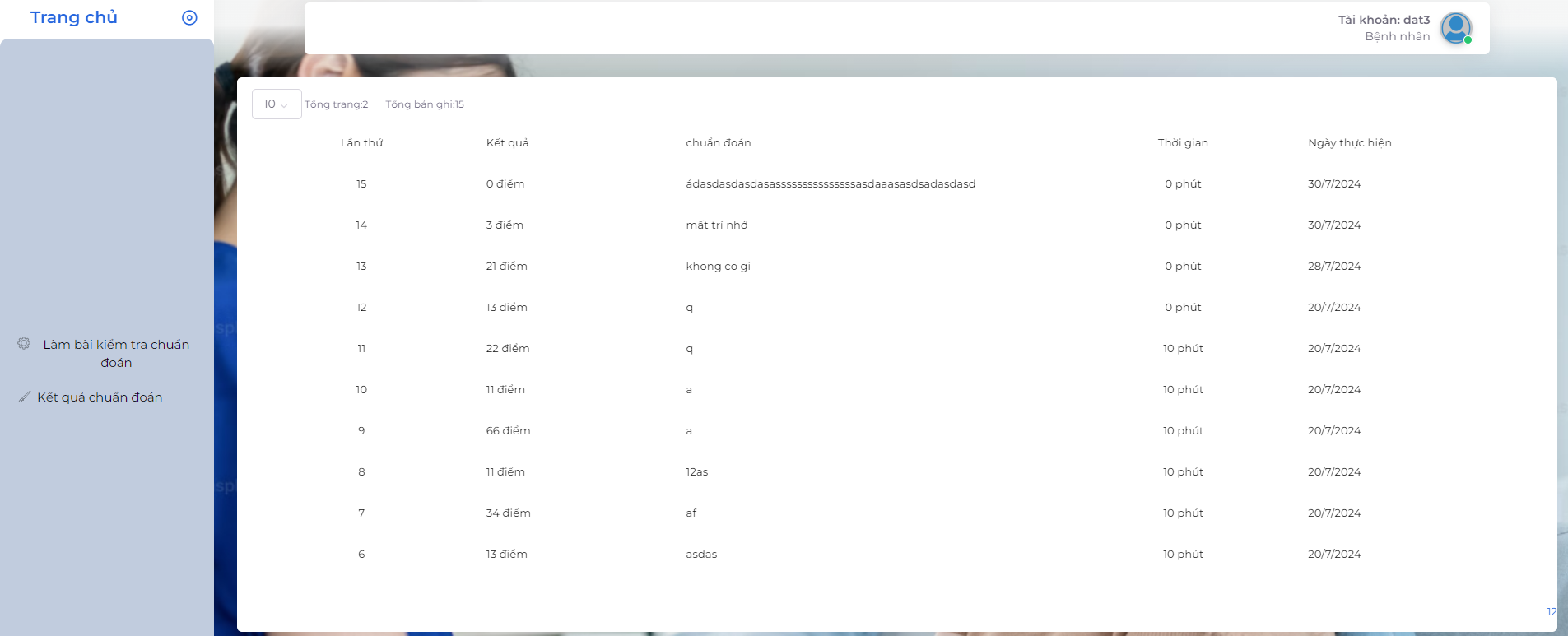
Hình 27: Minh họa kiểm tra bài đánh giá thứ nhất ở màn hình bác sĩ.



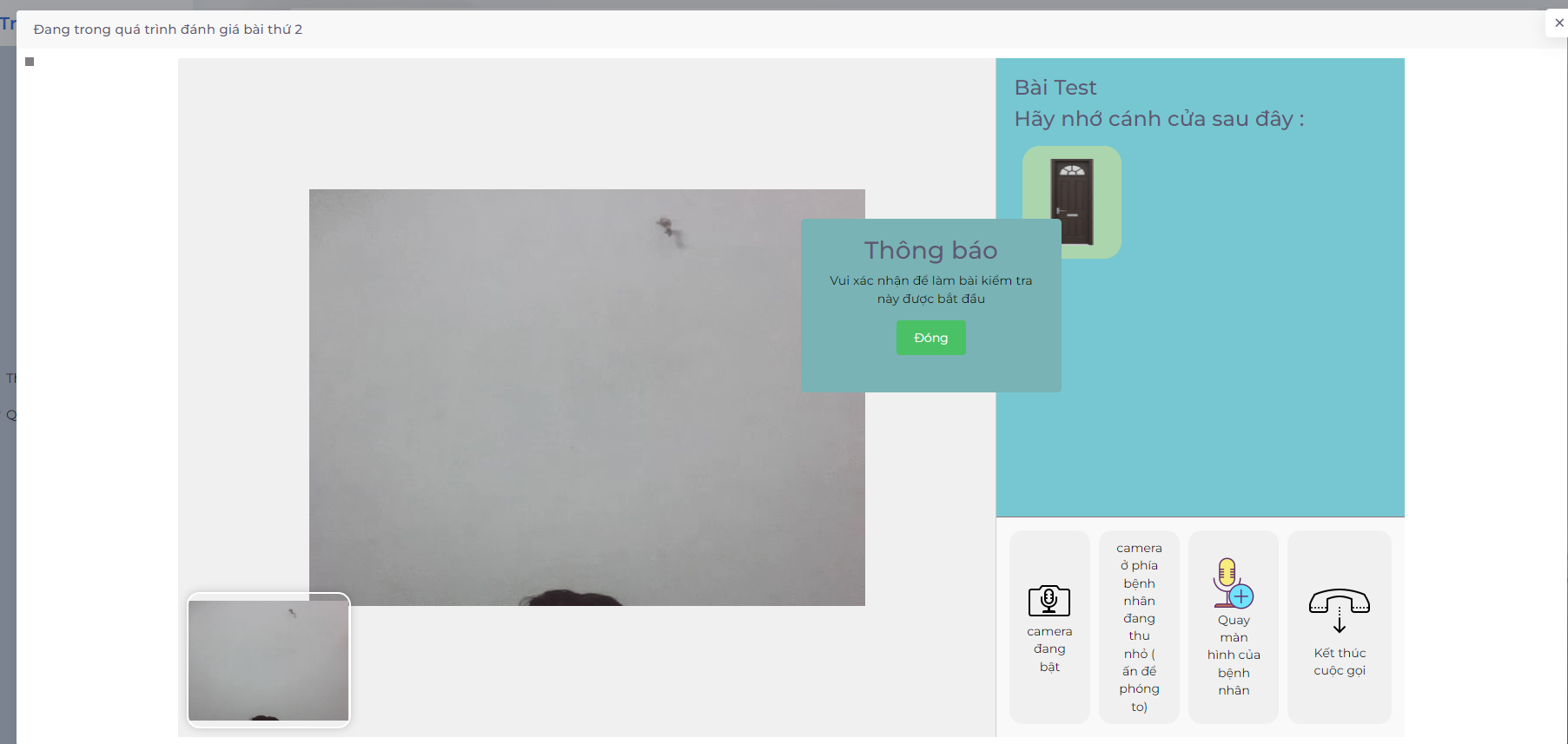
Hình 28: Minh họa kiểm tra bài đánh giá thứ nhất ở màn hình bệnh nhân.



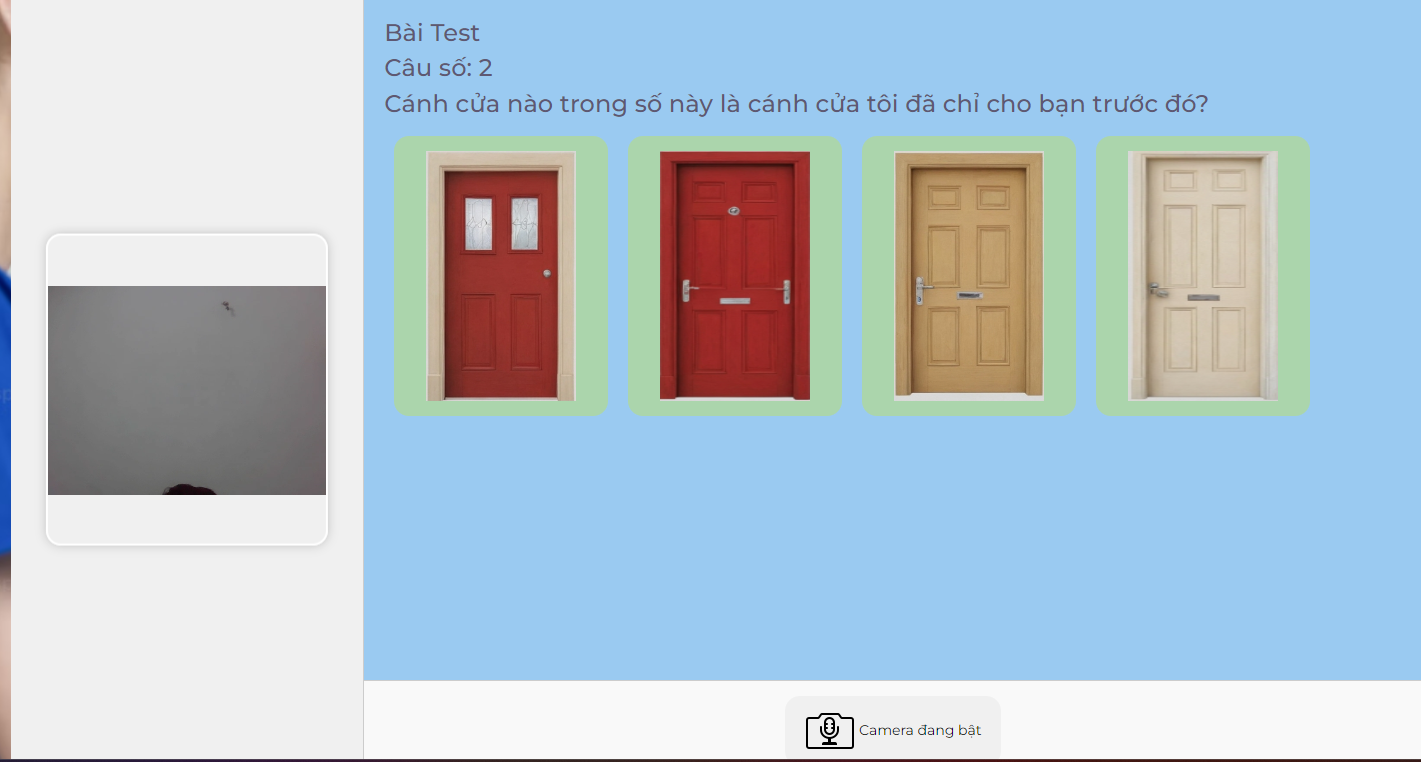
Hình 29: Minh họa cập nhật thông tin cá nhân của màn hình bác sĩ.



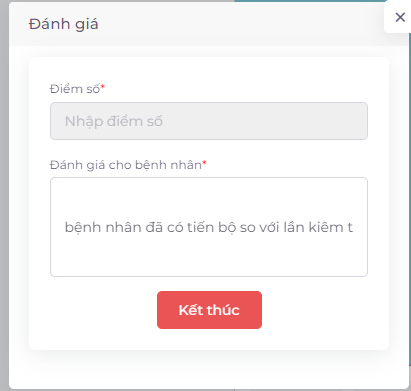
Hình 30:Minh họa kiểm tra kết quả các bài đánh giá ở màn hình bệnh nhân.



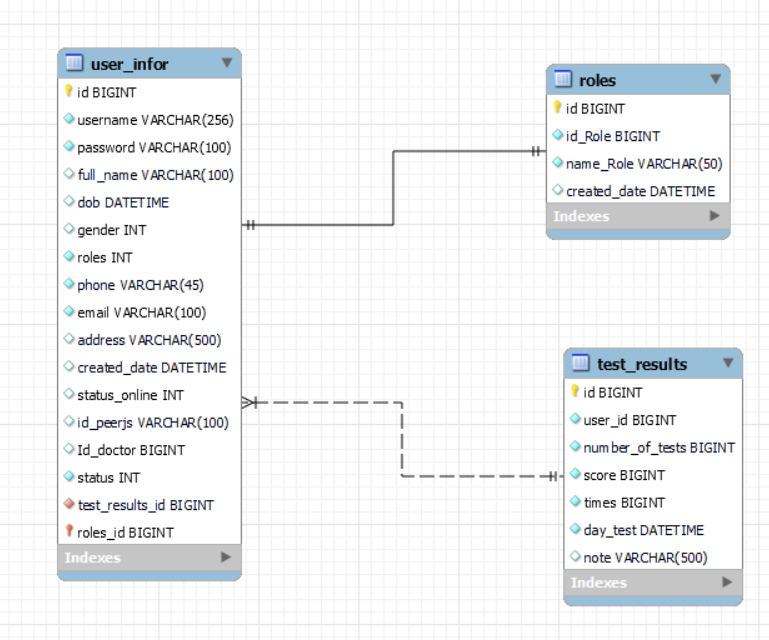
Hình 31: Minh họa kiểm tra bài đánh giá thứ hai ở màn hình bác sĩ.



Hình 32: Minh họa kiểm tra bài đánh giá thứ hai ở màn hình bệnh nhân.



Hình 33: Minh họa ghi chú đánh giá khi đã hoàn thành kiểm tra của bác sĩ.



Hình 34:Minh họa biểu đồ E-R Diagram của dự án

**Giải thích E-R Diagram:**

* *Thực thể user\_infor:*

Thuộc tính:

* id: Khóa chính, định danh duy nhất cho từng người dùng.
* username: Tên người dùng.
* password: Mật khẩu cho tài khoản.
* full\_name: Tên đầy đủ.
* dob: Ngày sinh.
* gender: Giới tính.
* roles: Chỉ định vai trò của người dùng, liên kết với bảng roles.
* phone: Số điện thoại.
* email: Địa chỉ email.
* address: Địa chỉ.
* created\_date: Ngày tạo tài khoản.
* status\_online: Trạng thái trực tuyến.
* id\_peerjs: ID cho dịch vụ PeerJS (nếu có).
* Id\_doctor: ID của bác sĩ liên quan (nếu có).
* status: Trạng thái hoạt động của tài khoản.
* Thực thể roles:

Thuộc tính:

* id: Khóa chính, định danh duy nhất cho từng vai trò.
* id\_Role: ID vai trò.
* name\_Role: Tên của vai trò.
* created\_date: Ngày tạo vai trò.
* Thực thể test\_results:

Thuộc tính:

* id: Khóa chính, định danh duy nhất cho từng kết quả kiểm tra.
* user\_id: ID người dùng liên quan đến kết quả, liên kết với bảng user\_infor.
* number\_of\_tests: Số lần bài kiểm tra đã thực hiện.
* score: Điểm số đạt được.
* times: Thời gian làm bài kiểm tra.
* day\_test: Ngày thực hiện bài kiểm tra.
* note: Ghi chú về kết quả.

**2.Quan hệ giữa các bảng**

* user\_infor và roles: Có mối quan hệ nhiều-một, nơi mỗi người dùng có thể có một vai trò duy nhất.
* user\_infor và test\_results: Có mối quan hệ một-nhiều, nơi mỗi người dùng có thể có nhiều kết quả kiểm tra khác nhau.

## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

Sinh viên liệt kê các công cụ, ngôn ngữ lập trình, API, thư viện, IDE, công cụ kiểm thử, v.v. mà mình sử dụng để phát triển ứng dụng. Mỗi công cụ phải được chỉ rõ phiên bản sử dụng. SV nên kẻ bảng mô tả tương tự như Bảng 1. Nếu có nhiều nội dung trình bày, sinh viên cần xoay ngang bảng.

bảng 14:Danh sách thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Mục đích** | **Công cụ** | **Địa chỉ URL** |
| IDE lập trình | Eclipse Oxygen 64 bit | http://www.eclipse.org/ |
| IDE lập trình | Visual Studio Code | https://code.visualstudio.com/ |
| Cơ sở dữ liệu | MySQL-8.0 | https://www.mysql.com/ |
| Hệ quản trị phiên bản | Git | https://git-scm.com/ |
| Hệ quản trị dự án | GitHub | https://github.com/ |
| Thư viện CSS | Bootstrap | https://getbootstrap.com/ |
| kiểm thử API | Postman | https://www.postman.com/ |

### Kết quả đạt được

Sản phẩm chính của dự án là Website “Xây dựng ứng dụng đánh giá sa sút trí tuệ người cao tuổi bằng hệ thống chẩn đoán từ xa” .Website được xây dựng với mục xây dựng và hoàn thiện một ứng dụng chẩn đoán từ xa với các tính năng hữu ích và đáp ứng nhu cầu của người dùng.

Minh hoạ các chức năng chính:

➢ Quản lý người dùng.  
➢ Chức năng đánh giá bệnh nhân.  
➢ Đánh giá các bài kiểm tra.  
➢ Thay đổi thông tin cá nhân.   
➢ Xem kết quả kiểm tra.

Minh họa một số chức năng quan trọng:

* Chức năng quản lý người dùng.

Mô tả:Chức năng nằm trong màn hình quản lý: tìm kiếm, thêm mới, sửa, xóa người dùng

* Chức năng đánh giá bệnh nhân.

Mô tả: nút tắt camera, chuyển tiếp các câu hỏi, xem hình ảnh bệnh nhân trực tiếp, đánh giá kết quả nhận xét.

* Chức năng kiểm tra kết quả.

Mô tả: kiểm tra số lần đã đánh giá, điểm số cùng với thời gian đánh giá tương ứng của các lần.

## Kiểm thử

### kiểm thử màn hình đăng nhập.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Mục đích | Các bước | Kết quả mong muốn | Kết quả hiện tại |
| 1 | Giao diện | Bố cục, form, chính tả | -Các label, combo, textbox có chiều dài, rộng, khoảng cách bằng nhau, Các trường bắt buộc nhập phải có dấu \*  - Kiểm tra về lỗi chính tả, cấu trúc câu, ngữ pháp trên màn hình | Pass |
| 2 | Chức năng đăng nhập | 1. Chọn đơn vị , Nhập tên đăng nhập và mật khẩu cặp tồn tại đã có trong DB  2. Click chọn button Đăng nhập | Đăng nhập thành công | Pass |
| 3 | Chức năng đăng nhập | 1. Chọn đơn vị , Nhập tên đăng nhập và mật khẩu cặp tồn tại chưa có trong DB  2. Click chọn button Đăng nhập | Đăng nhập không thành công, hiển thị lỗi | Pass |

bảng 15: Mô tả kiểm thử màn hình đăng nhập

### kiểm thử màn hình quản lý.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Mục đích | Các bước | Kết quả mong muốn | Kết quả hiện tại |
| 1 | Giao diện | Kiểm tra màn hình ở trạng thái mặc định | Màn hình chức năng gồm các thông tin:  Hiển thị các trường:  + Searchbox:Tài khoản, số điện thoại  + Button:Tìm kiếm,Thêm mới, Xóa,  + Link tìm kiếm nâng cao (Gồm các trường: Họ và tên, Trạng thái, ,ngày tạo,..)  -Grid kết quả tìm kiếm hiển thị các thông tin sau:  + STT  + Thao tác  + Họ tên  + Tài khoản  + Email  + Số điện thoại  ... | Pass |
| 2 | Tìm kiếm | 1. Đăng nhập tài khoản hợp lệ  2. Chọn chức năng Quản lý người dùng | - Nếu người dùng được phân quyền thì cho phép xem, tìm kiếm trong chức năng quản lý người dùng  - Nếu người dùng không được phân quyền thì không cho phép xem, tìm kiếm trong chức năng quản lý người dùng | Pass |
| 3 | Chức năng | 1. Đăng nhập tài khoản hợp lệ  2. Chọn chức năng Quản lý người dùng  3. Nhập dữ liệu có trong CSDL  4. Click biểu tượng Tìm kiếm | Hiển thị tất cả dữ liệu có trong database tồn tại của ô text vừa nhập | Pass |
| 4 | Chức năng | 1. Đăng nhập tài khoản hợp lệ  2. Chọn chức năng Quản lý người dùng  3. thêm/sửa/xóa/call người dùng tương ứng | Thực hiện thành công những hành động trên mà không có lỗi | Pass |

bảng 16:Mô tả kiểm thử màn hình quản lý

### kiểm thử màn hình đánh giá bệnh nhân.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Mục đích | Các bước | Kết quả mong muốn | Kết quả hiện tại |
| 1 | Giao diện | 1. Đăng nhập tài khoản hợp lệ | Hiển thị 3 bài test | Pass |
| 2 | Chức năng | 1. Đăng nhập tài khoản hợp lệ  2. Chọn bài test tương ứng | Hiển thị các bài test tương ứng | Pass |

bảng 17: Mô tả kiểm thử màn hình đánh giá bệnh nhân.

### kiểm thử chức năng thay đổi thông tin cá nhân của bản thân.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Mục đích | Các bước | Kết quả mong muốn | Kết quả hiện tại |
| 1 | Giao diện | 1. Đăng nhập tài khoản hợp lệ  2. Chọn thông tin cá nhân | Hiển thị form thay đổi thông tin cá nhân | Pass |
| 2 | Chức năng | 1. Đăng nhập tài khoản hợp lệ  2. Chọn thông tin cá nhân  3. nhập các thông tin cần thiết.  4. nhấn button lưu | Thông thay đổi thông tin thành công mà không thông báo lỗi. | Pass |

bảng 18: Mô tả kiểm thử chức năng thay đổi thông tin cá nhân của bản thân.

### kiểm thử màn hình kiểm tra của bệnh nhân.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stt | Mục đích | Các bước | Kết quả mong muốn | Kết quả hiện tại |
| 1 | Giao diện | 1. Đăng nhập tài khoản hợp lệ | Hiển thị 3 bài test | Pass |
| 2 | Chức năng | 1. Đăng nhập tài khoản hợp lệ  2. Chọn bài test tương ứng | Hiển thị các bài test tương ứng | Pass |

bảng 19: Mô tả kiểm thử màn hình kiểm tra của bệnh nhân.

# Các giải pháp và đóng góp nổi bật

# Kết luận và hướng phát triển

## Kết luận

Đề tài “Xây dựng ứng dụng đánh giá sa sút trí tuệ người cao tuổi bằng hệ thống chẩn đoán từ xa” cho thấy rõ ràng rằng hiện nay ở Việt Nam rất ít có một sản phẩm tương tự nào được phát triển và triển khai. Mặc dù trên thế giới đã có một số ứng dụng và hệ thống hỗ trợ đánh giá sa sút trí tuệ từ xa, việc tiếp cận các sản phẩm này tại Việt Nam gặp nhiều khó khăn do rào cản về ngôn ngữ, chi phí, và tính phù hợp với người dùng Việt Nam.

Tuy nhiên, trong quá trình thực hiện, cũng có những thách thức và hạn chế. Hệ thống hiện tại mới chỉ đáp ứng được các chức năng cơ bản và cần tiếp tục phát triển thêm các tính năng nâng cao để cải thiện độ chính xác và hiệu quả. Do hạn chế về thời gian, khả năng và kinh nghiệm nên không tránh khỏi những thiếu sót nhất định nên đề tài đã hoàn thành ở mức độ sau:

❖ Tìm hiểu ngôn ngữ lập trình Java, Javascript, typescript & mySql query.

❖ Tìm hiểu các framework Spring boot, Boottrap,angular...

❖ Áp dụng xây dựng ứng dụng đánh giá sa sút trí tuệ người cao tuổi bằng hệ thống chẩn đoán từ xa.

Tôi hy vọng rằng sản phẩm này sẽ được phát triển và cải tiến thêm, trở thành một công cụ hữu ích và đáng tin cậy trong việc chăm sóc sức khỏe cho người cao tuổi.

## Hướng phát triển

Định hướng công việc trong tương lai nhằm hoàn thiện và nâng cao chất lượng của ứng dụng đánh giá sa sút trí tuệ người cao tuổi bằng hệ thống chẩn đoán từ xa có thể được chia thành hai phần: hoàn thiện các chức năng hiện tại và phát triển các hướng đi mới.

* Hoàn thiện các chức năng hiện tại:
* Xây dựng trang Web quy mô lớn hơn với nhiều ứng dụng, chức năng.
* Sửa lỗi vặt của website hiện hành, thêm một số chức năng nâng cao độ chuyên nghiệp.của trang web cường độ chính xác của chẩn đoánViệc tích hợp thêm các công cụ phân tích trí tuệ nhân tạo và học máy có thể giúp tối ưu hóa quá trình xử lý dữ liệu và đưa ra kết quả chính xác hơn.
* Tối ưu hóa giao diện người dùng: Mặc dù đã chú trọng thiết kế giao diện thân thiện, nhưng việc tối ưu hóa hơn nữa để phù hợp với khả năng sử dụng của người cao tuổi là cần thiết. Điều này bao gồm việc cung cấp hướng dẫn rõ ràng hơn, giảm thiểu các bước thực hiện không cần thiết và cải thiện trải nghiệm người dùng tổng thể.
* Tích hợp sâu rộng với hệ thống y tế hiện có: Tăng cường khả năng tương tác với các hệ thống quản lý y tế đang hoạt động tại Việt Nam để cung cấp dữ liệu liên tục và nhất quán cho các bác sĩ, giúp họ theo dõi và chẩn đoán tốt hơn.
* Phát triển các hướng đi mới:
* Phát triển các chức năng cá nhân hóa: Nghiên cứu và phát triển các tính năng cho phép cá nhân hóa trải nghiệm người dùng, ví dụ như tùy chỉnh bài kiểm tra dựa trên dữ liệu lịch sử và đặc điểm cá nhân của từng người dùng.
* Mở rộng đối tượng sử dụng: Hướng đến việc mở rộng ứng dụng cho các đối tượng người dùng khác, như người trẻ tuổi có nguy cơ suy giảm trí nhớ sớm, hoặc phát triển phiên bản dành riêng cho các chuyên gia y tế để sử dụng trong việc theo dõi bệnh nhân.
* Nghiên cứu tích hợp công nghệ mới: Khám phá các công nghệ mới như thực tế ảo (VR) hoặc thực tế tăng cường (AR) để tạo ra các bài kiểm tra và bài tập cải thiện trí nhớ sinh động hơn, giúp tăng cường hiệu quả trong việc phát hiện và theo dõi sa sút trí tuệ.
* Khả năng mở rộng và hợp tác quốc tế: Nghiên cứu khả năng hợp tác với các tổ chức y tế quốc tế để mở rộng ứng dụng ra ngoài Việt Nam, học hỏi kinh nghiệm và công nghệ tiên tiến từ các hệ thống y tế khác, và đảm bảo rằng sản phẩm của chúng ta có thể cạnh tranh trên thị trường toàn cầu.

# Tài liệu tham khảo

[1] <https://inside.java/2023/09/19/the-arrival-of-java-21/>  
[2] <https://completejavascript.com/>  
[3] <https://spring.io/projects/spring-boot>   
[4] <https://react-bootstrap.netlify.app/>

[5] <https://angular.dev/tutorials/learn-angular/>