TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Xây dựng website nhắn tin trực tuyến cho nội bộ doanh nghiệp

Đỗ Thái Sơn
son.dt187273@sis.hust.edu.vn

Ngành Công nghệ thông tin và truyền thông
Chuyên ngành Công nghệ thông tin

Giảng viên hướng dẫn: TS. Hoàng Văn HIệp _____

Chữ kí GVHD

Khoa: Công nghệ thông tin

Trường: Công nghệ thông tin và Truyền thông

HÀ NỘI, 02/2023

LỜI CẢM ƠN

Đầu tiên, em xin cảm ơn thầy TS. Hoàng Văn Hiệp là người đã trực tiếp hướng dẫn và giúp đỡ em hoàn thiện đồ án tốt nghiệp này.

Tiếp theo, em muốn gửi lời cảm ơn đến nhà trường, các giảng viên cùng các bạn học cùng lớp đã tận tình hướng dẫn, giúp đỡ trong những năm học qua.

Cuối cùng con xin cảm ơn gia đình đã luôn là nguồn động lực để con tiến bước trên con đường đại học và ra ngoài xã hội khởi đầu với đồ án tốt nghiệp này.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Ngày nay ứng dụng công nghệ thông tin vào việc tin học hóa được xem là một trong những yếu tố quyết định trong hoạt động của các chính phủ, tổ chức, cũng như các công ty; nó đóng vai trò hết sức quan trọng, có thể tạo ra những bước đột phá mạnh mẽ. Với cuộc cách mạng khoa học công nghệ 4.0, ứng dụng nhắn tin đang chiếm lĩnh các kênh thông tin, giải trí, công việc, giao lưu không chỉ của của giới trẻ. Ứng dụng nhắn tin là một trang web hay nền tảng trực tuyến với nhiều hình thức, tính năng, giúp mọi người dễ dàng kết nối với nhau ở bất cứ nơi nào. Ở đó, không chỉ có các mối quan hệ ảo của những người cùng đam mê, sở thích. . . mà còn có cả những mối quan hệ công việc, đa ngành nghề. Việc sử dụng mạng xã hội tin nhắn mang lại nhiều lợi ích cho người dùng nói riêng và cho cộng đồng nói chung

Chính vì vậy, đồ án này hướng tới một hệ thống nhắn tin trực tuyến ứng dụng trong tổ chức, doanh nghiệp, trong đó tập trung hơn đến quy trình soạn thảo, gửi, hiển thị tin nhắn. Để thực hiện được điều đó, hệ thống sẽ được xây dựng sử dụng nền tảng Web để phát triển, bao gồm những chức năng cơ bản và tập trung vào quy trình xử lý tin nhắn.

Đóng góp chính của đồ án này là xây dựng một trang web nhắn tin được sử dụng với mục đích cụ thể cho công việc, học tập,. . . với đầy đủ các chức năng yêu cầu cơ bản của một ứng dụng nhắn tin trực tuyến như: kết bạn, nhắn tin 2 người, tạo nhóm nhắn nhiều người, tin nhắn nháp, tương tác tin nhắn, theo dõi luồng tin nhắn, thông báo,... Ứng dụng nhắn tin của em hướng tới mọi nhóm người dùng với các mục đích sử dụng cụ thể. Ứng dụng nhắn tin được xây dựng cho mục đích học tập, công việc, chia sẻ thông tin, kiến thức; giao lưu, gắn kết mọi người với nhau.

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Viết tắt	Tên tiếng Anh	Tên tiếng Việt
API	Application Programming Interface	Giao diện lập trình ứng dụng
UC	Use Case	
HTML	HyperText Markup Language	Ngôn ngữ đánh dấu siêu văn bản
RDBMS	Relational Database Management System	Hệ quản trị cơ sở dữ liệu quan hệ
SQL	Structured Query Language	Ngôn ngữ truy vấn có cấu trúc
FIFO	First In First Out	Vào trước sẽ được xử lý trước
CNTT		Công nghệ thông tin
ĐATN		Đồ án tốt nghiệp
SV		Sinh viên

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Vấn đề chia sẻ trao đổi thông tin, tài liệu từ trước đến nay luôn là một vấn đề quan trọng có ảnh hưởng lớn đến sự phát triển từ các đơn vị lớn như tập đoàn, các cơ quan hành chính nhà nước tới các đơn vị nhỏ như mô hình kinh doanh hộ gia đình, các doanh nghiệp vừa và nhỏ. Việc làm thế nào để người lao động có thể tiếp cận được với các thông tin, tài liệu có liên quan đến bản thân, công việc của mình cũng như tự chia sẻ thông tin, tài liệu, đề xuất với người khác một cách nhanh, chính xác đã trở thành một vấn đề quan trọng cần phải giải quyết.

Ban đầu vấn đề này được giải quyết bằng các phương pháp truyền thống như gặp mặt trực tiếp, tổ chức các cuộc họp nhóm,. . . Trong quá trình phát triển của công nghệ, các hình thức khác bắt đầu phát triển như gọi điện trực tiếp hay gửi tin nhắn sms. Tuy nhiên cách làm này vẫn còn nhiều hạn chế: chi phí cao, không hiệu quả trong việc chia sẻ thông tin hướng tài liệu phức tạp, khó khắn trong giao tiếp nhóm nhiều người.

Trong thời gian gần đây, nhờ sự phát triển của nền kinh tế kèm theo sự phát triển của khoa học kĩ thuật, rất nhiều các doanh nghiệp mới được thành lập cùng với đó là bùng nổ về số lượng thông tin, dữ liệu là vô cùng lớn. Từ đó nhu cầu về việc trao đổi thông tin, thảo luận trong các dự án, làm việc nhóm trở nên vô cùng quan trọng và cần thiết đối với sự thành công của một dự án. Việc máy tính được phát triển và được sử dụng rộng rãi trong xã hội cùng với sự phổ cập của mạng Internet toàn cầu đã tạo điều kiện cho một phương pháp mới làm trung gian trao đổi, thảo luận thông tin giữa người với người trong các tổ chức, nhóm, doanh nghiệp . Vì vậy việc trao đổi thông tin trực tuyến trên mạng Internet trở thành mục tiêu cần phải hướng tới.

Việc chuyển sang trao đổi thông tin trên mạng Internet đem lại nhiều lợi ích cho con người. Tận dụng được sự phổ cập của mạng Internet hiện nay, người dùng có thể tiếp cận được các thông tin công việc, học tập liên quan tới bản thân ngay tại nhà riêng hoặc ở bất cứ đâu có kết nối mạng Internet, tiết kiệm được nhiều chi phí, công sức. Bên cạnh đó, các doanh nghiệp, tổ chức, đoàn thể có thể tạo một kênh thông tin chung, riêng biệt cho các thành viên, nhân viên trong nhóm, tổ chức của mình để các cá nhân có thể trao đổi, chia sẻ thông tin phục vụ cho mục đích riêng. Lợi ích mà kênh trao đổi chung, trực tuyến giữa nhiều người với nhau được thiết lập là vô cùng lớn. Do đó việc tạo ra môi trường trao đổi thông tin trực tuyến giữa mọi người với nhau là vấn đề cần thiết.

1.2 Mục tiêu và phạm vi đề tài

Hiện nay, trên nền tảng Internet đã có nhiều sản phẩm, trang web nhắn tin trực tuyến. Một số trang web, ứng dụng nổi bật có thể kể đến như là zalo - ứng dụng nhắn tin nhanh đa nền tảng được phát triển bởi công ty VNG ở Việt Nam. Zalo là một mạng xã hôi với rất nhiều tính năng, tiện ích, trong đó nhắn tin trực tuyến là tính năng hay và được nhiều người dùng sử dụng nhất. Một trang web khác là Slack - một chương trình nhắn tin tức thời được thiết kế bởi Slack Technologies và thuộc sở hữu của Salesforce. Slack là một ứng dụng nhắn tin cho mục đích liên lạc chuyên nghiệp và tổ chức, được rất nhiều các công ty, doanh nghiệp lớn tin dùng nhớ các tính năng nổi bật về soạn thảo, bố cục tin nhắn, thông báo, hiển thị, ...

Đặc điểm chung của các trang web này là đã đáp ứng được đầy đủ các yêu cầu cơ bản của một trang web nhắn tin trực tuyến. Các trang web đều có giao diện sáng sủa, dễ dùng, có hướng dẫn rõ ràng cho người mới sử dụng. Các tính năng được phát triển đầy đủ. Các trang web cũng được quản lý bởi các công ty có đội ngũ nhân viên nhiều kinh nghiệm, luôn cập nhật, thêm mới các tính năng theo yêu cầu, phản hồi của khách hàng

Tuy nhiên, đa phần các ứng dụng nhắn tin hiện nay đều được tích hợp trong các ứng dụng mạng xã hội. Zalo là một ứng dụng mạng xã hội có rất nhiều các tính năng như đăng bài, nhắn tin, call video, ... nên nó có thể được sử dụng cho rất nhiều mục đích học tập, công việc, giải trí, ... Chính vì điều này nên nó không phù hợp để sử dụng trong môi trường công việc, học tập. Việc sử dụng một mạng xã hội có yếu tố giải trí có thể sẽ làm ảnh hưởng đến chất lượng công việc, học tập. Tin nhắn của zalo chỉ hỗ trợ ở dạng thường, không có tính năng soạn thảo, định dạng văn bản nên không thể hỗ trợ cho việc soạn thảo văn bản cho mục đích cụ thể

Vì vậy, mục tiêu của đồ án này sẽ hướng tới việc tạo ra một sản phẩm là ứng dụng nhắn tin trực tuyến thuần văn bản chuyên nghiệp trong các tổ chức, doanh nghiệp. Sản phẩm được xây dựng phục vụ cho mục đích trao đổi văn bản chuyên nghiệp, có mục đích rõ ràng, cụ thể. Đối tượng sử dụng có thể là các nhân viên trong công ty, thành viên trong cùng một dự án. Sản phẩm phục vụ cho mục đích trao đổi, thảo luận thông tin theo hướng thuần văn bản, phục vụ cho hoạt động của các dự án, tổ chức, doanh nghiệp

1.3 Định hướng giải pháp

Từ vấn đề đã nêu trên, tôi hướng tới việc sử dụng công nghệ web là công nghệ cốt lõi để phát triển sản phẩm trong đồ án này. Các công nghệ được sử dụng bao gồm nền tảng web là nơi sản phẩm được triển khai vì mục tiêu hướng tới một sản phẩm hoạt động trên mạng Internet. Cơ sở dữ liệu quan hệ được sử dụng để đảm

bảo tính toàn vẹn của dữ liệu có tính ràng buộc cao trong quá trình sử dụng và vận hành hệ thống. Cơ sở dữ liệu phi quan hệ được sử dụng để lưu trư các dữ liệu rời rạc, số lượng lớn và cần truy vấn nhanh.

Kiến trúc được áp dụng để xây dựng hệ thống là kiến trúc Microservice để đảm bảo phân tách vai trò các thành phần trong hệ thống được triển khai một cách rõ ràng, phù hợp với các ứng dụng trên nền tảng web. Mô hình Microservice phân chia phần mềm thành nhiều service nhỏ tồn tài độc lập, điều này đồng nghĩa với việc mọi hoạt động xử lý, lưu trữ và yêu cầu dữ liệu đều riêng biệt. Ưu điểm của mô hình Microservice là khả năng hoạt động độc lập, linh hoạt, có tính chuyên biệt cao do không bị ràng buộc bởi các yêu cầu chung, nên mỗi service nhỏ có thể tự do lựa chọn công nghệ, nền tảng phù hợp. Mô hình Microservice giúp thuận tiện trong nâng cấp, mở rộng, xử lý lỗi; Việc xử lý lỗi, nâng cấp, bảo trì service hoàn toàn độc lập sẽ không làm gián đoạn quá trình vận hành của cả hệ thống. Nhờ vậy, những phiên bản mới có thể được cập nhật thường xuyên. Quá trình kiểm thử cũng được đơn giản hóa. Với các service nhỏ, việc quản lý, tính toán, quản lý sẽ đơn giản, nhanh chóng hơn so với cả phần mềm

Từ những công nghệ đó, sản phẩm được hướng tới là một trang web nhắn tin trực tuyến cho phép người dùng chia sẻ, trao đổi, thu nhận thông tin, tài liệu từ các người dùng khác nhanh, hiệu quả. Sản phẩm cần phải đảm bảo được tính toàn vẹn dữ liệu trong quá trình vận hành.

Đóng góp chính của đồ án này là tập trung vào xây dựng một sản phẩm hỗ trợ người dùng trao đổi, chia sẻ thông tin hướng văn bản nhanh, chính xác và hiệu quả nhất. Kết quả đạt được là một trang web nhắn tin trực tuyến trong đó nội dung xử lý, soạn thảo, hiển thị tin nhắn sẽ là trọng tâm chính của sản phẩm. Cùng với đó sẽ bao gồm một số tính năng cần thiết đối với một trang web nhắn tin trực tuyến ứng dụng trong doanh nghiệp, tổ chức

1.4 Bố cục đồ án

Phần còn lại của báo cáo đồ án tốt nghiệp này được tổ chức như sau.

Chương 2 trình bày về hiện trạng hiện tại của các sản phẩm có nội dung tương tự với báo cáo đồ án này, phân tích các sản phẩm đó và đưa ra phần cải tiến để giải quyết vấn đề và hạn chế liên quan tới quy trình nghiệp vụ trong sản phẩm. Tiếp theo đến phần tổng quan các yêu cầu chức năng của sản phẩm được thiết kế, trong đó trình bày các biểu đồ use case tổng quát và phân mức mô tả các chức năng chính của sản phẩm kèm theo quy trình nghiệp vụ của sản phẩm. Kế đến là phần nội dung đặc tả các chức năng chính và các yêu cầu phi chức năng cho sản phẩm được phát triển.

Trong chương 3, các công nghệ và kĩ thuật được sử dụng trong việc tạo ra sản phẩm được đưa ra và phân tích ở đây. Trong đó bao gồm công nghệ về lưu trữ dữ liệu, các công cụ dùng để phát triển sản phẩm như ngôn ngữ lập trình, các framework hỗ trợ phát triển kèm theo ưu nhược điểm của từng công nghệ và lí do lựa chọn các công nghệ này trong quá trình phát triển sản phẩm.

Kế tiếp là chương 4 đề cập tới quá trình xây dựng phần mềm. Các nội dung được đề cập trong chương này bao gồm việc thiết kế về mặt kiến trúc sản phẩm, thiết kế tổng quan các thành phần chức năng. Kế đến là nội dung thiết kế chi tiết các bộ phận của sản phẩm gồm thiết kế giao diện, thiết kế lớp và cơ sở dữ liệu. Cuối cùng đề cập tới phần xây dựng sản phẩm cùng với kiểm thử và cách thức triển khai sản phẩm.

Chương 5 nêu ra các đóng góp chính của đề tài, trọng tâm là đóng góp việc xây dựng ra tính năng hỗ trợ quy trình soạn thảo tin nhắn văn bản hoàn chỉnh cho sản phẩm.

Chương kế tiếp, chương 6 nêu các kết luận trong quá trình thực hiện đồ án, bao gồm các bài học kinh nghiệm, phân tích và so sánh sản phẩm đạt được. Cùng với đó là hướng phát triển tương lai cho sản phẩm. Phần phụ lục cuối đồ án tốt nghiệp trình bày thêm đặc tả của một số use case của hệ thống.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

2.1 Tổng quan

Tiếp theo chương giới thiệu về đề tài, trong chương này tôi sẽ trình bày chi tiết hơn về các nội dung liên quan tới khảo sát đề tài. Kế tiếp đó là nội dung tổng quan về các chức năng chính của sản phẩm được phát triển, kèm theo quy trình nghiệp vụ quan trọng. Cuối cùng bao gồm đặc tả một số chức năng quan trọng cùng với đó là các yêu cầu phi chức năng đối với sản phẩm cần đạt được.

2.2 Khảo sát hiện trạng

Hiện nay trên nền tảng Internet đã có nhiều sản phẩm, nhiều giải pháp để giải quyết vấn đề chia sẻ, trao đổi thông tin trực tuyến. Các sản phẩm này đều đã tạo được nhiều tính năng tiện ích cho nhiều đối tượng người dùng khác nhau. Một số sản phẩm nổi bật sẽ được phân tích dưới đây.

Đầu tiên là sản phẩm Slack, một phần mềm nhắn tin tức thời được thiết kế bởi Slack Technologies và thuộc sở hữu của Salesforce. Slack là một ứng dụng nhắn tin được nhiều chuyên gia đánh giá là nằm trong top các phần mềm nhăn tin doanh nghiệp. Nó cung cấp các kênh công khai và riêng tư, chia sẻ tệp dễ dàng và tích hợp quan trọng với Zoom và Google Drive. Nó cũng rất thân thiện với người dùng và hoạt động trên được nhiều thiết bị khác nhau như: máy tính để bàn, điện thoại di động và thậm chí cả máy tính bảng,.. Với Slack người dùng có quyền truy cập ngay vào các cuộc hội thoại trước đó sau khi tham gia thành công một nhóm hiện có. Dễ dàng tìm thấy các cuộc thảo luận trước đây, cho phép người dùng bắt kịp tiến độ dự án và xem câu hỏi nào đã được đặt ra. Slack Có mức độ tùy chỉnh cao. Người dùng có thể tùy chỉnh các cảnh báo, từ tắt tiếng chúng đối với các kênh không quan trọng đối với công việc của họ cho đến bật và tắt hoàn toàn thông báo. Người dùng thậm chí có thể tắt tiếng tất cả các thông báo trong những giờ cụ thể trong ngày. Salck hỗ trợ người dùng chia sẻ tài liệu và tệp với mọi người theo thời gian thực trong cuộc hội thoại Slack. Có khả năng nhận xét về tài liệu và nhận phản hồi ngay lập tức, đồng thời có thể chia sẻ các tệp, tài liệu và ảnh từ máy tính để bàn hoặc dịch vụ lưu trữ đám mây của người dùng.

Sản phẩm thứ hai có thể kể đến là Zalo, ứng dụng nhắn tin nhanh đa nền tảng được phát triển bởi công ty VNG ở Việt Nam. Zalo là một nền tảng mãng xã hội với rất nhiều chức năng trong đó, chức năng nhắn tin là chức năng được người dùng yêu thích và sử dụng nhiều nhất. Rất nhiều các doanh nghiệp sử dụng zalo là phương thức giao tiếp, trao đổi thông tin trong công việc. Zalo co nhiều ưu điểm vượt trội có thể kể đến như hhả năng chia sẻ thông tin (file, hình ảnh với chất lượng HD) lên

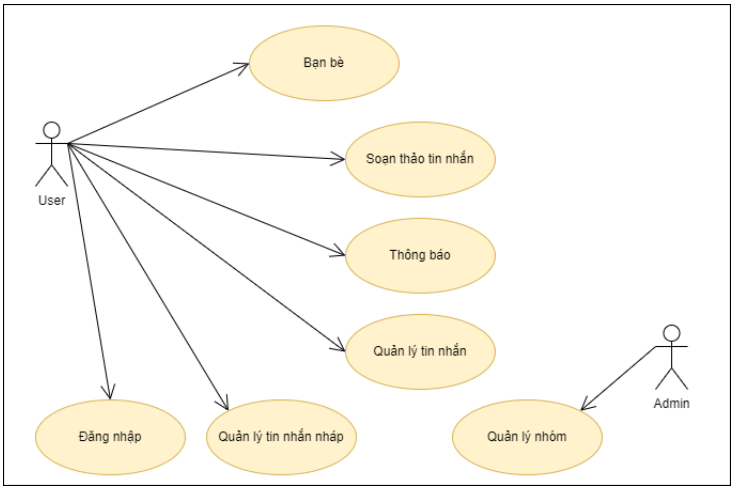
tới 20MB và có thể thực hiện với nhiều file cùng một lúc giúp Zalo là lựa chọn số một của dân công chức, sinh viên hay giới văn phòng. Việc đăng ký tài khoản zalo cực kỳ đơn giản, Không cần khai báo bất kỳ thông tin nào cũng không cần check mail để xác nhận, người dùng chỉ cần đăng kí bằng chính số điện thoại của mình , thậm chí cho dù có bị mất tài khoản, mật khẩu, người dùng cũng chỉ cần duy nhất số điện thoại để lấy lại. Tính năng trò chuyện nhóm, Zalo cho phép người dùng có thể tạo và tham gia các nhóm trò chuyện, chia sẻ thông tin với bạn bè, đồng nghiệp và gia đình một cách thuận tiện nhất; Trò chuyện và kết bạn cùng lúc 50 thành viên khác trong nhóm.

Từ những khảo sát trên đây, em nhận thấy các tính năng quan trọng cần được phát triển trong sản phẩm bao gồm các tính năng được nêu dưới đây.

- Tính năng đăng nhập/đăng xuất, đây là tính năng bắt buộc phải có để phân quyền người sử dụng, đồng thời cung cấp các tính năng cho từng đối tượng cụ thể.
- Các tính năng liên quan đến soạn thảo tin nhắn bao gồm: thêm/sửa/xóa tin nhắn, định dạng văn bản, đăng tải hình ảnh, đăng tải file, tag người dùng khác
- Các tính năng liên quan đến tương tác với tin nhắn bao gồm like tin nhắn, bình luận vào tin nhắn, đánh dấu tin nhắn quan trọng
- Các tính năng liên quan đến tìm kiếm bao gồm tìm kiếm người dùng theo, từ khóa, tìm kiếm người dùng trong nhóm
- Các tính năng liên quan đến tài khoản người dùng bao gồm: cập nhật profile, thay đổi ảnh đại diện, gửi lời mời kết bạn, chấp nhận/từ chối lời mời kết bạn
- Các tính năng liên quan đến nhóm bao gồm: cập nhật ảnh đại diện nhóm, thêm/xóa người trong nhóm, nâng quyền user lên admin trong nhóm, tạo/xóa nhóm
- Các tính năng liên quan đến tin nhắn nháp bao gồm: tạo/sửa/xóa tin nhắn nháp, chuyển tin nhắn nháp thành tin nhắn thường gửi vào nhóm
- Các tính năng liên quan đến hiển thị tin nhắn bao gồm: hiển thị tin nhắn theo luồng các tin nhắn của bản thân, hiển thị tin nhắn theo luồng các tin nhắn mà bản thân bình luận, hiện thị tin nhắn theo luồng các tin nhắn mà bản thân được người khác nhắc đến
- Các tính năng liên quan đến nhận thông báo bao gồm: thông báo bạn bè, thông báo tương tác tin nhắn(like, comment), thông báo nhóm

2.3 Tổng quan chức năng

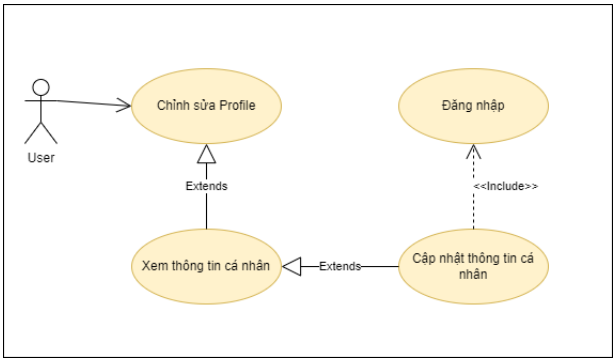
2.3.1 Biểu đồ use case tổng quát



Hình 2.1: Biểu đồ use case tổng quan

Hình 5.3 ở đây là biểu đồ use case tổng quát mô tả chức năng phần mềm. Người dùng muốn sử dụng hệ thống cần tạo tài khoản, sau đó đăng nhập vào hệ thống để sử dụng các chức năng của hệ thống.

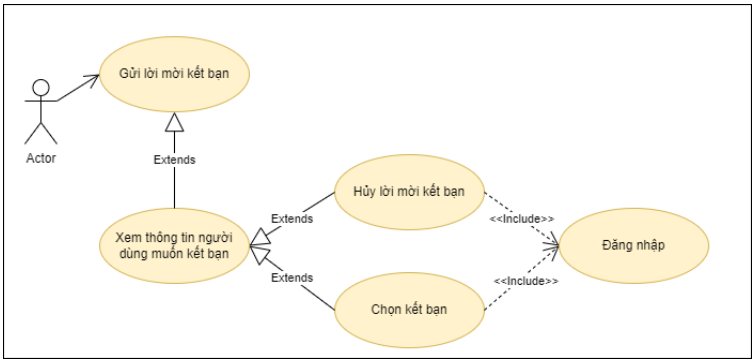
2.3.2 Biểu đồ use case phân rã cập nhật Profile cá nhân



Hình 2.2: Biểu đồ use case phân rã cập nhật Profile cá nhân

Hình 5.3 này mô tả use case phân rã cập nhật profile cá nhân của người dùng. Trong đó người dùng có thể tùy chọn thay đổi các thông tin cá nhân về bản thân như tên, số điện thoại, ngày sinh, ...

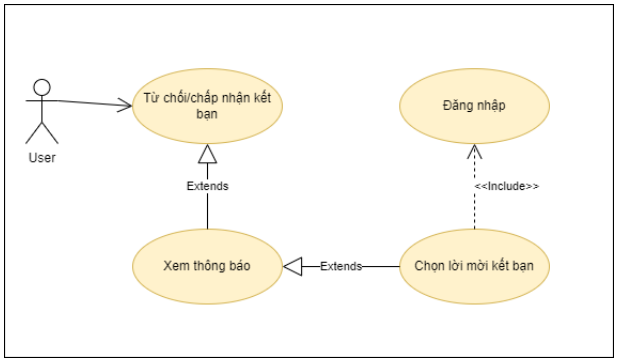
2.3.3 Biểu đồ use case phân rã gửi lời mời kết bạn



Hình 2.3: Biểu đồ use case phân rã gửi lời mời kết bạn

Hình 5.3 này mô tả use case phân rã gửi lời mời kết bạn cho người dùng khác. Trong đó sau khi xem thông tin cá nhân của người dùng khác, người dùng có thể gửi lời mời kết bạn cho người dùng đó và đợi phản hồi. Trong trường hợp đã gửi lời mời kết bạn và muốn hủy, người dùng có thể lựa chọn hủy lời mời kết bạn, lời mời kết bạn sẽ bị xóa.

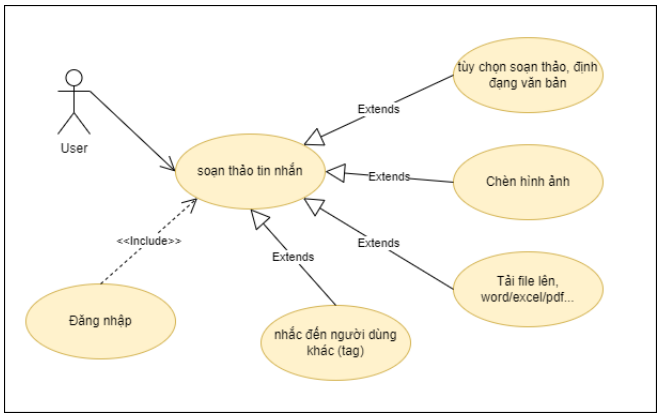
2.3.4 Biểu đồ use case phân rã chấp nhận/từ chối kết bạn



Hình 2.4: Biểu đồ use case phân rã chấp nhận/từ chối kết bạn

Hình 5.3 này mô tả use case phân rã chức năng chấp nhận/ từ chối kết bạn. Trong đó sau khi xem thông báo, người dùng sẽ thấy các lời mời kết bạn từ các người dùng khác. Người dùng có thể lựa chọn đồng ý kết bạn hoặc từ chối kết bạn. Nếu đồng ý kết bạn, 2 người sẽ trở thành bạn bè và có thể nhắn tin với nhau. Nếu từ chối lời mời kết bạn, lời mời kết bạn sẽ bị xóa.

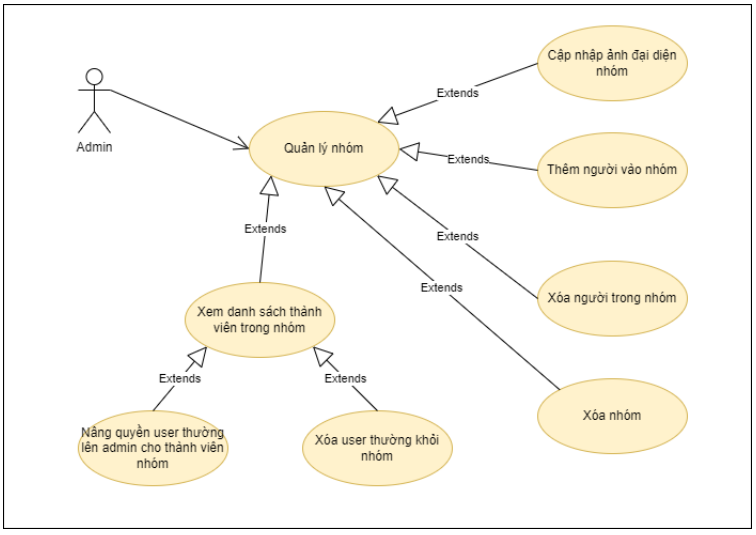
2.3.5 Biểu đồ use case phân rã soạn thảo tin nhắn



Hình 2.5: Biểu đồ use case phân rã soạn thảo tin nhắn

Hình 5.3 này mô tả use case phân rã chức năng soạn thảo tin nhắn. Khi soạn thảo tin nhắn, người dùng có quyền sử dụng các tùy chọn định dạng văn bản, chèn hình ảnh, tài file lên, nhắc/đánh dấu người dùng khác (tag).

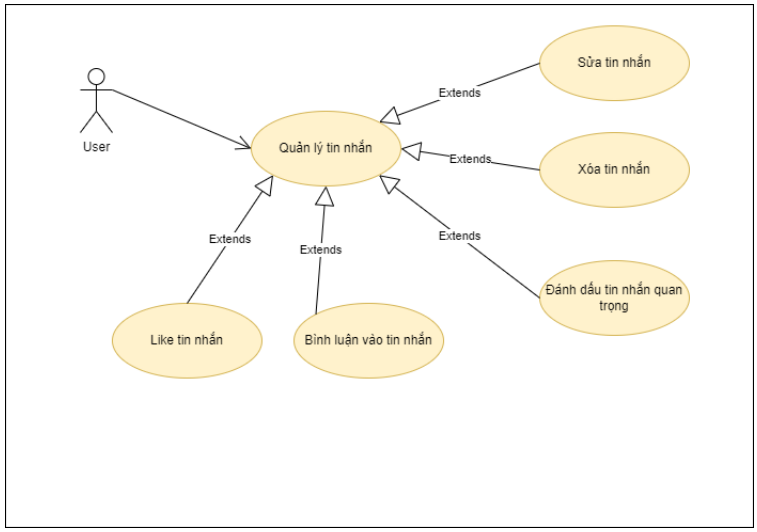
2.3.6 Biểu đồ use case phân rã quản lý nhóm



Hình 2.6: Biểu đồ use case phân rã quản lý nhóm

Hình 5.3 này mô tả use case phân rã chức năng quản lý nhóm cho admin trong nhóm. Admin có quyền thay đổi ảnh đại diện nhóm, thêm người mới vào nhóm, xóa nhóm. Trong nhóm có phân quyền 2 cấp là user và admin, admin có quyền nâng quyền cho người dùng từ user lên admin hoặc xóa người dùng user thường trong nhóm

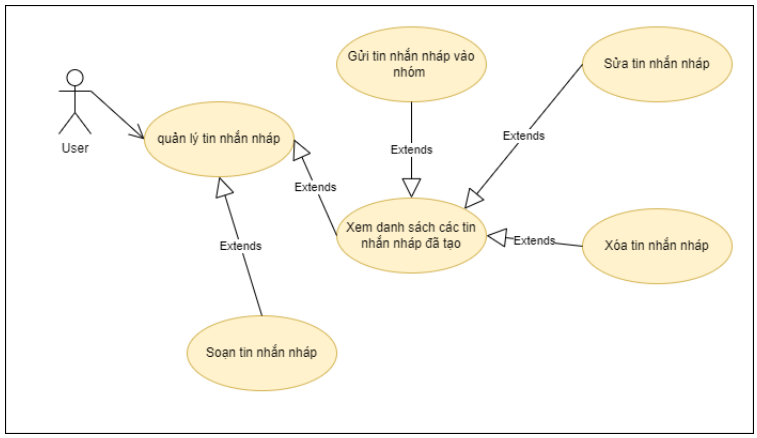
2.3.7 Biểu đồ use case phân rã quản lý tin nhắn



Hình 2.7: Biểu đồ use case phân rã quản lý tin nhắn

Hình 5.3 này mô tả use case phân rã chức năng quản lý tin nhắn của người dùng. Người dùng có quyền sửa, xóa, đánh dấu tin nhắn quan trọng, like, bình luận vào tin nhắn của mình. Đối với tin nhắn của người khác người dùng chỉ có quyền like và bình luận.

2.3.8 Biểu đồ use case phân rã quản lý tin nhắn nháp



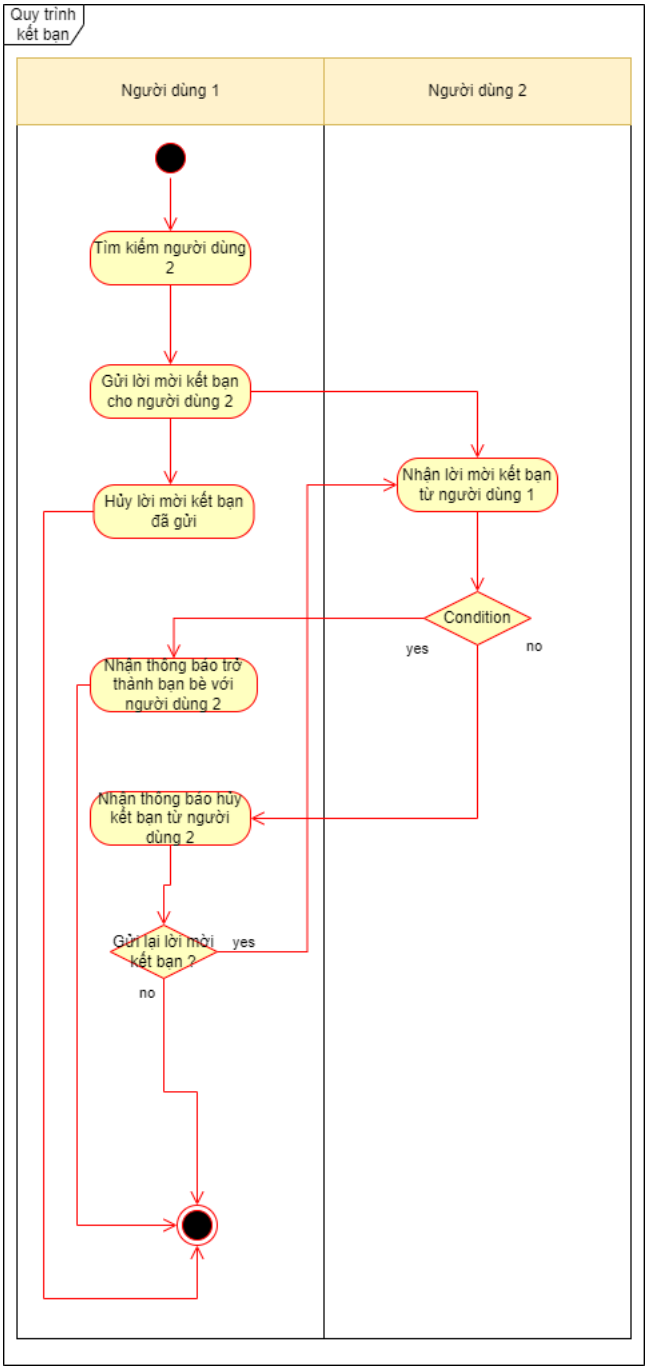
Hình 2.8: Biểu đồ use case phân rã quản lý tin nhắn nháp

Hình 5.3 này mô tả use case phân rã chức năng quản lý tin nhắn nháp của người dùng. Người dùng có quyền xem danh sách các tin nhắn nháp đã tạo, sửa/xóa các tin nhắn nháp đã tạo, gửi tin nhắn nháp vào nhóm, tạo tin nhắn nháp mới.

2.3.9 Quy trình nghiệp vụ

Trong hệ thống có nhiều quy trình nghiệp vụ dẫn tới các hoạt động của hệ thống, trong đó một trong những quy trình phức tạp là quy trình kết bạn. Quy trình này

có xuất phát điểm từ khi một người dùng gửi lời mời kết bạn cho một người dùng khác và người dùng nhận được lời mời kết bạn sẽ phản hồi lại. Mô tả của quy trình được biểu diễn trong biểu đồ hoạt động dưới đây (biểu diễn trên hình 5.3).



Hình 2.9: Biểu đồ hoạt động cho quy trình kết bạn

2.4 Đặc tả chức năng

2.4.1 Đặc tả use case 'Đăng nhập'

Bảng dưới đây đặc tả cho use case đăng nhập, đối tượng áp dụng bao gồm mọi người sử dụng hệ thống đã đăng kí tài khoản.

Mã Use Case	UCDN	**Tên Use Case**	Đăng nhập
Tác nhân	Người dùng		
Tiền điều kiện	Không		
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4	Người dùng Hệ thống Người dùng Hệ thống	Chọn tính năng đăng nhập Hiển thị giao diện đăng nhập Nhập thông tin tên tài khoản, mật khẩu, lựa chọn đăng nhập Thông báo đăng nhập thành công
Luồng sự kiện thay thế	4a 4b	Hệ thống Hệ thống	Thông báo lỗi nếu người dùng nhập sai tài khoản hoặc mật khẩu Thông báo lỗi nếu người dùng chưa nhập đủ thông tin tên tài khoản, mật khẩu
Hậu điều kiện	Không		

Bảng 2.1: Đăc tả Use Case Đăng Nhập

2.4.2 Đặc tả use case 'Quản lý tin nhắn'

Bảng dưới đây đặc tả cho use case quản lý tin nhắn, đối tượng áp dụng cho mọi người dùng đã đăng ký tài khoản, người dùng có thể tạo tin nhắn và sửa/xóa tin nhắn của mình.

Mã Use Case	UC01	**Tên Use Case**	Quản lý tin nhắn
Tác nhân	Người dùng		
Tiền điều kiện	Người dũng đã đăng nhập		
Mã : UC011 - Soạn tin nhắn			
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4	Người dùng Hệ thống Người dùng Hệ thống	Chọn soạn thảo tin nhắn Hiển thị giao diện soạn thảo tin nhắn Nhập nội dung tin nhắn, định dạng văn bản, chèn ảnh, tải file lên rồi ấn gửi Gửi tin nhắn thành công, hiển thị lại giao diện soạn thảo tin nhắn như ban đầu
Luồng sự kiện thay thế	4a	Hệ thống	Kết thúc use case nếu người dùng không ấn gửi tin nhắn
Mã : UC012 - Sửa tin nhắn			
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4 5 6	Người dùng Hệ thống Người dùng Hệ thống Người dùng Hệ thống	Chọn sửa tin nhắn Hiển thị giao diện sửa tin nhắn cùng với nội dung tin nhắn cũ Nhập nội dung tin nhắn, định dạng văn bản, chèn ảnh, tải file lên rồi xác nhận sửa tin nhắn Hiện thị thông báo bạn có chắc chắn muốn sửa tin nhắn Xác nhận sửa tin nhắn Hiển thị tin nhắn mới sau khi sửa
Luồng sự kiện thay thế	4a 6a	Hệ thống Hệ thống	Người dùng chọn nút hủy sửa tin nhắn, hệ thống hiển thị lại danh sách tin nhắn chứa tin nhắn ban đầu, kết thúc use case Người dùng chọn không, tin nhắn không được sửa,hệ thống hiển thị lại danh sách tin nhắn chứa tin nhắn ban đầu, kết thúc use case
Mã : UC013 - Xóa tin nhắn			
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4	Người dùng Hệ thống Người dùng Hệ thống	Chọn xóa tin nhắn Hiện thị thông báo bạn có chắc chắn muốn xóa Xác nhận xóa tin nhắn Hiển thị lại danh sách tin nhắn không chứa tin nhắn đã xóa
Luồng sự kiện thay thế	4a	Hệ thống	Người dùng xác nhận không xóa tin nhắn, hệ thống hiển thị lại danh sách tin nhắn chứa tin nhắn ban đầu, kết thúc use case
Hậu điều kiện	Không		

Bảng 2.2: Đăc tả Use Case Kết bạn

2.4.3 Đặc tả use case 'Quản lý thành viên nhóm'

Bảng dưới đây đặc tả cho use case quản lý nhóm, đối tượng áp dụng cho admin nhóm, admin có quyền thêm thành viên, xóa thành viên thường, nâng quyền user.

Mã Use Case	UC02	**Tên Use Case**	Quản lý thành viên nhóm
Tác nhân	Admin nhóm		
Tiền điều kiện	Người dũng đã đăng nhập		
Mã : UC021 - Thêm thành viên vào nhóm			
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4	Admin Hệ thống Admin Hệ thống	Chọn thêm thành viên vào nhóm Hiển thị giao diện thêm thành viên Nhập từ khóa tên thành viên, xác nhận thêm thành viên Hiển thị thêm thành viên thành công
Luồng sự kiện thay thế	4a	Hệ thống	Kết thúc use case nếu admin không lựa chọn thêm thành viên
Mã : UC022 - Xóa thành viên trong nhóm			
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4 5 6	Admin Hệ thống Admin Hệ thống Admin Hệ thống	Chọn xem danh sách thành viên Hiển thị danh sách thành viên cùng các tùy chọn Chọn xóa thành viên Hiện thị thông báo bạn có chắc chắn muốn xóa thành viên Xác nhận xóa thành viên Hiển thị lại danh sách thành viên mới
Luồng sự kiện thay thế	4a 6a	Hệ thống Hệ thống	Admin không chọn xóa thành viên, kết thúc use case Admin xác nhận không xóa thành viên, thành viên không bị xóa ,hệ thống hiển thị lại danh sách thành viên như ban đầu, kết thúc use case

Mã : UC023 - Nâng quyền user lên admin			
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1	Admin	Chọn xem danh sách thành viên
	2	Hệ thống	Hiện thị danh sách thành viên cùng các tùy chọn
	3	Admin	Chọn nâng quyền cho thành viên
	4	Hệ thống	Hiển thị thông báo xác nhận nâng quyền cho thành viên
	5	Admin	Xác nhận nâng quyền cho thành viên
	6	Hệ thống	Hiển thị lại danh sách thành viên mới
Luồng sự kiện thay thế	4a	Hệ thống	Admin không ấn nút nâng quyền thành viên, kết thúc use case
	6a	Hệ thống	Admin xác nhận không nâng quyền cho thành viên, hiển thị lại danh sách thành viên ban đầu, kết thúc use case
Hậu điều kiện	Không		

Bảng 2.3: Đăc tả Use Case Quản lý thành viên nhóm

2.4.4 Đặc tả use case 'Gửi lời mời kết bạn'

Bảng dưới đây đặc tả cho use case gửi lời mời kết bạn, đối tượng áp dụng cho mọi người dùng đã đăng ký tài khoản, người dùng có thể gửi lời mời kết bạn cho một người dùng khác và đợi phản hồi.

Mã Use Case	UC03	**Tên Use Case**	Gửi lời mời kết bạn
Tác nhân	Người dùng		
Tiền điều kiện	Người dũng đã đăng nhập		
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1	Người dùng	Nhập từ khóa, tên người dùng, chọn tìm kiếm
	2	Hệ thống	Hiển thị trang thông tin cá nhân người được tìm kiếm
	3	Người dùng	Người dùng xác nhận gửi lời mời kết bạn
	4	Hệ thống	Hệ thống hiển thị lời mời kết bạn đã được gửi, đợi phản hồi
	5	Người dùng	Người dùng chọn hủy lời mời kết bạn đã gửi
	6	Hệ thống	Hệ thống hiện thị lời mời kết bạn bị xóa, về giao diện ban đầu
Luồng sự kiện thay thế	2a	Hệ thống	Người dùng không chọn tìm kiếm, kết thúc use case
	4a	Hệ thống	Người dùng không chọn gửi lời mời kết bạn, kết thúc use case
	6a	Hệ thống	Người dùng không ấn hủy lời mời, hệ thống vẫn hiển thị giao diện lời mời kết bạn đã gửi, kết thúc use case
Hậu điền kiện	Không		

Bảng 2.4: Đăc tả Use Case Gửi lời mời kết bạn

2.4.5 Đặc tả use case 'Phản hồi lời mời kết bạn'

Bảng dưới đây đặc tả cho use case phản hồi lời mời kết bạn, đối tượng áp dụng cho mọi người dùng đã đăng ký tài khoản và nhận được lời mời kết bạn từ người dùng khác, người dùng có thể chấp nhận hoặc từ chối kết bạn.

Mã Use Case	UC04	**Tên Use Case**	Phản hồi lời mời kết bạn
Tác nhân	Người dùng		
Tiền điều kiện	Người dũng đã đăng nhập		
	STT	**Thực hiện bởi**	**Hành động**
Luồng sự kiện chính	1 2 3 4	Người dùng Hệ thống Người dùng Hệ thống	Chọn xem thông báo Hiển thị thông báo, bao gồm các thông báo lời mời kết bạn Người dùng xác nhận phản hồi lời mời kết bạn Hệ thống hiển thị thông báo hai người dùng đã trở thành bạn bè
Luồng sự kiện thay thế	2a 4a	Hệ thống Hệ thống	Người dùng không chọn xem thông báo, kết thúc use case Người dùng không đồng ý kết bạn, hệ thống hiển thị thông báo lời mời kết bạn đã bị hủy, kết thúc use case
Hậu điền kiện	Không		

Bảng 2.5: Đăc tả Use Case Phản hồi kết bạn

2.5 Yêu cầu phi chức năng

2.5.1 Tính an toàn

Trong quá trình vận hành, có thể xảy ra các sự cố gây ra thiệt hại nghiêm trọng đến cơ sở dữ liệu như hỏng hóc ổ cứng, cần phải có cơ chế sao lưu các dữ liệu được lưu trữ này để có khả năng phục hồi được dữ liệu khi sự cố được khắc phục. Có thể sử dụng các phương pháp lưu trữ sao lưu hoặc lưu trữ lịch sử chỉnh sửa dữ liệu trong quá trình vận hành để khôi phục được dữ liệu.

2.5.2 Tính bảo mật

Cần phải đảm bảo tính bảo mật, đảm bảo đối tượng sử dụng chỉ được truy cập những phần dữ liệu được phép. Đảm bảo phân quyền đúng đối tượng và có cơ chế bảo vệ những truy cập trái phép đến các tài nguyên.

2.5.3 Giao diện

Giao diện cần phải được xây dựng phù hợp với người sử dụng. Đảm bảo các yếu tố dễ dàng, thân thiện với người dùng. Các phần của giao diện phải đồng bộ, thống nhất, đơn giản, dễ tiếp cận, dễ sử dụng, có hướng dẫn chi tiết.

Trong quá trình vận hành, có thể xảy ra các sự cố gây ra thiệt hại nghiêm trọng đến cơ sở dữ liệu như hỏng hóc ổ cứng, cần phải có cơ chế sao lưu các dữ liệu được lưu trữ này để có khả năng phục hồi được dữ liệu khi sự cố được khắc phục. Có thể sử dụng các phương pháp lưu trữ sao lưu hoặc lưu trữ lịch sử chỉnh sửa dữ liệu trong quá trình vận hành để khôi phục được dữ liệu.

2.5.4 Một số đặc tính cần có của sản phẩm

Đầu tiên là về tính sẵn có. Hệ thống luôn phải sẵn sàng hoạt động kể từ khi triển khai để đảm bảo quá trình sử dụng của người dùng ít khi hoặc không bao giờ gặp vấn đề về gián đoạn.

Tiếp đến là tính khả dụng, hệ thống cần phải hoạt động ổn định, thực hiện các yêu cầu từ người dùng một cách nhanh chóng, chính xác, đảm bảo được các quá trình tìm kiếm, xử lý, hiển thị cho người dùng một cách hiệu quả.

Kế đến là tính đúng đắn, hệ thống cần đảm bảo các quá trình hoạt động sẽ luôn đảm bảo được toàn vẹn dữ liệu, các kết quả thu được luôn phải được đảm báo tính chính xác trong các hành vi thực hiện.

2.5.5 Hiệu năng

Cơ sở dữ liệu cần được tối ưu, tránh dư thừa trong quá trình lưu trữ dữ liệu. Các kết quả truy vấn dữ liệu không được vượt quá 5 giây. Đảm bảo phần tài nguyên còn trống đạt tối thiểu 20

2.5.6 Các yêu cầu khác

Dữ liệu cần được lưu trữ lâu dài, trong trường hợp phần dữ liệu đã lâu không được sử dụng sẽ được chuyển sang lưu trữ ở một kho dữ liệu khác để tránh quá tải đến nơi lưu trữ dữ liệu hiện tại, vừa đảm bảo sao lưu dữ liệu, vừa để sử dụng cơ sở dữ liệu lưu trữ thường ngày hiệu quả. Khi cần sẽ truy cập vào kho dữ liệu đó để lấy ra và sử dụng.

Hệ thống được tiến hành bảo trì định kì theo chu kì 3 tháng.

2.6 Kết chương

Chương này đã phân tích chi tiết về khảo sát hiện trạng của bài toán đang xét tại thời điểm hiện tại, các giải pháp đã được triển khai cùng với đó là những hạn chế còn gặp, từ đó đề xuất ra các cải tiến. Tiếp đến là triển khai phân tích thiết kế về các chức năng cần có trong sản phẩm được phát triển, kèm theo đó là các yêu cầu về nghiệp vụ, yêu cầu chức năng và phi chức năng cho sản phẩm.

Từ những phân tích trên đây, tôi tiến hành sử dụng các công nghệ và giải pháp được trình bày trong chương tiếp theo – Chương 3.

CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

3.1 Tổng quan

Trong chương trước, tôi đã đề cập tới các nội dung về phân tích và khảo sát bài toán được giải quyết, kèm theo đó là những giải pháp thiết kế để giải quyết bài toán cùng với phân tích nghiệp vụ.

Vì vậy trong chương này, tôi sẽ giới thiệu những công nghệ, phần mềm được sử dụng để phát triển và giải quyết bài toán trong đồ án này cùng với những nguyên do lựa chọn chúng để giải quyết bài toán hiệu quả.

3.2 Công nghệ lưu trữ dữ liệu

3.2.1 Hệ quản trị cơ sở dữ liệu quan hệ

Hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System – RDBMS) là một hệ thống quản trị cơ sở dữ liệu dựa trên nền tảng mô hình quản hệ được phát biểu và giới thiệu bởi Edgar F. Codd vào năm 1970. [4]

Mô hình dữ liệu quan hệ có đặc trưng các dữ liệu được tổ chức biểu diễn dưới dạng bảng. Mỗi bảng biểu diễn các nội dung của một đối tượng hay một quan hệ nào đó. Mỗi bảng được chia ra thành các hàng và cột biểu diễn các giá trị, và là đơn vị lưu trữ dữ liệu.

Mô hình quan hệ có một số khái niệm cơ bản để xác định và biểu diễn dữ liệu. Đầu tiên là thuộc tính, thuộc tính mô tả một tính chất, hay đặc trưng của đối tương, được biểu diễn trong bảng dưới dạng cột và thuộc tính yêu cầu tên kèm kiểu giá trị cần lưu trữ. Tiếp đến là quan hệ, hay lược đồ, bao gồm một tập hợp các thuộc tính có liên quan và tên của quan hệ xác lập trên các thuộc tính đó. Kế tiếp là bộ giá trị, hay bản ghi, là các thông tin của một đối tượng trong quan hệ kể trên, được biểu diễn dưới dạng dòng trong bảng, kèm theo tên của từng thuộc tính trong bảng. Tiếp đến là một thành phần quan trọng trong mô hình quan hệ, đó là khóa. Khóa được định nghĩa là tập tối thiểu các thuộc tính để xác định duy nhất một bộ giá trị, điều này có nghĩa là các bộ giá trị sẽ phân biệt nhau thông qua khóa và không có hai bộ khác nhau nào có cùng khóa. Ngoài ra còn có một số phép toán để thực hiện trên mô hình quan hệ này, một số phép toán có thể kể đến như phép hợp, phép giao, phép lấy tích đề-các,. . .

Mô hình có nhiều ưu điểm khi dựa trên lý thuyết tập hợp để xây dựng ra các bảng, đồng thời tạo khả năng xây dựng được các truy vấn phong phú, phức tạp dựa trên các quan hệ được định nghĩa trong mô hình. Các quan hệ được định nghĩa có mối quan hệ rõ ràng, đảm bảo được các yếu tố về dữ liệu khi thiết kế tốt bao gồm

chống dữ thừa dữ liệu, đảm bảo toàn vẹn dữ liệu,. . . Tuy nhiên mô hình cũng mang nhiều tính hạn chế. Do biểu diễn rõ ràng và chặt chẽ nên cấu trúc dữ liệu không linh hoạt, khó khăn khi thay đổi sau khi đã thiết lập xong. Ngoài ra các truy vấn cần phải có ngữ nghĩa rõ ràng, chính xác, không thể hoạt động với các truy vấn mập mờ, không chính xác, không sử dụng ngôn ngữ được định nghĩa qua các phép toán.

Hệ quản trị cơ sở dữ liệu quan hệ là hệ quản trị cơ sở dữ liệu được xây dựng trên mô hình quan hệ, trong đó cung cấp cơ sở dữ liệu để lưu trữ dữ liệu theo mô hình quan hệ, cung cấp khả năng truy vấn vào dữ liệu một cách nhanh chóng, tối ưu và hiệu quả. Hệ quản trị này cung cấp khả năng thực hiện các truy vấn tới dữ liệu thông qua ngôn ngữ SQL (Structured Query Language – Ngôn ngữ truy vấn có cấu trúc) với khả năng tạo ra và thực hiện các truy vấn phức tạp. Đồng thời do được xây dựng trên nền mô hình quan hệ nên hệ quản trị cơ sở dữ liệu quan hệ cũng mang những ưu và nhược điểm tương tự với mô hình quan hệ.

Một số hệ quản trị cơ sở dữ liệu quan hệ nổi tiếng thường được sử dụng là Microsoft SQLServer, Oracle, IBM DB2, MySQL và Microsoft Access.

Từ những ưu nhược điểm của hệ quản trị cơ sở dữ liệu quan hệ, từ các yêu cầu liên quan tới lưu trữ, xử lý dữ liệu và bảo mật, an toàn, tính toàn vẹn dữ liệu, em lựa chọn hệ quản trị cơ sở dữ liệu quan hệ làm hệ thống để quản lý các dữ liệu có tính ràng buộc cao, chặt chẽ của sản phẩm.

3.2.2 Hệ quản trị cơ sở dữ liệu quan hệ MySQL

MySQL là hệ quản trị cơ sở dữ liệu quan hệ, được phát hành lần đầu tiên vào năm 1995 bởi nhà phát triển MySQL AB. Sau đó MySQL được mua lại bởi công ty Sun Microsystem vào năm 2008, và một lần nữa được chuyển quyền sở hữu sau khi Oracle mua lại Sun Microsystem. Cho đến hiện tại, MySQL vẫn đang nằm dưới sự quản lý của Oracle. Hình 5.3 là logo của MySQL.

Hình 3.1: Logo của MySQL

MySQL là một hệ quản trị cơ sở dữ liệu mã nguồn mở, được xây dựng hoạt động theo mô hình client-server (khách – chủ). Hiện tại MySQL được xây dựng hoạt động trên nhiều hệ điều hành khác nhau như Windows, MacOS, Linux, Unix và nhiều hệ điều hành khác. Đối với ngôn ngữ lập trình, MySQL cũng đã được hỗ trợ trên nhiều ngôn ngữ khác nhau như C++, Java, PHP, Ruby,. . . Ngoài phiên bản thông thường miễn phí, MySQL còn cung cấp các phiên bản nâng cao trả phí hướng tới các nhóm đối tượng khác như các doanh nghiệp, tập đoàn.

Hiện tại, MySQL là một hệ quản trị cơ sở dữ liệu rất phổ biến vì nhiều lí do khác nhau. Được phát triển dưới dạng mã nguồn mở, MySQL rất linh hoạt cùng với việc cho phép người dùng có khả năng can thiệp và chỉnh sửa theo mong muốn, đồng thời việc cài đặt và sử dụng cũng đơn giản do có được nhiều sự hỗ trợ từ các hệ điều hành cùng với các ngôn ngữ lập trình. Một lí do khác là hiệu năng của MySQL tốt, phù hợp với để lưu trữ dữ liệu cho các hệ thống từ đơn giản tới phức tạp với tốc độ xử lý truy vấn cao. Kèm với đó là tính năng phân quyền người sử dụng, các tính năng bảo mật và mã hóa tốt, đảm bảo đúng đối tượng mới có thể truy cập vào hệ thống, đặt tính an toàn của hệ thống lên hàng đầu. MySQL đã tồn tại và phát triển trong một thời gian dài nên cùng với các ưu điểm trên, độ phổ biến của MySQL đã lan rộng. Một vài trang web nổi tiếng cũng đã hoặc từng sử dụng MySQL để lưu trữ dữ liệu của mình như Facebook, Google.

Từ những ưu điểm kể trên, em lựa chọn MySQL là hệ quản trị cơ sở dữ liệu được sử dụng trong đồ án này.

3.2.3 Cơ sở dữ liệu phi quan hệ

Cơ sở dữ liệu NoSQL là Cơ sở dữ liệu được xây dựng dành riêng cho mô hình dữ liệu và có sơ đô linh hoạt để xây dựng các ứng dụng hiện đại. Cơ sở dữ liệu NoSQL được công nhận rộng rãi vì khả năng dễ phát triển, chức năng cũng như hiêu năng ở quy mô lớn. Các loại cơ sở dữ liệu này được tối ưu hóa dành riêng cho các ứng dụng yêu cầu mô hình dữ liệu linh hoạt có lượng dữ liệu lớn và độ trễ thấp, có thể đạt được bằng cách giảm bớt một số hạn chế về tính nhất quán của dữ liệu của các cơ sở dữ liệu khác.

Các cơ sở dữ liệu NoSQL được thiết kế cho các mẫu truy cập dữ liệu, bao gồm các ứng dụng có độ trễ thấp. Cơ sở dữ liệu tìm kiếm NoSQL được thiết kế để phục vụ phân tích dữ liệu có cấu trúc chưa hoàn chỉnh. Cơ sở dữ liệu NoSQL thường phải đánh đổi bằng cách nới lỏng một số thuộc tính ACID này của cơ sở dữ liệu quan hệ để có mô hình dữ liệu linh hoạt hơn có khả năng thay đổi quy mô theo chiều ngang. Việc này biến các cơ sở dữ liệu NoSQL thành lựa chọn tuyệt vời cho các trường hợp sử dụng cần thông lượng cao, độ trễ thấp cần thay đổi quy mô theo

chiều ngang vượt qua giới hạn của một phiên bản duy nhất. Cơ sở dữ liệu NoSQL thường có tính phân mảnh cao do các mẫu truy cập khóa-giá trị có khả năng tăng quy mô bằng cách sử dụng kiến trúc được phân phối để tăng thông lượng, đem đến hiệu năng ổn định với quy mô gần như không giới hạn.

Từ những ưu nhược điểm của cơ sở dữ liệu phi quan hệ, từ các yêu cầu liên quan tới lưu trữ, xử lý dữ liệu và tốc độ truy vấn em lựa chọn cơ sở dữ liệu phi quan hệ để quản lý các dữ liệu có tính rời rạc, số lượng lớn và cần truy vấn nhanh của sản phẩm.

3.2.4 Cơ sở dữ liệu phi quan hệ MongoDB

MongoDB lần đầu ra đời bởi MongoDB Inc., tại thời điểm đó là thế hệ 10, vào tháng Mười năm 2007, nó là một phần của sản phẩm PaaS (Platform as a Service) tương tự như Windows Azure và Google App Engine. Sau đó nó đã được chuyển thành nguồn mở từ năm 2009. MongoDB đã trở thành một trong những NoSQL database nổi trội nhất bấy giờ, được dùng làm backend cho rất nhiều website như eBay, SourceForge và The New York Times. Hình 5.3 là logo của MongoDB.

Hình 3.2: Logo của MongoDB

MongoDB là một database hướng tài liệu (document), một dạng NoSQL database. Vì thế, MongoDB sẽ tránh cấu trúc table-based của relational database để thích ứng với các tài liệu như JSON có một schema rất linh hoạt gọi là BSON. MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ các các kích cỡ và các document khác nhau. Các dữ liệu được lưu trữ trong document kiểu JSON nên truy vấn sẽ rất nhanh.

3.3 Ngôn ngữ NodeJS

Được tạo ra bởi Ryan Dahl năm 2009, Nodejs là một nền tảng (Platform) phát triển độc lập được xây dựng trên V8 JavaScript Engine – trình thông dịch thực thi mã JavaScript giúp chúng ta có thể xây dựng được các ứng dụng web như các trang video clip, các forum và đặc biệt là trang mạng xã hội phạm vi hẹp một cách nhanh chóng và dễ dàng mở rộng. NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ Window cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất. Hình 5.3 là logo của

NodeJS.

Hình 3.3: Logo của NodeJS

Node.js chứa một thư viện built-in cho phép các ứng dụng hoạt động như một Webserver mà không cần phần mềm như Nginx, Apache HTTP Server hoặc IIS. Node.js cung cấp kiến trúc hướng sự kiện (event-driven) và non-blocking I/O API, tối ưu hóa thông lượng của ứng dụng và có khả năng mở rộng cao. Mọi hàm trong Node.js là không đồng bộ (asynchronous). Do đó, các tác vụ đều được xử lý và thực thi ở chế độ nền (background processing). Đặc điểm nổi bật của Node.js là nó nhận và xử lý nhiều kết nối chỉ với một single-thread. Điều này giúp hệ thống tốn ít RAM nhất và chạy nhanh nhất khi không phải tạo thread mới cho mỗi truy vấn

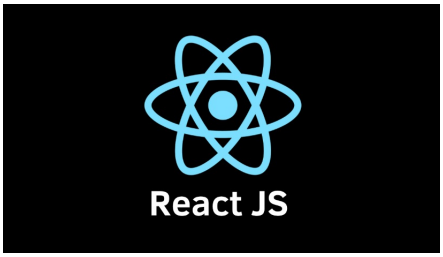
3.4 Framework của NodeJS: ExpressJS

ExpressJS là một framework ứng dụng web có mã nguồn mở và miễn phí được xây dựng trên nền tảng Node.js. ExpressJS được sử dụng để thiết kế và phát triển các ứng dụng web một cách nhanh chóng. ExpressJS cung cấp nhiều tính năng phổ biến của NodeJS dưới dạng hàm có thể dễ dàng sử dụng ở bất kỳ đâu trong chương trình, điều này sẽ giúp rút ngắn thời gian để viết code. ExpressJS cung cấp một cơ chế định tuyến nâng cao giúp duy trì trạng thái của trang web.Hình 5.3 là logo của ExpressJS.

Hình 3.4: Logo của ExpressJS

3.5 Thư viện ReactJS

ReactJS là một opensource được phát triển bởi Facebook, ra mắt vào năm 2013, bản thân nó là một thư viện Javascript được dùng để để xây dựng các tương tác với các thành phần trên website. Một trong những điểm nổi bật nhất của ReactJS đó là việc render dữ liệu không chỉ thực hiện được trên tầng Server mà còn ở dưới Client. Hình 5.3 là logo của ReactJS.



Hình 3.5: Logo của ReactJS

Nhằm giúp tăng tốc quá trình phát triển và giảm thiểu những rủi ro có thể xảy ra trong khi coding, React cung cấp cho chúng ta khả năng tái sử dụng code bằng cách đưa ra 2 khái niệm quan trọng bao gồm: JSX và Virtual DOM.

Trọng tâm chính của bất kỳ website cơ bản nào đó là những HTML documents. Trình duyệt Web đọc những document này để hiển thị nội dung của website trên máy tính, tablet, điện thoại. Trong suốt quá trình đó, trình duyệt sẽ tạo ra một thứ gọi là Document Object Model (DOM) – một tree đại diện cho cấu trúc website được hiển thị như thế nào. Lập trình viên có thể thêm bất kỳ dynamic content nào vào những dự án của họ bằng cách sử dụng ngôn ngữ JavaScript để thay đổi cây DOM. JSX (nói ngắn gọn là JavaScript extension) là một React extension giúp chúng ta dễ dàng thay đổi cây DOM bằng các HTML-style code đơn giản. ReactJS browser hiện nay hỗ trợ toàn bộ những trình duyệt Web hiện đại nên chúng ta có thể tự tin sử dụng JSX trên bất kỳ trình duyệt nào mà chúng ta đang làm việc.

Các website thông thường sẽ sử dụng HTML để cập nhật lại cây DOM cho chính bản nó (quá trình thay đổi diễn ra tự nhiên trên trang mà người dùng không cần phải tải lại trang), cách làm này sẽ ổn cho các website nhỏ, đơn giản, static website. Nhưng đối với các website lớn, đặc biệt là những website thiên về xử lý các tương tác của người dùng nhiều, điều này sẽ làm ảnh hưởng performance website cực kỳ nghiêm trọng bởi vì toàn bộ cây DOM phải reload lại mỗi lần người dùng nhấn vào tính năng yêu cầu phải tải lại trang). Tuy nhiên, nếu ta sử dụng JSX thì sẽ giúp cây DOM cập nhật cho chính DOM đó, ReactJS đã khởi tạo một thứ gọi là Virtual DOM (DOM ảo). Virtual DOM là bản copy của DOM thật trên trang đó, và ReactJS sử dụng bản copy đó để tìm kiếm đúng phần mà DOM thật cần cập

nhật khi bất kỳ một sự kiện nào đó khiến thành phần trong nó thay đổi. Với việc cập nhật đúng chỗ như vậy, nó tiết kiệm cho chúng ta rất nhiều tài nguyên cũng như thời gian xử lý. Ở các website lớn và phức tạp thì việc này là vô cùng cần thiết và quan trọng để làm tăng trải trải nghiệm của khách hàng và performance được cải thiện đáng kể.

3.6 Công nghệ lưu trữ và cache dữ liệu Redis

Redis là một kho lưu trữ cấu trúc dữ liệu bên trong bộ nhớ của máy tính được phát triển và giới thiệu lần đầu vào năm 2009 bởi Salvatore Sanfilippo và được xây dựng bằng ngôn ngữ lập trình C. Redis được xây dựng để hoạt động trên các nền tảng hệ điều hành họ Unix. Redis hỗ trợ phát triển với nhiều ngôn ngữ lập trình khác nhau như C, C++, Java, PHP, Python, Ruby, Scala. Hình 5.3 là logo của Redis.

Hình 3.6: Logo của Redis

Redis là kho lưu trữ cơ sở dữ liệu dưới dạng key - value (khóa – giá trị). Do đặc trưng lưu trữ trong RAM – bộ nhớ trong nên tốc độ truy xuất nhanh. Tuy nhiên đặc tính của bộ nhớ trong là dữ liệu sẽ bị mất nếu như xảy ra sự cố như mất điện nên Redis có tính năng xuất dữ liệu ra ổ đĩa theo chu kì hoặc theo một khối lượng công việc nhất định để đảm bảo khả năng phục hồi nếu như có sự cố xảy ra. Redis là một giải pháp hàng đầu khi cần triển khai cache trong bộ nhớ (in-memory) để hạn chế độ trễ khi truy cập dữ liệu, đồng thời tăng thông lượng và giảm tải của database

3.7 Kết chương

Trong chương này, tôi đã trình bày lý thuyết và các công nghệ được sử dụng cùng với lí do lựa chọn các công nghệ này trong quá trình phát triển sản phẩm. Những công nghệ đó được áp dụng và phát triển sản phẩm, dẫn đến một sản phẩm hoàn chỉnh sẽ được trình bày trong chương tiếp theo – Chương 4.

CHƯƠNG 4. THỰC NGHIỆM VÀ ĐÁNG GIÁ

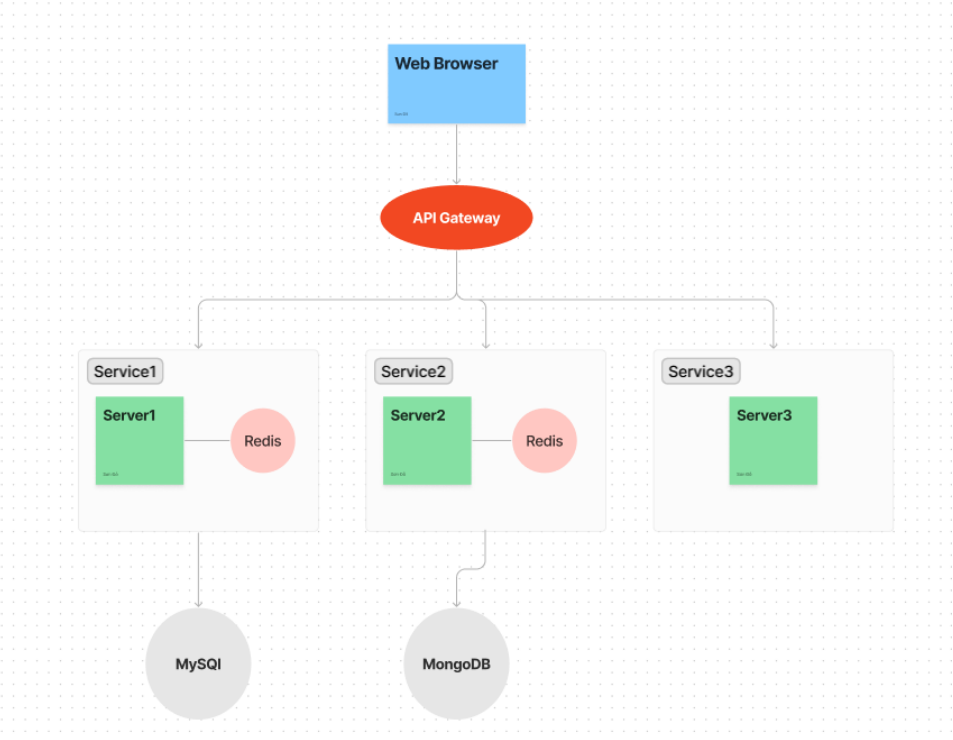
4.1 Tổng quan

Trong chương trước tôi đã nêu ra và phân tích nguyên do lựa chọn các công nghệ, thư viện, tính năng được áp dụng trong phát triển sản phẩm cho đồ án này. Tiếp theo trong chương này em sẽ đi vào phần thiết kế chi tiết hệ thống và sử dụng những công nghệ trên để giải quyết những vấn đề và tiến hành xây dựng hệ thống sản phẩm trong các phần kế tiếp.

4.2 Thiết kế kiến trúc

4.2.1 Lựa chọn kiến trúc phần mềm

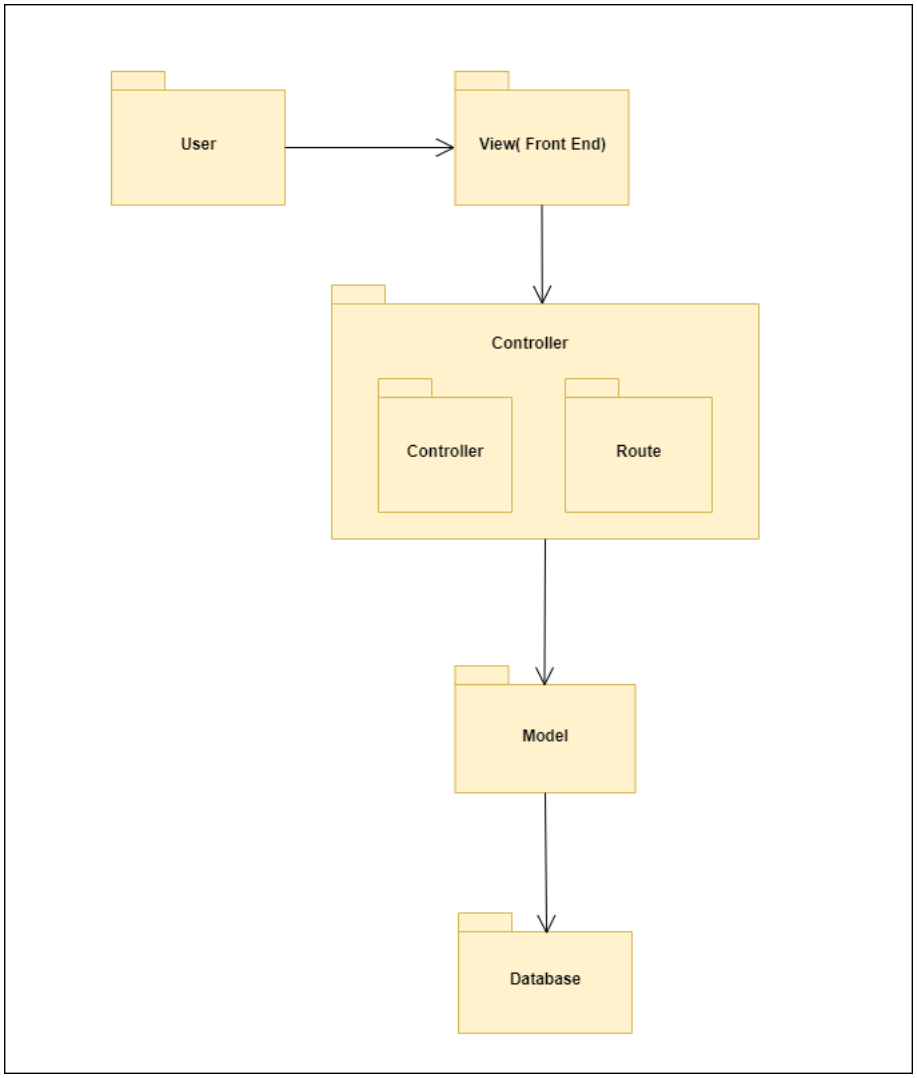
Kiến trúc được tôi lựa chọn để xây dựng sản phẩm là kiến trúc Client - Server và mô hình Microservice. Trong mô hình kiến trúc này, bao gồm hai thành phần: các servers và nhiều clients. Các Servers sẽ cung cấp dịch vụ cho nhiều clients. Ngược lại, các client gửi request tới server để xử lý và trả các service tương ứng cho phía client đó. Song song đó, máy chủ vẫn luôn tiếp tục tiếp nhận requests từ clients. Các request từ client gửi tới server sẽ được gửi thông qua một trung gian là API Gateway, nó có nhiệm vụ điều phối request nào sẽ được gửi tới server nào. Các servers hoạt động độc lập, thực thi các tác vụ riêng không gây ảnh hưởng tới nhau. Hình 5.3



Hình 4.1: Mô hình Client - Server kết hợp Microservice

- Web Browser: là trình duyệt web, là nơi có vai trò lưu trữ các thành phần giao diện như nút bấm, các biểu mẫu,. . . Thành phần này có tác dụng tạo ra các phần giao diện với mục đích hiển thị và giúp người sử dụng có thể tương tác được với hệ thống thông qua các giao diện này.
- API gateway là nơi tiếp nhận các request do web browser gửi tới, nó có nhiệm vụ điều phối các requests nào sẽ được gửi tới server nào
- Server 1 kết nối tới hệ quản trị cơ sở dữ liệu MySQL. Server 1 có nhiệm vụ tiếp nhận xử lý các requests làm thay đổi, cập nhật dữ liệu trong hệ quản trị cơ sở dữ liệu mySQL. Server 1 sử dụng Redis để cache dữ liệu, đặc biệt là với các API có lượng dữ liệu trả về lớn
- Server 2 kết nối tới hệ quản trị cơ sở dữ liệu MongoDB. Server 2 có nhiệm vụ tiếp nhận xử lý các requests làm thay đổi, cập nhật dữ liệu trong hệ quản trị cơ sở dữ liệu MongoDB. Server 2 sử dụng Redis để cache dữ liệu, đặc biệt là với các API có lượng dữ liệu trả về lớn
- Server 3 có nhiệm vụ chạy web socket. Server 3 có nhiệm vụ xử lý các requests yêu cầu realtime, gửi và nhận dữ liệu ngay tức thì
- Hệ quản trị cơ sở dữ liệu MySQL được sử dụng để lưu trữ các loại dữ liệu yêu cầu chặt chẽ, có tính ràng buộc cao, đảm bảo tính toàn vẹn dữ liệu
- Hệ quản trị cơ sở dữ liệu MongoDB được sử dụng để lưu trữ các loại dữ liệu có số lượng lớn, có tính rời rạc, cần tốc độ truy vấn nhanh

4.2.2 Thiết kế tổng quan



Hình 4.2: Biểu đồ gói tổng quan

Hình 5.3 trên đây minh họa cho biểu đồ gói tổng quan của sản phẩm được phát triển. Do sử dụng mô hình kiến trúc Client-Server làm mô hình gốc, biểu đồ gói tổng quan cũng thể hiện điều đó. Các gói bao gồm những thành phần phụ trách các nhiệm vụ tương đồng và có mối liên hệ chặt chẽ với nhau. Trong đó các gói ở lớp trên phụ thuộc vào lớp dưới và không phụ thuộc theo chiều ngược lại.

Đầu tiên là gói về User, gói này không nằm trong hệ thống mà đại diện cho nhóm đối tượng người sử dụng hệ thống, có khả năng tương tác với hệ thống dựa trên những gì được đưa ra bởi gói View trong quá trình hoạt động.

Kế đến là gói View, gói này có nhiệm vụ lưu trữ, quản lý các đối tượng giao diện như nút bấm, biểu mẫu,. . . Tiếp đến gói này cung cấp khả năng tương tác cho các đối tượng người sử dụng được mô tả trong gói User, nhận những yêu cầu từ người sử dụng chuyển xuống cho gói Controller xử lý, và ngược lại nhận những phản hồi,

những dữ liệu từ gói Controller để xử lý hiển thị ra cho người sử dụng. Gói này phụ thuộc vào gói Controller để xử lý được hoạt động của mình.

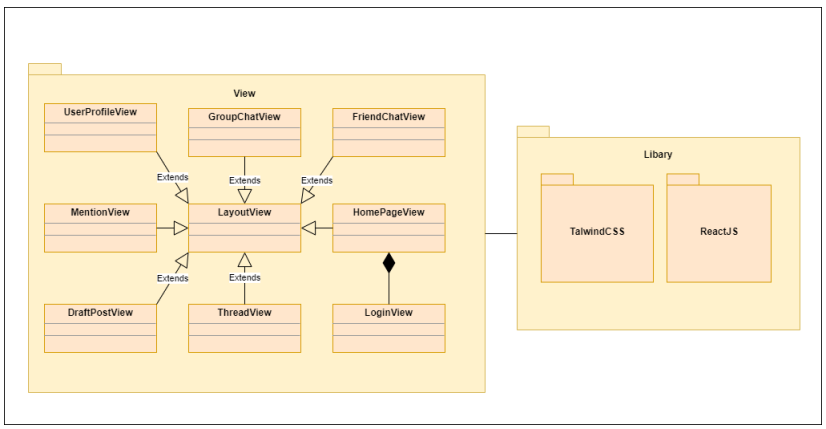
Tiếp đến là gói Controller, gói này đảm nhận vai trò nhận yêu cầu được chuyển về từ người dùng, tiến hành điều hướng, xử lý các nghiệp vụ logic cần thiết, và chuyển hướng các yêu cầu xuống dưới cho gói Model phụ trách tiếp các yêu cầu đó. Sau đó, gói sẽ nhận các trả về từ gói Model là các dữ liệu cần thiết, tiếp tục xử lý nghiệp vụ logic và chuyển kết quả lên phía trên cho gói View thực hiện tiếp nhiệm vụ của mình. Trong gói Controller được chia nhỏ thành hai gói Controller và gói Route, trong đó gói Router bên trong có chức năng điều hướng các requests, yêu cầu dữ liệu từ gói View, và gói Controller sẽ làm nhiệm vụ chính bên trong thông qua việc xử lý các tác vụ logic, tính toán. Gói này phụ thuộc vào gói Model để xử lý được hoạt động của mình.

Phía dưới là gói Model, gói này có vai trò là nơi thiết lập liên kết giữa hệ thống với hệ quản trị cơ sở dữ liệu, là nơi xử lý những thao tác liên quan đến dữ liệu như truy vấn nội dung, chỉnh sửa thông tin dữ liệu trong hệ thống. Gói này sẽ nhận những yêu cầu từ gói Controller và sau đó tiến hành các truy vấn, các yêu cầu thực hiện đúng mục đích chuyển xuống cho gói Database xử lý, sau đó nhận lại kết quả, xử lý dữ liệu và truyền lên phía trên cho gói Controller. Gói này phụ thuộc vào gói Database để xử lý được hoạt động của mình.

Cuối cùng là gói Database, gói này chứa hệ quản trị cơ sở dữ liệu, có vai trò lưu trữ và thực hiện các truy vấn dữ liệu. Gói sẽ nhận những yêu cầu truy vấn từ gói Model, sau đó thực thi các truy vấn và gửi trả lại kết quả lên cho gói Model tiếp tục xử lý.

Trên đây là tổng quan về biểu đồ gói cho hệ thống được phát triển. Trong phần tới em sẽ đi sâu vào chi tiết thiết kế gói cho hệ thống.

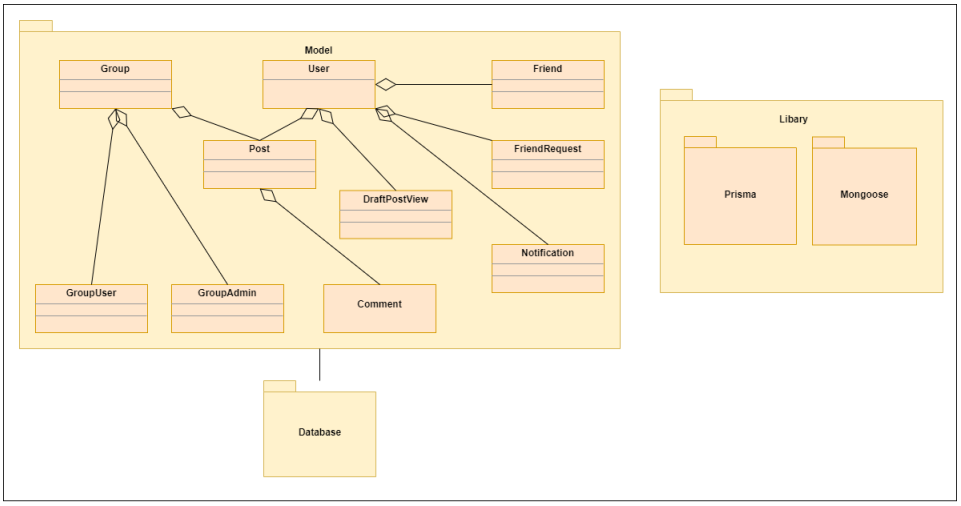
4.2.3 Thiết kế chi tiết gói



Hình 4.3: Biểu đồ gói chi tiết cho gói View

Trong hình 4.3 này mô tả biểu đồ gói chi tiết cho gói View trong hệ thống. Gói View này có phụ thuộc vào một số thư viện bên ngoài như thư viện ReactJS và thư viện TalwindCSS để giúp tạo giao diện nhanh chóng hơn, hỗ trợ sử dụng và tạo ra những tính năng dùng Javascript nhanh chóng và dễ dàng hơn.

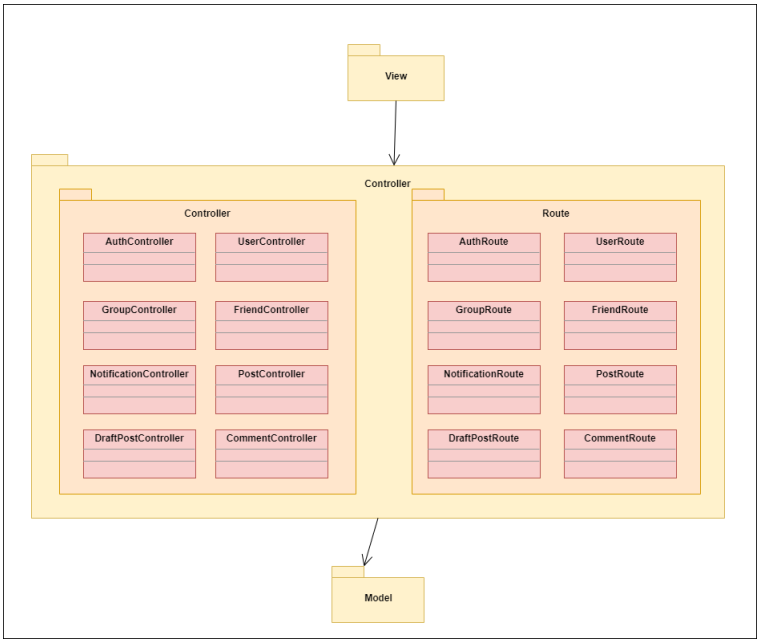
Gói View này được chia ra các lớp theo các nhóm đối tượng hiển thị. Trong đó có một lớp LayoutView bao gồm khuôn mẫu định dạng hiển thị tiêu chuẩn, các lớp giao diện khác sẽ kế thừa và chỉnh sửa các thành phần nội dung phù hợp với lớp đó. Tại đây có lớp LoginView về giao diện đăng nhập sẽ hợp thành bên trong HomePageView là giao diện về trang chủ. Người dùng sau khi đăng nhập thành công, hệ thống sẽ chuyển đổi từ giao diện đăng nhập sang giao diện trang chủ.



Hình 4.4: Biểu đồ gói chi tiết cho gói Model

Kế tiếp là thiết kế chi tiết cho biểu đồ gói Model được minh họa trên hình 4.4. Gói Model có phụ thuộc vào một số thư viện quan trọng như là Prisma và Mongoose, các thư viện này có vai trò giúp thiết lập và tạo kết nối giữa gói Model và gói Database để có khả năng truy cập đồng thời thực hiện các truy vấn với hệ quản trị cơ sở dữ liệu MySQL và MongoDB được đặt nằm trong gói Database này.

Gói Model bao gồm các lớp liên quan tới quản lý tài khoản người dùng. Lớp User có quan hệ thu nạp với các lớp Friend đại diện cho bạn bè, lớp FriendRequest đại diện cho lời mời kết bạn, lớp Notification đại diện cho thông báo, lớp Post và DraftPost đại diện cho tin nhắn và tin nhắn nháp. Tiếp theo là các lớp liên quan tới nhóm như lớp Group, lớp GroupUser, lớp GroupAdmin. Lớp Group có quan hệ thu nạp với các lớp GroupUser, GroupAdmin, Post.



Hình 4.5: Biểu đồ gói chi tiết cho gói Controller

Cuối cùng là biểu đồ gói chi tiết cho gói Controller, được minh họa trên hình 4.4. Trong gói này, gói được chia nhỏ thành hai gói bên trong, gồm gói Controller trong và gói Route. Gói này có các lớp có tính tương ứng với các lớp trong gói View và gói Model có liên hệ với gói. Ví dụ lớp GroupController là lớp điều khiển luồng xử lý logic của nhóm, có mối liên hệ với lớp GroupChatView bên gói View và liên hệ với lớp Group trong lớp Model để tạo thành một liên kết xuyên suốt.

Gói Controller bên trong chứa những lớp xử lý nghiệp vụ logic cơ bản ứng với các lớp bên dưới gói Model và các lớp trên bên gói View. Trong đó có lớp AuthController là lớp xác thực tài khoản người dùng, lớp UserController thực thi các tác vụ tài khoản người dùng, lớp GroupController thực thi các tác vụ nhóm, lớp FriendController thực thi các tác vụ bạn bè, lớp NotificationController hiển thị thông báo, lớp PostController soạn thảo. lưu trữ tin nhắn, lớp DraftPost thực thi tác vụ tin nhắn nháp, lớp CommentController xử lý bình luận trong tin nhắn. Gói Route bên trong chứa các Route con tương ứng với các gói Controller, có nhiệm vụ điều hướng các request từ phía Frontend(View) tới các Controller để xử lý.

4.3 Thiết kế chi tiết

4.3.1 Thiết kế giao diện

Các chuẩn hóa thiết kế giao diện với các đối tượng được mô tả dưới đây.

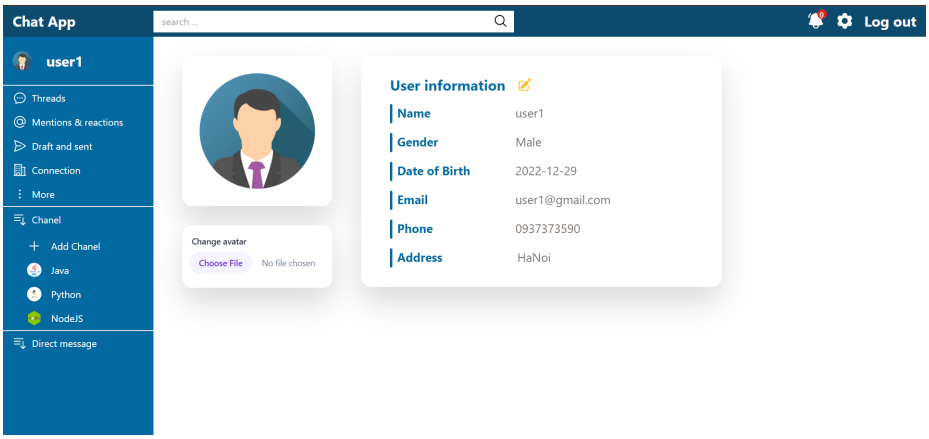
Về nút bấm, nút bấm có kích thước phù hợp với nội dung bên trong nút bấm. Màu sắc của nút bấm được thể hiện cho các chức năng tương ứng, trong đó màu xanh lá bao gồm các chức năng lên quan đến xác nhận, chuyển trạng thái, còn màu

đỏ thể hiện việc thực hiện các hành động như xóa hay từ chối một nội dung nào đó.

Các thông báo được đưa ra màn hình ở góc trên bên phải của thanh TopBar. Thông báo được hiển thị theo thời gian, người dùng cuộn lên để xem các thông báo cũ.

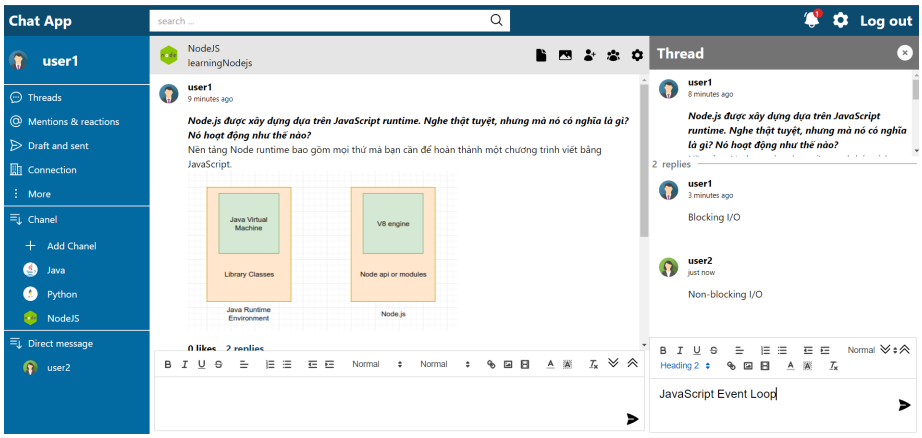
Giao diện có màu sắc tươi sáng, tông màu chính là màu trắng và màu xanh dương, một số màu phụ đạo chủ yếu là màu xanh nhạt để giúp cho hiển thị trở nên rõ ràng. Các biểu mẫu được thiết kế căn lề phân tách giữa cột tên các trường và cột điền nội dung. Một số biểu mẫu được thiết kế dưới dạng một popup hiển thị để không cần chuyển trang trong quá trình sử dụng hệ thống.

Dưới đây là một số hình ảnh minh họa thiết kế giao diện. Đầu tiên là hình 4.6 minh họa trang thông tin cá nhân người dùng. Người dùng truy cập trang thông tin cá nhân để xem hồ sơ, chỉnh sửa hồ sơ.



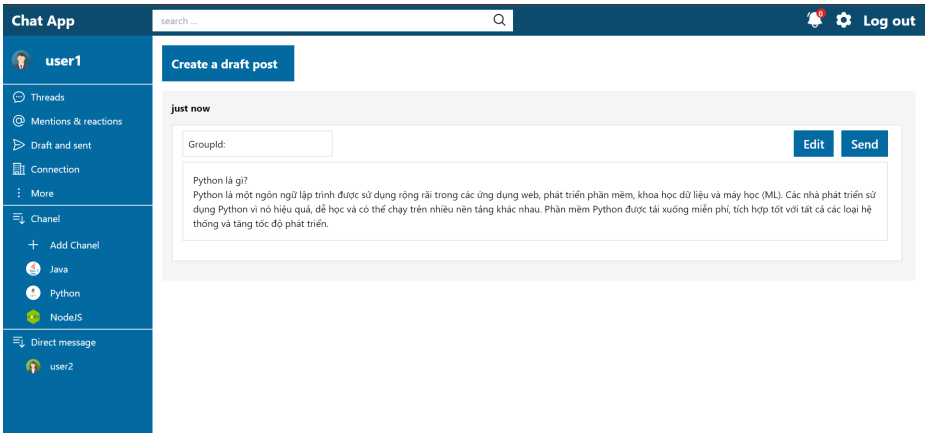
Hình 4.6: Giao diện trang Profile người dùng

Tiếp đến, hình 4.7 minh họa giao diện trang chat nhóm. Trang này hiển thị các tin nhắn trong nhóm theo thời gian.



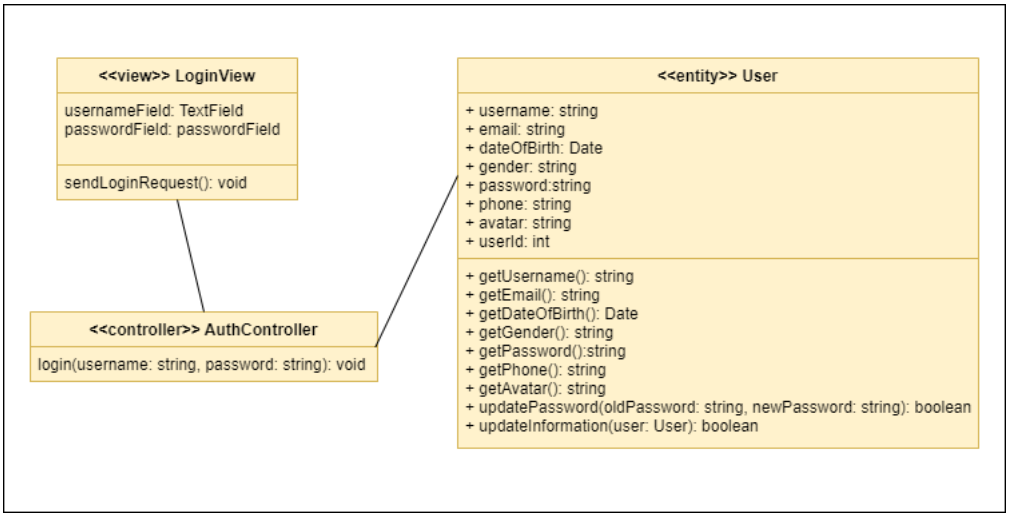
Hình 4.7: Giao diện nhóm chat

Cuối cùng, hình 4.8 là giao diện minh họa cho trang tin nhắn nháp, trang này hiển thị các tin nhắn nháp người dùng đã tạo, nút bấm để tạo tin nhắn nháp mới.



Hình 4.8: Giao diện trang tin nhắn nháp

4.3.2 Thiết kế lớp



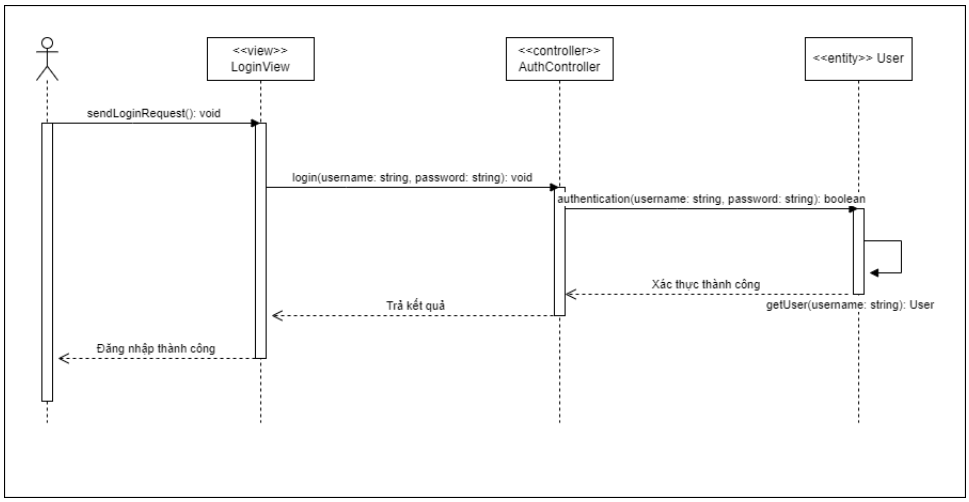
Hình 4.9: Biểu đồ lớp thiết kế chi tiết cho lớp User và AuthController

Hình 4.9 là thiết kế chi tiết cho lớp User và AuthController bao gồm các thuộc tính và các phương thức được sử dụng trong các lớp này. Trong đó lớp AuthController có chứa phương thức login để xác thực tài khoản của người dùng xem đó có phải là một tài khoản hợp lệ bên trong hệ thống.

Lớp User là lớp mô hình hóa thực thể người dùng trong hệ thống, lớp này có những thuộc tính mô tả các thông tin cá nhân cơ bản của một người dùng, bao gồm "email", được sử dụng làm tên đăng nhập của tài khoản, "password" là trường mật khẩu, hai trường này là để định danh một tài khoản. Các trường về thông tin cá nhân bao gồm "phone" lưu trữ số điện thoại, "address" để lưu trữ địa chỉ của người dùng, "dateOfBirth" là trường về ngày sinh, "gender" lưu trữ về giới tính. "password" để

lưu trữ mật khẩu người dùng, "avatar" lưu trữ đường dẫn tới thư mục chứa ảnh đại diện người dùng. Lớp User có các phương thức để truy xuất các trường dữ liệu của nhân viên được mô tả trong hình vẽ. Kế đến là hai phương thức để cập nhật dữ liệu cho ứng viên, đó là "updatePassword" là phương thức để cập nhật lại mật khẩu, và "updateInformation" để cập nhật lại các thông tin cá nhân khác. Ngoài ra lớp User có một số phương thức lớp dùng để lấy ra các nhân viên, tạo tài khoản nhân viên mới và xác thực tài khoản của nhân viên đăng nhập vào hệ thống.

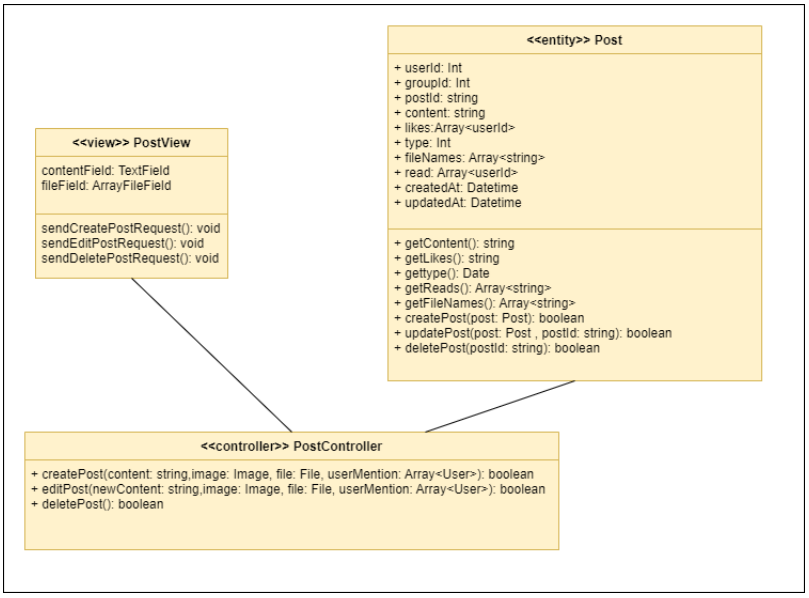
Để minh họa cho hoạt động của lớp User và AuthController, biểu đồ trình tự mô tả cho hoạt động đăng nhập của nhân viên trong hình 4.10 sẽ mô tả luồng truyền thông điệp trong hệ thống để tiến hành use case đăng nhập dưới đây. Luồng này sẽ mô tả hoạt động của người dùng để tiến hành đăng nhập vào hệ thống.



Hình 4.10: Biểu đồ trình tự mô tả usecase đăng nhập với người dùng

Trong biểu đồ 4.9 trên đây, chúng ta xuất phát từ người sử dụng hệ thống. Tác nhân là người dùng truy cập vào trang hiển thị đăng nhập thuộc lớp LoginView sau đó điền đầy đủ các thông tin và gửi yêu cầu đăng nhập. Sau khi nhận được yêu cầu đăng nhập, lớp AuthController sẽ thực thi phương thức đăng nhập với các thông tin được tác nhân là người dùng nhập gửi vào hệ thống, và gọi đến phương thức xác thực người dùng bên lớp User. Tại lớp User này, lớp sẽ thực thi xác nhận người dùng dựa trên thông tin Controller chuyển xuống, sau khi xác thực thành công sẽ lấy ra thông tin của nhân viên đó và gửi trả lên phía trên cho controller. Từ controller sẽ trả ngược lại kết quả để hiển thị ra view và cuối cùng thông báo cho tác nhân quá trình thực hiện đăng nhập thành công.

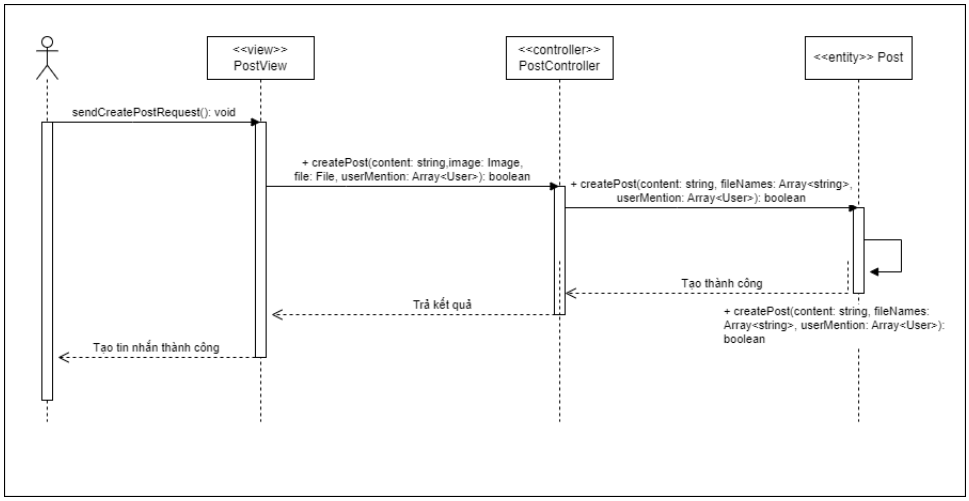
Tiếp đến là thiết kế chi tiết cho lớp Post, lớp đại diện cho các tin nhăn của người dùng trong hệ thống. Biểu đồ thiết kế chi tiết cho lớp Post và lớp controller tương ứng PostController được minh họa trong hình 4.10 sau đây.



Hình 4.11: Biểu đồ lớp thiết kế chi tiết cho lớp Post và PostController

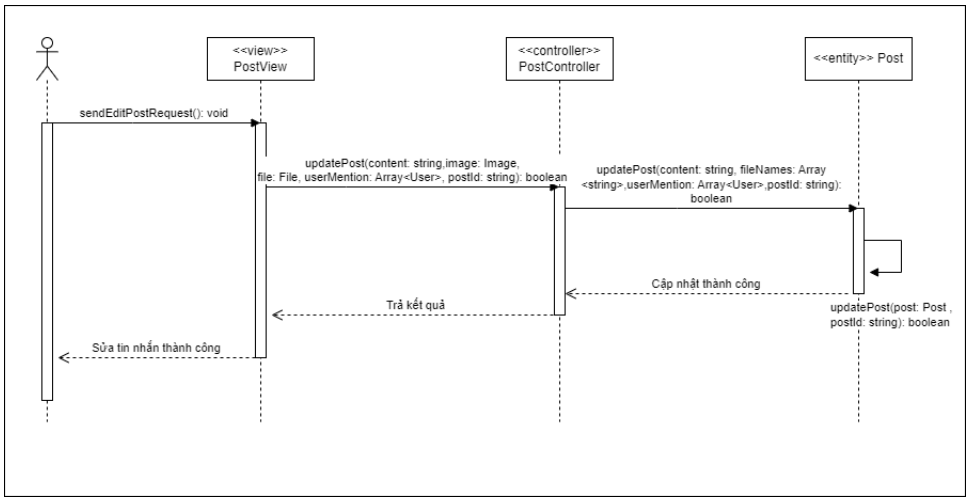
Tại đây, lớp thực thể Post mô hình hóa cho tin nhắn. Lớp này cần có những thông tin bao gồm postId là định danh tin nhắn, kế đến là "userId" là đinh danh người tạo tin nhắn. Kế đến là thuộc tính "groupId" xác định tin nhắn thuộc nhóm nào. Các thuộc tính về nội dung bao gồm thuộc tính "type" xác định trạng thái tin nhắn quan trọng hay không quan trọng, thuộc tính "reads" là một mảng để lưu trữ đinh danh các người dùng đã xem tin, thuộc tính "fileNames" là mảng lưu đường dẫn các tệp người dùng tải lên, thuộc tính "content" lưu nội dung tin nhắn thuộc tính "likes" là mảng lưu đinh danh những người dùng đã like tin nhắn, "createdAt" và "updatedAt" lưu thời gian tạo và cập nhật tin nhắn. Phần phương thức của lớp bao gồm các phương thức để lấy ra thông tin các thuộc tính đối với một tin nhắn, cùng với đó là phương thức về cập nhật tin nhắn, xóa tin nhắn.

Lớp controller tương ứng của lớp Post là PostController. Lớp này cung cấp các phương thức để tạo tin nhắn mới, cập nhật tin nhắn, xóa tin nhắn. Để minh họa cho hoạt động của lớp Post, hình 4.12, 4.13, 4.14 dưới đây sẽ mô tả lại trình tự thực hiện của các quá trình tạo tin nhắn, sửa tin nhắn, xóa tin nhắn.



Hình 4.12: Biểu đồ trình tự mô tả usecase tạo tin nhắn

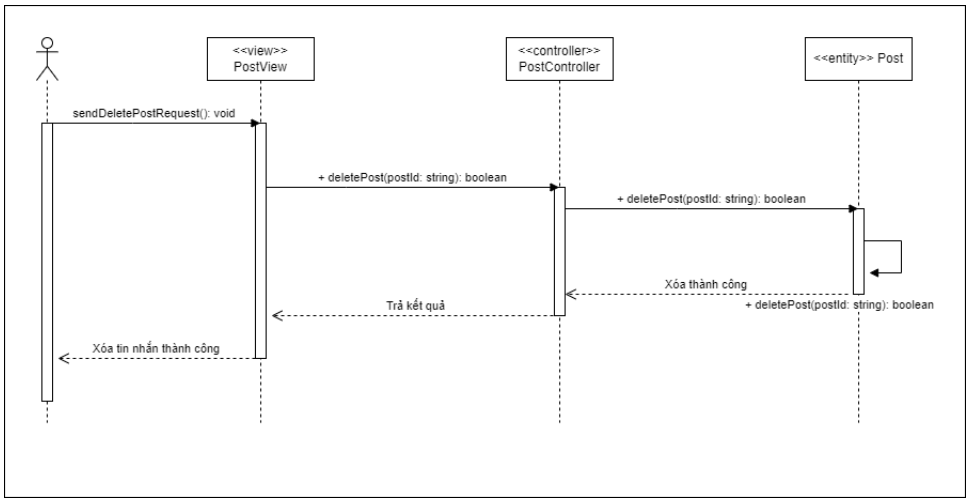
Trong biểu đồ 4.12 trên đây, chúng ta xuất phát từ người sử dụng hệ thống. Tác nhân là người dùng truy cập vào trang soạn thảo tin nhắn thuộc lớp PostView sau đó điền đầy đủ các thông tin và gửi yêu cầu tạo tin nhắn. Sau khi nhận được yêu cầu tạo tin nhắn, lớp PostController sẽ thực thi phương thức tạo tin nhắn với các thông tin được tác nhân là người dùng nhập gửi vào hệ thống, và gọi đến phương thức tạo tin nhắn bên lớp Post. Tại lớp Post này, lớp sẽ thực thi tạo tin nhắn dựa trên thông tin Controller chuyển xuống, sau khi tạo thành công sẽ lấy ra thông tin của tin nhắn vừa tạo đó và gửi trả lên phía trên cho controller. Từ controller sẽ trả ngược lại kết quả để hiển thị ra view và cuối cùng thông báo cho tác nhân quá trình thực hiện tạo tin nhắn thành công.



Hình 4.13: Biểu đồ trình tự mô tả usecase sửa tin nhắn

Trong biểu đồ 4.13 trên đây, chúng ta xuất phát từ người sử dụng hệ thống. Tác nhân là người dùng truy cập vào trang xem tin nhắn thuộc lớp PostView sau đó lựa chọn chỉnh sửa tin nhắn và điền đầy đủ các thông tin mới rồi gửi yêu cầu sửa tin

nhắn. Sau khi nhận được yêu cầu sửa tin nhắn, lớp PostController sẽ thực thi tìm kiếm tin nhắn theo đinh danh và sửa tin nhắn đó với các thông tin được tác nhân là người dùng nhập gửi vào hệ thống, và gọi đến phương thức sửa tin nhắn bên lớp Post. Tại lớp Post này, lớp sẽ thực thi sửa tin nhắn dựa trên thông tin Controller chuyển xuống, sau khi sửa thành công sẽ lấy ra thông tin sửa tin nhắn và gửi trả lên phía trên cho controller. Từ controller sẽ trả ngược lại kết quả để hiển thị ra view và cuối cùng thông báo cho tác nhân quá trình thực hiện sửa tin nhắn thành công.

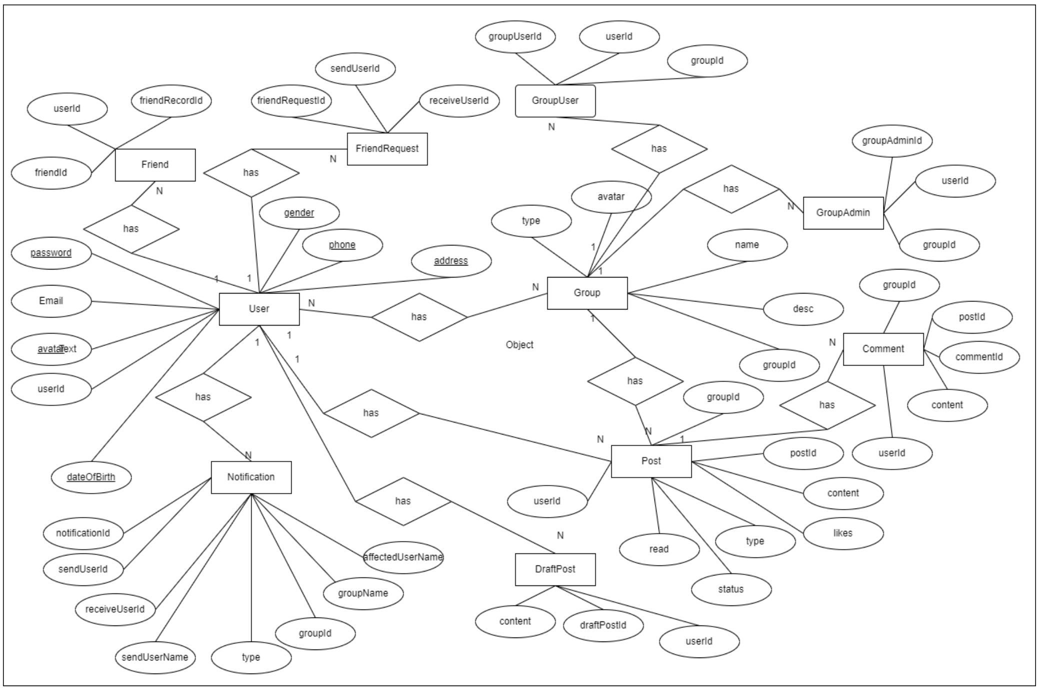


Hình 4.14: Biểu đồ trình tự mô tả usecase xóa tin nhắn

Trong biểu đồ 4.14 trên đây, chúng ta xuất phát từ người sử dụng hệ thống. Tác nhân là người dùng truy cập vào trang xem tin nhắn thuộc lớp PostView sau đó lựa chọn xóa tin nhắn rồi gửi yêu cầu xóa tin nhắn. Sau khi nhận được yêu cầu xóa tin nhắn, lớp PostController sẽ thực thi phương thức xóa tin nhắn với các thông tin được tác nhân là người dùng nhập gửi vào hệ thống, và gọi đến phương thức xóa tin nhắn bên lớp Post. Tại lớp Post này, lớp sẽ thực thi tìm kiếm tin nhắn cần xóa theo định danh và xóa tin nhắn đó dựa trên thông tin Controller chuyển xuống, sau khi xóa thành công sẽ lấy ra thông tin xoa tin nhắn và gửi trả lên phía trên cho controller. Từ controller sẽ trả ngược lại kết quả để hiển thị ra view và cuối cùng thông báo cho tác nhân quá trình thực hiện sửa tin nhắn thành công.

4.3.3 Thiết kế cơ sở dữ liệu

Từ hoạt động của các thực thể tương tác với nhau trong hệ thống, cùng với sự liên kết và thực thi các quy trình nghiệp vụ, em thiết kế sơ đồ thực thể liên kết thể hiện mối liên kết giữa các đối tượng trong hình 5.3



Hình 4.15: Sơ đồ thực thể liên kết

Các thực thể trong hệ thống bao gồm người dùng (User), nhóm (Group), Bạn bè (Friend), tin nhắn (Post), tin nhắn nháp (DraftPost), nhóm trưởng (GroupAdmin), nhóm viên (GroupUser) và các thông báo (Notification).

Các thuộc tính của từng thực thể đã được mô tả chi tiết trong sơ đồ thực thể liên kết trong hình 5.3. Giữa các thực thể này xuất hiện các liên kết để mô tả mối quan hệ giữa chúng. Đầu tiên là thực thể người dùng, thực thể này có liên kết mô tả một người dùng sẽ có nhiều bạn bè. Một người dùng có thể gửi nhiều lời mời kết bạn tới cho nhiều người dùng khác và cũng có thể nhận được nhiều lời mời kết bạn từ nhiều người dùng khác nhau.

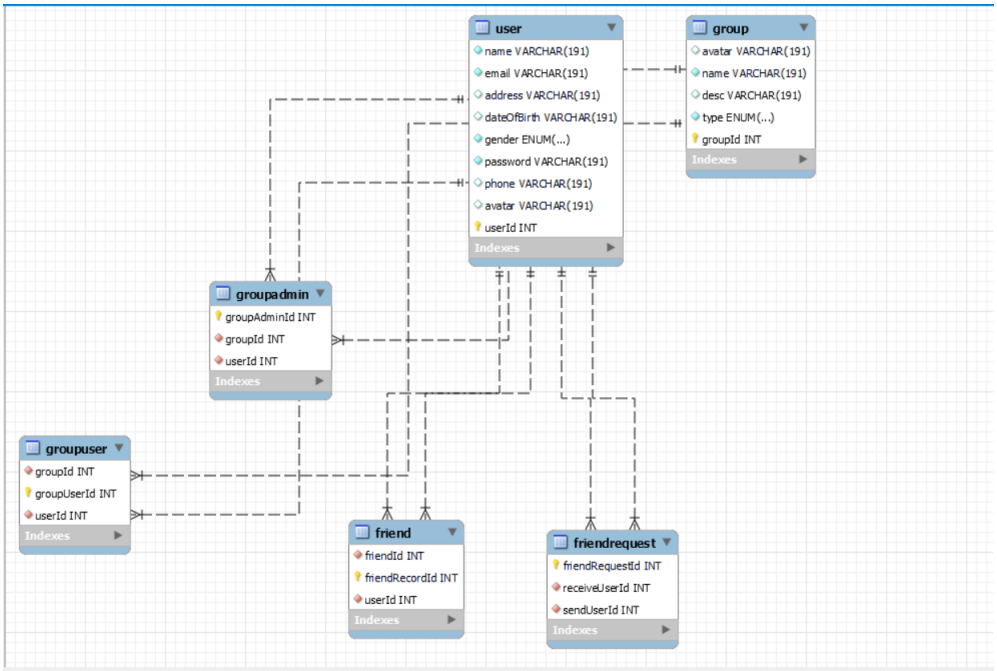
Tiếp đến là các liên kết của thực thể nhóm (Group), thực thể nhóm có liên kết n-n với thực thể người dùng (User) vì trong một nhóm có nhiều người dùng và một người dùng cũng có thể tham gia nhiều nhóm. Trong một nhóm sẽ có nhiều trường nhóm (GroupAdmin) và nhiều nhóm viên (GroupUser) vì vậy mối quan hệ giữa thực thể nhóm (Group) với các thực thể nhóm trưởng (GroupAdmin), nhóm viên (GroupUser) là quan hệ 1-n

Sang đến thực thể tin nhắn (Post), thực thể này có mối quan hệ n-1 với thực thể người dùng (User) vì 1 người dùng có thể soạn thảo nhiều tin nhắn. Một người dùng cũng có thể soạn sắn nhiều tin nhắn nháp nên mối quan hệ giữa thực thể người dùng (User) và thực thể tin nhắn nháp (DraftPost) là quan hệ 1-n

Cuối cùng là thực thể thông báo (Notification), một người dùng sẽ nhận được

rất nhiều thông báo bao gồm: thông báo kết bạn, thông báo nhóm, thông báo tin nhắn. Vì vậy mối quan hệ giữa thực thể người dùng (User) và thực thể thông báo (Notification) là mối quan hệ 1-n

Chương trình của em sử dụng 2 loại co sở dữ liệu là MySQL và MongoDB. Hệ quản trị cơ sở dữ liệu MySQL được sử dụng để lưu trữ các dữ liệu ràng buộc như người dùng, nhóm, bạn bè. NoSQL MongoDB được sử dụng để lưu trữ các dữ liệu ràng buộc, số lượng lớn như tin nhắn, thông báo, bình luận.



Hình 4.16: Thiết kế cơ sử dữ liệu MySQL

Từ sơ đồ thực thể liên kết, em chuyển sang thiết kế cơ sở dữ liệu cho hệ thống được phát triển. Hệ thống được xây dựng trên hệ quản trị cơ sở dữ liệu quan hệ sử dụng MySQL. Thiết kế cơ sở dữ liệu bao gồm 6 bảng.

Bảng User chứa các thông tin cơ bản, xác thực tài khoản người dùng bao gồm: định danh người dùng, tên, địa chỉ mail, địa chỉ nhà, ngày sinh, giới tính, số điện thoại, mật khẩu, ảnh đại diện. Trường định danh người dùng (UserId) là khóa chính tham chiếu tới các khóa ngoại trong các bảng nhóm trưởng (GroupAdmin), nhóm viên (GroupUser), bạn bè (Friend), lời mời kết bạn (FriendRequest).

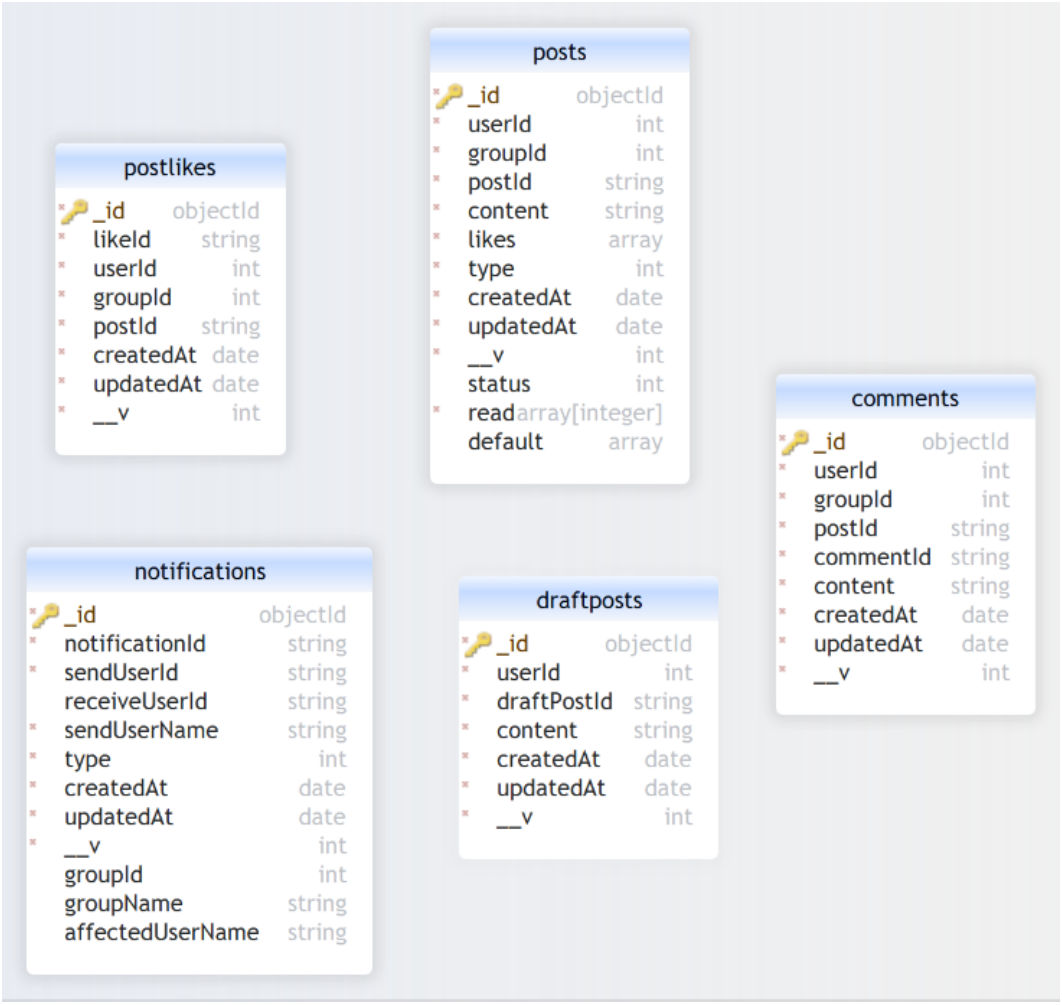
Bảng Group chứa các thông tin cơ bản về nhóm chat bao gồm: định danh nhóm, tên nhóm , mô tả nhóm, loại nhóm (có 2 loại nhóm là nhóm chat 2 người và nhóm chat nhiều người), ảnh đại diện nhóm. Trường định danh nhóm (GroupId) là khóa chính tham chiếu tới các khóa ngoại trong các bảng nhóm trưởng (GroupAdmin), nhóm viên (GroupUser).

Bảng GroupAdmin chứa các thông tin cơ bản về nhóm trưởng bao gồm: định danh nhóm trưởng, đinh danh người dùng, định danh nhóm. Trường định danh nhóm trưởng (GroupAdminId) là khóa chính. Trường định danh nhóm (GroupId) là khóa ngoại được tham chiếu từ bảng nhóm (Group). Trường định danh người dùng (UserId) là khóa ngoại được tham chiếu từ bảng người dùng (User).

Bảng GroupUser chứa các thông tin cơ bản về nhóm viên bao gồm: định danh nhóm viên, đinh danh người dùng, định danh nhóm. Trường định danh nhóm viên (GroupUserId) là khóa chính. Trường định danh nhóm (GroupId) là khóa ngoại được tham chiếu từ bảng nhóm (Group). Trường định danh người dùng (UserId) là khóa ngoại được tham chiếu từ bảng người dùng (User).

Bảng Friend chứa các thông tin cơ bản về bạn bè bao gồm: định danh bạn bè, đinh danh người dùng, định danh bạn bè của người dùng. Trường định danh bạn bè (FriendRecordId) là khóa chính. Trường định danh người dùng (UserId) là khóa ngoại được tham chiếu từ bảng người dùng (User). Trường định danh bạn bè của người dùng (FriendId) là khóa ngoại được tham chiếu từ bảng người dùng (User).

Bảng FriendRequest chứa các thông tin cơ bản về lời mời kết bạn bao gồm: định danh lời mời kết bạn, đinh danh người gửi, định danh người nhận. Trường định danh lời mời kết bạn (friendRequestId) là khóa chính. Trường định danh người gửi (SendUserId) là khóa ngoại được tham chiếu từ bảng người dùng (User). Trường định danh người nhận (ReceiveUserId) là khóa ngoại được tham chiếu từ bảng người dùng (User).



Hình 4.17: Thiết kế cơ sử dữ liệu MongoDB

Thiết kế cơ sở dữ liệu bao gồm 5 bảng. Mỗi bảng đều được bổ sung thêm hai trường "createdAt" và "updatedAt" có kiểu dữ liệu date để lưu trữ lại thời điểm tạo bản ghi và thời điểm cập nhật bản ghi.

Bảng Post chứa các thông tin cơ bản về tin nhắn bao gồm: định danh tin nhắn, định danh người soạn tin nhắn, định danh nhóm, nội dung tin nhắn, kiểu tin nhắn (có 2 loại tin nhắn là tin nhắn thường và tin nhắn quan trọng),người đã xem tin nhắn (đây là một mảng các định danh của những người dùng đã xem tin nhắn).

Bảng DraftPost chứa các thông tin cơ bản về tin nhắn nháp bao gồm: định danh tin nhắn nháp, định danh người soạn tin nhắn, nội dung tin nhắn.

Bảng Comment chứa các thông tin cơ bản về bình luận bao gồm: định danh bình luận, định danh người soạn bình luận, định danh nhóm, đinh danh tin nhắn chứa bình luận, nội dung bình luận.

Bảng PostLike chứa các thông tin cơ bản về like tin nhắn bao gồm: định danh like, định danh người like tin nhắn, định danh nhóm, đinh danh tin nhắn được like.

4.4 Xây dựng ứng dụng

4.4.1 Thư viện và công cụ sử dụng

Mục đích	Công cụ	Phiên bản
IDE lập trình	Microsoft Visual Studio Code	1.69.2
Validate dữ liệu	@hapi/joi	17.6.0
ORM hỗ trợ kết nối CSDL	@prisma/client	4.4.0
Mã hóa mật khẩu tài khoản	bcrypt	5.0.1
Khai báo biến môi trường	dotenv	16.0.1
Framework của nodeJS	express	4.18.1
Tạo mã xác thực	jsonwebtoken	8.5.1
ORM kết nối MongoDB	mongoose	6.3.4
Tải, lưu trữ ảnh tại server	multer	1.4.5
Cache dữ liệu	redis	4.5.0
Tạo chuỗi định danh	uuidv4	6.12.3
Gửi nhận dữ liệu realtime	socket.io	4.5.1
Chạy lại chương trình	nodemon	2.0.16
Hỗ trợ gọi API tại frontend	axios	0.27.2
Định dạng thời gian	javascript-time-ago	2.0.3
Soạn thảo văn bản	react-quilljs	1.3.3
Định dạng CSS cho trang web	tailwindcss	3.1.8
Mở rộng của javascript, OOP	typescript	4.8.4

Bảng 4.1: Danh sách thư viện và công cụ sử dụng

Bảng trên đây liệt kê những danh sách thư viện và các công cụ quan trọng nhất được sử dụng để tiến hành phát triển sản phẩm. Các công cụ và thư viện được chia ra thành các nhóm bao gồm nhóm hỗ trợ và lập trình các chức năng (gồm liên quan tới ngôn ngữ lập trình, framework, công cụ để lập trình), nhóm thiết lập và liên kết với cơ sở dữ liệu, nhóm bổ trợ xử lý giao diện (liên quan tới các thư viện hiển thị, tạo nút bấm) và nhóm bổ trợ các hoạt động xử lý hoạt động cốt lõi, triển khai hệ thống (thư viện server), ngoài ra còn sử dụng thêm công cụ quản lý mã nguồn (git) để lưu trữ và quản lý các phiên bản giúp quá trình bảo trì và nâng cấp trở nên dễ dàng hơn.

4.4.2 Kết quả đạt được

Kết quả thu được khi thực hiện xây dựng sản phẩm về trang web nhắn tin trực tuyến trong doanh nghiệp được thực hiện trong đồ án này là một hệ thống phần mềm được triển khai trên nền tảng web. Hệ thống được chia thành các phần, bao

gồm các hệ thống kết nối bên ngoài gồm khu vực hệ quản trị cơ sở dữ liệu dùng hệ quản trị MySQL và MongoDB để lưu trữ dữ liệu của hệ thống, khu vực kho lưu trữ dữ liệu trong quá trình cache dữ liệu là Redis, đây là hai thành phần kết nối ra bên ngoài, cần cài đặt riêng.

Các thành phần bên trong được chia thành các gói theo cấu trúc của framework ExpressJS để phát triển. Đầu tiên là gói config bao gồm những thiết lập cho hệ thống bao gồm các cài đặt ban đầu, khai báo các thư viện và các biến môi trường, nơi đây cũng quản lí danh sách các biến môi trường được sử dụng trong vận hành hệ thống. Kế đến là gói db lưu trữ lại thiết kế của cơ sở dữ liệu được sử dụng trong hệ thống, trong đây còn có mô tả về việc tạo dữ liệu mẫu cho sản phẩm được phát triển. Cuối cùng là gói app chứa mã nguồn của sản phẩm được đóng gói.

Trong gói app được chia ra thành các gói nhỏ để lưu trữ những thành phần quan trọng trong quá trình phát triển. Đầu tiên là gói routes để lưu trữ những khai báo đường dẫn, middlewares có tác dụng điều hướng và phân luồng, đón nhận những yêu cầu của người dùng được phát triển, tương ứng với gói routes trong quá trình thiết kế biểu đồ gói tổng quan. Kế đến là gói models dùng để lưu trữ những model mô tả dữ liệu các lớp và các đối tượng tham gia hệ thống, tương ứng với gói model trong thiết kế biểu đồ gói tổng quan. Tiếp đến là gói views dùng để lưu trữ giao diện để hiển thị ra các thiết bị cuối phía người sử dụng, tương ứng với gói views trong thiết kế biểu đồ gói. Kể đến sau đó là gói services, cung cấp những dịch vụ hỗ trợ cho hoạt động của controller, được mô tả đến trong biểu đồ gói tổng quan. Ngoài ra có gói assets lưu trữ những tệp định dạng cho giao diện dưới dạng scss giúp cho việc thiết kế giao diện trở nên hiệu quả hơn. Còn có gói jobs dùng để lưu trữ những tác vụ thêm, trong đồ án này là nơi để tạo ra những tác vụ ngầm được thực thi trong hệ thống.

4.4.3 Minh họa các chức năng chính

Sinh viên lựa chọn và đưa ra màn hình cho các chức năng chính, quan trọng, và thú vị nhất. Mỗi giao diện cần phải có lời giải thích ngắn gọn. Khi giải thích, sinh viên có thể kết hợp với các chú thích ở trong hình ảnh giao diện.

4.5 Kiểm thử

Để kiểm tra việc hệ thống hoạt động đúng đắn có như mong muốn theo thiết kế ban đầu hay không, chúng ta cần phải kiểm thử các chức năng của hệ thống được phát triển. Mục tiêu cần kiểm tra xem các chức năng có hoạt động đúng hay không, có gặp phải lỗi trong quá trình vận hành hay không. Các kết quả hiển thị từ hệ thống có đúng như mong muốn hay không, các thông báo lỗi có chuẩn xác không. Một số chức năng quan trọng được kiểm thử sẽ được nêu ra trong các phần

tiếp sau đây.

4.5.1 Kiểm thử chức năng đăng nhập

Trường hợp 1: Người dùng nhập đúng tên đăng nhập và mật khẩu, sau đó nhấn nút đăng nhập. Kết quả mong muốn: người dùng đăng nhập vào hệ thống và hệ thống thông báo đăng nhập thành công.

Trường hợp 2: Người dùng nhập sai tên đăng nhập hoặc nhập sai mật khẩu, sau đó nhấn nút đăng nhập. Kết quả mong muốn: người dùng không đăng nhập được vào hệ thống và hệ thống thông báo đăng nhập thất bại, đồng thời hiển thị lại trang đăng nhập.

Trường hợp 3: Người dùng nhập thiếu một trong hai trường tên đăng nhập hoặc mật khẩu, sau đó nhấn nút đăng nhập. Kết quả mong muốn: người dùng không đăng nhập được vào hệ thống và hệ thống thông báo đăng nhập thất bại, đồng thời hiển thị lại trang đăng nhập.

Trong việc kiểm thử chức năng đăng nhập này, kĩ thuật kiểm thử bảng quyết định được sử dụng để thiết kế ra các trường hợp có thể sinh ra từ việc nhập nội dung các trường cần thiết để tiến hành đăng nhập vào hệ thống.

4.5.2 Kiểm thử chức năng kết bạn

Trường hợp 1: Người dùng truy cập vào trang profile của người dùng khác, nếu đã là bạn bè sẽ hiện thị thông tin là bạn bè

Trường hợp 2: Người dùng truy cập vào trang profile của người dùng khác, nếu chưa là bạn bè sẽ hiện thị thông tin kết bạn, ấn kết bạn để gửi lời mời kết bạn

Trường hợp 3: Người dùng truy cập vào trang profile của người dùng khác, nếu chưa là bạn bè nhưng đã gửi lời mời kết bạn thì sẽ hiển thị thông tin đang đợi lời mời kết bạn được chấp nhận, người dùng có thể ấn hủy để xóa lời mời kết bạn đã gửi> Nếu người dùng ẩn hủy sẽ hiện thị lại thông tin ban đầu là kết bạn

Trường hợp 4: Người dùng truy cập vào trang profile của người dùng khác, nếu người dùng đã gửi lời mời kết bạn và được người đó chấp nhận thì sẽ hiện thị thông tin là bạn bè

Trường hợp 5: Người dùng truy cập vào trang profile của người dùng khác, nếu người dùng đã gửi lời mời kết bạn và bị người đó từ chối thì sẽ hiện thị lại thông tin ban đầu là kết bạn

4.5.3 Kiểm thử chức năng soạn tin nhắn

Trường hợp 1: Người dùng nhập dữ liệu dạng text, sử dụng các định dạng tùy chọn văn bản như chữ đậm, chữ nghiêng, ... sau đó ấn gửi, tin nhắn sẽ được hiển

thị với nội dung text đúng định dạng đã gửi

Trường hợp 2: Người dùng tải hình ảnh từ máy cá nhân lên và ấn gửi, tin nhắn sẽ được hiển thị với hình ảnh đã tải

Trường hợp 3: Người dùng tải các file(word, excel, pdf, ...) lên và ấn gửi, tin nhắn sẽ được hiển thị với các file đính kèm, nếu là các file định dạng ảnh hay pdf có thể ấn vào để xem ngay trên trình duyệt, đối với các file định dạng word, excel, ... cần ấn vào để tải về máy xem nội dung file

Trường hợp 4: Người dùng nhắc đến một người dùng khác trong nhóm chat và ấn gửi, tin nhắn hiển thị lên sẽ có gắn tên người được nhắc đến và người đó cũng sẽ nhận được thông báo

4.5.4 Kiểm thử chức năng nhận thông báo

Trường hợp 1: Có người dùng khác bình luận vào tin nhắn của mình sẽ nhận được thông báo bình luận mới.

Trường hợp 2: Có người dùng khác like tin nhắn của mình sẽ nhận được thông báo like tin nhắn.

Trường hợp 3: Có người dùng khác thêm mình vào nhóm mới sẽ nhận được thông báo thêm mới vào nhóm.

Trường hợp 4: Có người dùng khác quyền cao hơn xóa mình khỏi nhóm sẽ nhận được thông báo bị xóa khỏi nhóm.

Trường hợp 5: Có người dùng khác gửi lời mời kết bạn tới mình sẽ nhận được thông báo lời mời kết bạn Nếu ấn đồng ý thì 2 người sẽ trở thành bạn bè, người kia cũng sẽ nhận được thông báo bạn bè. Nếu ấn từ chối lời mời kết bạn sẽ bị xóa.

Trường hợp 6: Có người dùng khác xóa bạn bè với mình sẽ nhận được được thông báo xóa bạn bè

4.5.5 Tổng kết phần kiểm thử chức năng

Các chức năng được kiểm thử theo nhiều kĩ thuật kiểm thử khác nhau, bao gồm cả kĩ thuật kiểm thử hộp đen và kiểm thử hộp trắng, cùng với đó là một số kĩ thuật dùng để xây dựng các kịch bản kiểm thử cho các tính năng trong hệ thống như kiểm thử sử dụng phân tích giá trị biên, sử dụng bảng quyết định để tạo ra những tham số đầu vào xây dựng lên những trường hợp kiểm thử cụ thể mà vẫn đảm bảo được tính hiệu quả. Tổng kết quá trình kiểm thử, tổng cộng có hơn ba mươi trường hợp kiểm thử được xây dựng trong quá trình kiểm thử chức năng của hệ thống. Không có trường hợp nào trong đó có kết quả không như mong muốn.

4.6 Triển khai

Để triển khai được hệ thống trên thực tế, cần phải có năm server để tiến hành chạy hệ thống. Server đầu tiên là nơi cài đặt hệ quản trị cơ sở dữ liệu, tại đó có cài hệ quản trị cơ sở dữ liệu MySQL là nơi để lưu trữ các dữ liệu trong quá trình hoạt động và vận hành của hệ thống. Server thứ hai là nơi cài đặt cơ sở dữ liệu noSQL MongoDB là nơi để lưu trữ các dữ liệu rời rạc với số lượng lớn cần truy vấn nhanh. Các server ba, bốn, năm là nơi để chạy hệ thống, đóng vai trò tiếp nhận các yêu cầu của người dùng, sau đó xử lý và trả lại kết quả cho người dùng thông qua các thiết bị đầu cuối. Server 3 kết nối tới Server 1, thực hiện thay đổi cập nhật dữ liệu trong cơ sở dữ liệu MySQL. Server 4 kết nôi tới server 2, thực hiện thay đổi cập nhật dữ liệu trong cơ sở dữ liệu mongoDB. Server 5 có nhiệm vụ thực thị web socket, truyền nhận dữ liệu realtime, chạy độc lập. . Các server cần được đặt cùng trong một mạng nội bộ để thuận tiện trong quá trình giao tiếp. Server chạy hệ thống để tiếp nhận xử lý của người dùng cần kết nối mạng internet để có thể nhận được yêu cầu của người sử dụng.

Sau khi thiết lập xong các phần cứng cần thiết để tiến hành hoạt động, cần phải cài các phần mềm để vận hành hệ thống. Cài đặt cơ sở dữ liệu, tiến hành chạy khởi tạo các bảng để lưu trữ dữ liệu trong server đóng vai trò xử lý dữ liệu. Trên server chạy hệ thống, triển khai cài đặt các thư viện được sử dụng, sau đó chạy phần mã nguồn của chương trình để hệ thống hoạt động. Hệ thống sau khi vận hành có thể bắt đầu tiếp nhận các yêu cầu từ người dùng thông qua nền tảng web. Hệ thống được triển khai với mục tiêu có thể đáp ứng nhu cầu cho hàng trăm người có thể truy cập cùng lúc, được vận hành và bảo trì thường xuyên. Sẽ có những phiên bản cập nhật theo định kì vận hành của hệ thống. Hệ thống hiện chưa được thử nghiệm triển khai trên thực tế.

4.7 Kết chương

Trong chương này tôi đã đề cập đến việc lựa chọn mô hình kiến trúc phát triển sản phẩm. Tiếp đến là phần thiết kế từ tổng quan đến chi tiết cho các thành phần bên trong của sản phẩm được phát triển cùng với đó là xây dựng và kết quả thu được. Hai phần tiếp nêu về cách thức và phương án kiểm thử một số tính năng của sản phẩm cùng với mô hình để triển khai trong thực tế như thế nào. Trong chương kế tiếp – chương 5, tôi sẽ trình bày về nội dung đóng góp chính của sản phẩm được thực hiện trong đồ án này.

CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT

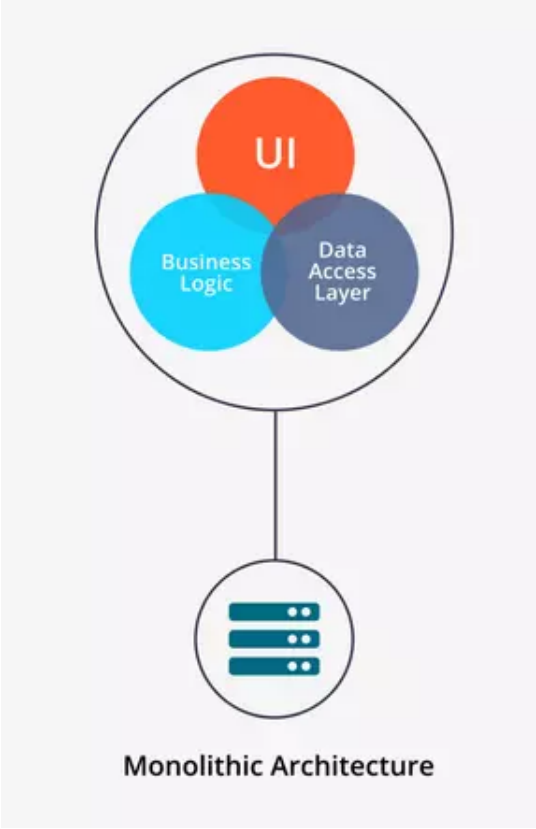
5.1 Tổng quan

Phần tiếp theo đây, tôi sẽ trình bày những giải pháp và đóng góp trong quá trình thực hiện đồ án tốt nghiệp này. Trong đó trọng tâm chính là mô hình kiến trúc thiết kế để xây dựng hệ thống.

5.2 Kiến trúc Microservice

5.2.1 Đặt vấn đề

Trong quá khứ, các lập trình viên thường xây dựng hệ thống theo một khối (Monolithic Applications). Ứng dụng phần mềm doanh nghiệp được thiết kế để đáp ứng nhiều yêu cầu kinh doanh của doanh nghiệp. Do đó, các phần mềm cung cấp hàng trăm các tính năng và tất cả những tính năng này đều được gói trong một ứng dụng monolithic. Việc triển khải, sử lỗi, mở rộng và nâng cấp những phần mềm khổng lồ này trở thành một trong những vấn đề lớn.



Hình 5.1: Mô hình monolithic

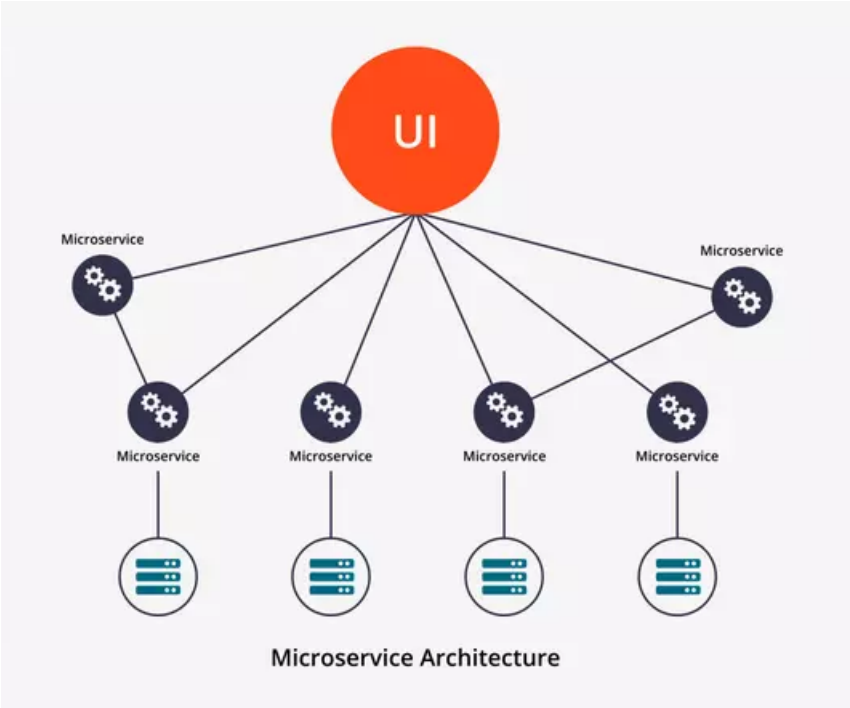
Việc thiết kế hệ thống phần mềm theo mô hình Monolithic đem lại nhiều ưu điểm có thể kể đến như dễ phát triển vì các stack công nghệ thống nhất ở tất cả các layer, dễ test do toàn bộ project được đóng gói trong một package nên dễ dàng

chạy test integrantion và test end-to-end. Việc deploy cũng vô cùng đơn giản và nhanh chóng vì ta chỉ cần quan tâm đến 1 package đơn nhất. Hệ thống dễ scale hơn vì chúng ta có thể có nhiều instance cho load banlancer. Các lập trình viên trong nhóm có thể chia sẻ ít nhiều về skill do cùng sử dụng chung công nghệ, các công nghệ đơn giản và đa số là dễ học. Thiết kế hệ thống theo mô hình Monolithic yêu cầu cơ sở hạ tầng đơn giản, thậm chí một container đơn giản cũng đủ để chạy ứng dụng.

Tuy nhiên, theo thời gian công nghệ phát triển, mô hình monolithic truyền thống dần trở nên không phù hợp cho các hệ thống lớn do chúng tạo ra nhất nhiều nhược điểm. Theo thời gian thì project trở nên phức tạp và lớn dần. Các tính năng mới sẽ mất nhiều thời gian hơn để phát triển và tái cấu trúc các tính năng hiện có sẽ nhiều khó khăn hơn. Các component được liên kết chặt chẽ với nhau dẫn đến side effect không mong muốn như khi thay đổi một component ảnh hưởng đến một component khác. Việc áp dụng công nghệ mới trở nên khó khăn vì toàn bộ ứng dụng phải thay đổi, do đó nhiều ứng dụng một khối thường phụ thuộc một công nghệ cũ và lỗi thời. Các service quan trọng không thể scale riêng dẫn đến lãng phí tài nguyên vì toàn bộ ứng dụng phải scale theo. Các ứng dụng một khối lớn sẽ có thời gian khởi động lâu và tốn tài nguyên CPU cũng như bộ nhớ. Các team tham gia vào dự án phải phụ thuộc lẫn nhau và rất khó để mở rộng quy mô team.

Từ những nhược điểm trên của mô hình monolithic truyền thống, mô hình microservice ra đời đã giải quyến được các vấn đề nêu trên

5.2.2 Mô hình microservice



Hình 5.2: Mô hình microservice

Đây là loại kiến trúc đang dần trở nên phổ biến trong những năm gần đây nhờ khả năng mudule hóa và khả năng mở rộng. Kiến trúc microservice có thể cung cấp hầu hết các tính năng của một ứng dụng một khối. Ngoài ra, nó cung cấp nhiều tính năng và linh hoạt hơn, do đó nó là sự lựa chọn ưu việt cho ứng dụng phức tạp. Không giống như kiến trúc một khối, khá khó để khái quát hóa kiến trúc microservice vì nó có thể thay đổi nhiều tùy thuộc vào trường hợp sử dụng và triển khai. Nhưng nhìn chung thì chúng cũng có một số đặc điểm tương đồng. Các component trong kiến trúc microservice liên kết khá lỏng lẻo (loose coupling). Các component có thể được phát triển, test, deploy và scale độc lập mà không ảnh hưởng đến các component khác. Các component không cần phải được phát triển cùng một stack công nghệ. Điều này có nghĩa là một component có thể chọn stack công nghệ và ngôn ngữ lập trình của riêng nó. Các component chủ yếu là nhẹ và chúng làm theo chức năng riêng biệt. Ví dụ dịch vụ xác thực sẽ chỉ quan tâm xác thực người dùng vào hệ thống.

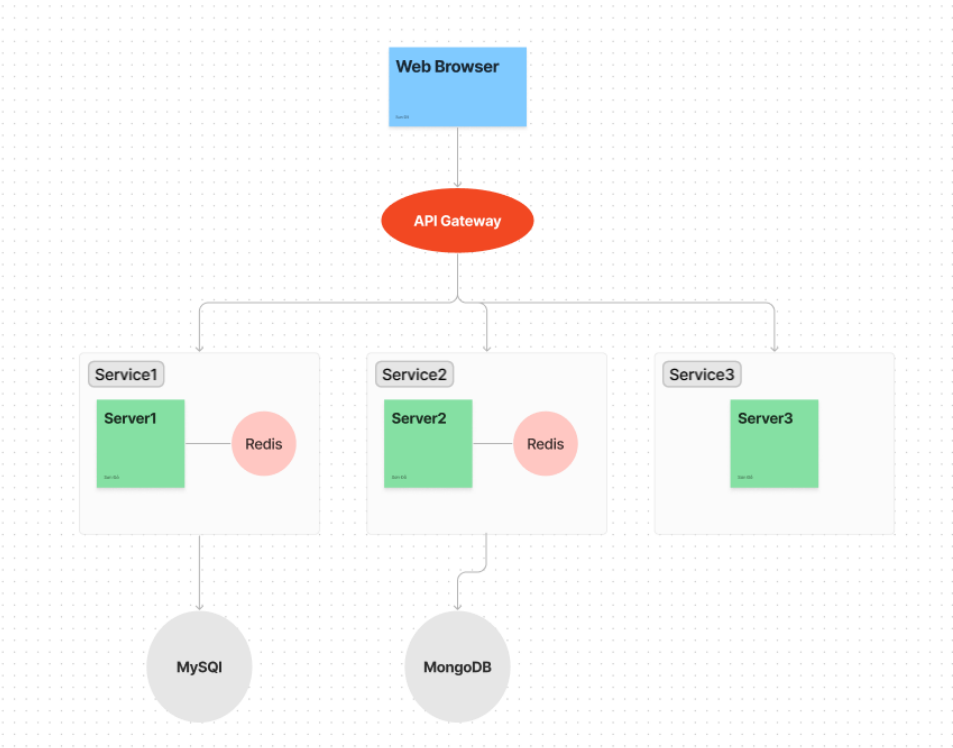
Mô hình microservice đã giải quyết được khá nhiều các vấn đề nghiêm trọng mà mô hình monolithic chưa thể xử lý. Vòng đời phát triển của dự án trở nên nhanh hơn, tính năng mới được phát triển nhanh hơn và tính năng cũ được cấu trúc lại dễ hơn. Việc áp dụng các công nghệ mới dễ dàng hơn trước, các component có thể được nâng cấp độc lập với nhau. Các mô hình scale phức tạp và hiệu quả hơn có thể

được thiết lập. Các service quan trọng có thể scale hiệu quả hơn. Các component riêng sẽ khởi động nhanh hơn và cải thiện thời gian khởi động của cả hệ thống. Các team tham gia sẽ ít phụ thuộc lẫn nhau. Kiến trúc này rất thích hợp cho các dự án có số lương thành viên lớn.

Tuy nhiên mô hình microservice cũng có những hạn chế. Phức tạp hơn về mặt tổng thể vì các component khác nhau có các stack công nghệ khác nhau nên buộc team phải tập trung đầu tư thời gian để theo kịp công nghệ. Khó thực hiện test end-to-end và integration test vì có nhiều stack công nghệ khác nhau. Deploy toàn bộ ứng dụng phức tạp hơn vì có nhiều container và nền tảng ảo hóa liên quan. Ứng dụng được scale hiệu quả hơn nhưng thiết lập nâng cấp sẽ phức tạp hơn vì nó sẽ yêu cầu nâng cao nhiều tính năng như truy tìm dịch vụ (service discovery), định tuyến DNS,... Yêu cầu một team-size lớn để maintain ứng dụng vì có nhiều component và công nghệ khác nhau. Các thành viên trong team chia sẻ các skill khác nhau dựa trên component họ làm nên sẽ tạo ra sự khó khăn khi thay thế và chia sẻ kiến thức. Stack công nghệ phức tạp và khó để học hơn.

5.3 Thiết kế chi tiết kiến trúc hệ thống tương ứng theo mô hình Microservice

Đề tài đồ án tốt nghiệp của tôi là xây dựng một trang web nhắn tin trực tuyến trong doanh nghiệp. Một trang web nhắn tin trực tuyến sẽ có lượng cơ sở dữ liệu về tin nhắn, thông báo, người dùng là rất lớn, khi số lương người dùng tăng lên, lượng requests tới server cũng tăng dẫn đến ảnh hưởng tới khả năng xử lý của server. Từ những ưu điểm của mô hình Microservice so với mô hình mô hình Monolithic, em đã quyết định lựa chọn mô hình Microservice là mô hình kiến trúc sẽ được sử dụng để thiết kế trang web nhắn tin trực tuyến trong đồ án tốt nghiệp này.



Hình 5.3: Mô hình Client - Server kết hợp Microservice

- Web Browser: là trình duyệt web, là nơi có vai trò lưu trữ các thành phần giao diện như nút bấm, các biểu mẫu,. . . Thành phần này có tác dụng tạo ra các phần giao diện với mục đích hiển thị và giúp người sử dụng có thể tương tác được với hệ thống thông qua các giao diện này.
- API gateway là nơi tiếp nhận các request do web browser gửi tới, nó có nhiệm vụ điều phối các requests nào sẽ được gửi tới server nào
- Server 1 kết nối tới hệ quản trị cơ sở dữ liệu MySQL. Server 1 có nhiệm vụ tiếp nhận xử lý các requests làm thay đổi, cập nhật dữ liệu trong hệ quản trị cơ sở dữ liệu mySQL. Server 1 sử dụng Redis để cache dữ liệu, đặc biệt là với các API có lượng dữ liệu trả về lớn
- Server 2 kết nối tới hệ quản trị cơ sở dữ liệu MongoDB. Server 2 có nhiệm vụ tiếp nhận xử lý các requests làm thay đổi, cập nhật dữ liệu trong hệ quản trị cơ sở dữ liệu MongoDB. Server 2 sử dụng Redis để cache dữ liệu, đặc biệt là với các API có lượng dữ liệu trả về lớn
- Server 3 có nhiệm vụ chạy web socket. Server 3 có nhiệm vụ xử lý các requests yêu cầu realtime, gửi và nhận dữ liệu ngay tức thì
- Hệ quản trị cơ sở dữ liệu MySQL được sử dụng để lưu trữ các loại dữ liệu yêu cầu chặt chẽ, có tính ràng buộc cao, đảm bảo tính toàn vẹn dữ liệu
- Hệ quản trị cơ sở dữ liệu MongoDB được sử dụng để lưu trữ các loại dữ liệu

có số lượng lớn, có tính rời rạc, cần tốc độ truy vấn nhanh

CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Kết luận

Từ những nội dung đã được trình bày trong đồ án này, chúng ta đã thấy được vấn đề truyền nhận thông tin trực tuyến đóng vai trò quan trọng trong cộng việc. Từ đó chỉ ra được việc áp dụng thành tựu khoa học kĩ thuật, cụ thể ở đây là sử dụng hệ thống máy tính và mạng Internet có thể giúp cho thực hiện trao đổi, truyền nhận thông tin trong công việc hiệu quả hơn mà còn tiết kiệm chi phí xét về lâu dài. Sử dụng các công nghệ và thiết bị trên giúp cho con người dễ dàng giao tiếp hơn, giúp quản lí các thông tin một cách đáng tin cậy, dễ dàng trong vấn đề so sánh, tiếp cận đến từng nội dung công việc, đồng thời nâng cao tính tương tác giữa các bên trong quá trình làm việc.

Các sản phẩm hiện tại trong nước và ngoài nước đã giải quyết được tốt các vấn đề. Những sản phẩm kể trên đã thực hiện tốt việc quản lý người dùng, tin nhắn, tương tác. Sản phẩm của đồ án này đã chọn lựa các tính năng hay, quan trọng từ các các sản phẩm nổi bật trên thị trường , cùng với đó thêm vào một số tính năng phụ giúp cho việc trao đổi thông tin, văn bản trong doanh nghiệp hiệu quả hơn.

6.2 Hướng phát triển

Một sản phẩm, trong quá trình phát triển và vận hành, luôn cần được nâng cao hơn cả về các tính năng và trải nghiệm để đem lại hiệu quả tốt hơn, giúp tạo ra và giữ được những người đã sử dụng tiếp tục sử dụng sản phẩm đó. Chính vì vậy bản thân hệ thống được phát triển trong đồ án này cũng cần những hướng đi mới để cải thiện được các chức năng sao cho tiện dụng với người sử dụng nhất.

Trước hết, để sản phẩm trở nên hoàn thiện hơn, cần phải bổ sung thêm tính năng call video 2 người trực tiếp và call video nhóm. Việc sử dụng văn bản để truyền nhận thông tin cũng đáp ứng được rất nhiều tiêu chí nhưng vẫn còn một số hạn chế. Giao tiếp bằng hình ảnh và giọng nói sẽ giúp cho việc trao đổi thông tin trở nên nhanh và dễ dàng hơn. Các cuộc họp nhóm, call video sẽ tạo không gian giao tiếp tốt, mọi người trong nhóm, dự án có thể thảo luận trực tiếp về các vấn đề khó khắn đang gặp phải, cùng nhau tranh luận để đưa ra được hướng giải quyết nhanh và tốt hơn. Hơn nữa giao tiếp bằng hình ảnh và giọng nói cũng giúp người sử dụng tăng cảm giác gần gũi, nghiêm chỉnh chánh nhàm chán, mất tập trung.

Một ứng dụng nhắn tin trực tuyến sẽ có số lượng dữ liệu và tin nhắn, thông báo là vô cùng lớn, vì vậy việc xử lý, cache dữ liệu như nào cho nhanh và hiệu quả nhất là vô cùng quan trọng. Sản phẩm của em có sử dụng redis để cach dữ liệu với các

API có số lương request trả về lớn nhưng điều này vẫn là chưa đủ. Khi số lượng người dùng tăng lên , lượng requests gửi tứi server và dữ liệu cũng sẽ tăng lên rất nhiều ảnh hưởng lớn tới tốc độ xử lý. Vì vậy cần phải có các thuật toán xử lý phân bổ dữ liệu hợp lý, các cách phân bổ, điều phối request để giảm tải tối đa cho server

Hơn nữa, các ứng dụng nhắn tin trực tuyến trong doanh nghiệp, tổ chức sẽ yêu cầu tính bảo mật thông tin cao, sản phẩm hiện tại của em chưa đáp ứng được yêu cầu này. Một ứng dụng nhắn tin trong doanh nghiệp sẽ cần phải có các phương pháp mã hóa, ẩn tin nhắn giúp đảm bảo an toàn thông tin. Điều này là vô cùng quan trọng vì các thông tin trong dự án là vô cùng quan trọng, nếu để rơi vào tay kẻ xấu hay công ty đối thủ cạnh tranh sẽ gây ra những hậu quả vô cùng nghiêm trọng.

Tổng kết lại, hệ thống còn có nhiều hướng đi để phát triển góp phần đem lại trải nghiệm tốt hơn, hiệu quả hơn và thân thiện hơn với người dùng trong tương lai.

Tài liệu tham khảo

[1] Eric Freeman, Elisabeth Robson, *Head First JavaScript Programming: A Brain-Friendly Guide 1st Edition* (March 26, 2014)

[2] Công ty công nghệ Slack Technologies thuộc sở hữu của Salesforce, *Về Slack*. [Online]. Available: `https://slack.com/customer-stories` (visited on 10/10/2022)

[3] Microsoft Corporation, *Microservice architecture style*. [Online]. Available: `https://learn.microsoft.com/en-us/azure/architecture/guide/architecture-styles/microservices` (visited on 10/10/2022)

[4] Redis Ltd, *Introduction to redis*. [Online]. Available: `https://redis.io/docs/about/` (visited on 10/10/2022).

[5] NodeJS Ltd, *Introduction to NodeJS*. [Online]. Available: `https://nodejs.org/en/docs/` (visited on 10/10/2022).

[6] ReactJS Ltd, *Introduction to ReactJS*. [Online]. Available: `https://reactjs.org/docs/getting-started.html` (visited on 10/10/2022)

PHỤ LỤC