

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN

CAO VĂN DÂN

XÂY DỰNG WEBSITE NHẬN TIN TRỰC TUYẾN NOTIP

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

HƯNG YÊN - 2024

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN

CAO VĂN DÂN

XÂY DỰNG WEBSITE NHẮN TIN TRỰC TUYẾN NOTIP

KHOA: CÔNG NGHỆ THÔNG TIN
CHUYÊN NGÀNH: KỸ THUẬT PHẦN MỀM

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC

NGƯỜI HƯỚNG DẪN
NGUYỄN HỮU ĐÔNG

HƯNG YÊN – 2024

NHẬN XÉT

Nhận xét của giảng viên hướng dẫn:

[illegible]

GIẢNG VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

LỜI CAM ĐOAN

Em xin cam đoan đồ án “Xây dựng Website nhắn tin trực tuyến Notip” là kết quả thực hiện của bản thân em dưới sự hướng dẫn của thầy Nguyễn Hữu Đông.

Những phân sử dụng tài liệu tham khảo trong đồ án đã được nêu rõ trong phần tài liệu tham khảo. Các kết quả trình bày trong đồ án và chương trình xây dựng được hoàn toàn là kết quả do bản thân em thực hiện.

Nếu vi phạm lời cam đoan này, em xin chịu hoàn toàn trách nhiệm trước khoa và nhà trường.

Hưng Yên, ngày ... tháng ... năm ...

Sinh viên

LỜI CẢM ƠN

Để có thể hoàn thành đồ án này, lời đầu tiên em xin phép gửi lời cảm ơn tới bộ môn Công nghệ Web và ứng dụng, Khoa Công nghệ thông tin – Trường Đại học Sư phạm Kỹ thuật Hưng Yên đã tạo điều kiện thuận lợi cho em thực hiện đồ án tốt nghiệp này.

Đặc biệt em xin chân thành cảm ơn thầy Nguyễn Hữu Đông đã rất tận tình hướng dẫn, chỉ bảo em trong suốt thời gian thực hiện đồ án vừa qua.

Em cũng xin chân thành cảm ơn tất cả các Thầy, các Cô trong Trường đã tận tình giảng dạy, trang bị cho em những kiến thức cần thiết, quý báu để giúp em thực hiện được đồ án này.

Mặc dù em đã có cố gắng, nhưng với trình độ còn hạn chế, trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em hi vọng sẽ nhận được những ý kiến nhận xét, góp ý của các Thầy giáo, Cô giáo về những kết quả triển khai trong đồ án.

Em xin trân trọng cảm ơn!

MỤC LỤC

NHẬN XÉT	3
DANH MỤC CÁC THUẬT NGỮ	8
DANH MỤC CÁC BẢNG	9
DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ.....	10
CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI.....	12
1.1. Lý do chọn đề tài	12
1.2. Mục tiêu của đề tài.....	13
1.2.1 Mục tiêu tổng quát	13
1.2.2 Mục tiêu cụ thể	13
1.3. Giới hạn và phạm vi của đề tài	13
1.3.1 Đối tượng nghiên cứu	13
1.3.2 Phạm vi nghiên cứu	13
1.4. Nội dung thực hiện	14
1.5. Phương pháp tiếp cận.....	14
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT.....	16
2.1. Phương pháp phát triển phần mềm hướng đối tượng	16
2.2. Phương pháp phân tích thiết kế hướng đối tượng	18
2.3. Hệ quản trị cơ sở dữ liệu MySQL	20
2.4. Công nghệ áp dụng	21
2.4.1 Tổng quan về Angular	21
2.4.2 Tổng quan ASP.NET Core	27
CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	31
3.1 Phát biểu bài toán.....	31
3.2 Phân tích và thiết kế yêu cầu phần mềm.....	32
3.2.1 Các yêu cầu chức năng	32
3.2.2 Biểu đồ lớp thực thể.....	45
3.2.3 Các yêu cầu phi chức năng	49
3.3 Thiết kế hệ thống	49

3.3.1 Thiết kế cơ sở dữ liệu	49
3.3.2 Thiết kế lớp đối tượng	53
CHƯƠNG 4: TRIỂN KHAI WEBSITE.....	70
4.1 Triển khai các chức năng cho phân hệ người dùng	70
4.1.1 Trang chủ	70
4.2 Triển khai các chức năng cho phân hệ quản trị nội dung	75
4.3 Kiểm thử và triển khai ứng dụng.....	76
4.3.1 Kiểm thử	76
4.3.2 Đóng gói ứng dụng	77
4.3.3 Triển khai ứng dụng.....	77
KẾT LUẬN	78
TÀI LIỆU THAM KHẢO.....	80

DANH MỤC CÁC THUẬT NGỮ

STT	Từ viết tắt	Cụm từ tiếng anh	Diễn giải
1	HTML	HyperText Markup Language	Ngôn ngữ Đánh dấu Siêu văn bản
2	CSS	Cascading Style Sheets	Được dùng để miêu tả cách trình bày các tài liệu viết bằng ngôn ngữ HTML và XHTML
3	JS	JavaScript	Là một ngôn ngữ lập trình thông dịch được phát triển từ các ý niệm nguyên mẫu
4	RDBMS	Relational Database Management System	Một hệ quản trị cơ sở dữ liệu quan hệ
5	MVVM	Model-View-ViewModel	Là 1 mẫu kiến trúc phần mềm
6	API	Application Programming Interface	Là cách tổ chức và thiết kế hệ thống giao tiếp giữa các thành phần phần mềm
7	HTTP	HyperText Transfer Protocol	Là một giao thức truyền tải siêu văn bản

DANH MỤC CÁC BẢNG

Bảng 1: Bảng danh sách các chức năng Quản lý hệ thống	32
Bảng 2 Các chức năng của phân hệ trang người dùng.....	40
Bảng 3: Bảng thông tin TypeChats	45
Bảng 4: Bảng thông tin ChatRoom	46
Bảng 5: Bảng thông tin Message	46
Bảng 6: Bảng thông tin User.....	47
Bảng 7: Bảng thông tin Role.....	47
Bảng 8: Bảng thông tin UserRole	48
Bảng 9 Bảng thông tin LoginUserHistorys.....	48
Bảng 10: Bảng thông tin Friends	48
Bảng 11: Bảng thông tin FriendStatuses.....	48
Bảng 12: Bảng danh sách các yêu cầu phi chức năng	49
Bảng 13: Bảng thuộc tính User	50
Bảng 14: Bảng thuộc tính Role	50
Bảng 15: Bảng thông tin UserRole	51
Bảng 16: Bảng thuộc tính Chats.....	51
Bảng 17: Bảng thuộc tính TypeChats	51
Bảng 18: Bảng thuộc tính Messages	52
Bảng 19: Bảng thuộc tính UserChats.....	52
Bảng 20: Bảng thuộc tính Friends.....	53
Bảng 21: Bảng thuộc tính FriendStatuses	53
Bảng 22: Bảng test case chức năng đăng nhập	76

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

Hình 1: Biểu đồ Use Case phân hệ quản trị hệ thống	32
Hình 2 Biểu đồ Use case phân rõ quản lý phòng chat.....	34
Hình 3: Biểu đồ Use Case phân rõ quản lý người dùng.....	36
Hình 4: Biểu đồ Use Case phân rõ quản lý nhân viên.....	38
Hình 5: Biểu đồ Use case tổng quát trang người dùng	41
Hình 6: Biểu đồ phân rõ Use case quản lý tin nhắn	42
Hình 7: Biểu đồ Use case phân rõ quản lý bạn bè.....	43
Hình 8: Biểu đồ lớp thực thể hệ thống.....	45
Hình 9: Mô hình Cơ sở dữ liệu	49
Hình 10: Biểu đồ VOPC thêm người dùng	54
Hình 11: Biểu đồ VOPC sửa người dùng	54
Hình 12: Biểu đồ VOPC Xem toàn bộ phòng chat.....	55
Hình 13: Biểu đồ VOPC Xem chi tiết phòng chat.....	55
Hình 14 Biểu đồ tuần tự cho Use Case Đăng nhập.....	56
Hình 15: Biểu đồ tuần tự thêm phòng chat	56
Hình 16: Biểu đồ tuần tự thêm người dùng vào phòng chat.....	57
Hình 17: Biểu đồ tuần tự xóa người dùng khỏi phòng chat.....	57
Hình 18: Biểu đồ tuần tự của Use case Đặt lại tên phòng chat.....	58
Hình 19: Biểu đồ tuần tự của Use Xóa phòng chat.....	58
Hình 20: Biểu đồ lớp chi tiết cho Use case Đăng nhập	59
Hình 21: Biểu đồ lớp chi tiết cho Use case Tạo phòng chat.....	59
Hình 22: Biểu đồ lớp chi tiết cho Use case Thêm người dùng vào phòng chat.....	60
Hình 23: Biểu đồ lớp chi tiết cho Use case Xóa thành viên khỏi phòng chat.....	60
Hình 24: Biểu đồ lớp chi tiết cho Use case Sửa tên phòng chat	61
Hình 25: Biểu đồ lớp chi tiết cho Use case Xóa phòng chat	61
Hình 26: Giao diện trang Đăng nhập	62
Hình 27: Giao diện trang Đăng ký	62
Hình 28: Giao diện dashboard phân hệ người dùng	63

Hình 29: Giao diện danh sách phòng chat	64
Hình 30: Giao diện tin nhắn phòng chat	65
Hình 31: Giao diện trang Lời mời kết bạn	65
Hình 32: Giao diện trang Danh sách bạn bè	66
Hình 33: Giao diện trang đăng nhập	66
Hình 34: Giao diện trang dashboard quản trị	67
Hình 35: Giao diện trang quản lý phòng chat	67
Hình 36: Giao diện thông tin chi tiết phòng chat	68
Hình 37: Giao diện quản lý khách hàng	68
Hình 39: Giao diện quản lý nhân viên	69
Hình 40: Xây dựng hiển thị Navbar sử dụng template Angular	70
Hình 41: Xây dựng hiển thị Header bằng template Angular	71
Hình 42: Xây dựng hiển thị danh sách phòng chat bằng Angular	72
Hình 43: Xây dựng hiển thị tin nhắn phòng chat bằng Angular	73
Hình 44: Các models được triển khai	73
Hình 45: Triển khai các lớp tầng Services	74
Hình 46: Triển khai các lớp tầng Controller	74
Hình 47: Trang quản lý phòng chat	75
Hình 48: Trang quản lý khách hàng	75
Hình 49: Trang quản lý nhân viên	76

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1. Lý do chọn đề tài

Hiện nay, thế giới đang chứng kiến sự phát triển vượt bậc của Internet và các ứng dụng trên Internet. Cùng với sự phát triển đó cộng với nhu cầu trao đổi và thông tin liên lạc một cách nhanh chóng và tiện lợi đã thúc đẩy sự phát triển các phần mềm để trao đổi thông tin một cách tức thì. Điển hình cho các phần mềm đó là ứng dụng “chat”. Các ứng dụng cho phép người dùng gửi và nhận các thông điệp nhanh chóng một cách trực tiếp với nhau. Vì thế những ứng dụng như “Messenger”, “Yahoo” ... ra đời. Nhưng vấn đề đặt ra là với các phần mềm “chat” như trên thì một yêu cầu gần như bắt buộc là việc phải cài đặt ứng dụng để có thể sử dụng chúng. Từ đó nảy sinh ra ý tưởng đưa các ứng dụng chat lên Web. Một câu hỏi đặt ra là tại sao lại là web và sự tiện lợi có được là gì khi đưa những ứng dụng “chat” lên web. Câu trả lời rất đơn giản: với Web bạn có thể thao tác bất cứ đâu có Internet mà không yêu cầu phải cài đặt ứng dụng. Điều đó mang lại sự thuận tiện cho người dùng và đồng thời là sự tiết kiệm tài nguyên máy tính một cách đáng kể. Thử tưởng tượng nếu bạn phải làm việc với một máy tính được kết nối Internet nhưng lại không cài sẵn một chương trình ứng dụng chat mà bạn cần cho việc trao đổi thông tin với người khác. Thì việc phải tải ứng dụng và cài đặt ứng dụng đó lên máy tính rõ ràng là rắc rối và phiền phức hơn rất nhiều so với việc dùng một trang web có cùng chức năng.

Cùng với đó, với sự phát triển bùng nổ của công nghệ web và sự hỗ trợ ngày càng mạnh của các ngôn ngữ lập trình. Việc tạo ra một trang web có khả năng hoạt động với các chức năng như một ứng dụng được cài trên máy tính là hoàn toàn có thể. Vì vậy, việc đưa một ứng dụng trên máy tính lên thành một trang web trở thành một nhu cầu thiết thực và cần thiết.

Với những lý do trên, đồ án sẽ tập trung để giải quyết vấn đề xây dựng một ứng dụng web chat với đề tài “Xây dựng website nhắn tin trực tuyến Notip”. Với mục tiêu đề tài là xây dựng một trang web có khả năng tương tự như những phần mềm chat được cài đặt trên máy tính đã có trước đây. Cùng với đó đồ án cũng tập trung hướng tới những công nghệ phổ biến hiện nay là framework Asp.net Core

API và Angular nhằm tạo một website tiện dụng với đầy đủ các chức năng của một ứng dụng web chat.

1.2. Mục tiêu của đề tài

1.2.1 Mục tiêu tổng quát

- Xây dựng được website nhắn tin trực tuyến.
- Dễ dàng gửi tin nhắn tới người khác.

1.2.2 Mục tiêu cụ thể

- Về kiến thức:
 - Hiểu và trình bày được các kiến thức về phân tích thiết kế phần mềm, ASP.NET Core API, Angular, Javascript, cơ sở dữ liệu MySQL và phân tích thiết kế hướng đối tượng
 - Áp dụng thành thạo các kiến thức để xây dựng một sản phẩm website
- Về kỹ năng:
 - Sử dụng thành thạo ASP.NET Core API, Angular, Javascript, MySQL, phân tích thiết kế phần mềm và hướng đối tượng để phát triển hai phân hệ của website: người dùng và quản trị.
 - Nâng cao kỹ năng sử dụng các công cụ hỗ trợ phát triển phần mềm.
 - Phát triển khả năng làm việc độc lập và thực hiện đầy đủ các bước trong quy trình phát triển phần mềm để xây dựng một website đáp ứng được yêu cầu thực tế.
- Về sản phẩm:
 - Hoàn thành một website nhắn tin trực tuyến Notip sử dụng ASP.NET Core API và Angular.

1.3. Giới hạn và phạm vi của đề tài

1.3.1 Đối tượng nghiên cứu

- Đối tượng nghiên cứu: Website nhắn tin
- Khách thể nghiên cứu: những người thường xuyên sử dụng web chat, messenger, zalo.

1.3.2 Phạm vi nghiên cứu

- Thu thập thông tin từ các tài liệu có liên quan đến website chat.

- Tham khảo các website trên mạng.
- Ý nghĩa khoa học: phát triển xây dựng Website nhắn tin trực tuyến một cách mạch lạc, công khai minh bạch, giúp người sử dụng có thể liên lạc với người khác một cách nhanh chóng, tiện lợi, rõ ràng, minh bạch.

1.4. Nội dung thực hiện

- Thu thập thông tin:
 - Thu thập thông tin về nhu cầu của người dùng và các website tương tự.
- Lập kế hoạch:
 - Liệt kê các chức năng của Website.
 - Lập kế hoạch thiết kế Cơ sở dữ liệu.
 - Lập kế hoạch thiết kế giao diện.
- Lập trình và thiết kế Website:
 - Thiết kế giao diện Website.
 - Lập trình các chức năng.
- Kiểm tra và chỉnh sửa:
 - Thử nghiệm các chức năng và chỉnh sửa nếu có vấn đề.
 - Triển khai ứng dụng:
 - Đưa Website vào hoạt động trên môi trường internet.
- Sao lưu và bảo trì:
 - Sao lưu Website hàng ngày và bảo trì khi cần thiết.

1.5. Phương pháp tiếp cận

Về mặt lý thuyết:

- Tìm hiểu kỹ thuật lập trình, cách thức hoạt động và phương pháp hướng đối tượng trong Angular, Web API.
- Tìm hiểu cách thức hoạt động của Client – Server.
- Tìm hiểu cách lưu dữ liệu của hệ quản trị cơ sở dữ liệu MySQL.

Về mặt lập trình:

- Thiết kế theo ngôn ngữ lập trình front-end Angular, back-end ASP.NET Core API, hệ quản trị cơ sở dữ liệu MySQL.

- Xây dựng trang web đúng theo mô tả đúng nghiệp vụ của hệ thống dựa trên các yêu cầu thực tế.

Về mặt hoạt động:

- Chương trình thực hiện đầy đủ các chức năng của một trang web nhắn tin trực tuyến tới người dùng khi truy cập.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Phương pháp phát triển phần mềm hướng đối tượng

➤ Phương pháp lập trình hướng đối tượng là gì?

Lập trình hướng đối tượng (Object-oriented programming) là một phương pháp sử dụng các đối tượng tương tác để giải quyết những nhiệm vụ phức tạp trong lập trình. Mỗi đối tượng sẽ có những thuộc tính và hành vi khác nhau.

Chẳng hạn, bạn có thể hình dung lon sữa đặc là một đối tượng. Phương pháp lập trình hướng đối tượng sẽ tạo ra những lon sữa đặc có nắp khui (thuộc tính) và chức năng tự mở nắp (hành vi). Người dùng chỉ việc yêu cầu đối tượng đó thực hiện chức năng của mình.

Hai ngôn ngữ hỗ trợ lập trình hướng đối tượng nổi tiếng là Java và C++. Ngoài ra, một số ngôn ngữ khác cũng hỗ trợ hướng đối tượng, bao gồm Objective C, Perl, Python, JavaScript, Simula, Modula, Ada, Smalltalk...

➤ Ưu điểm của phương pháp lập trình hướng đối tượng

Một số ưu điểm của lập trình hướng đối tượng bao gồm:

- Nâng cao hiệu năng phát triển phần mềm

Ba yếu tố quan trọng của lập trình hướng đối tượng là:

- **Tính mô-đun:** nó tách biệt các nhiệm vụ trong quá trình phát triển phần mềm dựa trên những đối tượng cụ thể. Mỗi đối tượng sẽ có một nhiệm vụ khác nhau.
- **Tính mở rộng:** các đối tượng có thể được mở rộng dễ dàng, bao gồm mở rộng thuộc tính và các hành vi mới.
- **Tính tái sử dụng:** các đối tượng cũng có thể được sử dụng lại trong một ứng dụng hoặc nhiều ứng dụng khác nhau.

Ba yếu tố trên của lập trình hướng đối tượng giúp hiệu năng phát triển phần mềm cũng được cải thiện rất nhiều, so với kỹ thuật lập trình truyền thống dựa trên thủ tục.

- Nâng cao khả năng bảo trì phần mềm

Chính vì những lý do nêu trên, phần mềm được lập trình theo hướng đối tượng cũng dễ bảo trì hơn. Vì thiết kế là mô-đun, nên việc thay đổi một phần của

chương trình sẽ không ảnh hưởng đến những phần còn lại. Điều này rất có lợi trong trường hợp dự án của bạn có quy mô lớn, đòi hỏi nhiều thay đổi.

- **Phần mềm phát triển nhanh hơn**

Tính tái sử dụng của lập trình hướng đối tượng cho phép phát triển phần mềm nhanh hơn. Các phần mềm được lập trình theo hướng đối tượng thường có thư viện đối tượng phong phú, các đoạn mã được tối ưu hóa và có thể tái sử dụng ở các dự án khác trong tương lai.

Lập trình hướng đối tượng giúp phát triển phần mềm nhanh hơn

- **Giảm thiểu chi phí phát triển**

Việc tái sử dụng phần mềm cũng làm giảm thiểu chi phí phát triển cho nhà sản xuất. Thông thường, phần lớn công sức chỉ tập trung vào việc phân tích đối tượng và thiết kế phần mềm. Do đó, tổng chi phí phát triển phần mềm cũng giảm đi đáng kể.

- **Chất lượng phần mềm cao hơn**

Thực tế, chất lượng của phần mềm phụ thuộc vào nhiều yếu tố khác nhau. Chẳng hạn, kinh nghiệm và trình độ của nhóm phát triển cũng sẽ ảnh hưởng đến sản phẩm đầu ra. Tuy nhiên, phương pháp này có xu hướng tạo ra những phần mềm chất lượng cao hơn.

Việc phát triển phần mềm nhanh hơn, chi phí thấp hơn giúp các nhà sản xuất dành nhiều thời gian, tài nguyên vào việc kiểm thử. Điều đó đồng nghĩa rằng phần mềm cuối cùng thường ít lỗi hơn, chất lượng tốt hơn.

- **Nhược điểm của phương pháp lập trình hướng đối tượng**

Bên cạnh những ưu điểm nêu trên, lập trình hướng đối tượng cũng có những nhược điểm như sau:

- **Đường cong học tập sâu**

Lập trình hướng đối tượng là một phương pháp đòi hỏi khá nhiều tư duy. Do đó, nó có thể không dễ dàng với một số người, đặc biệt là những người mới vào nghề. Các lập trình viên cần một khoảng thời gian để học và tập làm quen với nó.

Phương pháp này phức tạp vì phần mềm phải dựa trên sự tương tác của đối tượng. Do đó, lập trình viên cần phải hiểu bản chất của những khái niệm cơ bản

như: lớp, đối tượng, phương thức, thuộc tính. Đồng thời, ta cũng cần nắm được bốn tính chất cơ bản của lập trình hướng đối tượng. Đó là: Tính trừu tượng (Abstraction), Tính đóng gói (Encapsulation), Tính kế thừa (Inheritance) và Tính đa hình (Polymorphism).

Lập trình viên cần có nhiều thời gian để làm chủ phương pháp này

- **Chương trình chậm và có kích thước lớn hơn**

Phần mềm được lập trình theo hướng đối tượng thường chậm hơn các phần mềm dựa trên thủ tục. Lý do là vì các phần mềm này thường yêu cầu nhiều câu lệnh hơn để thực thi. Lập trình viên phải viết ra nhiều dòng mã để đảm bảo những thuộc tính của phương pháp này. Do đó, kích thước cũng chương trình cũng lớn hơn.

- **Phương pháp lập trình hướng đối tượng không phù hợp với mọi loại vấn đề.**

Mỗi phương pháp khác nhau sẽ phù hợp với một vấn đề khác nhau. Lập trình hướng đối tượng cũng vậy.

Thực tế, có những vấn đề mặc định sẽ được giải quyết tốt hơn nếu lập trình viên sử dụng phương pháp lập trình chức năng (Functional Programming), lập trình logic, hoặc lập trình thủ tục. Nếu ta áp dụng lập trình hướng đối tượng, có thể sẽ không đem lại hiệu quả tốt.

2.2. Phương pháp phân tích thiết kế hướng đối tượng

Khái niệm OOAD (Object Oriented Analysis and Design)

- Phân tích và thiết kế hướng đối tượng là một kỹ thuật tiếp cận phổ biến dùng để phân tích, thiết kế một ứng dụng, hệ thống. Nó dựa trên bộ các nguyên tắc chung, đó là một tập các hướng dẫn để giúp chúng ta tránh khỏi một thiết kế xấu. 5 nguyên tắc SOLID trong thiết kế hướng đối tượng:
- Một lớp chỉ nên có một lý do để thay đổi, tức là một lớp chỉ nên xử lý một chức năng đơn lẻ, duy nhất thôi. Nếu đặt nhiều chức năng vào trong một lớp sẽ dẫn đến sự phụ thuộc giữa các chức năng với nhau và mặc dù sau đó ta chỉ thay đổi ở một chức năng thì cũng phá vỡ các chức năng còn lại.
- Các lớp, module, chức năng nên dễ dàng Mở (Open) cho việc mở rộng (thêm chức năng mới) và Đóng (Close) cho việc thay đổi.

- Lớp dẫn xuất phải có khả năng thay thế được lớp cha của nó.
- Chương trình không nên buộc phải cài đặt một interface mà nó không sử dụng đến.
- Các module cấp cao không nên phụ thuộc vào các module cấp thấp. Cả hai nên phụ thuộc thông qua lớp trừu tượng. Lớp trừu tượng không nên phụ thuộc vào chi tiết. Chi tiết nên phụ thuộc vào trừu tượng.

Khái niệm UML

UML (Ngôn ngữ Mô hình hóa Thống nhất), là một ngôn ngữ chuẩn để mô hình hóa các hệ thống phần mềm. UML được sử dụng rộng rãi trong lĩnh vực phát triển phần mềm để giúp các nhà phát triển, kiến trúc sư và các bên liên quan hiểu rõ và thiết kế hệ thống một cách hiệu quả.

UML bao gồm nhiều loại biểu đồ khác nhau, mỗi loại phục vụ một mục đích cụ thể trong quá trình phát triển phần mềm. Các biểu đồ này có thể được chia thành ba loại chính: biểu đồ cấu trúc, biểu đồ hành vi và biểu đồ tương tác.

1. Biểu đồ cấu trúc (Structural Diagrams):

- **Biểu đồ lớp (Class Diagram):** Mô tả cấu trúc tĩnh của hệ thống bằng cách hiển thị các lớp, thuộc tính, phương thức và các mối quan hệ giữa chúng.
- **Biểu đồ đối tượng (Object Diagram):** Hiển thị các đối tượng và mối quan hệ của chúng tại một thời điểm cụ thể.
- **Biểu đồ thành phần (Component Diagram):** Mô tả các thành phần và các mối quan hệ giữa chúng.
- **Biểu đồ triển khai (Deployment Diagram):** Hiển thị cấu trúc vật lý của hệ thống, bao gồm các nút và mối quan hệ giữa chúng.

2. Biểu đồ hành vi (Behavioral Diagrams):

- **Biểu đồ use case (Use Case Diagram):** Mô tả các trường hợp sử dụng (use cases) và các tác nhân (actors) tương tác với hệ thống.
- **Biểu đồ trạng thái (State Diagram):** Hiển thị các trạng thái của một đối tượng và các sự kiện gây ra sự thay đổi trạng thái.
- **Biểu đồ hoạt động (Activity Diagram):** Mô tả luồng công việc hoặc các hoạt động trong hệ thống.

3. Biểu đồ hành vi (Behavioral Diagrams):

- **Biểu đồ use case (Use Case Diagram):** Mô tả các trường hợp sử dụng (use cases) và các tác nhân (actors) tương tác với hệ thống.
- **Biểu đồ trạng thái (State Diagram):** Hiển thị các trạng thái của một đối tượng và các sự kiện gây ra sự thay đổi trạng thái.
- **Biểu đồ hoạt động (Activity Diagram):** Mô tả luồng công việc hoặc các hoạt động trong hệ thống.

2.3. Hệ quản trị cơ sở dữ liệu MySQL

MySQL là hệ quản trị dữ liệu miễn phí, được tích hợp sử dụng chung với Apache, PHP. Chính yếu tố phát triển trong cộng đồng mã nguồn mở nên MySQL đã qua rất nhiều sự hỗ trợ của những lập trình viên yêu thích mã nguồn mở. MySQL cũng có cùng một cách truy xuất và mã lệnh tương tự với ngôn ngữ SQL. Nhưng MySQL không bao quát toàn bộ những câu truy vấn cao cấp như SQL. Về bản chất MySQL chỉ đáp ứng việc truy xuất đơn giản trong quá trình vận hành của website nhưng hầu hết có thể giải quyết các bài toán trong PHP.

MySQL là một RDBMS nhanh và dễ dàng để sử dụng. MySQL đang được sử dụng nhiều công việc kinh doanh từ lớn tới nhỏ. MySQL được phát triển, được công bố, được hỗ trợ bởi MySQL AB, là một công ty của Thụy Điển. MySQL trở thành khá phổ biến vì nhiều lý do:

- MySQL là mã nguồn mở. Vì thế, để sử dụng nó, bạn chẳng phải mất một xu nào.
- MySQL là một chương trình rất mạnh mẽ.
- MySQL sử dụng một Frm chuẩn của ngôn ngữ dữ liệu nổi tiếng là SQL.
- MySQL làm việc trên nhiều Hệ điều hành và với nhiều ngôn ngữ như PHP, PERL, C, C++, Java, ...
- MySQL làm việc nhanh và khỏe ngay cả với các tập dữ liệu lớn.
- MySQL hỗ trợ các cơ sở dữ liệu lớn, lên tới 50 triệu hàng hoặc nhiều hơn nữa trong một Bảng . Kích cỡ file mặc định được giới hạn cho một Bảng là 4 GB, nhưng bạn có thể tăng kích cỡ này (nếu hệ điều hành của bạn có thể xử lý nó) để đạt tới giới hạn lý thuyết là 8 TB. MySQL là có thể điều chỉnh.

Giấy phép GPL mã nguồn mở cho phép lập trình viên sửa đổi phần mềm MySQL để phù hợp với môi trường cụ thể của họ

2.4. Công nghệ áp dụng

2.4.1 Tổng quan về Angular

❖ Angular là gì?

Angular là một mã nguồn mở viết bằng TypeScript và được sử dụng để thiết kế giao diện web (front – end). Angular được xây dựng, phát triển từ những năm 2009 và đang duy trì cho đến nay bởi Google. Đây được xem là framework front end mạnh mẽ và chuyên dụng dành cho các lập trình viên sử dụng HTML cao cấp.

Angular được ứng dụng rộng rãi để xây dựng các project Single Page Application (ứng dụng trang đơn).

Hiện nay, Angular được các công ty lớn lựa chọn sử dụng như: Upwork, Forbes, General Motors,... Đây sẽ là cơ hội việc làm rất lớn nếu bạn sử dụng thành thạo Angular. Tuy nhiên, trước đó bạn cần nắm vững các kiến thức nền tảng về JavaScript, CSS và HTML, cách làm việc với kiến trúc Model-View-Controller

❖ Lịch sử phát triển của Angular

Phần mềm này được xây dựng, phát triển và cho ra đời vào năm 2009 bởi Misko Hevery cùng với một người bạn khác là Adam Abrons. Angular được coi là một dự án riêng cho đến khi Misko Hevery tham gia vào dự án Google Feedback với tư cách là lập trình viên bán thời gian. Trong thời gian khoảng 6 tháng, Misko cùng 2 người khác nữa đã viết lên 17.000 dòng mã khác nhau cho dự án Google Feedback.

Tuy nhiên, số lượng mã ngày càng nhiều gây phát sinh thêm vấn đề sửa lỗi kiểm soát. Misko đã mạnh dạn cá cược với quản lý rằng nếu sử dụng GetAngular thì có thể viết lại toàn bộ những mã này trong 2 tuần. Kết quả 17.000 mã đã giảm xuống 1.500 mã. Tuy thua cuộc nhưng nhờ sự kiện này mà Angular được Google phát triển nhân rộng và ngày càng phổ biến.

❖ Ưu và nhược điểm của Angular

Mỗi mã nguồn đều có những ưu – nhược điểm nhất định và Angular cũng vậy. Cụ thể sau đây là những ưu – nhược điểm của mã nguồn này mà bạn cần nắm.

- **Ưu điểm**

Angular mang đến nhiều ưu điểm nổi bật cho nhiều lập trình viên, cụ thể như:

Angular được các chuyên gia đánh giá cao, mã nguồn này giúp các Single Page Application làm việc dễ dàng, nhanh chóng.

Nhờ khả năng Binding data lên trên các nền tảng HTML nên code front-end thường rất thân thiện với người dùng.

Bạn có thể thuận tiện Unit Test.

Component có thể tái sử dụng dễ dàng hơn.

Angular có khả năng hỗ trợ cho các lập trình viên có thể viết code được ít hơn cùng với nhiều chức năng hơn. Từ đó giúp tiết kiệm thời gian lập trình và tăng hiệu suất công việc.

AngularJS tương thích với nhiều nền tảng khác nhau. Bạn có thể dùng được trên nhiều loại trình duyệt khác nhau cả trong máy tính và thiết bị điện thoại di động.

- **Nhược điểm**

Ngoài những ưu điểm nổi bật đã nêu ở trên, Angular còn tồn tại một vài nhược điểm cần được khắc phục, cụ thể như:

Tính bảo mật: Bản chất của Angular là một framework front-end. Thông thường, tính bảo mật của front-end thường không cao bằng back-end. Chính vì thế, bạn cần xây dựng một hệ thống kiểm tra dữ liệu sao cho việc trả về được tốt nhất khi sử dụng API.

Khả năng an toàn: Website có thể trở nên không an toàn nếu bạn sử dụng một số trình duyệt sở hữu tính năng Disable JavaScript.

- ❖ **Tại sao nên sử dụng Angular?**

Nếu bạn là một người mới, những lý do sau sẽ khiến bạn hối hận vì không sử dụng Angular sớm hơn:

Angular giúp nâng cao năng suất của các lập trình viên

Trong vài năm trở lại đây, vấn đề phát sinh website đã có nhiều bước tiến vượt bậc và được chú trọng hơn rất nhiều. Ở bản ECMAScript (ES) 2015, người dùng thường gọi với cái tên ES6, cùng với những class hoặc arrow function. Angular 2+ với những tính năng mới cập nhật này đã giúp việc code với Angular trở nên chi tiết và dễ dàng hơn rất nhiều.

Angular giúp nâng cao năng suất của các lập trình viên

Bên cạnh đó, với việc ứng dụng TypeScript đây còn được xem là một phiên bản nâng cấp hoặc ngôn ngữ nổi trội của JavaScript. Angular phối hợp với TypeScript tạo nên một công cụ hỗ trợ xử lý những vấn đề tồn đọng của AngularJS, có thể kể đến như kiểm tra kiểu dữ liệu, giúp refactor code an toàn hơn,... Do đó mà việc debug trở nên dễ dàng, lập trình viên cũng hiểu được chi tiết mã nguồn của họ hơn.

- **Cấu trúc phát triển rõ ràng**

Đối với các lập trình viên chú trọng ở một frameworks là cấu trúc phát triển ứng dụng của chúng, Angular lại mang cho mình một cấu trúc rất chi tiết, thông qua ba yếu tố chính như: class, các dependency được bổ sung vào và mô hình MVVM (model-view-viewmodel).

Angular dùng class trong ES6 với nhiều thuộc tính để xây dựng toàn bộ các cấu trúc quan trọng. Ví dụ bạn muốn tạo một Angular component – Tạo một class kèm thêm vào những thuộc tính thiết yếu. Còn nếu bạn muốn tạo Angular module – Hãy tạo một class và bổ sung cho chúng những thuộc tính cần thiết.

Về lý thuyết sẽ là như vậy, Angular mang đến một cấu trúc rõ ràng để phát triển từng tính năng cho ứng dụng của bạn. Những dependency vượt trội được dùng trong ứng dụng ở thời điểm cần thiết, nếu muốn tích hợp bất kì dependency nào. Ví dụ: HTTP, router, bạn chỉ cần cho nó vào bên trong constructor của class.

Ngoài ra, mô hình MVVM cũng hỗ trợ Angular rất nhiều trong phát triển ứng dụng client-side, thông thường bạn sẽ phải quan tâm đến 3 yếu tố chính: giao diện người dùng, mã nguồn điều khiển giao diện, mô hình dữ liệu (data) dành cho giao diện.

Angular với MVVM xác định rõ ràng các yếu tố trên nhờ vào mô hình MVVM:

Phần giao diện (view) được định nghĩa ở một template gồm cả HTML dành cho một component nhất định. Template ở đây gồm tất cả layout hay bất cứ yếu tố nào nằm trong layout đó.

Các thuộc tính của component class được gọi là model. Hiểu đơn giản là dữ liệu, theo đó để phần view dùng để thực thi.

View/model là class giám sát cả view tương tự model. Là phần code sẽ tiến hành truy xuất dữ liệu, bên cạnh đó cũng đảm bảo các tương tác của người dùng trên view.

Chính vì ứng dụng những tích cực của các thành phần trên, Angular làm cho vấn đề phát triển ứng dụng trở nên tối ưu và thuận tiện hơn.

- **Extensive binding**

Đa số các ứng dụng web làm việc với dữ liệu (data). Application đảm nhiệm truy xuất dữ liệu từ server, sau đó sẽ hiển thị dữ liệu đó lên view cho người dùng và sử dụng template. Khi người dùng tương tác thì những dữ liệu đó sẽ được thay đổi, view sẽ xác minh và lưu lại ở server. Data binding trong Angular hỗ trợ người dùng tiến hành quá trình trên rất đơn giản.

Đơn thuần từ việc liên đới yếu tố HTML trong template cùng những tính năng trong class và dữ liệu sẽ tự động cập nhật trên màn hình. Đối với những tương tác đòi hỏi phải thay đổi dữ liệu của người dùng, Angular sẽ áp dụng cách two-way binding. Khi đó những thay đổi xuất phát ở view sẽ tự động cập nhật thuộc tính model nằm ở class.

Ngoài ra, Angular cũng giúp property binding cho phép người dùng điều khiển DOM thông qua ràng buộc thuộc tính HTML với thuộc tính của component class, đến lúc này data sẽ tự động cập nhật bên trong view.

Cuối cùng, Angular cũng hỗ trợ event binding, có nghĩa là người dùng có thể xử lý bất kì event nào tại view, giống HTML event. Cơ bản bạn sẽ phải gắn event và method vào trong class. Những lúc event xuất hiện, method tương thích sẽ tiến

hành. Extensive binding giúp đỡ thiệu được hiển thị, quản lý DOM, thực thi các event một cách mượt mà và nhanh chóng.

- **Hỗ trợ đầy đủ tính năng điều hướng**

Thông thường những ứng dụng web không có duy nhất một view, một page, mà sẽ có nhiều view khác nhau được cung cấp với những chức năng tương đồng. Điển hình như: website chứa các trang giới thiệu, hướng dẫn, thông tin,... view cần hiển thị đúng nơi đúng thời điểm. Đây cũng chính là mục đích mà routing hướng tới.

Những tính năng này được Angular cung cấp đầy đủ, router được kích hoạt dựa theo tương tác của người dùng (user).

Bạn có thể bổ sung dữ liệu đến những router, từ đó view sẽ hiển thị nội dung theo cách dynamic và bảo vệ router, điều này làm cho người dùng chỉ có thể truy cập sau khi đăng nhập hoặc được cấp quyền truy cập. Đồng thời, chúng cũng giúp ngăn việc người dùng rời khỏi trang sau loạt thao tác chưa hoàn thành và được phép rời đi khi họ đã thực hiện xác nhận,...

Ngoài ra, Angular cũng giúp child-router cho việc điều hướng trong một router. Hành động điều hướng bên trong ứng dụng Angular này được đánh giá là rất hiệu quả và linh hoạt.

Angular giúp giảm tối đa kích thước và tăng tối đa hiệu suất của ứng dụng

Kích thước cùng hiệu năng có mối quan hệ rất chặt chẽ và quan trọng khi bạn làm việc trên nền tảng website. Với component nhỏ sẽ làm tăng tốc độ việc khởi động – thời gian download, compile trên trình duyệt cũng được giảm đi. Giảm kích thước component cùng tăng hiệu suất là một điểm nổi bật cũng như mục tiêu mà Angular muốn hướng tới cho các lập trình viên.

Angular giúp giảm tối đa kích thước và tăng tối đa hiệu suất của ứng dụng

Giảm kích thước ứng dụng có thể thực hiện bằng nhiều phương pháp khác nhau. Bạn có thể thực hiện thu nhỏ kích thước tối đa của từng component đến mức tối thiểu được cho phép. Sau đó các component đưa vào theo trình tự bên trong Angular module thông qua việc để cho các nhóm logic có mối quan hệ với nhau sẽ được download đồng thời. Cuối cùng, lazy loading nằm trong các route thực hiện

download những module được dùng trong việc hiển thị nội dung mà người dùng yêu cầu.

Tại đây sẽ có một trình biên dịch với tên là AOT, chúng sẽ vận hành một lần trong thời gian build ứng dụng. Ngay lần đầu tiên, phiên bản chưa được biên dịch của ứng dụng và render ứng dụng tới người dùng bởi trình duyệt.

- **Tài liệu và cộng đồng (community)**

Tài liệu cho Angular 2+ vô cùng rõ ràng và đầy đủ, có tất cả giới thiệu từ cơ bản đến nâng cao, hỗ trợ người dùng làm quen với Angular. Bên cạnh đó, chúng cũng có cả Tutorial Basic được phát triển bởi đội ngũ Angular, giúp người dùng hiểu được các thuộc tính cơ bản của framework.

Ngoài ra, Angular được phát triển bởi Google nên có một cộng đồng người dùng vô cùng lớn làm cho Angular không ngừng phát triển và lớn mạnh.

- **Các đặc trưng của Angular là gì?**

Angular có các đặc trưng nổi bật sau đây:

Angular có khả năng tạo ra các ứng dụng client-side dựa trên mô hình Model-View-Controller (MVC).

Angular được các lập trình viên sử dụng để có thể phát triển dựa trên JavaScript.

Các mã JavaScript có thể dễ dàng tự động xử lý sao cho phù hợp với các trình duyệt nhất nhờ khả năng tương thích cao của Angular.

Angular được sử dụng rộng rãi khi có mã nguồn mở và miễn phí.

- ❖ **Angular hoạt động như thế nào?**

Sau khi trang được nhúng với AngularJS, mã HTML được phân tích cú pháp và hiển thị. Mã HTML này sẽ chứa một thẻ có thuộc tính `ng-app=""`. Thuộc tính này sau đó sẽ được sử dụng để bắt đầu khởi tạo ứng dụng AngularJS. Thẻ tiếp theo có thuộc tính `ng-model="name"` tạo biến `name` trong ứng dụng AngularJS ở trên. Từ đó trở đi, giá trị của biến luôn bằng giá trị của trường cuối cùng với thẻ thứ 2 của thuộc tính. Chúng được sử dụng bất cứ khi nào ứng dụng có thể phát hiện ra sự thay đổi giá trị bên trong tên biến và nối giá trị vào nội dung HTML và đặt chúng vào thẻ thứ hai.

❖ Những tính năng của Angular

Phần mềm này sở hữu những tính năng nổi bật, là công cụ dùng để phát triển giao diện web tuyệt vời:

Controller: Tính năng hỗ trợ xử lý dữ liệu cho đối tượng \$scope. Với tính năng này, bên view sẽ dùng các dữ liệu có sẵn tại scope để tiến hành hiển thị tương ứng.

Data-binding: Tính năng tự động đồng bộ dữ liệu giữa hai chiều view và model khi view có thay đổi.

Service: Được coi là singleton object dùng để cung cấp các phương án dữ liệu có sẵn như: (\$http, \$controller, \$sce, \$compile, \$document, \$parse, \$httpBackend,...) được khởi tạo 1 lần duy nhất.

Scope: Là đối tượng có nhiệm vụ giao tiếp giữa hai phía controller và view trong ứng dụng.

Filter: Có khả năng lọc tập hợp con có trong item ở các mảng và trả nhanh về các mảng mới.

Directive: Được sử dụng để tạo ra các thẻ HTML riêng và thường sở hữu các directive sẵn như: ngModel, ngBind,...

Deeplink: Liên kết sâu này hỗ trợ lập trình viên trong việc mã hóa trạng thái của ứng dụng URL và có thể bookmark với nhiều công cụ tìm kiếm khác. Hầu hết có thể phục hồi lại từ những địa chỉ URL có cùng trạng thái từ ứng dụng này.

Dependency Injection: Thường được tích hợp trong phần mềm AngularJS hỗ trợ tạo lập những ứng dụng có nhiều tiềm năng phát triển, dễ hiểu và kiểm tra.

2.4.2 Tổng quan ASP.NET Core

❖ ASP.NET Core là gì?

ASP.NET Core là một web framework mã nguồn và được tối ưu hóa cho cloud để phát triển các ứng dụng web chạy trên nhiều nền tảng như Windows, Linux và Mac. Hiện tại, nó bao gồm MVC framework được kết hợp các tính năng của MVC và Web API thành một web framework duy nhất.

- Các ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc trên .NET Framework hoàn chỉnh.

- Nó đã được thiết kế để cung cấp một framework tối ưu cho các ứng dụng để triển khai tới cloud hoặc chạy on-premises.
- Nó bao gồm những modular với các thành phần tối thiểu, do đó bạn giữ được tính linh hoạt trong quá trình xây dựng các giải pháp của mình.
- Bạn có thể phát triển và chạy các ứng dụng đa nền tảng từ ASP.NET Core trên Windows, Mac và Linux.

❖ Ưu điểm của ASP.NET Core

ASP.NET Core đi kèm với những ưu điểm sau:

- ASP.NET Core có một số thay đổi kiến trúc dẫn đến modular framework nhỏ hơn.
- ASP.NET Core không còn dựa trên System.Web.dll. Nó dựa trên một tập hợp nhiều yếu tố của Nuget packages.
- Điều này cho phép bạn tối ưu ứng dụng của mình chỉ cần những NuGet packages cần thiết.
- Lợi ích của diện tích bề mặt ứng dụng nhỏ hơn thì bảo mật chặt chẽ hơn, giảm dịch vụ, cải thiện hiệu suất và giảm chi phí.

Với ASP.NET Core, bạn có thể nhận được các cải tiến sau:

- Xây dựng và chạy các ứng dụng ASP.NET Core đa nền tảng trên Windows, Mac và Linux.
- Được xây dựng trên **.NET Core**, hỗ trợ side-by-side app versioning.
- Công cụ mới giúp đơn giản hóa việc phát triển web hiện đại.
- Liên kết đơn các web stack như Web UI và API Web.
- Cấu hình dựa trên môi trường đám mây sẵn có.
- Được xây dựng dựa trên cho DI (Dependency Injection).
- Tag Helpers làm cho các Razor makup trở nên tự nhiên hơn với HTML.
- Có khả năng host trên IIS hoặc self-host.

2.4.3 Tổng quan Websocket

❖ Websocket là gì?

WebSoket là công nghệ hỗ trợ giao tiếp hai chiều giữa client và server bằng cách sử dụng một TCP socket để tạo một kết nối hiệu quả và ít tốn kém. Mặc dù được thiết kế để chuyên sử dụng cho các ứng dụng web, lập trình viên vẫn có thể đưa chúng vào bất kì loại ứng dụng nào.

- WebSockets mới xuất hiện trong HTML5, là một kỹ thuật Reverse Ajax. WebSockets cho phép các kênh giao tiếp song song hai chiều và hiện đã được hỗ trợ trong nhiều trình duyệt (Firefox, Google Chrome và Safari). Kết nối được mở thông qua một HTTP request (yêu cầu HTTP), được gọi là liên kết WebSockets với những header đặc biệt. Kết nối được duy trì để bạn có thể viết và nhận dữ liệu bằng JavaScript như khi bạn đang sử dụng một TCP socket đơn thuần.
- Dữ liệu truyền tải thông qua giao thức HTTP (thường dùng với kỹ thuật Ajax) chứa nhiều dữ liệu không cần thiết trong phần header. Một header request/response của HTTP có kích thước khoảng 871 byte, trong khi với WebSocket, kích thước này chỉ là 2 byte (sau khi đã kết nối). Vậy giả sử bạn làm một ứng dụng game có thể tới 10,000 người chơi đăng nhập cùng lúc, và mỗi giây họ sẽ gửi/nhận dữ liệu từ server. Hãy so sánh lượng dữ liệu header mà giao thức HTTP và WebSocket trong mỗi giây:
 - HTTP: $871 \times 10,000 = 8,710,000$ bytes = 69,680,000 bits per second (66 Mbps)
 - WebSocket: $2 \times 10,000 = 20,000$ bytes = 160,000 bits per second (0.153 Kbps) Như bạn thấy chỉ riêng phần header thôi cũng đã chiếm một phần lưu lượng đáng kể với giao thức HTTP truyền thống.

❖ Ưu điểm

- WebSockets cung cấp khả năng giao tiếp hai chiều mạnh mẽ, có độ trễ thấp và dễ xử lý lỗi. Không cần phải có nhiều kết nối như phương pháp Comet long-polling và cũng không có những nhược điểm như Comet streaming.
- API cũng rất dễ sử dụng trực tiếp mà không cần bất kỳ các tầng bổ sung nào, so với Comet, thường đòi hỏi một thư viện tốt để xử lý kết nối lại, thời gian

chờ timeout, các Ajax request (yêu cầu Ajax), các tin báo nhận và các dạng truyền tải tùy chọn khác nhau (Ajax long-polling và jsonp polling).

❖ **Nhược điểm**

Những nhược điểm của WebSockets gồm có:

- Nó là một đặc tả mới của HTML5, nên nó vẫn chưa được tất cả các trình duyệt hỗ trợ.
- Không có phạm vi yêu cầu nào. Do WebSocket là một TCP socket chứ không phải là HTTP request, nên không dễ sử dụng các dịch vụ có phạm vi yêu cầu, như SessionInViewFilter của Hibernate. Hibernate là một framework kinh điển cung cấp một bộ lọc xung quanh một HTTP request. Khi bắt đầu một request, nó sẽ thiết lập một contest (chứa các transaction và liên kết JDBC) được ràng buộc với luồng request. Khi request đó kết thúc, bộ lọc hủy bỏ contest này.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1 Phát biểu bài toán

Để đáp ứng yêu cầu người dùng, hệ thống website quản lý phòng chat online bao gồm 2 phân hệ chính: phân hệ cho người quản trị đến quản trị nội dung và quản trị hoạt động người dùng; phân hệ cho người dùng để cho khách hàng có thể nhắn tin trực tuyến. Các yêu cầu chi tiết của hai phân hệ như sau:

Phân hệ quản trị:

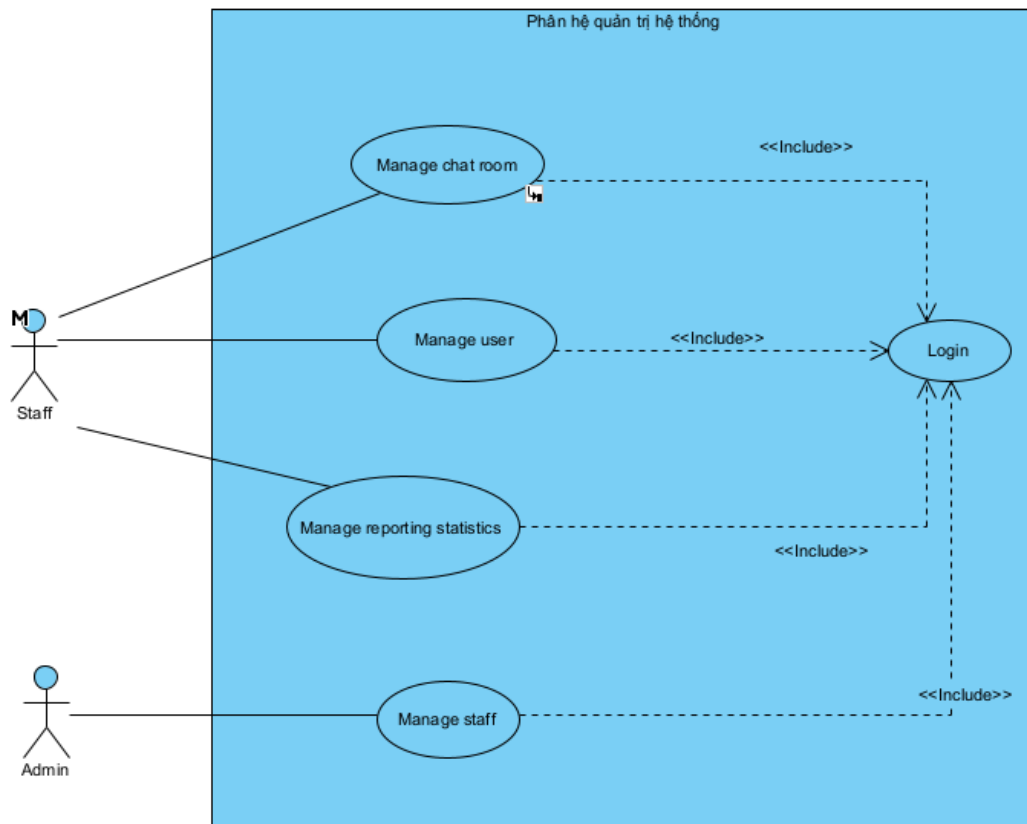
1. **Quản lý tài khoản:** Cho phép quản trị viên tạo, chỉnh sửa và khóa tài khoản người dùng.
2. **Quản lý nội dung:** Cung cấp giao diện để quản lý nội dung trên trang web, bao gồm tin nhắn, hình ảnh và các tài liệu khác.
3. **Quản lý phòng chat, nhóm chat:** Cho phép tạo, chỉnh sửa và xóa các phòng chat hoặc nhóm chat. Quản lý người dùng trong các phòng chat này.

Phân hệ người dùng:

1. **Đăng ký và đăng nhập:** Cho phép người dùng đăng ký tài khoản và đăng nhập để sử dụng dịch vụ nhắn tin trực tuyến
2. **Tìm kiếm và kết nối với người dùng khác:** Tích hợp chức năng tìm kiếm người dùng khác và gửi lời mời kết bạn hoặc thêm vào các nhóm chat.
3. **Trò chuyện và gửi tin nhắn:** Cho phép gửi tin nhắn văn bản, hình ảnh và tài liệu. Cung cấp tính năng nhắn tin riêng tư và trong nhóm.
4. **Bảo mật và quyền riêng tư:** Đảm bảo tính riêng tư của người dùng và cung cấp các cài đặt bảo mật.
5. **Giao diện thân thiện:** Tạo một giao diện thân thiện, dễ sử dụng và thích hợp cho cả ứng dụng trên web.

3.2 Phân tích và thiết kế yêu cầu phần mềm

3.2.1 Các yêu cầu chức năng



Hình 1: Biểu đồ Use Case phân hệ quản trị hệ thống

a) Chức năng của phân hệ quản trị hệ thống

Bảng 1: Bảng danh sách các chức năng Quản lý hệ thống

STT	Tên chức năng	Mô tả
I	Quản lý chat room	
1	Xem danh sách phòng chat.	Hiển thị ra danh sách phòng chat
2	Xem chi tiết phòng chat.	Hiển thị thông tin chi tiết phòng chat
3	Thêm người dùng vào phòng chat	Thêm người dùng vào phòng chat.
4	Xóa người dùng ra khỏi	Xóa người dùng ra khỏi phòng chat

	phòng chat	
5	Xóa phòng chat	Xóa phòng chat khỏi hệ thống
II	Quản lý người dùng	
1	Hiển thị thông tin người dùng	Hiển thị thông tin của người dùng.
2	Cập nhật thông tin người dùng	Sửa lại thông tin người dùng trong hệ thống.
3	Xóa người dùng.	Xóa người dùng khỏi hệ thống.
4	Tìm kiếm người dùng.	Tìm kiếm người dùng theo tên, email, số điện thoại
III	Quản lý nhân viên	
1	Hiển thị thông tin nhân viên.	Hiển thị thông tin của nhân viên.
2	Thêm nhân viên mới .	Thêm nhân viên mới vào hệ thống.
3	Cập nhật thông tin nhân viên.	Sửa lại thông tin nhân viên trong hệ thống.
4	Xóa nhân viên.	Xóa nhân viên khỏi hệ thống.
5	Tìm kiếm nhân viên.	Tìm kiếm nhân viên

- Use case quản lý phòng chat



Hình 1 Biểu đồ Use case phân rã quản lý phòng chat

❖ Luồng sự kiện quản lý phòng chat

- Xóa thành viên khỏi phòng chat:

1. Yêu cầu chức năng xóa thành viên khỏi phòng chat
2. **SYSTEM** Hiển thị giao diện danh sách thành viên của phòng chat
3. Người dùng chọn những thành viên muốn xóa khỏi phòng chat
4. Xác nhận xóa thành viên khỏi phòng chat
- 4.a. Hủy thao tác xóa thành viên khỏi phòng chat
 1. Hệ thống đóng giao diện chức năng xóa thành viên và trở lại giao diện phòng chat
5. **SYSTEM** Cập nhật thông tin phòng chat sau khi xóa người dùng
6. Chuyển người dùng trở lại giao diện phòng chat
7. Yêu cầu chức năng xóa thành viên khỏi phòng chat

- Thêm người dùng vào phòng chat

1. Người dùng yêu cầu chức năng Thêm người dùng vào phòng chat
2. **SYSTEM** Hiển thị giao diện danh sách bạn bè
3. Người dùng chọn những bạn bè muốn thêm vào phòng chat
4. Xác nhận thêm bạn bè vào phòng chat

4.a. Hủy thao tác thêm người dùng vào phòng chat

1. Hệ thống đóng giao diện chức năng thêm người dùng vào phòng chat và trở lại giao diện phòng chat

5. **SYSTEM** Cập nhật thông tin người dùng mới được thêm vào phòng chat

- Xóa phòng chat

1. Người dùng chọn phòng chat cần xóa
2. **SYSTEM** Hiển thị thông báo xác nhận xóa phòng chat
3. Người dùng lựa chọn xác nhận xóa phòng chat

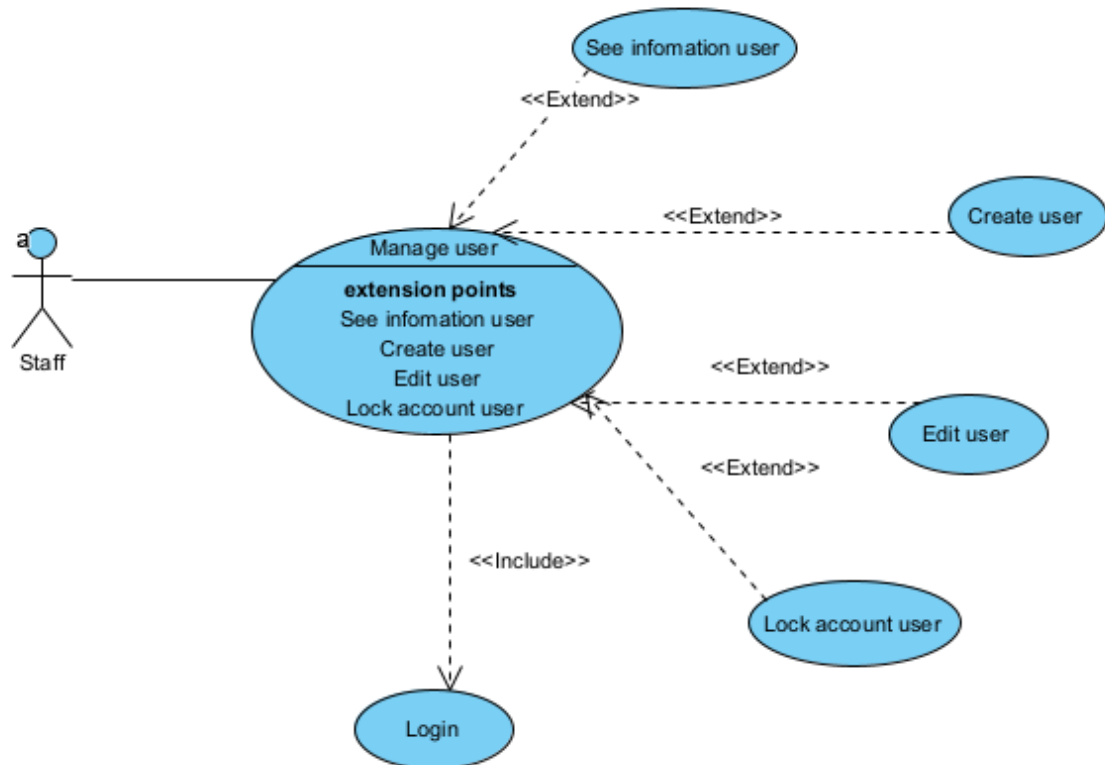
3.1. **if** Xác nhận xóa phòng chat

- 3.1.1. **SYSTEM** Xóa phòng chat đã chọn và thông báo thao tác thành công
end if

3.2. **if** Hủy thao tác xóa thông tin phòng chat

- 3.2.1. **SYSTEM** Đóng hộp thoại yêu cầu xác nhận xóa phòng chat
end if

- Use case quản lý khách hàng



Hình 3: Biểu đồ Use Case phân rã quản lý người dùng

❖ Luồng sự kiện quản lý người dùng

- Thêm người dùng

1. Yêu cầu chức năng tạo mới người dùng
2. Nhập thông tin người dùng muốn tạo
3. **SYSTEM** Kiểm tra tính hợp lệ thông tin nhập vào

3.a. Dữ liệu nhập vào không hợp lệ

1. **SYSTEM** Hiển thị thông báo và quay lại bước 2

4. **SYSTEM** Lưu thông tin người dùng mới vào hệ thống
5. **SYSTEM** Hiển thị lại danh sách người dùng mới

- Sửa thông tin người dùng

1. Yêu cầu chức năng sửa người dùng
2. **SYSTEM** Hiển thị giao diện sửa thông tin người dùng và hiển thị thông tin người

dùng đó

3. Nhân viên thực hiện sửa thông tin người dùng

4. **SYSTEM** Kiểm tra tính hợp lệ của thông tin nhập vào

4.a. Thông tin người dùng không hợp lệ

1. **SYSTEM** Hệ thống thông báo lỗi và quay trở lại bước 3

5. Xác nhận sửa thông tin người dùng

5.a. Hủy thao tác sửa thông tin người dùng

1. **SYSTEM** Hệ thống đóng giao diện chức năng sửa thông tin người dùng và trở lại giao diện chức năng quản lý người dùng

6. **SYSTEM** Cập nhật thông tin người dùng mới vào trong hệ thống

7. **SYSTEM** Trở về trang quản lý người dùng và lấy danh sách các người dùng hiển thị lên giao diện

- Khóa tài khoản người dùng

1. Nhân viên chọn người dùng cần khóa tài khoản

2. Hiện thị hộp thoại thông báo xác nhận khóa tài khoản người dùng

3. Nhân viên chọn xác nhận khóa tài khoản người dùng

3.1. **if** Xác nhận khóa tài khoản người dùng

3.1.1. Khóa tài khoản người dùng đã chọn và thông báo thành công

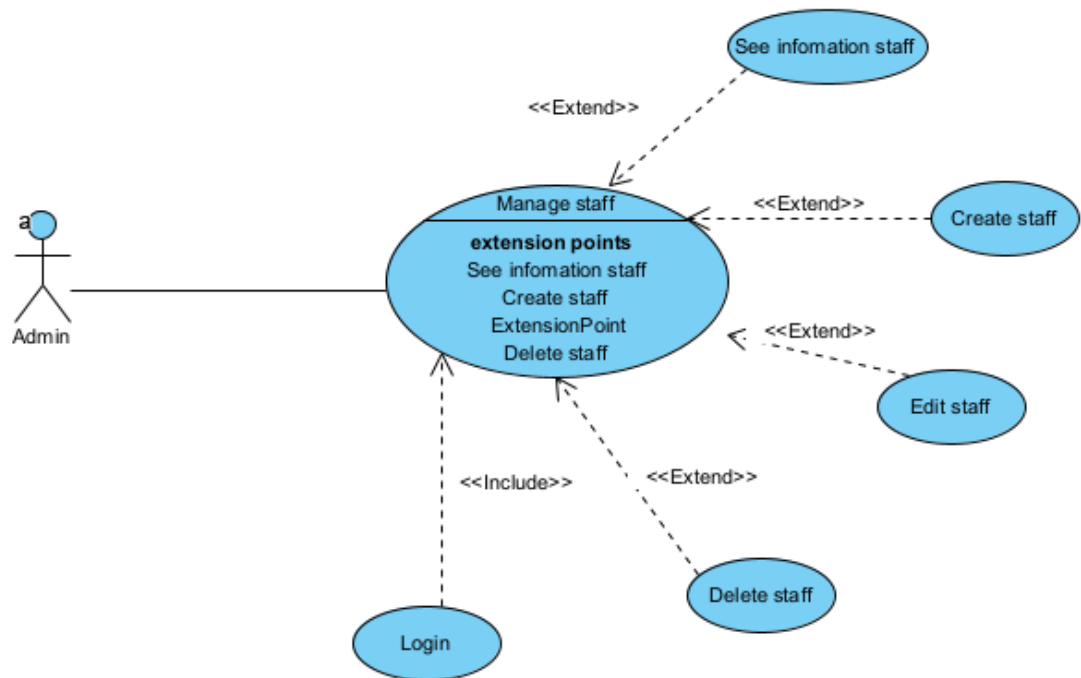
end if

3.2. **if** Hủy thao tác khóa tài khoản người dùng

3.2.1. Đóng hộp thoại yêu cầu xác nhận khóa tài khoản khách hàng

end if

- Use case quản lý nhân viên



Hình 4: Biểu đồ Use Case phân rã quản lý nhân viên

- ❖ Luồng sự kiện quản lý nhân viên

- Tạo mới nhân viên

1. Yêu cầu chức năng tạo mới nhân viên
2. Nhập thông tin nhân viên muốn tạo
3. **SYSTEM** Kiểm tra tính hợp lệ thông tin nhập vào
 - 3.1. **if** Dữ liệu nhập vào không hợp lệ
 - 3.1.1. Hiện thị thông báo và quay lại bước 2
 - end if**
4. Lưu thông tin nhân viên mới vào hệ thống
5. Hiện thị lại danh sách nhân viên

- Sửa thông tin nhân viên

1. Chọn nhân viên cần sửa
2. **SYSTEM** Hiện thị giao diện sửa thông tin nhân viên và hiện thị thông tin nhân viên đó

3. Admin nhập thông tin nhân viên cần sửa

4. **SYSTEM** Kiểm tra tính hợp lệ thông tin nhập vào

4.a. Thông tin nhân viên không hợp lệ

1. **SYSTEM** Hệ thống thông báo lỗi và quay trở lại bước 3

5. Xác nhận sửa thông tin nhân viên

5.a. Hủy thao tác sửa thông tin nhân viên

1. **SYSTEM** Hệ thống đóng giao diện chức năng sửa thông tin nhân viên và trở lại giao

diện chức năng quản lý nhân viên

6. **SYSTEM** Cập nhật thông tin nhân viên mới vào trong hệ thống

7. **SYSTEM** Trở về trang quản lý nhân viên và lấy danh sách các nhân viên hiển thị lên giao diện

- Xóa nhân viên

1. Chọn nhân viên cần xóa

2. Hiện thị hộp thoại thông báo xác nhận xóa nhân viên

3. Admin chọn xác nhận xóa nhân viên

3.1. **if** Xác nhận xóa nhân viên

3.1.1. Xóa nhân viên đã chọn và thông báo thành công

end if

3.2. **if** Hủy thao tác xóa nhân viên

3.2.1. Đóng hộp thoại yêu cầu xác nhận xóa nhân viên

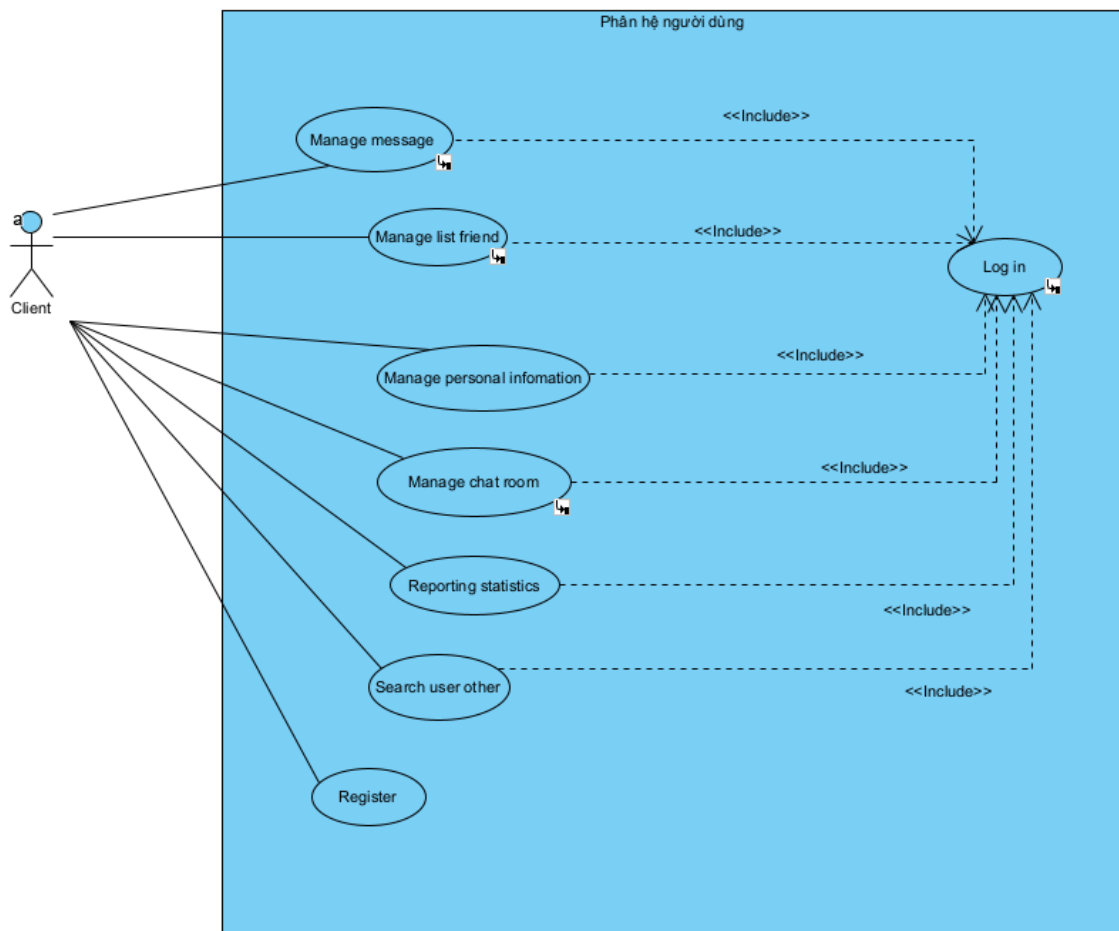
end if

b) Chức năng của phân hệ người dùng

Bảng 1 Các chức năng của phân hệ trang người dùng

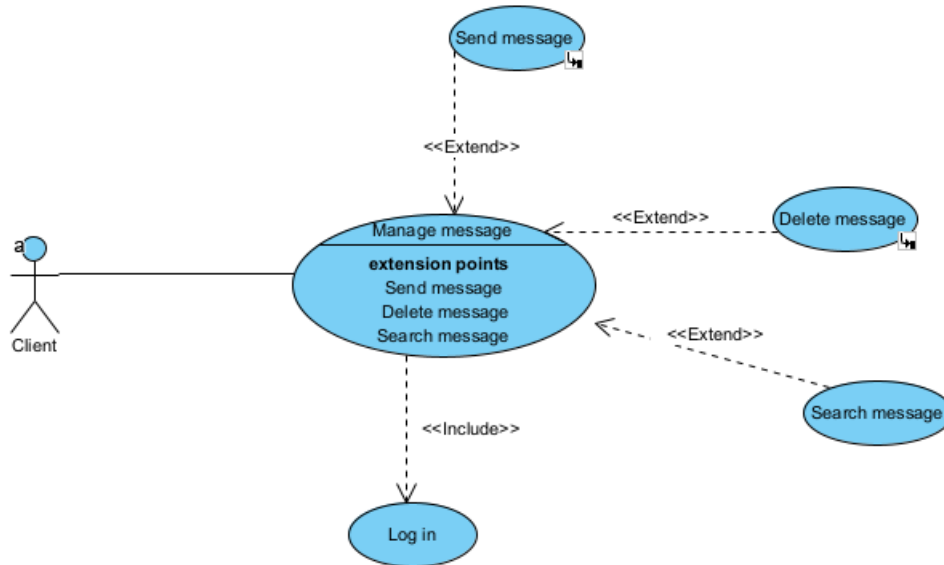
STT	Chức năng	Mô tả
1	Đăng ký	- Tạo tài khoản
2	Đăng nhập	- Đăng nhập vào hệ thống
3	Tạo phòng chat	- Tạo phòng chat riêng tư hoặc chat nhóm
4	Gửi tin nhắn	- Gửi tin nhắn đến phòng chat chỉ định
5	Quản lý tài khoản	- Sửa thông tin tài khoản - Thay đổi ảnh đại diện
6	Quản lý bạn bè	- Gửi yêu cầu kết bạn - Xác nhận kết bạn - Xóa bạn bè - Tìm kiếm người dùng
7	Quản lý phòng chat	- Đổi tên phòng chat - Đổi ảnh đại diện phòng chat - Thêm người dùng vào phòng chat - Rời phòng chat

- Use case tổng quát phân hệ người dùng



Hình 5: Biểu đồ Use case tổng quát trang người dùng

- Use case quản lý tin nhắn



Hình 6: Biểu đồ phân rã Use case quản lý tin nhắn

❖ Luồng sự kiện quản lý tin nhắn

- Gửi tin nhắn

1. Người dùng chọn phòng chat, bạn bè muốn nhắn tin
2. **SYSTEM** Hiển thị phòng chat và những tin đã nhắn trong phòng chat đó
3. Người dùng soạn tin nhắn
4. Người dùng gửi tin nhắn
5. **SYSTEM** Tin nhắn được lưu vào hệ thống và hiển thị trong phòng chat đó

- Xóa tin nhắn:

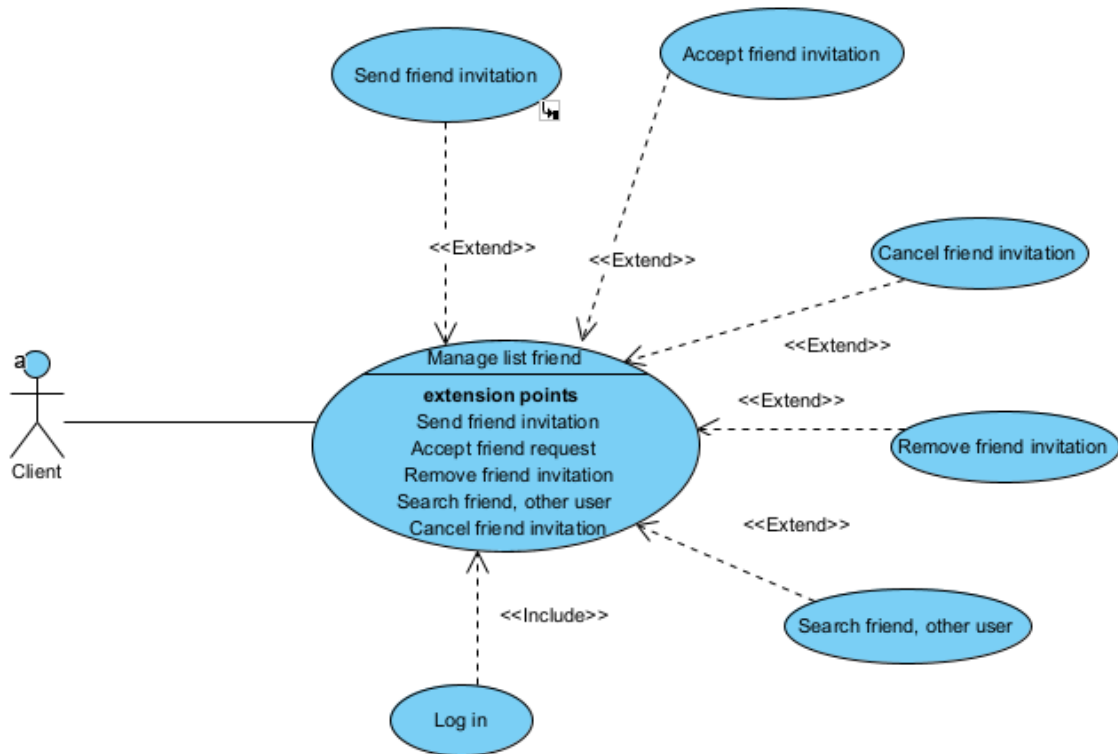
1. Người dùng click vào icon thùng rác của tin nhắn muốn xóa
2. Hiển thị thông báo xác nhận xóa tin nhắn
3. Người dùng đồng ý xóa tin nhắn
 - 3.1. **if** Người xóa tin nhắn là người gửi tin nhắn đó
 - 3.1.1. Tin nhắn bị xóa trong hệ thống**end if**
 - 3.2. **SYSTEM**

3.3. if Người xóa tin nhắn không phải là người gửi tin nhắn đó

3.3.1. Tin nhắn bị ẩn trong chat room người xóa, những người khác trong chat room vẫn được hiển thị

end if

- Use case quản lý danh sách bạn bè



Hình 7: Biểu đồ Use case phân rã quản lý bạn bè

❖ Luồng sự kiện quản lý bạn bè

- Gửi lời mời kết bạn

1. Chọn người dùng muốn gửi lời mời kết bạn
2. Gửi yêu cầu kết bạn tới người dùng đó
3. **SYSTEM** Lưu lời mời kết bạn đó vào hệ thống

- Hủy lời mời kết bạn

1. Chọn người dùng muốn hủy lời mời kết bạn
2. Xóa lời mời kết bạn

3. **SYSTEM** Cập nhật lại hệ thống

- Chấp nhận lời mời kết bạn

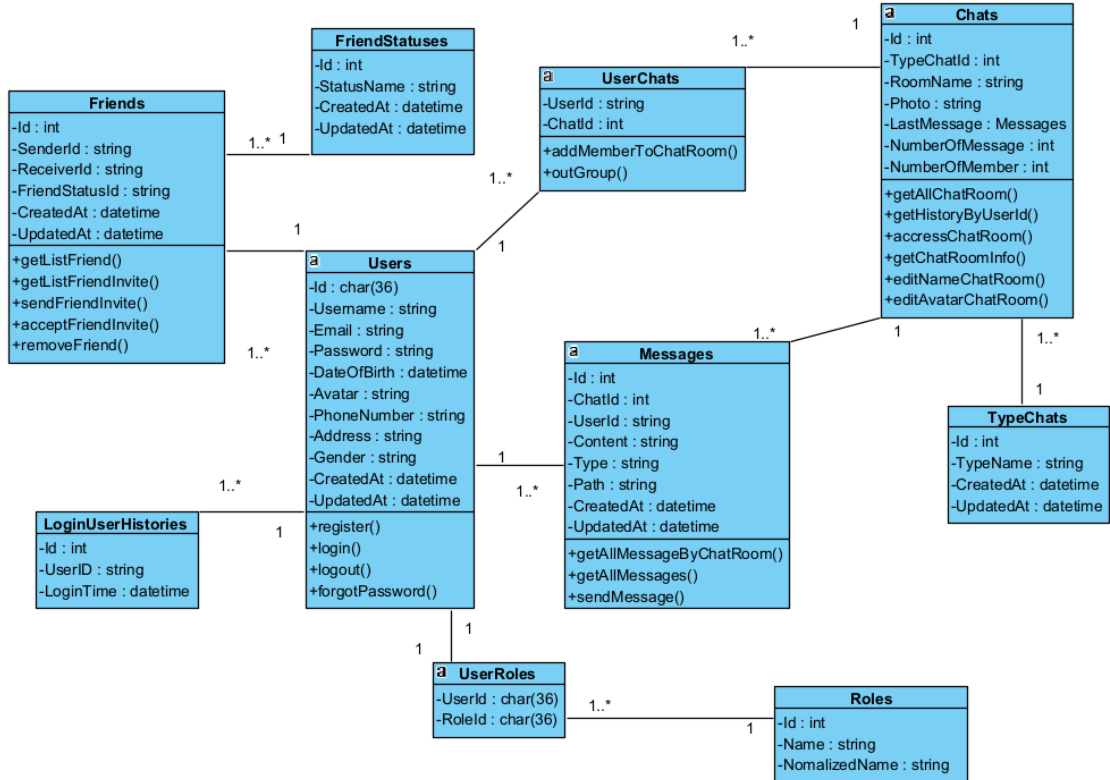
1. Chọn người dùng đã gửi lời mời kết bạn
2. Chấp nhận yêu cầu kết bạn đó
3. **SYSTEM** Thay đổi trạng thái yêu cầu kết bạn thành bạn bè
4. **SYSTEM** Cập nhật vào hệ thống

- Tìm kiếm bạn bè, những người dùng khác

1. Yêu cầu chức năng tìm kiếm người dùng, bạn bè
2. Nhập tên người dùng, bạn bè trong ô input tìm kiếm
3. **SYSTEM** Kiểm tra tính hợp lệ dữ liệu nhập
 - 3.a. Dữ liệu nhập không hợp lệ
 1. **SYSTEM** Hiển thị thông báo tìm kiếm không thành công, quay lại bước 2
4. Tìm kiếm tên người dùng theo dữ liệu nhập
5. **SYSTEM** Hiển thị danh sách người dùng có tên trùng hoặc có chứa ký tự trùng với dữ liệu nhập

3.2.2 Biểu đồ lớp thực thể

🚦 Biểu đồ lớp thực thể hệ thống



Hình 8: Biểu đồ lớp thực thể hệ thống

🚦 Mô tả các thực thể:

❖ **TypeChats:**

Bảng 2: Bảng thông tin TypeChats

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi loại phòng chat sẽ có một mã khác nhau
2	TypeName	Tên loại phòng chat
3	CreatedAt	Ngày tạo
4	UpdatedAt	Ngày cập nhật

❖ **ChatRoom:**

Bảng 4: Bảng thông tin ChatRoom

STT	Tên thuộc tính	Mô tả
1	Id	Mã phòng chat
2	RoomTypeId	Xác định kiểu phòng chat
3	RoomName	Tên của phòng chat
4	LastestMessage	Lấy tin nhắn cuối cùng của phòng chat
5	NumberOfMessage	Xác định số lượng tin đã nhắn của phòng chat
6	NumberOfMember	Xác định số lượng thành viên của phòng chat

❖ **Message:**

Bảng 5: Bảng thông tin Message

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi tin nhắn sẽ có một mã khác nhau
2	RoomId	Xác định tin nhắn thuộc phòng chat nào
3	SenderId	Mã người gửi tin nhắn
4	Content	Nội dung tin nhắn
5	Path	Đường dẫn file đính kèm (nếu có)
8	CreatedAt	Thời gian tin nhắn được gửi

❖ **User:**

Bảng 6: Bảng thông tin User

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi người dùng sẽ có một mã khác nhau
2	Name	Tên người dùng
3	Email	Email người dùng
4	Password	Mật khẩu người dùng
5	DateOfBirth	Ngày sinh của người dùng
6	Created	Ngày tạo tài khoản
7	LastActive	Thời gian đăng nhập lần cuối
8	Gender	Giới tính người dùng
12	Address	Đất nước người dùng ở
13	Avatar	Ảnh đại diện người dùng
18	Phone	Số điện thoại người dùng
19	Status	Trạng thái tài khoản người dùng

❖ **Role**

Bảng 7: Bảng thông tin Role

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi role sẽ có mã khác nhau
2	Name	Tên role
3	NormalizedName	Chuẩn hóa tên role

❖ **UserRole:**

Bảng 8: Bảng thông tin UserRole

STT	Tên thuộc tính	Mô tả
1	UserId	Mã người dùng
2	RoleId	Mã role

❖ **LoginUserHistorys**

Bảng 9: Bảng thông tin LoginUserHistorys

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi lượt đăng nhập sẽ có 1 mã khác nhau
2	UserId	Xác định mã người dùng
3	LoginTime	Thời gian người dùng đăng nhập

❖ **Friends**

Bảng 10: Bảng thông tin Friends

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi mối quan hệ sẽ có một mã khác nhau
2	userId	Xác định mã người dùng
3	loginTime	Thời gian người dùng đăng nhập

❖ **FriendStatuses**

Bảng 11: Bảng thông tin FriendStatuses

STT	Tên thuộc tính	Mô tả
1	Id	Mỗi trạng thái mối quan hệ sẽ có một mã khác nhau
2	StatusName	Tên trạng thái mối quan hệ

3	CreatedAt	Thời gian tạo
4	UpdatedAt	Thời gian cập nhật

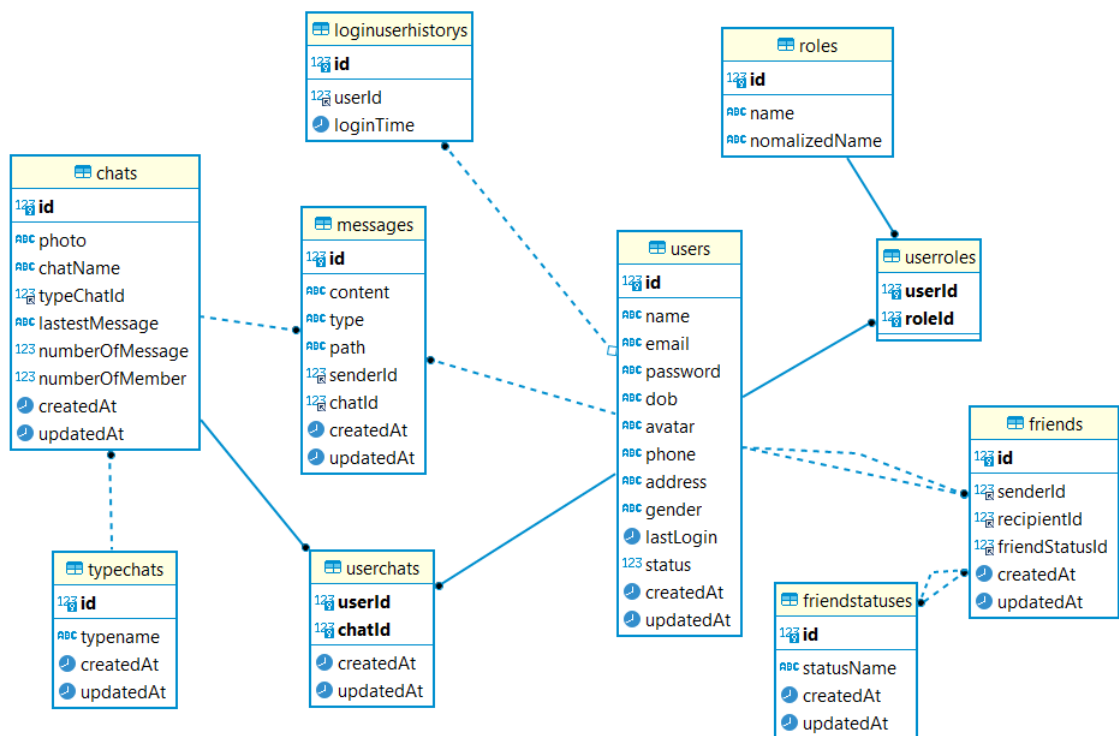
3.2.3 Các yêu cầu phi chức năng

Bảng 12: Bảng danh sách các yêu cầu phi chức năng

STT	Yêu cầu	Mô tả
1	Hiệu năng	Website có tốc độ phản hồi nhanh, không có hiện tượng đơ, lag,...
2	Độ tin cậy	Đạt mức bảo mật an toàn.
3	Tính tiện dụng	Dễ dàng sử dụng, thuận tiện cho người dùng.

3.3 Thiết kế hệ thống

3.3.1 Thiết kế cơ sở dữ liệu



Hình 9: Mô hình Cơ sở dữ liệu

❖ **Bảng User:**

Bảng 13: Bảng thuộc tính User

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Char(36)	Khóa chính	Mã người dùng
UserName	Varchar (50)	Not null	Tên người dùng
Bod	Datetime		Ngày sinh
Address	Varchar (50)		Địa chỉ
Avatar	Longtext		Ảnh đại diện
Email	Varchar (50)		Email người dùng
NomalizedEmail	Varchar (256)		Chuẩn hóa email
PasswordSalt	Varchar (256)	Not null	Salt để mã hóa password
PasswordHash	Varchar (256)	Not null	Password được mã hóa
PhoneNumber	Varchar(11)		Số điện thoại người dùng

❖ **Bảng Role:**

Bảng 14: Bảng thuộc tính Role

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Char(36)	Khóa chính	Mã Role
Name	Varchar (50)	Not null	Tên Role
NomalizedName	Varchar (50)		Chuẩn hóa tên Role

❖ **Bảng UserRole:**

Bảng 15: Bảng thông tin UserRole

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
UserId	Char(36)	Khóa ngoại	Mã người dùng
RoleId	Char(36)	Khóa ngoại	Mã Role

❖ **Bảng Chats:**

Bảng 16: Bảng thuộc tính Chats

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Char(32)	Khóa chính	Mã phòng chat
Name	Varchar(255)	Not null	Tên phòng chat
TypeChatId	Int	Khóa ngoại	Mã loại phòng chat
Photo	Varchar(255)		Ảnh phòng chat
LastMessage	Int	Khóa ngoại	Tin nhắn cuối cùng của phòng chat
NumberOfMessage	Int		Số lượng tin nhắn của phòng chat
NumberOfMember	Int		Số lượng thành viên của phòng chat

❖ **Bảng TypeChats**

Bảng 17: Bảng thuộc tính TypeChats

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Char(32)	Khóa chính	Mã loại phòng

			chat
TypeName	Varchar (50)	Not null	Tên loại phòng chat
CreatedAt	Datetime		Ngày tạo
UpdatedAt	Datetime		Ngày cập nhật

❖ **Bảng Messages:**

Bảng 18: Bảng thuộc tính Messages

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Int	Khóa chính	Mã tin nhắn
Content	Longtext	Not null	Nội dung tin nhắn
Type	Varchar(50)	Not null	Loại tin nhắn
Path	Varchar(255)		Đường dẫn thư mục(nếu tin nhắn là file)
SenderId	Varchar(36)	Not null	Id người gửi
ChatId	Varchar(36)	Not null	Id phòng chat
CreatedAt	Datetime		Ngày tạo
UpdatedAt	Datetime		Ngày cập nhật

❖ **Bảng UserChats**

Bảng 19: Bảng thuộc tính UserChats

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
UserId	Varchar(36)	Khóa ngoại	Mã người dùng
ChatId	Varchar(36)	Khóa ngoại	Mã phòng chat
CreatedAt	Datetime		Ngày tạo
UpdatedAt	Datetime		Ngày cập nhật

❖ **Bảng Friends**

Bảng 20: Bảng thuộc tính Friends

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Int	Not null	Mã mối quan hệ
SenderId	Varchar(36)	Khóa ngoại	Mã người gửi
ReceiverId	Varchar(36)	Khóa ngoại	Mã người nhận
FriendStatusId	Int	Khóa ngoại	Mã trạng thái mối quan hệ
CreatedAt	Datetime		Ngày tạo
UpdatedAt	Datetime		Ngày cập nhật

❖ **Bảng FriendStatuses**

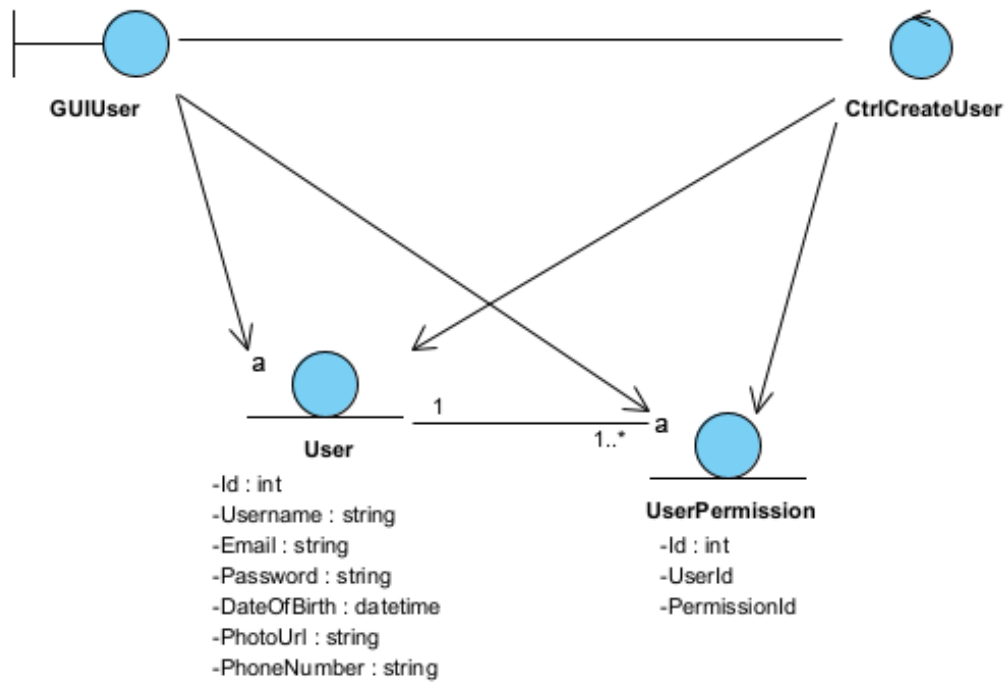
Bảng 21: Bảng thuộc tính FriendStatuses

Tên thuộc tính	Kiểu dữ liệu	Ràng buộc	Mô tả
Id	Int	Khóa chính	Mã trạng thái mối quan hệ
StatusName	Varchar(55)	Not null	Tên trạng thái mối quan hệ
CreatedAt	Datetime		Ngày tạo
UpdatedAt	Datetime		Ngày cập nhật

3.3.2 Thiết kế lớp đối tượng

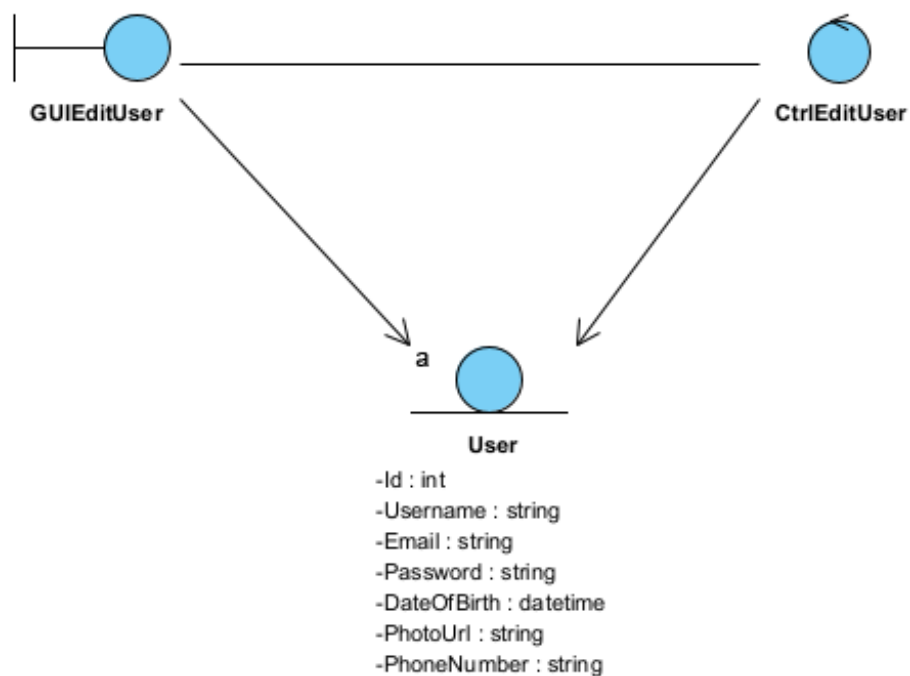
a) **Biểu đồ lớp VOPC của các ca sử dụng**

- ❖ Biểu đồ lớp VOPC trang quản trị viên
 - Biểu đồ VOPC thêm người dùng



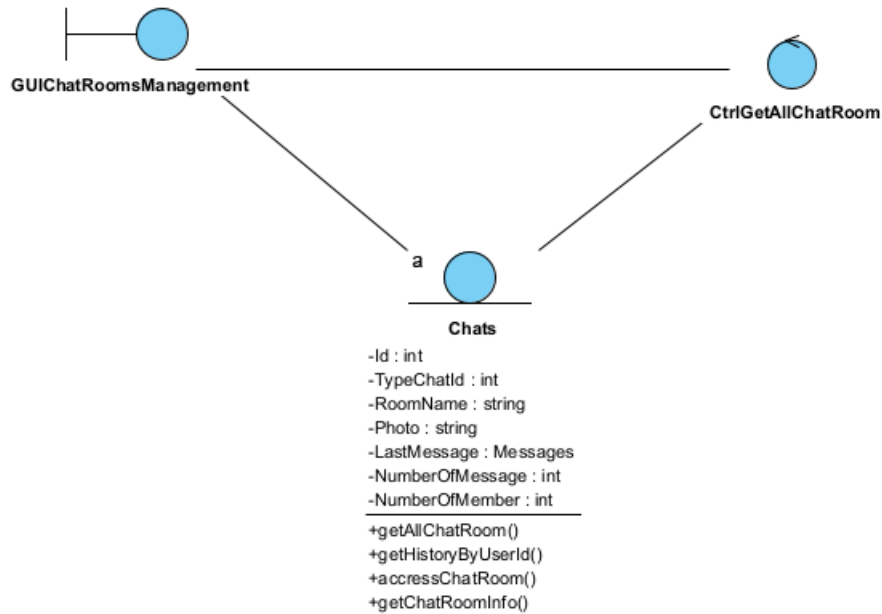
Hình 10: Biểu đồ VOPC thêm người dùng

- Biểu đồ VOPC sửa người dùng



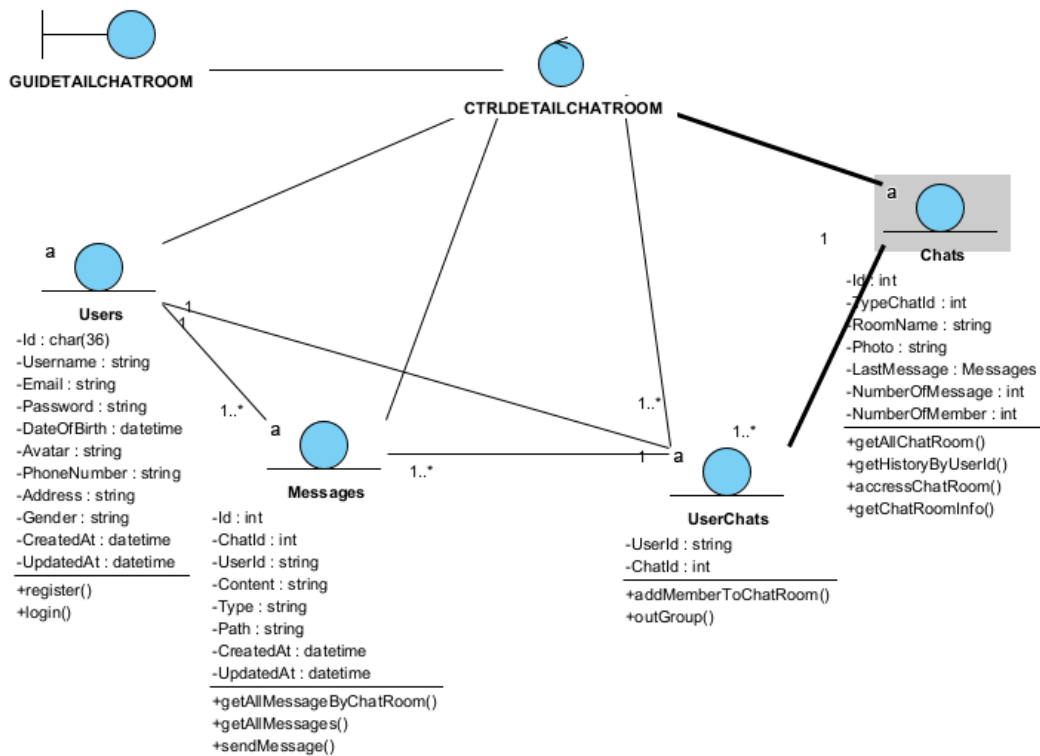
Hình 11: Biểu đồ VOPC sửa người dùng

- Biểu đồ VOPC Xem toàn bộ phòng chat



Hình 12: Biểu đồ VOPC Xem toàn bộ phòng chat

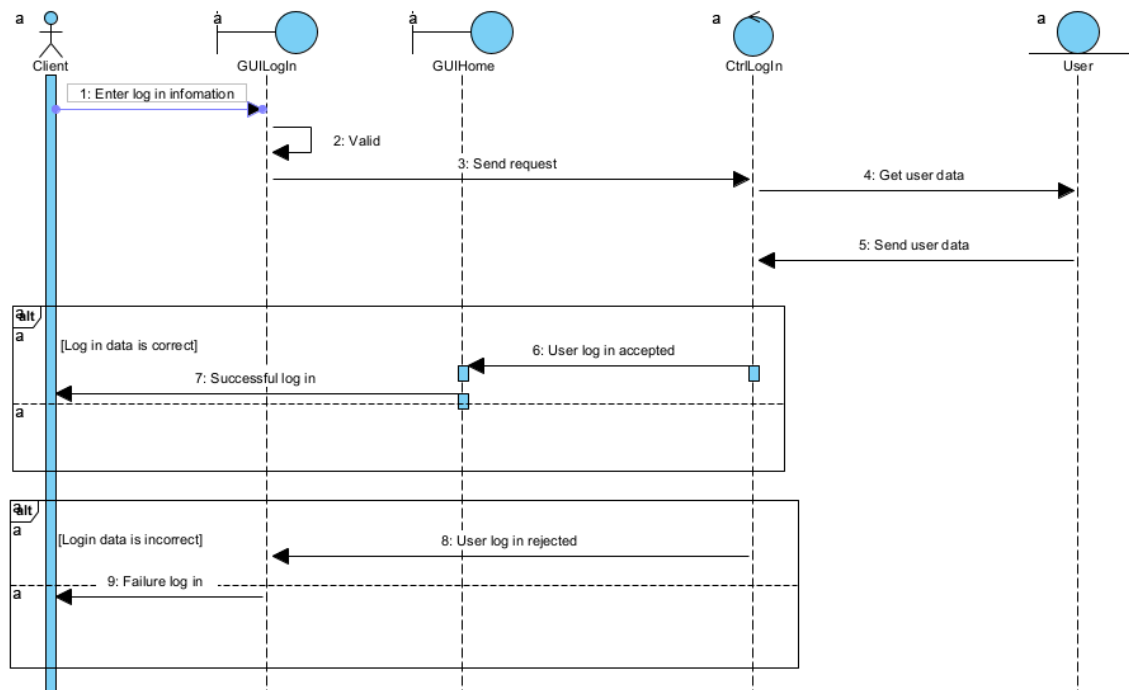
- Biểu đồ VOPC Xem chi tiết phòng chat



Hình 13: Biểu đồ VOPC Xem chi tiết phòng chat

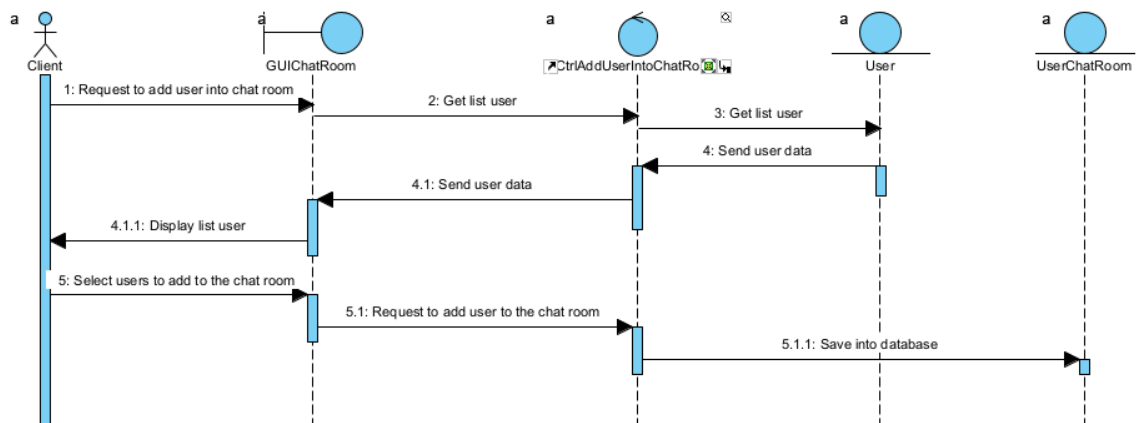
b) Biểu đồ tuần tự

- Biểu đồ tuần tự của Use Case Đăng nhập



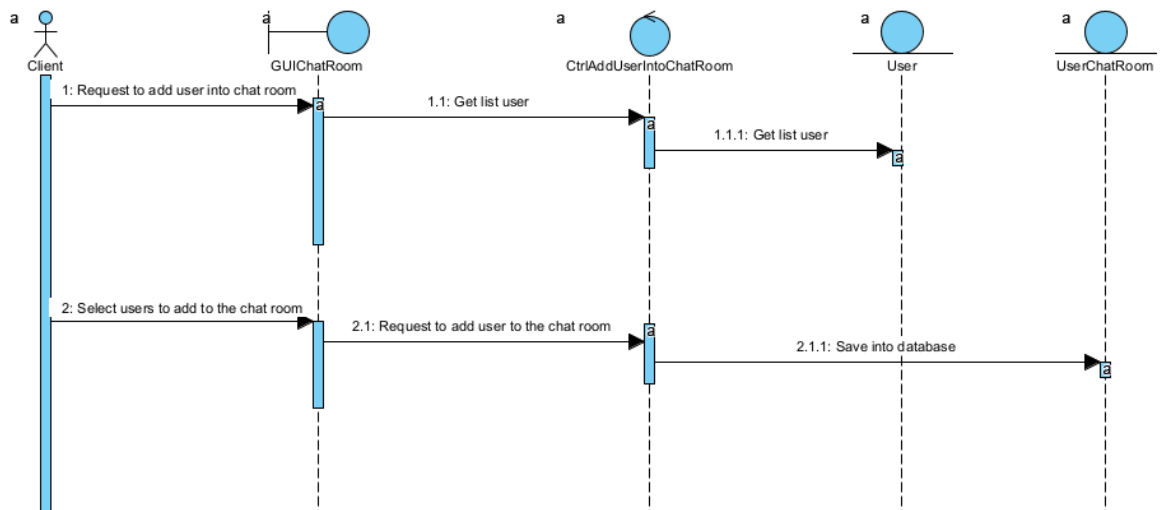
Hình 14 Biểu đồ tuần tự cho Use Case Đăng nhập

- Biểu đồ tuần tự của Use Case Thêm phòng chat



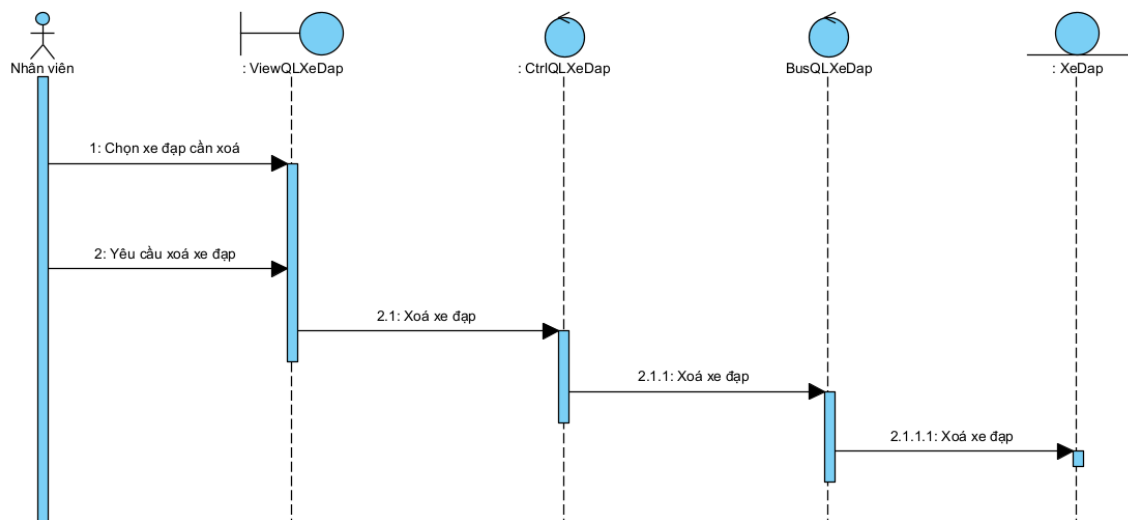
Hình 15: Biểu đồ tuần tự thêm phòng chat

- Biểu đồ tuần tự của Use Case Thêm người dùng vào phòng chat



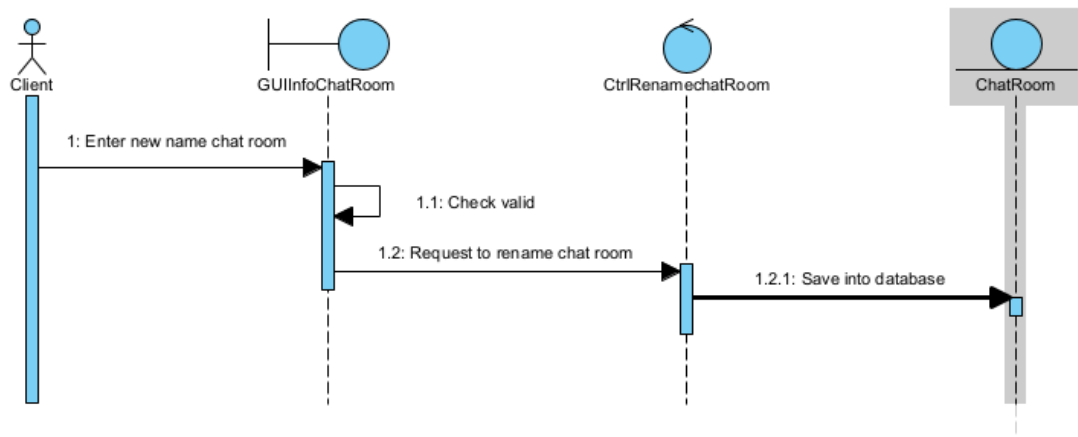
Hình 16: Biểu đồ tuần tự thêm người dùng vào phòng chat

- Biểu đồ tuần tự của Use Case Xóa người dùng khỏi phòng chat



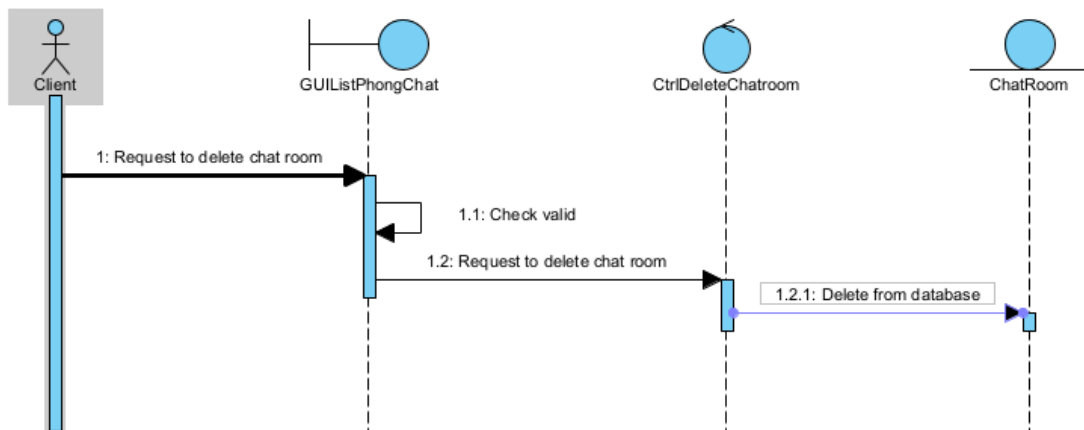
Hình 17: Biểu đồ tuần tự xóa người dùng khỏi phòng chat

- Biểu đồ tuần tự của Use case Đặt lại tên phòng chat



Hình 18: Biểu đồ tuần tự của Use case Đặt lại tên phòng chat

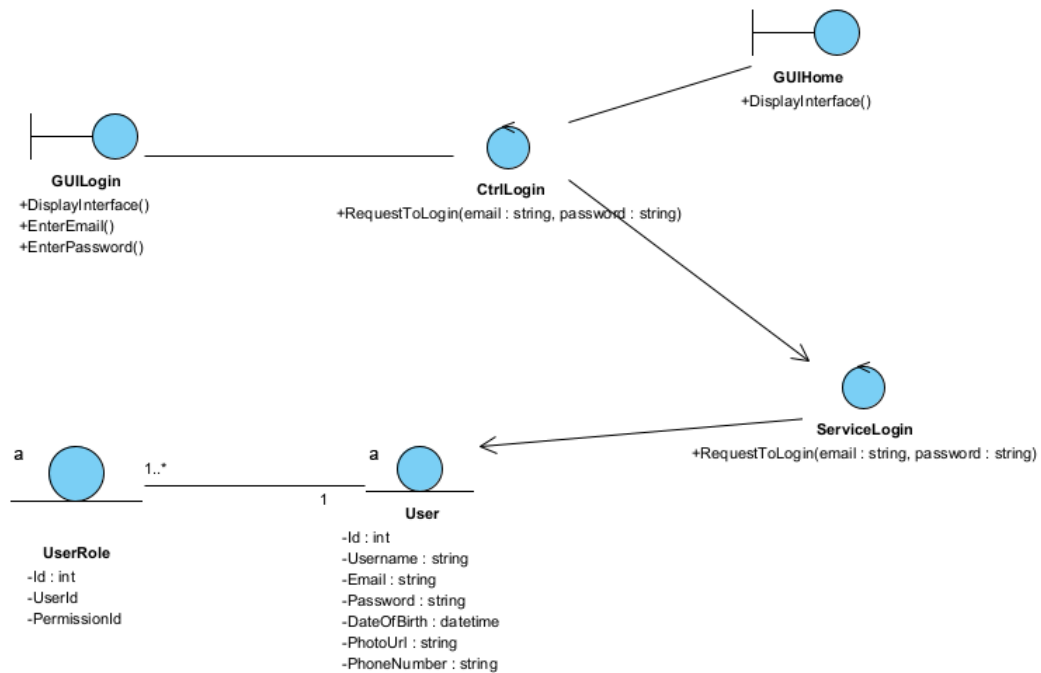
- Biểu đồ tuần tự của Use case Xóa phòng chat



Hình 19: Biểu đồ tuần tự của Use Xóa phòng chat

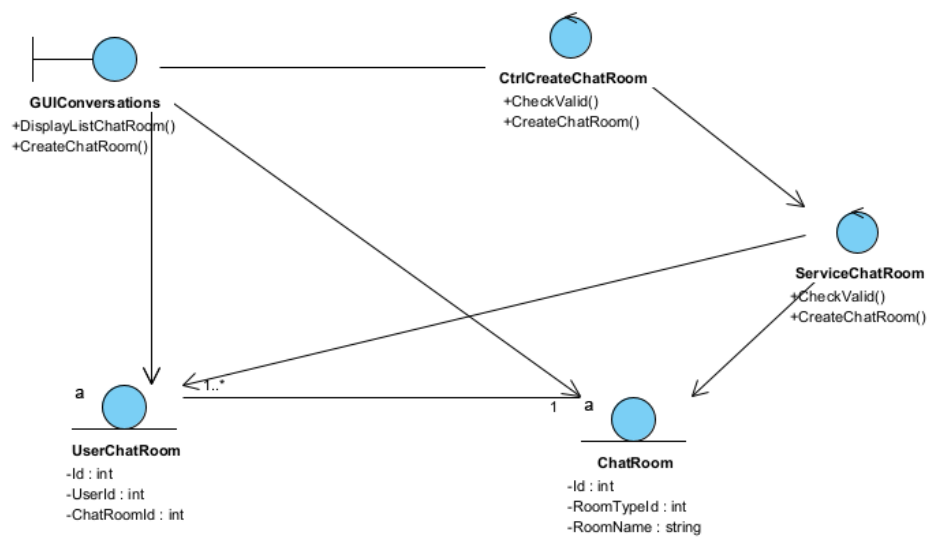
c) Biểu đồ lớp chi tiết

- Biểu đồ lớp chi tiết cho Use case đăng nhập



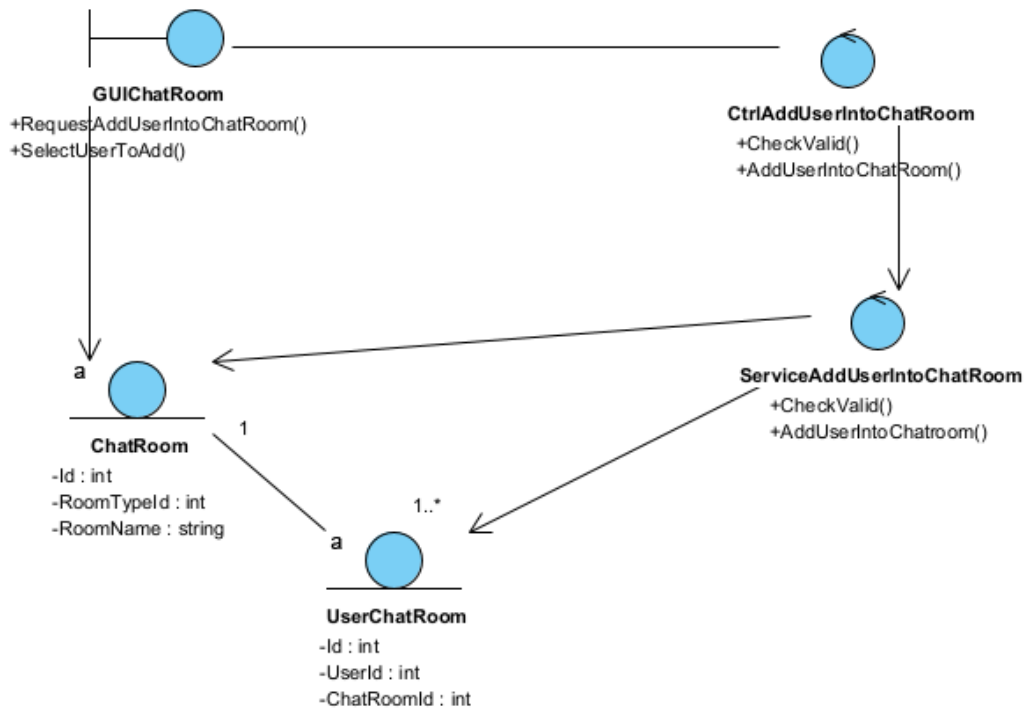
Hình 20: Biểu đồ lớp chi tiết cho Use case Đăng nhập

- Biểu đồ lớp chi tiết cho Use case Tạo phòng chat



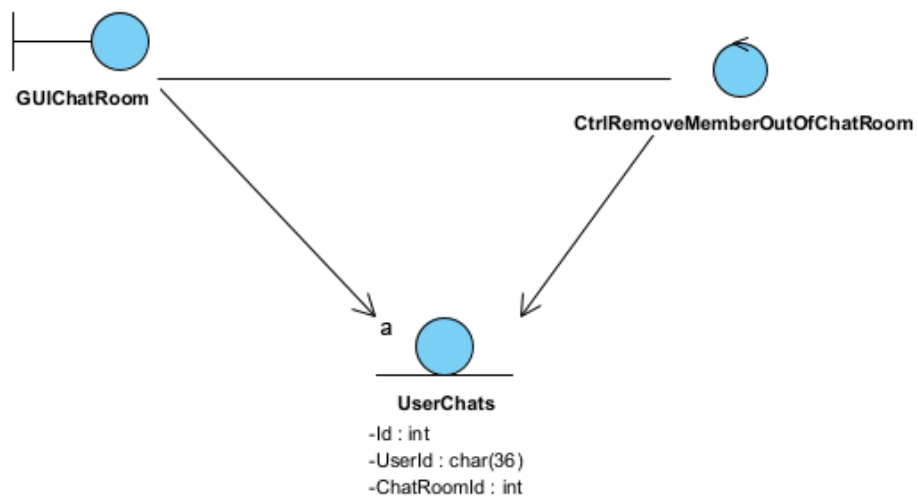
Hình 21: Biểu đồ lớp chi tiết cho Use case Tạo phòng chat

- Biểu đồ lớp chi tiết cho Use case Thêm người dùng vào phòng chat



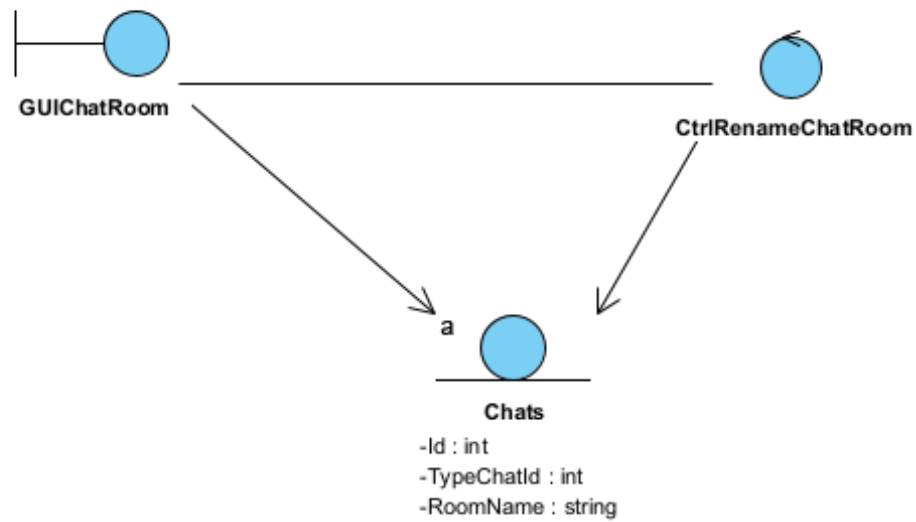
Hình 22: Biểu đồ lớp chi tiết cho Use case Thêm người dùng vào phòng chat

- Biểu đồ lớp chi tiết cho Use case Xóa thành viên khỏi phòng chat



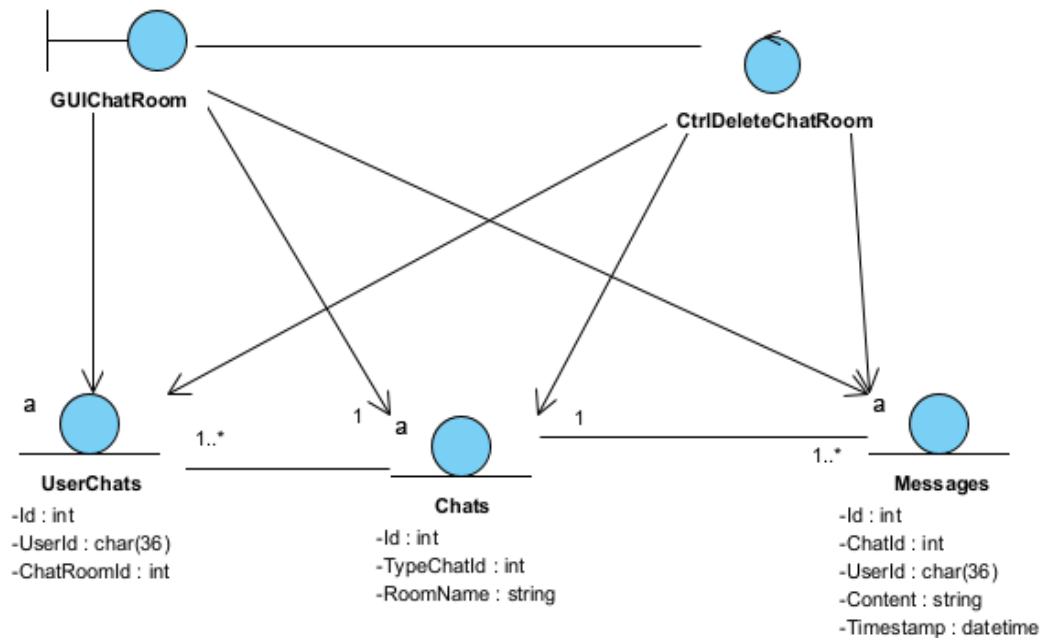
Hình 23: Biểu đồ lớp chi tiết cho Use case Xóa thành viên khỏi phòng chat

- Biểu đồ lớp chi tiết cho Use case Sửa tên phòng chat



Hình 24: Biểu đồ lớp chi tiết cho Use case Sửa tên phòng chat

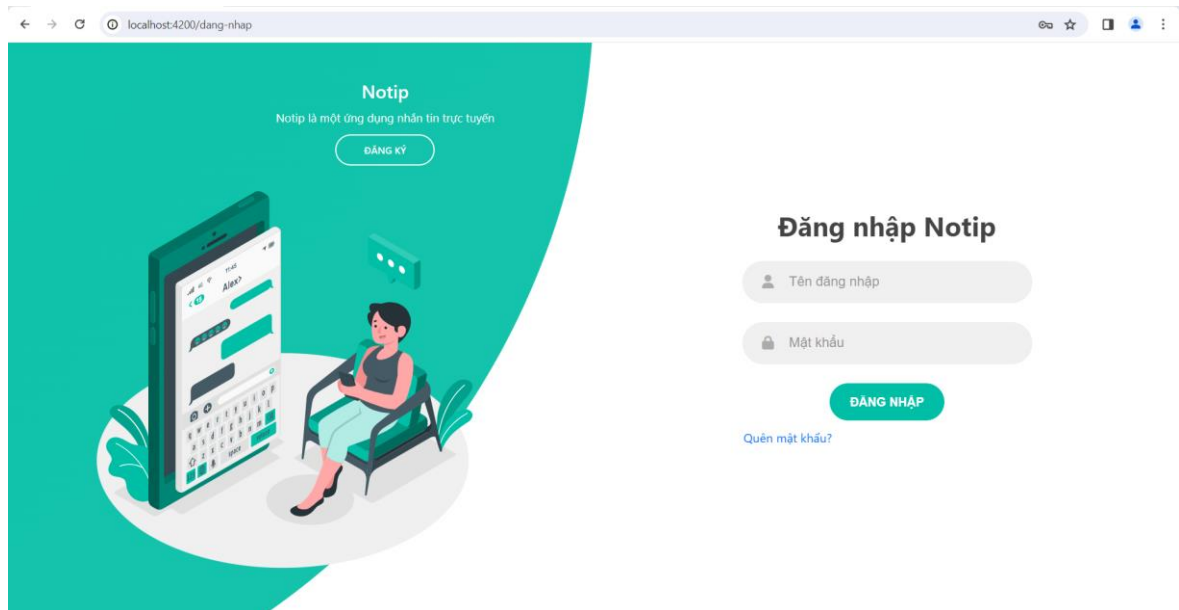
- Biểu đồ lớp chi tiết cho Use case Xóa phòng chat



Hình 25: Biểu đồ lớp chi tiết cho Use case Xóa phòng chat

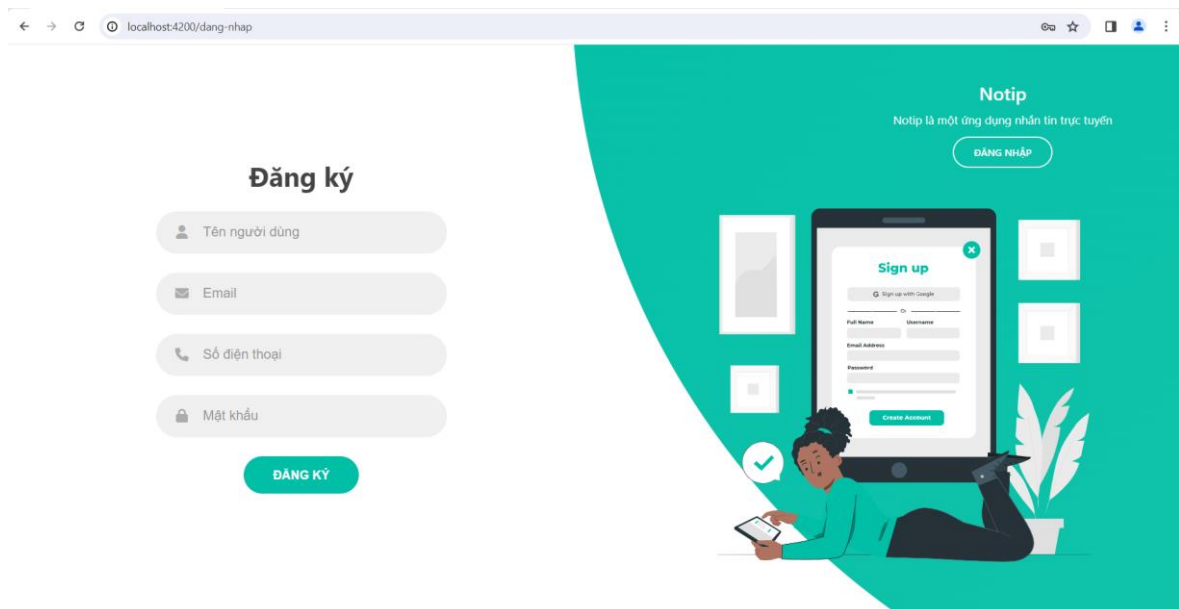
3.3.3 Thiết kế giao diện

❖ Giao diện phân hệ người dùng



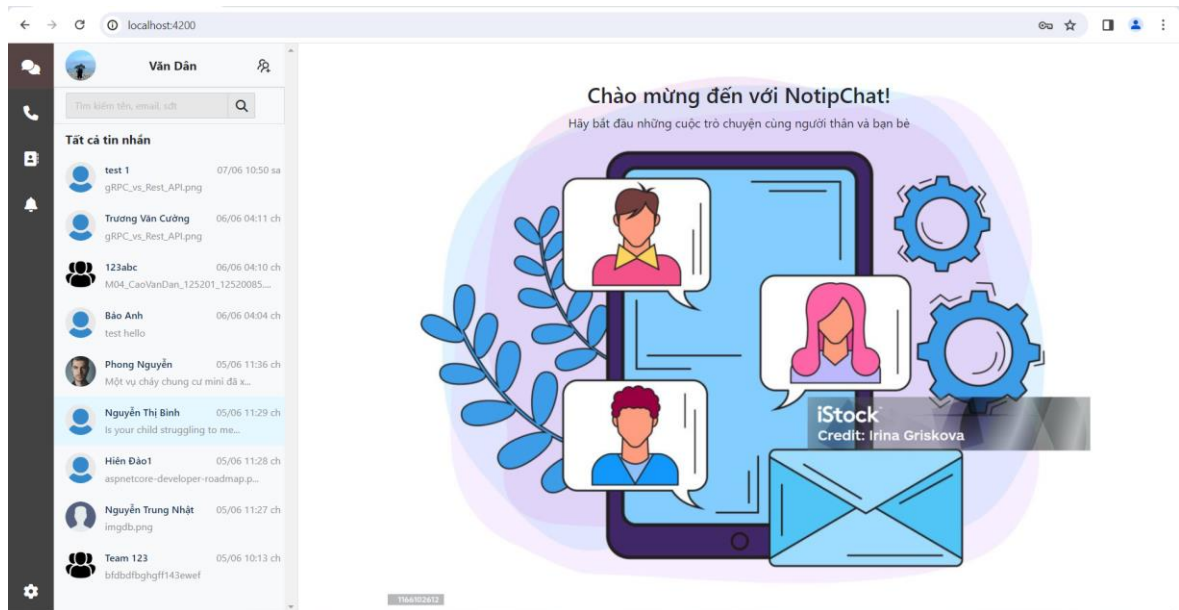
Hình 26: Giao diện trang Đăng nhập

Đầu tiên, người dùng phải đăng nhập để có thể sử dụng các dịch vụ của website



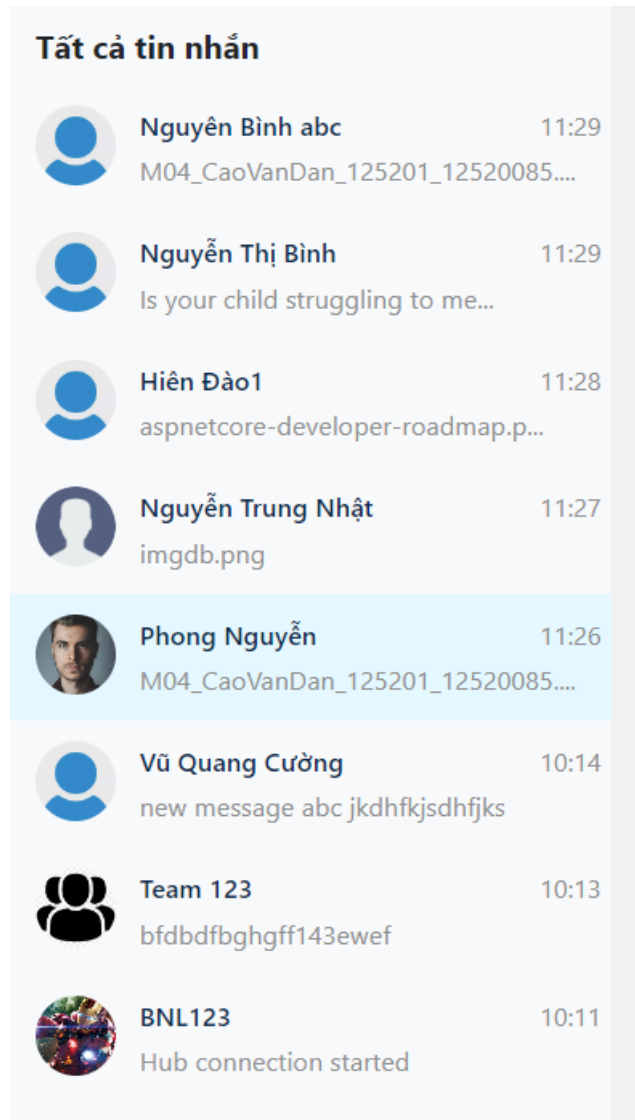
Hình 27: Giao diện trang Đăng ký

Nếu chưa có tài khoản, người dùng có thể tạo một tài khoản với các thông tin tên người dùng, địa chỉ email, số điện thoại và mật khẩu



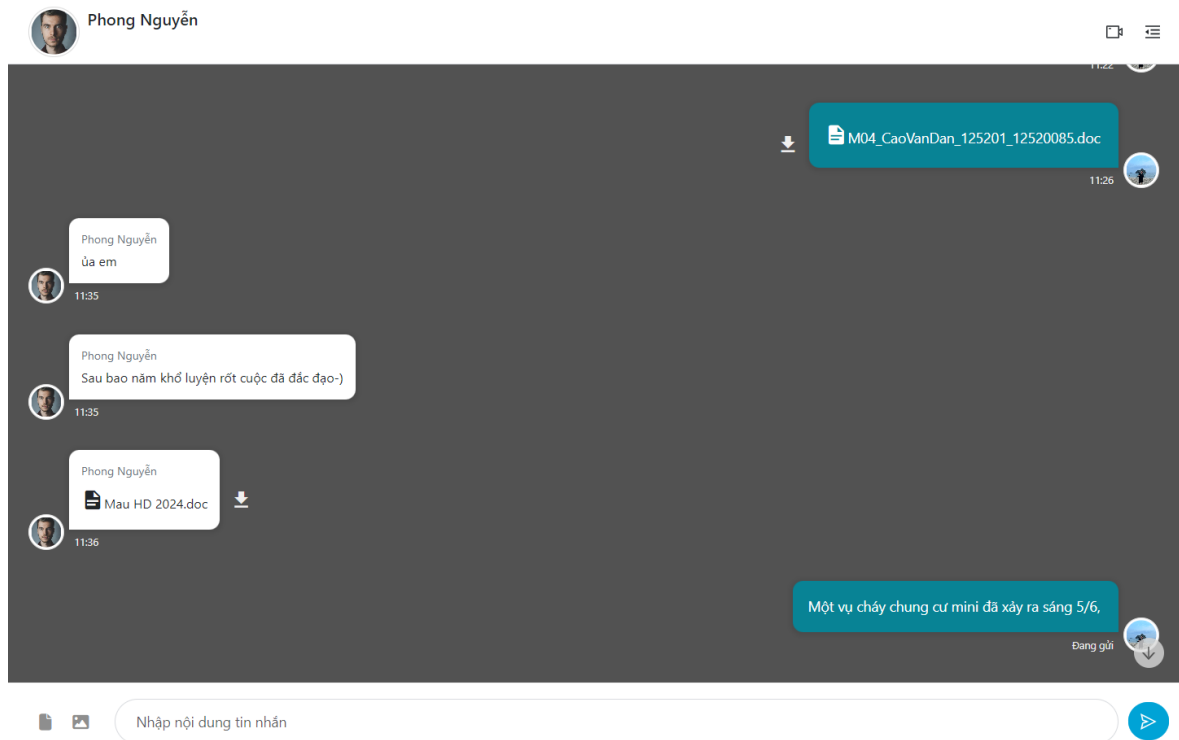
Hình 28: Giao diện dashboard phân hệ người dùng

Trang dashboard người dùng gồm 2 thành phần chính: danh sách phòng chat của người dùng và tin nhắn của phòng chat được chọn



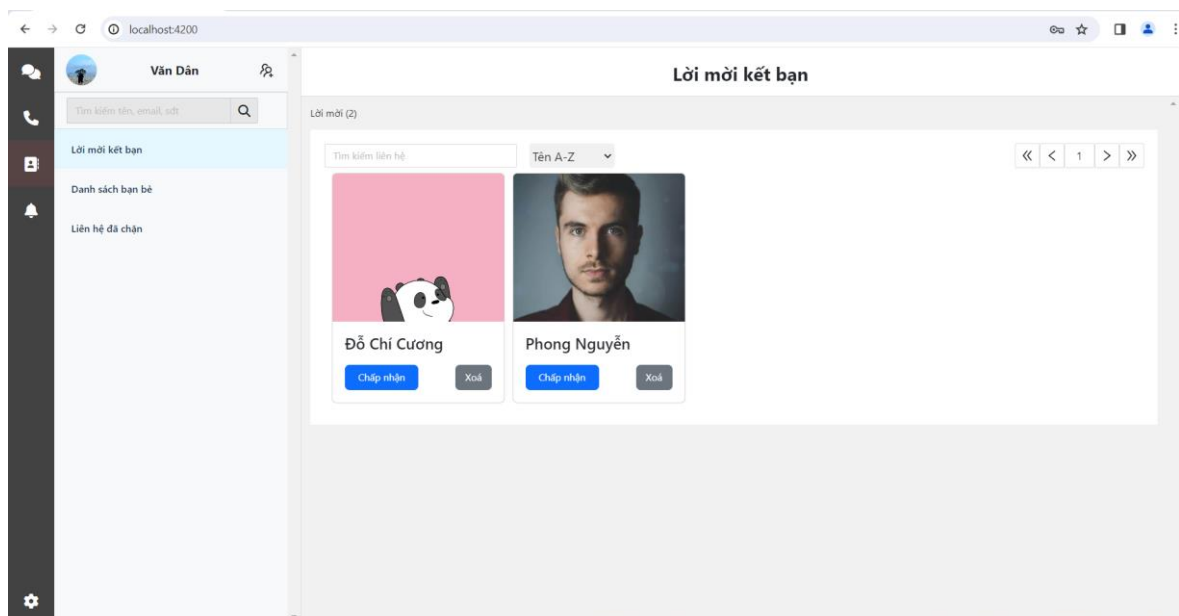
Hình 29: Giao diện danh sách phòng chat

Danh sách phòng chat gồm những phòng chat người dùng đã tham gia, có thể là chat nhóm hoặc chat riêng tư



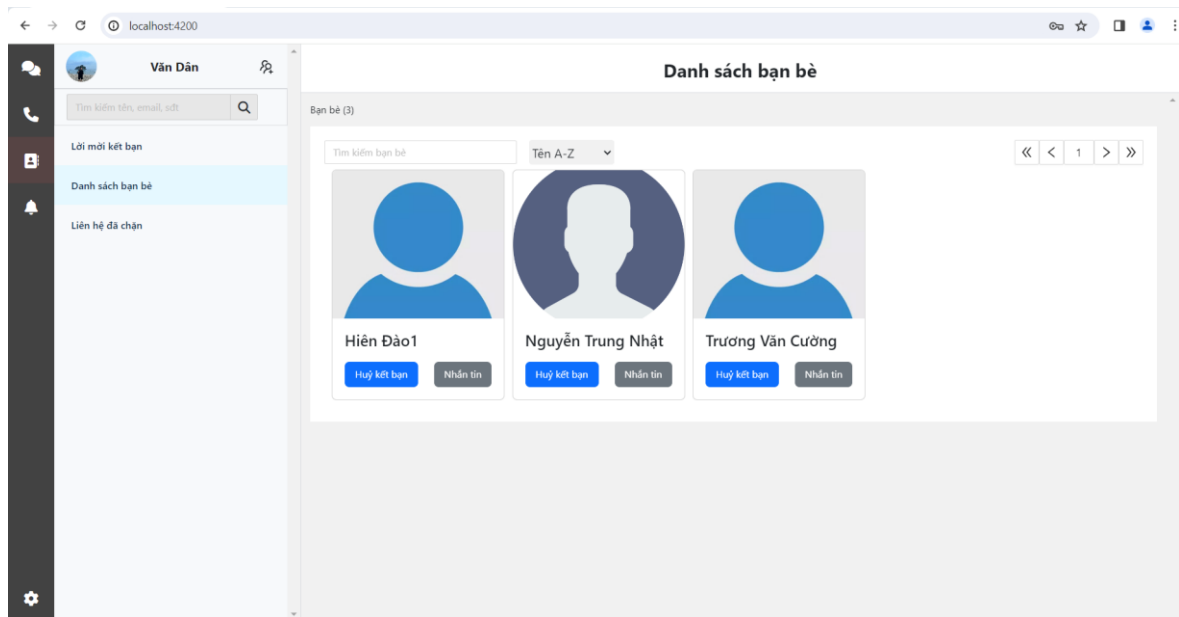
Hình 30: Giao diện tin nhắn phòng chat

Tin nhắn phòng chat hiển thị tất cả những tin được nhắn trong phòng chat đó, tên, ảnh phòng chat



Hình 31: Giao diện trang Lời mời kết bạn

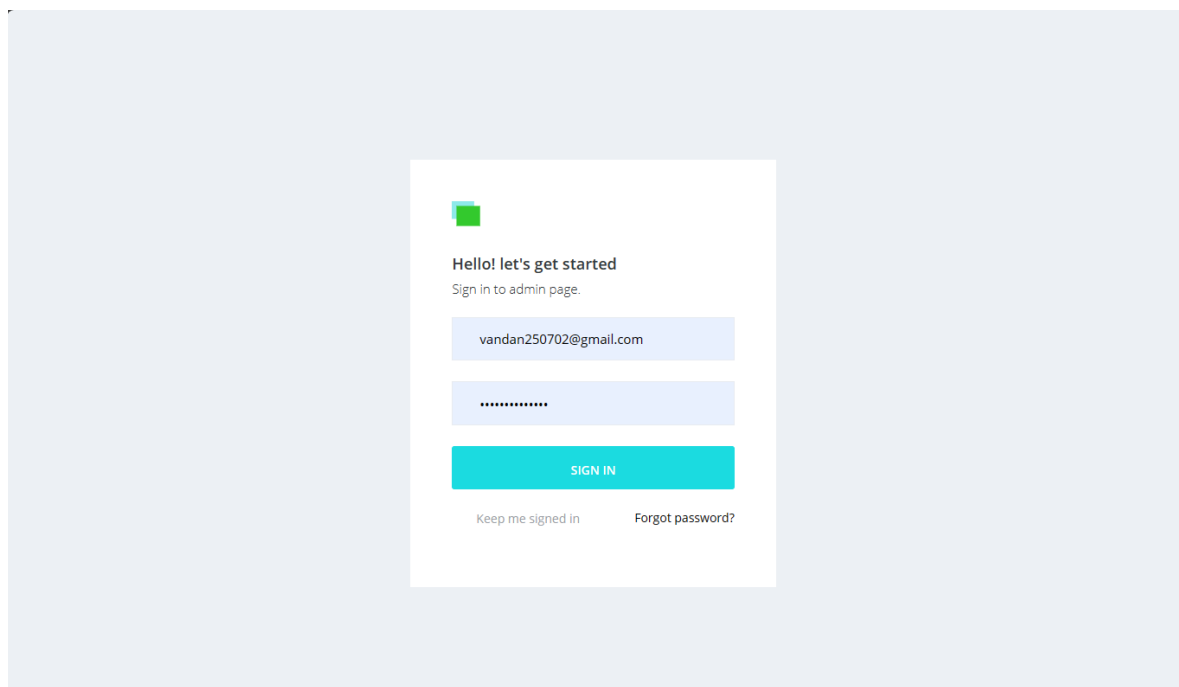
Trang Lời mời kết bạn hiển thị danh sách lời mời kết bạn của người dùng



Hình 32: Giao diện trang Danh sách bạn bè

Trang Danh sách bạn bè hiển thị danh sách bạn bè của người dùng

❖ Giao diện phân hệ quản trị



Hình 33: Giao diện trang đăng nhập

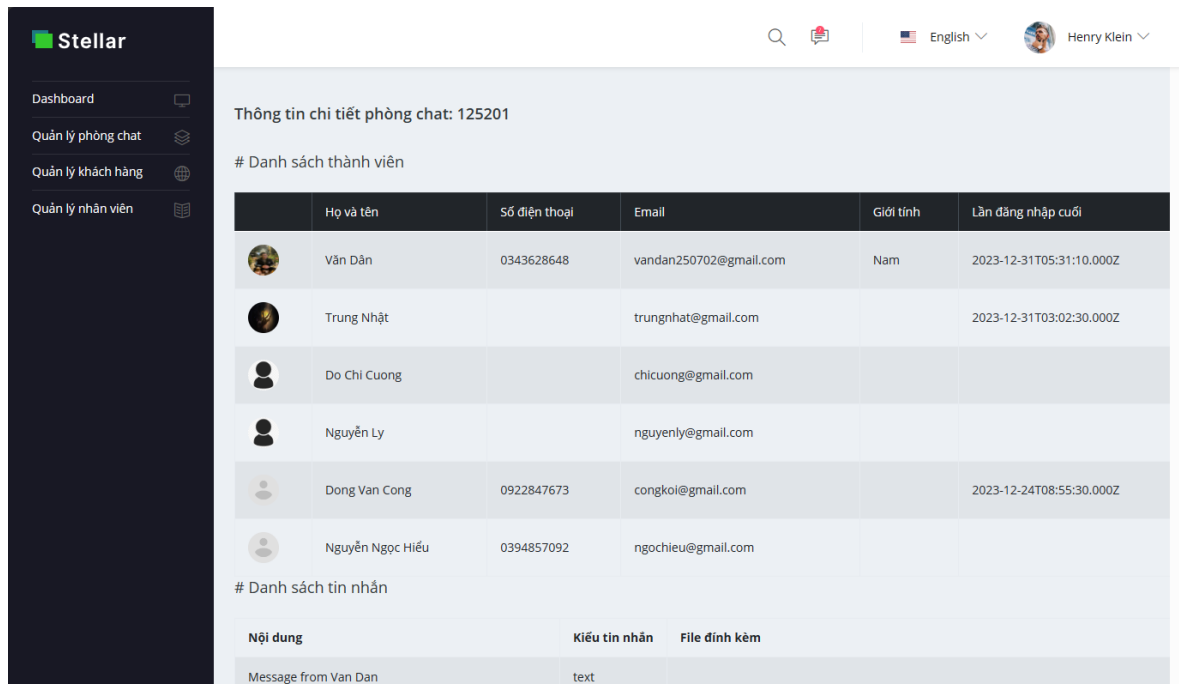


Hình 34: Giao diện trang dashboard quản trị

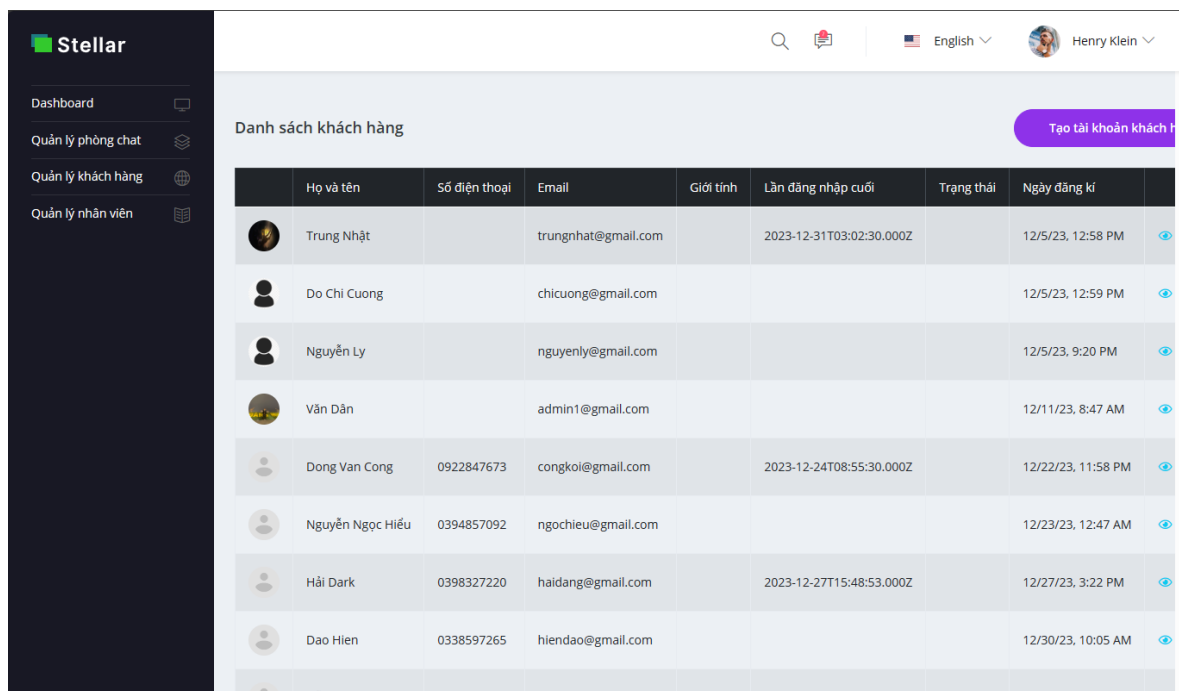
The screenshot displays the 'Danh sách phòng chat' (Chat Room List) interface. It includes a sidebar with navigation options and a top bar with search, language, and user profile settings. The main area contains a table listing various chat rooms with their details.

	Tên phòng chat	Loại phòng chat	Số lượng tin đã nhắn	Số lượng thành viên	Thời gian tạo	Thời gian hoạt động cuối
	125201	Nhóm chat	57	6	12/5/23, 10:07 PM	12/23/23, 2:07 PM
	BNL	Nhóm chat	1	2	12/5/23, 10:10 PM	12/17/23, 9:31 AM
	BNL	Nhóm chat	31	5	12/5/23, 10:23 PM	12/24/23, 11:21 PM
	new group	Nhóm chat	6	3	12/14/23, 11:28 PM	12/23/23, 3:18 PM
	Văn Dân - Dong Van Cong	Riêng tư	1	2	12/24/23, 12:46 AM	12/24/23, 3:22 PM
	Web1	Nhóm chat	3	4	12/27/23, 1:05 AM	12/27/23, 1:41 PM
	Văn Dân - Trung Nhật	Riêng tư	4	2	12/27/23, 1:41 PM	12/27/23, 1:47 PM
	group chat	Nhóm chat	1	6	12/28/23, 11:05 PM	12/31/23, 9:50 AM

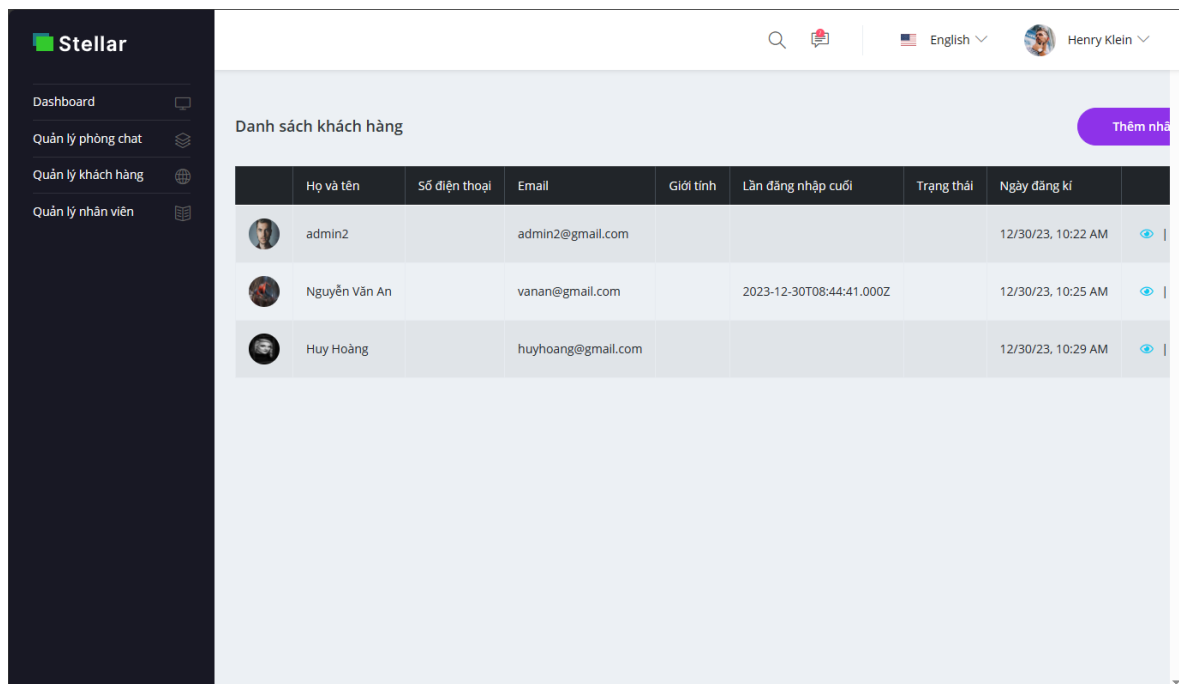
Hình 35: Giao diện trang quản lý phòng chat



Hình 36: Giao diện thông tin chi tiết phòng chat



Hình 37: Giao diện quản lý khách hàng



Hình 39: Giao diện quản lý nhân viên

CHƯƠNG 4: TRIỂN KHAI WEBSITE

4.1 Triển khai các chức năng cho phân hệ người dùng

4.1.1 Trang chủ

a) Phía front end

- Xây dựng bố cục trang Dashboard bằng các thẻ HTML
- Bố cục trang Dashboard gồm các phần: header, navbar, danh sách phòng chat, tin nhắn phòng chat.

```
<div class="navbar">
  <div>
    <div
      *ngFor="let item of tabControls; index as index"
      class="tab-icon"
      [ngClass]="{ active: tabIndexSelected == index }"
    >
      <div (click)="clickTab(index)">
        <i [class]='mdi ' + item.iconClass"></i>
      </div>
    </div>

  </div>
  <div style="cursor: pointer; font-size: 1.2rem;">
    <i class="fa fa-solid fa-gear"></i>
  </div>
</div>
```

Hình 40: Xây dựng hiển thị Navbar sử dụng template Angular

```
<div class="body-vchat-left">
  <div *ngIf="currentUser != null" class="body-vchat-left-header">
    <div (click)="openModalProfile()" class="box-avatar">
      
    </div>
    <div class="box-fullname">
      <span>{{ currentUser.name.split( separator: " ").pop() }}</span>
    </div>
    <div class="box-add-user-group">
      <span
        (click)="openModalAddGroup()"
        class="icon-control"
        title="Tạo nhóm"
      >
        <i class="mdi mdi-account-group"></i>
      </span>
    </div>
  </div>
  <div class="tab-header">
    <div class="tab-header-search">
      <input type="text" class="tab-header-search-input" id="search-contact" autocomplete="false" placeholder="Tìm kiếm tên, email,
      <button
        type="button"
        class="tab-header-search-btn"
      ><i class="fa fa-solid fa-magnifying-glass"></i></button>
    </div>
  </div>
</div>
```

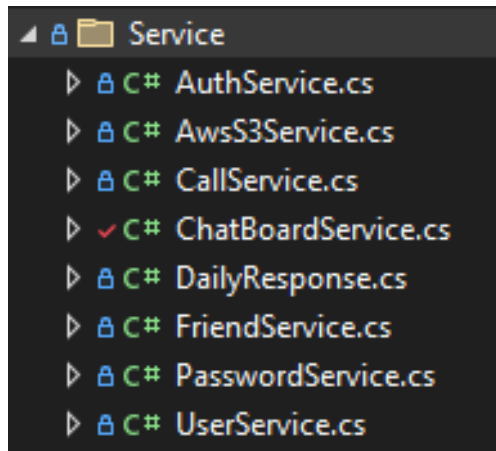
Hình 41: Xây dựng hiển thị Header bằng template Angular

```
<div class="box-left-content-title">
  <strong class="size-13">Tất cả tin nhắn</strong>
</div>
<div class="box-left-content-body">
  <ul class="list-unstyled">
    <li *ngFor="let item of datas; index as index">
      <div
        (click)="openMessage(item.id)"
        class="box-group"
        [ngClass]="{ active: chatId == item.id }"
      >
        <div class="user-avatar">
          
        </div>
        <div class="box-group-detail">
          <div class="box-group-title">
            <div class="box-group-title-name">
              <span>
                {{ item.chatName }}
              </span>
            </div>
            <div class="box-group-title-time">
              <span>{{ item.updatedAt | chatDate }}</span>
            </div>
          </div>
          <div class="box-group-last-message">
            <ng-container *ngIf="item.lastestMessage != null">
              <ng-container
                *ngIf="item.lastestMessage.length > 30; else elseTemplate"
              >
                <span> {{ item.lastestMessage.substring(0, 30) }}... </span>
              </ng-container>
            </div>
          </div>
        </div>
      </li>
    </ul>
  </div>
</div>
```

👤 Văn Dân

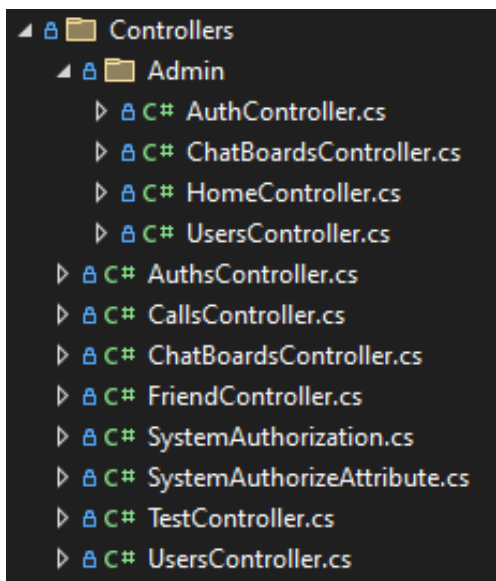
Hình 42: Xây dựng hiển thị danh sách phòng chat bằng Angular

➤ Triển khai các lớp tầng Services



Hình 45: Triển khai các lớp tầng Services

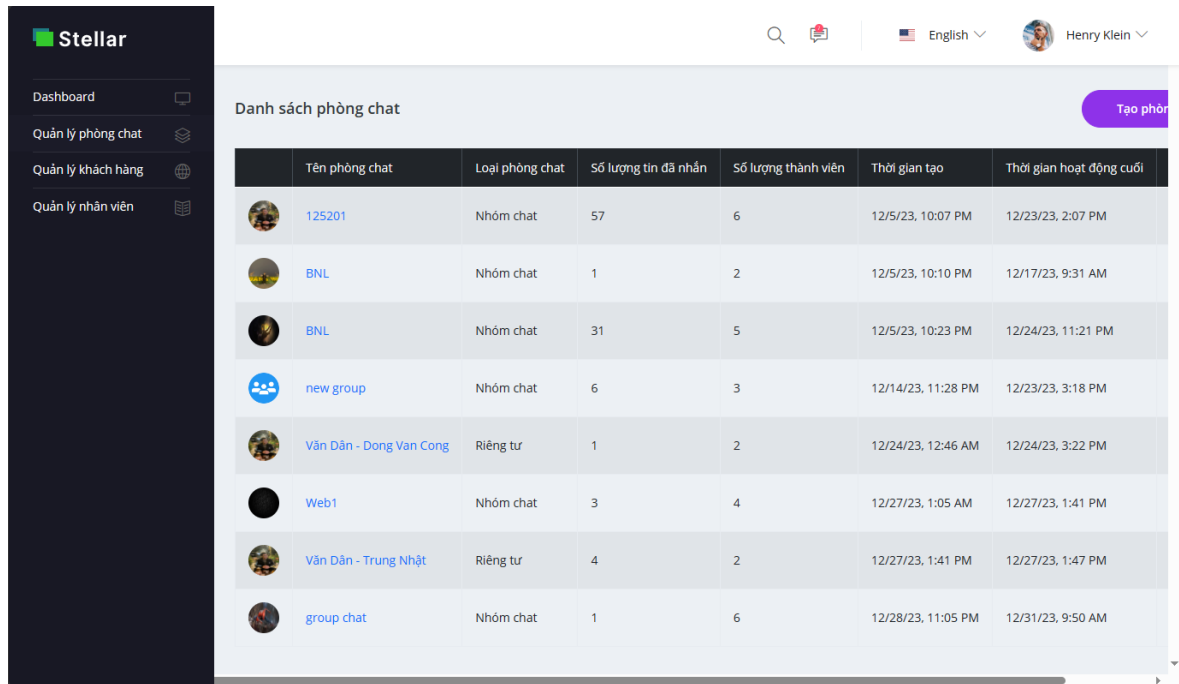
➤ Triển khai các lớp tầng Controller



Hình 46: Triển khai các lớp tầng Controller

4.2 Triển khai các chức năng cho phân hệ quản trị nội dung

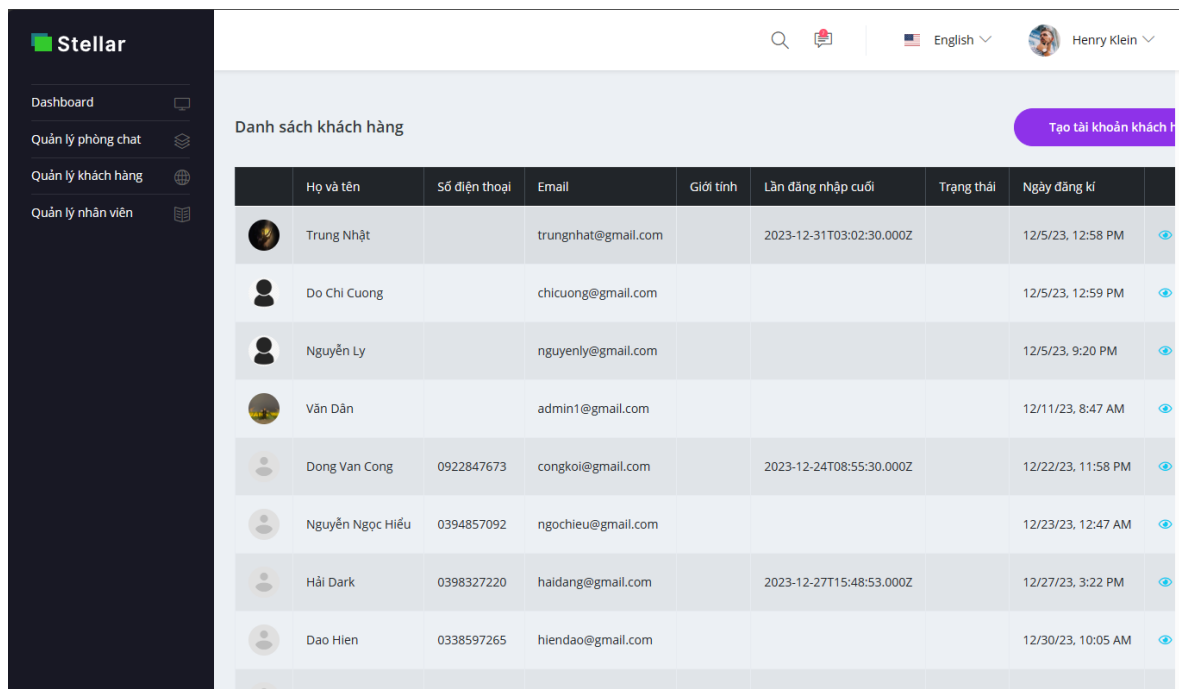
❖ Trang Quản lý phòng chat



	Tên phòng chat	Loại phòng chat	Số lượng tin đã nhận	Số lượng thành viên	Thời gian tạo	Thời gian hoạt động cuối
	125201	Nhóm chat	57	6	12/5/23, 10:07 PM	12/23/23, 2:07 PM
	BNL	Nhóm chat	1	2	12/5/23, 10:10 PM	12/17/23, 9:31 AM
	BNL	Nhóm chat	31	5	12/5/23, 10:23 PM	12/24/23, 11:21 PM
	new group	Nhóm chat	6	3	12/14/23, 11:28 PM	12/23/23, 3:18 PM
	Văn Dân - Dong Van Cong	Riêng tư	1	2	12/24/23, 12:46 AM	12/24/23, 3:22 PM
	Web1	Nhóm chat	3	4	12/27/23, 1:05 AM	12/27/23, 1:41 PM
	Văn Dân - Trung Nhật	Riêng tư	4	2	12/27/23, 1:41 PM	12/27/23, 1:47 PM
	group chat	Nhóm chat	1	6	12/28/23, 11:05 PM	12/31/23, 9:50 AM

Hình 47: Trang quản lý phòng chat

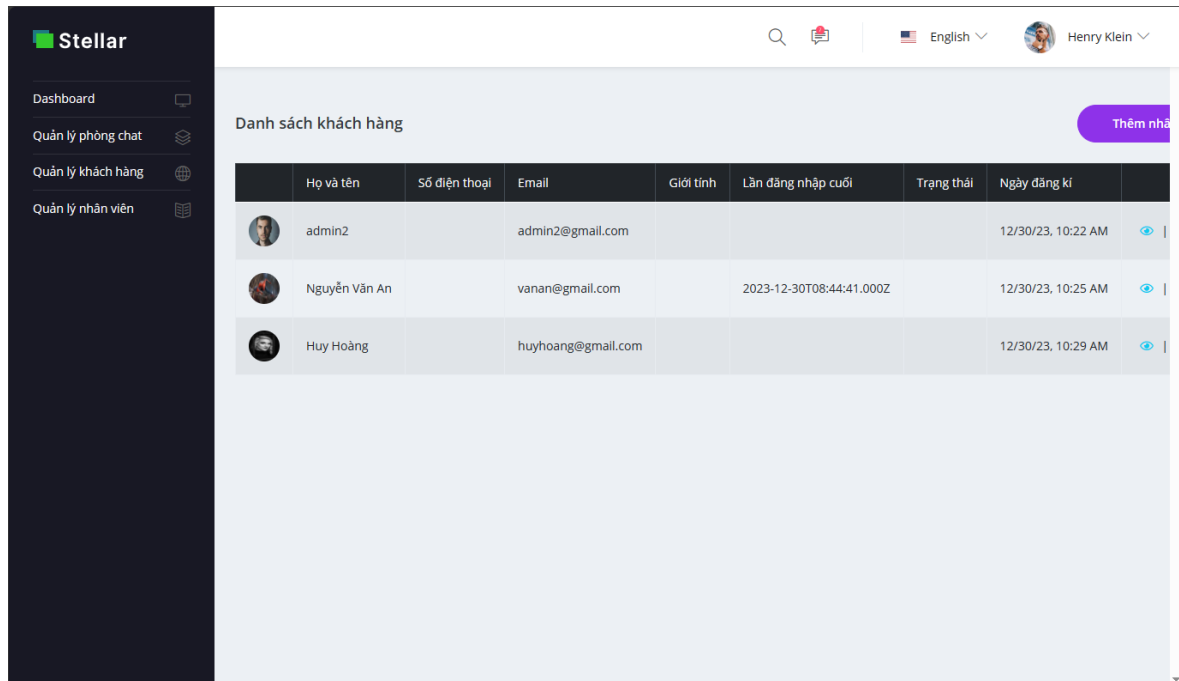
❖ Trang quản lý khách hàng



	Họ và tên	Số điện thoại	Email	Giới tính	Lần đăng nhập cuối	Trạng thái	Ngày đăng kí
	Trung Nhật		trungnhat@gmail.com		2023-12-31T03:02:30.000Z		12/5/23, 12:58 PM
	Do Chi Cuong		chicuong@gmail.com				12/5/23, 12:59 PM
	Nguyễn Ly		nguyenly@gmail.com				12/5/23, 9:20 PM
	Văn Dân		admin1@gmail.com				12/11/23, 8:47 AM
	Dong Van Cong	0922847673	congkoi@gmail.com		2023-12-24T08:55:30.000Z		12/22/23, 11:58 PM
	Nguyễn Ngọc Hiếu	0394857092	ngochieu@gmail.com				12/23/23, 12:47 AM
	Hải Dark	0398327220	haidang@gmail.com		2023-12-27T15:48:53.000Z		12/27/23, 3:22 PM
	Dao Hien	0338597265	hiendao@gmail.com				12/30/23, 10:05 AM

Hình 48: Trang quản lý khách hàng

❖ Trang quản lý nhân viên



Hình 49: Trang quản lý nhân viên

4.3 Kiểm thử và triển khai ứng dụng

4.3.1 Kiểm thử

❖ Đăng nhập

Bảng 22: Bảng test case chức năng đăng nhập

Input	Không tìm thấy tài khoản và mật khẩu trong Database	Tìm thấy tài khoản và mật khẩu trong Database	Tài khoản hoặc mật khẩu trống	Output
Tài khoản và mật khẩu	Xảy ra			Sai tài khoản hoặc mật khẩu
Tài khoản và mật khẩu		Xảy ra		Đăng nhập thành công
Tài khoản và mật khẩu			Xảy ra	Vui lòng nhập lại ô bỏ trống

4.3.2 Đóng gói ứng dụng

- Nén project source (.zip)
- Xuất file databale (.sql)

4.3.3 Triển khai ứng dụng

❖ System:

- Processor: Intel® Core™ i7-4800MQ @ 1.60GHz 1.80GHz
- RAM: 8.00GB
- System type: 64-bit

❖ Window 10

❖ Phần mềm:

- Visual studio
- Visual studio code

KẾT LUẬN

- Trong quá trình nghiên cứu và thực hiện đề tài, em đã đạt được những kết quả sau:
 - ✓ Nắm vững kiến thức về lập trình Angular và ASP.NET Core API: Đã học và áp dụng thành công kiến thức về lập trình sử dụng Angular và tạo API với C# ASP.NET. Điều này giúp xây dựng nền tảng mạnh mẽ cho dự án của mình.
 - ✓ Hoàn thiện website nhắn tin: Đã thành công trong việc hoàn thiện một trang web nhắn tin trực tuyến với những chức năng cần thiết, quan trọng và nâng cao. Sản phẩm mang lại trải nghiệm trò chuyện trực tuyến tích cực và thuận lợi.
 - ✓ Giao diện thân thiện và dễ sử dụng: Một trong những ưu điểm lớn của sản phẩm là giao diện người dùng thân thiện và dễ sử dụng.
- Những hạn chế của đề tài:
 - ✓ Hiệu suất và tối ưu hóa: Trang web có thể gặp vấn đề về hiệu suất khi số lượng người dùng và phòng chat tăng lên do chưa được tối ưu hóa hoàn toàn. Điều này có thể ảnh hưởng đến tốc độ tải trang và trải nghiệm người dùng.
 - ✓ Bảo mật: Mặc dù đã áp dụng một số biện pháp bảo mật, nhưng trang web vẫn cần được kiểm tra và cải thiện thêm về bảo mật để đảm bảo an toàn cho dữ liệu người dùng và giao dịch trực tuyến.
- Hướng phát triển tiếp theo:
 - ✓ Phát triển thêm tính năng: Tiếp tục phát triển và tích hợp các tính năng nâng cao như video call, và hệ thống quản lý khách hàng (CRM).
 - ✓ Tối ưu hóa hiệu suất: Cải thiện hiệu suất của trang web bằng cách tối ưu hóa mã nguồn, sử dụng bộ nhớ đệm (cache), và áp dụng các kỹ thuật tối ưu hóa cơ sở dữ liệu.

- ✓ **Nâng cao bảo mật:** Tiến hành các kiểm tra bảo mật thường xuyên, áp dụng các biện pháp bảo mật tiên tiến như mã hóa dữ liệu, xác thực hai yếu tố (2FA), và bảo vệ chống lại các cuộc tấn công mạng.
- ✓ **Tích hợp trí tuệ nhân tạo (AI):** Sử dụng AI để cá nhân hóa trải nghiệm người dùng, phân tích tin nhắn và loại bỏ những nội dung tiêu cực.
- ✓ **Phát triển ứng dụng di động:** Tạo phiên bản ứng dụng di động để người dùng có thể nhắn dễ dàng hơn trên các thiết bị di động, từ đó mở rộng phạm vi tiếp cận khách hàng.

TÀI LIỆU THAM KHẢO

- [1] Đề cương bài giảng Công nghệ Web và ứng dụng, Trường đại học SPKT Hưng Yên.
- [2] Đề cương môn học Hệ quản trị cơ sở dữ liệu, Trường đại học SPKT Hưng Yên.
- [3] Đề cương môn học PHÂN TÍCH THIẾT KẾ HƯỚNG ĐỐI TƯỢNG VỚI UML, Trường đại học SPKT Hưng Yên.
- [4] Shyam Seshadri, Angular Up & Running, 2018.
- [5] Troelsen, Andrew, and Philip Japikse. Pro C# 10 with .NET 6: Foundational [3]. Principles and Practices in Programming. Apress, 2022. [4]. Liberty, Jesse. Programming C#: Building .NET Applications with C#. O'Reilly Media, 2021.
- [6] Freeman, Adam. Pro ASP.NET Core 6. Apress, 2022.