

ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Thiết kế xây dựng mạng xã hội EGOSNET với kiến trúc microservices

PHẠM ĐỨC QUÝ

quy.pd183816@sis.hust.edu.vn

Ngành Công nghệ thông tin và truyền thông

Giảng viên hướng dẫn: TS. Nguyễn Thị Thanh Nga

Chữ ký GVHD

Khoa: Kỹ thuật máy tính

Trường: Công nghệ Thông tin và Truyền thông

HÀ NỘI, 01/2024

LỜI CẢM ƠN

Vậy là hành trình ở Bách Khoa Hà Nội cũng sắp khép lại với rất nhiều kỷ niệm vui buồn.

Em xin gửi lời cảm ơn tới TS. Nguyễn Thị Thanh Nga đã đồng hành, giúp đỡ và chỉ dẫn em trong suốt thời gian thực hiện hoàn thiện Đồ án tốt nghiệp. Em cảm ơn cô đã đưa ra những góp ý, nhận xét để em có thể hoàn thành sản phẩm một cách tốt nhất.

Em xin gửi lời cảm ơn đến gia đình, những người bạn và những người đồng nghiệp đã đồng hành, luôn bên cạnh và hỗ trợ lẫn nhau trong thời gian hoàn thành đồ án này.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Mạng xã hội (social network) là hệ thống thông tin cung cấp cho cộng đồng người sử dụng mạng các dịch vụ lưu trữ, cung cấp, sử dụng, tìm kiếm, chia sẻ và trao đổi thông tin với nhau, bao gồm dịch vụ tạo trang thông tin điện tử cá nhân, diễn đàn (forum), trò chuyện (chat) trực tuyến, chia sẻ âm thanh, hình ảnh và các hình thức dịch vụ tương tự khác.

Hiểu được điều đó, đồ án mạng xã hội này đặt ra mục tiêu xây dựng một nền tảng trực tuyến đa chức năng, nhằm tạo ra một không gian kết nối chặt chẽ giữa cộng đồng người dùng. Quá trình phát triển bao gồm các chức năng cơ bản như đăng ký, đăng nhập, quản lý hồ sơ cá nhân và tương tác xã hội, như kết bạn và theo dõi. Các tính năng nâng cao như chia sẻ nội dung đa dạng, tham gia thảo luận, và quản lý quyền riêng tư cũng được tích hợp để tạo ra một trải nghiệm người dùng đầy đủ và linh hoạt.

Sinh viên thực hiện
(Ký và ghi rõ họ tên)

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI.....	1
1.1 Đặt vấn đề.....	1
1.2 Mục tiêu và phạm vi đề tài.....	1
1.3 Định hướng giải pháp.....	2
1.4 Bố cục đồ án	2
CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU.....	3
2.1 Khảo sát hiện trạng	3
2.2 Tổng quan chức năng	4
2.2.1 Biểu đồ use case tổng quát	4
2.2.2 Biểu đồ use case phân rã hiển thị danh sách bài viết	7
2.2.3 Biểu đồ use case phân rã nhắn tin	14
2.2.4 Biểu đồ use case phân rã quản lý bài viết.....	19
2.2.5 Biểu đồ use case phân rã quản lý bình luận.....	26
2.2.6 Biểu đồ use case phân rã quản lý nhóm	33
2.2.7 Biểu đồ use case phân rã quản lý thông báo	39
2.2.8 Biểu đồ use case phân rã quản lý trang cá nhân	43
2.3 Yêu cầu phi chức năng	49
CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG.....	50
3.1 VueJS	50
3.2 ExpressJS	50
3.3 MongoDB	50
3.4 Google Firebase	51
3.5 Redis	51
3.6 Minio	52

3.7 Docker.....	52
3.8 Microservices.....	52
3.9 RabbitMQ.....	53
CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG	54
4.1 Thiết kế kiến trúc.....	54
4.1.1 Lựa chọn kiến trúc phần mềm	54
4.1.2 Thiết kế chi tiết gói	57
4.1.3 Thiết kế chi tiết kiến trúc microservices cho hệ thống.....	61
4.2 Thiết kế chi tiết.....	69
4.2.1 Thiết kế giao diện	69
4.2.2 Thiết kế lớp	72
4.2.3 Thiết kế cơ sở dữ liệu	85
4.3 Xây dựng ứng dụng.....	90
4.3.1 Thư viện và công cụ sử dụng.....	90
4.3.2 Kết quả đạt được	91
4.4 Kiểm thử.....	94
4.4.1 Kiểm thử chức năng đăng bài	94
4.4.2 Kiểm thử chức năng bình luận	94
4.5 Triển khai	95
4.5.1 Tạo server.....	95
4.5.2 Tạo các file Docker và docker-compose.....	96
4.5.3 Lưu trữ source code trên Github	98
4.5.4 Build mã nguồn trên server	98
4.5.5 Trỏ tên miền	98

CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT	99
5.1 Thiết kế hệ thống theo kiến trúc microservices	99
5.1.1 Đặt vấn đề	99
5.1.2 Thiết kế giải pháp	100
5.1.3 Kết quả đạt được	101
5.2 Chức năng thông báo	101
5.2.1 Tổng quan vấn đề	101
5.2.2 Thiết kế giải pháp	101
5.2.3 Kết quả đạt được	104
CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	105
6.1 Kết luận	105
6.2 Hướng phát triển.....	105
TÀI LIỆU THAM KHẢO.....	106

DANH MỤC HÌNH VẼ

Hình 2.1	Biểu đồ usecase Tổng quát	4
Hình 2.2	Biểu đồ hoạt động Tìm kiếm bài viết	6
Hình 2.3	Biểu đồ tuần tự tìm kiếm bài viết	6
Hình 2.4	Biểu đồ usecase hiển thị danh sách bài viết	7
Hình 2.5	Biểu đồ hoạt động thích bài viết	9
Hình 2.6	Biểu đồ tuần tự thích bài viết	9
Hình 2.7	Biểu đồ hoạt động bình luận bài viết	11
Hình 2.8	Biểu đồ tuần tự bình luận bài viết	11
Hình 2.9	Biểu đồ hoạt động chia sẻ bài viết	13
Hình 2.10	Biểu đồ tuần tự chia sẻ bài viết	13
Hình 2.11	Biểu đồ usecase nhắn tin	14
Hình 2.12	Biểu đồ hoạt động nhắn tin	15
Hình 2.13	Biểu đồ tuần tự hiển thị danh sách người dùng	16
Hình 2.14	Biểu đồ tuần tự xem tin nhắn người dùng	17
Hình 2.15	Biểu đồ tuần tự nhắn tin cho người dùng	19
Hình 2.16	Biểu đồ usecase quản lý bài viết	19
Hình 2.17	Biểu đồ hoạt động xóa bài viết	21
Hình 2.18	Biểu đồ tuần tự xóa bài viết	21
Hình 2.19	Biểu đồ hoạt động sửa bài viết	23
Hình 2.20	Biểu đồ tuần tự sửa bài viết	23
Hình 2.21	Biểu đồ hoạt động xem chi tiết bài viết	25
Hình 2.22	Biểu đồ tuần tự xem chi tiết bài viết	25
Hình 2.23	Biểu đồ usecase quản lý bình luận	26
Hình 2.24	Biểu đồ hoạt động sửa bình luận	27
Hình 2.25	Biểu đồ tuần tự sửa bình luận	28
Hình 2.26	Biểu đồ hoạt động xóa bình luận	29
Hình 2.27	Biểu đồ tuần tự xóa bình luận	30
Hình 2.28	Biểu đồ hoạt động thích bình luận	32
Hình 2.29	Biểu đồ tuần tự thích bình luận	32
Hình 2.30	Biểu đồ usecase quản lý nhóm	33
Hình 2.31	Biểu đồ hoạt động tạo nhóm	34
Hình 2.32	Biểu đồ tuần tự tạo nhóm	35
Hình 2.33	Biểu đồ hoạt động tham gia nhóm	36
Hình 2.34	Biểu đồ tuần tự tham gia nhóm	37
Hình 2.35	Biểu đồ hoạt động xem chi tiết nhóm	38

Hình 2.36 Biểu đồ tuần tự xem chi tiết nhóm	39
Hình 2.37 Biểu đồ usecase quản lý thông báo	39
Hình 2.38 Biểu đồ hoạt động đánh dấu là đã đọc	41
Hình 2.39 Biểu đồ tuần tự đánh dấu là đã đọc	41
Hình 2.40 Biểu đồ hoạt động xem chi tiết thông báo	42
Hình 2.41 Biểu đồ tuần tự xem chi tiết thông báo	43
Hình 2.42 Biểu đồ usecase quản lý trang cá nhân	43
Hình 2.43 Biểu đồ hoạt động chỉnh sửa thông tin cá nhân	45
Hình 2.44 Biểu đồ tuần tự chỉnh sửa thông tin cá nhân	45
Hình 2.45 Biểu đồ hoạt động nhắn tin người dùng	46
Hình 2.46 Biểu đồ tuần tự nhắn tin người dùng	47
Hình 2.47 Biểu đồ hoạt động theo dõi người dùng	48
Hình 2.48 Biểu đồ tuần tự theo dõi người dùng	48
 Hình 4.1 Mô hình Microservices	54
Hình 4.2 Mô hình Microservices trong hệ thống	55
Hình 4.3 Biểu đồ gói của hệ thống	56
Hình 4.4 Biểu đồ gói chi tiết Feed Management	57
Hình 4.5 Biểu đồ gói chi tiết Group Management	58
Hình 4.6 Biểu đồ gói chi tiết Message Management	59
Hình 4.7 Biểu đồ gói chi tiết Notification Management	60
Hình 4.8 Biểu đồ gói chi tiết User Management	61
Hình 4.9 Thiết kế chi tiết kiến trúc microservices cho hệ thống	61
Hình 4.10 Luồng bình luận và gửi thông báo cho người dùng	63
Hình 4.11 Luồng tạo bài viết	64
Hình 4.12 Luồng lấy danh sách bài viết	65
Hình 4.13 Luồng xác thực người dùng	66
Hình 4.14 Thiết kế giao diện trang chủ	69
Hình 4.15 Thiết kế giao diện chi tiết bài viết	70
Hình 4.16 Thiết kế giao diện đăng bài	70
Hình 4.17 Thiết kế giao diện nhóm	71
Hình 4.18 Thiết kế giao diện trang cá nhân	71
Hình 4.19 Thiết kế giao diện nhắn tin	72
Hình 4.20 Chi tiết sơ đồ lớp User	72
Hình 4.21 Chi tiết sơ đồ lớp Friend	74
Hình 4.22 Chi tiết sơ đồ lớp Feed	75
Hình 4.23 Chi tiết sơ đồ lớp Comment	77
Hình 4.24 Chi tiết sơ đồ lớp Reaction	78

Hình 4.25 Chi tiết sơ đồ lớp Group	80
Hình 4.26 Chi tiết sơ đồ lớp Message	82
Hình 4.27 Chi tiết sơ đồ lớp Notification	84
Hình 4.28 Sơ đồ thực thể liên kết	86
Hình 4.29 Giao diện màn hình đăng nhập cho người dùng cuối	91
Hình 4.30 Giao diện màn hình trang chủ	91
Hình 4.31 Giao diện màn hình chi tiết bài viết	91
Hình 4.32 Giao diện màn hình tạo bài viết	92
Hình 4.33 Giao diện màn hình nhóm	92
Hình 4.34 Giao diện màn hình trang cá nhân	92
Hình 4.35 Giao diện màn hình nhắn tin	93
Hình 4.36 Giao diện màn hình xem thông báo ở trang chủ	93
Hình 4.37 Giao diện màn hình xem thông báo ở trang chi tiết	93
Hình 4.38 Thông tin tổng quan của server	95
Hình 4.39 Dockerfile	96
Hình 4.40 Docker compose file	97
Hình 4.41 Lưu trữ source code trên github	98
Hình 4.42 Cấu hình DNS	98
Hình 5.1 Cấu trúc hệ thống	100
Hình 5.2 Mô hình hoạt động phổ biến của FCM	102
Hình 5.3 Luồng hoạt động của chức năng thông báo	102
Hình 5.4 Kết quả đạt được của chức năng thông báo	104
Hình 5.5 Danh sách thông báo	104

DANH MỤC BẢNG BIỂU

Bảng 2.1	Bảng đặc tả usecase thích bài viết.	5
Bảng 2.2	Bảng đặc tả usecase thích bài viết.	8
Bảng 2.3	Bảng đặc tả usecase bình luận bài viết.	10
Bảng 2.4	Bảng đặc tả usecase chia sẻ bài viết.	12
Bảng 2.5	Bảng đặc tả usecase hiển thị danh sách người dùng	16
Bảng 2.6	Bảng đặc tả usecase xem tin nhắn người dùng.	17
Bảng 2.7	Bảng đặc tả usecase nhắn tin cho người dùng.	18
Bảng 2.8	Bảng đặc tả usecase xóa bài viết	20
Bảng 2.9	Bảng đặc tả usecase sửa bài viết.	22
Bảng 2.10	Bảng đặc tả usecase xem chi tiết bài viết.	24
Bảng 2.11	Bảng đặc tả usecase sửa bình luận	27
Bảng 2.12	Bảng đặc tả usecase xóa bình luận.	29
Bảng 2.13	Bảng đặc tả usecase thích bình luận.	31
Bảng 2.14	Bảng đặc tả usecase tạo nhóm.	34
Bảng 2.15	Bảng đặc tả usecase tham gia nhóm.	36
Bảng 2.16	Bảng đặc tả usecase xem chi tiết nhóm.	38
Bảng 2.17	Bảng đặc tả usecase đánh dấu là đã đọc.	40
Bảng 2.18	Bảng đặc tả usecase xem chi tiết thông báo.	42
Bảng 2.19	Bảng đặc tả usecase chỉnh sửa thông tin cá nhân.	44
Bảng 2.20	Bảng đặc tả usecase nhắn tin.	46
Bảng 2.21	Bảng đặc tả usecase theo dõi người dùng.	47
Bảng 4.1	Bảng Mô tả một số luồng khác	68
Bảng 4.2	Bảng đặc tả lớp MUser	73
Bảng 4.3	Bảng đặc tả lớp CUser	73
Bảng 4.4	Bảng đặc tả lớp VUser	73
Bảng 4.5	Bảng đặc tả lớp MFriend	74
Bảng 4.6	Bảng đặc tả lớp CFriend	74
Bảng 4.7	Bảng đặc tả lớp VFriend	75
Bảng 4.8	Bảng đặc tả lớp MFeed	76
Bảng 4.9	Bảng đặc tả lớp CFeed	76
Bảng 4.10	Bảng đặc tả lớp VFeed	76
Bảng 4.11	Bảng đặc tả lớp MComment	77
Bảng 4.12	Bảng đặc tả lớp CComment	77
Bảng 4.13	Bảng đặc tả lớp VComment	78

Bảng 4.14	Bảng đặc tả lớp MLike	79
Bảng 4.15	Bảng đặc tả lớp CLike	79
Bảng 4.16	Bảng đặc tả lớp VLike	79
Bảng 4.17	Bảng đặc tả lớp MGroup	80
Bảng 4.18	Bảng đặc tả lớp MMod	81
Bảng 4.19	Bảng đặc tả lớp MUserGroups	81
Bảng 4.20	Bảng đặc tả lớp CGroup	81
Bảng 4.21	Bảng đặc tả lớp VGroup	82
Bảng 4.22	Bảng đặc tả lớp MChannel	83
Bảng 4.23	Bảng đặc tả lớp MMessage	83
Bảng 4.24	Bảng đặc tả lớp CMessage	83
Bảng 4.25	Bảng đặc tả lớp VMessage	83
Bảng 4.26	Bảng đặc tả lớp MNotification	84
Bảng 4.27	Bảng đặc tả lớp CNotification	85
Bảng 4.28	Bảng đặc tả lớp VNotification	85
Bảng 4.29	Bảng User	86
Bảng 4.30	Bảng Friend	87
Bảng 4.31	Bảng Feed	87
Bảng 4.32	Bảng Comment	87
Bảng 4.33	Bảng Reaction	88
Bảng 4.34	Bảng Group	88
Bảng 4.35	Bảng Mod	88
Bảng 4.36	Bảng UserGroup	89
Bảng 4.37	Bảng Channel	89
Bảng 4.38	Bảng Member	89
Bảng 4.39	Bảng Message	89
Bảng 4.40	Bảng Notification	90
Bảng 4.41	Danh sách thư viện và công cụ sử dụng	90
Bảng 4.42	Bảng kiểm thử chức năng đăng bài	94
Bảng 4.43	Bảng kiểm thử chức năng bình luận	94

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Thuật ngữ	Ý nghĩa
API	Giao diện lập trình ứng dụng (Application Programming Interface)
EUD	Phát triển ứng dụng người dùng cuối(End-User Development)
FCM	Dịch vụ đám mây đa nền tảng dành cho tin nhắn và thông báo (Firebase Cloud Messaging)
GWT	Công cụ lập trình Javascript bằng Java của Google (Google Web Toolkit)
HTML	Ngôn ngữ đánh dấu siêu văn bản (HyperText Markup Language)
IaaS	Dịch vụ hạ tầng

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Đặt vấn đề

Trong thời đại hiện đại, mạng xã hội đã trở thành một phần không thể thiếu trong cuộc sống hàng ngày của chúng ta. Việc triển khai mạng xã hội không chỉ đơn thuần là một xu hướng công nghệ, mà còn là một thách thức và bài toán đầy thách thức cần phải giải quyết. Tình hình này xuất phát từ nhu cầu ngày càng cao của cộng đồng sử dụng internet, nơi mọi người muốn chia sẻ thông tin, tương tác và kết nối với nhau một cách nhanh chóng và thuận tiện.

Một trong những vấn đề lớn nhất liên quan đến triển khai mạng xã hội là sự cần thiết của nó trong việc duy trì và phát triển mối quan hệ xã hội. Xã hội ngày nay đang đổi mới với tình trạng giảm đối thoại trực tiếp và tăng cường giao tiếp trực tuyến. Điều này đặt ra một thách thức đối với môi trường xã hội truyền thống và đồng thời mở ra một lĩnh vực mới, nơi mạng xã hội có thể giúp tạo ra một không gian giao tiếp và kết nối mới.

Một khía cạnh quan trọng khác của vấn đề này là tầm quan trọng của mạng xã hội trong việc chia sẻ thông tin và tiếp cận kiến thức. Trong một thế giới ngày càng phức tạp, mạng xã hội có thể trở thành một công cụ mạnh mẽ giúp đưa thông tin đến đúng đối tượng mục tiêu. Điều này không chỉ tăng cường sự thông tin, mà còn tạo ra cơ hội mới cho các tổ chức, doanh nghiệp và cá nhân để chia sẻ ý kiến, kiến thức chuyên sâu và ý nghĩa của họ.

Nếu vấn đề triển khai mạng xã hội được giải quyết một cách hiệu quả, nó có thể mang lại lợi ích không chỉ cho cá nhân và cộng đồng mà còn cho doanh nghiệp và xã hội nói chung. Do đó, em sẽ xây dựng một mạng xã hội nơi mọi người có thể chia sẻ thông tin, tăng cường sê kết nối đối tượng. Mọi người có thể đăng bài, bình luận, tương tác với nhau hoặc có thể nhắn tin trực tiếp trao đổi thông tin với nhau.

1.2 Mục tiêu và phạm vi đề tài

Hiện nay, trên môi trường Internet đã xuất hiện nhiều mạng xã hội rất đa dạng. Tuy nhiên, bên cạnh những ưu điểm thì các website này vẫn tồn tại những bất cập, hạn chế nhất định. Vì vậy mục tiêu của đồ án này em muốn tạo ra một trang website mạng xã hội giúp mọi người chia sẻ thông tin nhanh hơn, giao tiếp và kết nối của nhiều hơn. Ngoài ra, hệ thống cũng được mong đợi trở thành một nơi mà mọi người có thể thoải mái tương tác, trao đổi, chia sẻ thông tin nhiều hơn.

Với mục đích là tương tác, trao đổi và chia sẻ thông tin. Hệ thống sẽ có các chức năng như đăng bài, bình luận, thích bài viết. Ngoài ra còn có chức năng tạo nhóm,

chia sẻ các thông tin chỉ liên quan đến mục đích của nhóm. Chức năng nhắn tin, giúp trao đổi thông tin một cách riêng tư hơn giữa các người dùng.

1.3 Định hướng giải pháp

Sau khi nêu ra các vấn đề cần giải quyết ở trên, giải pháp em đưa ra là xây dựng một Website mạng xã hội mang tên **Egosnet**. Hệ thống sử dụng framework VueJS để xây dựng, lập trình giao diện người dùng và framework ExpressJS cho việc xây dựng, lập trình server. Lý do em lựa chọn VueJS vì đây là một framework linh động và dễ sử dụng, cho phép xây dựng website một cách đơn giản, hiệu suất cao, tốc độ tải trang nhanh chóng. Còn về phía server, lý do em chọn ExpressJS vì đây là một framework mã nguồn mở phát triển trên nền tảng NodeJS mỏng nhẹ, dễ học, dễ sử dụng, hiệu suất cao và cung cấp các tính năng mạnh mẽ trong việc phát triển các API.

Website **Egosnet** được mong đợi là một nền tảng mạng xã hội, là một nơi mà mọi người có thể đăng bài viết, chia sẻ các quan điểm cá nhân, chia sẻ cuộc sống của mình lên mạng xã hội, người dùng có thể tương tác với người dùng khác thông qua bình luận hay thích bài viết. Ngoài ra, ứng dụng còn có các nhóm người dùng, nó được tạo ra nhằm mục đích hoạt động về các chủ đề khác nhau mang lại những nét riêng cho mạng xã hội.

1.4 Bố cục đồ án

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

Trong chương 2, em sẽ trình bày về khảo sát thực trạng vấn đề và giải pháp đã tồn tại. Sau đó đánh giá, so sánh và phân tích tổng quan những chức năng chính của hệ thống sẽ xây dựng. Phần cuối chương sẽ là những yêu cầu phi chức năng của hệ thống.

Chương 3 sẽ là những công nghệ được em sử dụng trong sản phẩm đồ án này.

Chương 4 trình bày về quá trình triển khai hệ thống bắt đầu từ kiến trúc đến những thiết kế sơ đồ gói, sơ đồ lớp, các biểu đồ liên quan sau đó là cách thức đưa hệ thống lên môi trường Internet. Và cuối cùng là một số hình ảnh thể hiện kết quả đạt được của hệ thống.

Chương 5 sẽ là những giải pháp, chức năng nổi bật của hệ thống.

Chương 6 sẽ là phần kết luận, ưu điểm, nhược điểm, hạn chế của đồ án so với các website và hướng phát triển của đồ án trong tương lai.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

Hiểu được vấn đề được đặt ra, trong chương 2 này em sẽ tiến hành một số khảo sát để nắm bắt được nhu cầu sử dụng mạng xã hội hiện nay của người dùng, thời gian sử dụng mạng xã hội. Sau đó, tiến hành phân tích, nhận xét và so sánh ưu, nhược điểm của các hệ thống, sản phẩm tương tự. Cuối cùng sẽ là tổng quan hệ thống, các chức năng chính, biểu đồ usecase, đặc tả cho những usecase, quy trình nghiệp vụ và cuối cùng là đưa ra các yêu cầu phi chức năng của hệ thống.

2.1 Khảo sát hiện trạng

Hiện nay, Khi thời đại công nghệ ngày càng phát triển, nhu cầu giao tiếp, trao đổi thông tin, giải trí của con người ngày càng nhiều. Do đó, mạng xã hội ra đời như một lẽ tất yếu. Mạng xã hội đã trở thành một phần không thể thiếu trong cuộc sống của chúng ta, định hình cách chúng ta giao tiếp, tương tác và kết nối với nhau. Nó đã mở ra cánh cửa cho sự tiến bộ to lớn trong công nghệ và truyền thông, tạo điều kiện thuận lợi cho việc chia sẻ kiến thức, ý tưởng và thậm chí cả những trải nghiệm cá nhân. Trong thời đại kỹ thuật số ngày nay, mạng xã hội là phương tiện trao đổi quan trọng giữa các cá nhân và cộng đồng, thúc đẩy cảm giác kết nối và thống nhất toàn cầu.

Ở Việt Nam, mạng xã hội cũng đã phát triển theo cấp số nhân. Những người trẻ tuổi thường sử dụng nó để kết nối với bạn bè và gia đình, tham gia vào các hoạt động như diễn đàn trực tuyến và nhóm trên mạng xã hội. Điều này đặc biệt phổ biến trong giới trẻ, những người luôn tìm kiếm những trải nghiệm mới và cập nhật. Ngoài ra, thanh thiếu niên ở độ tuổi vị thành niên đang sử dụng mạng xã hội để khám phá thế giới xung quanh họ và duy trì kết nối với cha mẹ và bạn bè của họ. Mạng xã hội như là một thế giới ảo được tạo ra trên mạng vậy, ta không cần phải gặp mặt mọi người cũng có thể trò chuyện, trao đổi thông tin với nhau. Nó rất tiện lợi, ngoài việc kết nối con người với nhau, với những nhà bán hàng, nó còn là nơi để quảng bá sản phẩm giúp cho sản phẩm của họ có thể tiếp cận nhiều đối tượng khách hàng hơn.

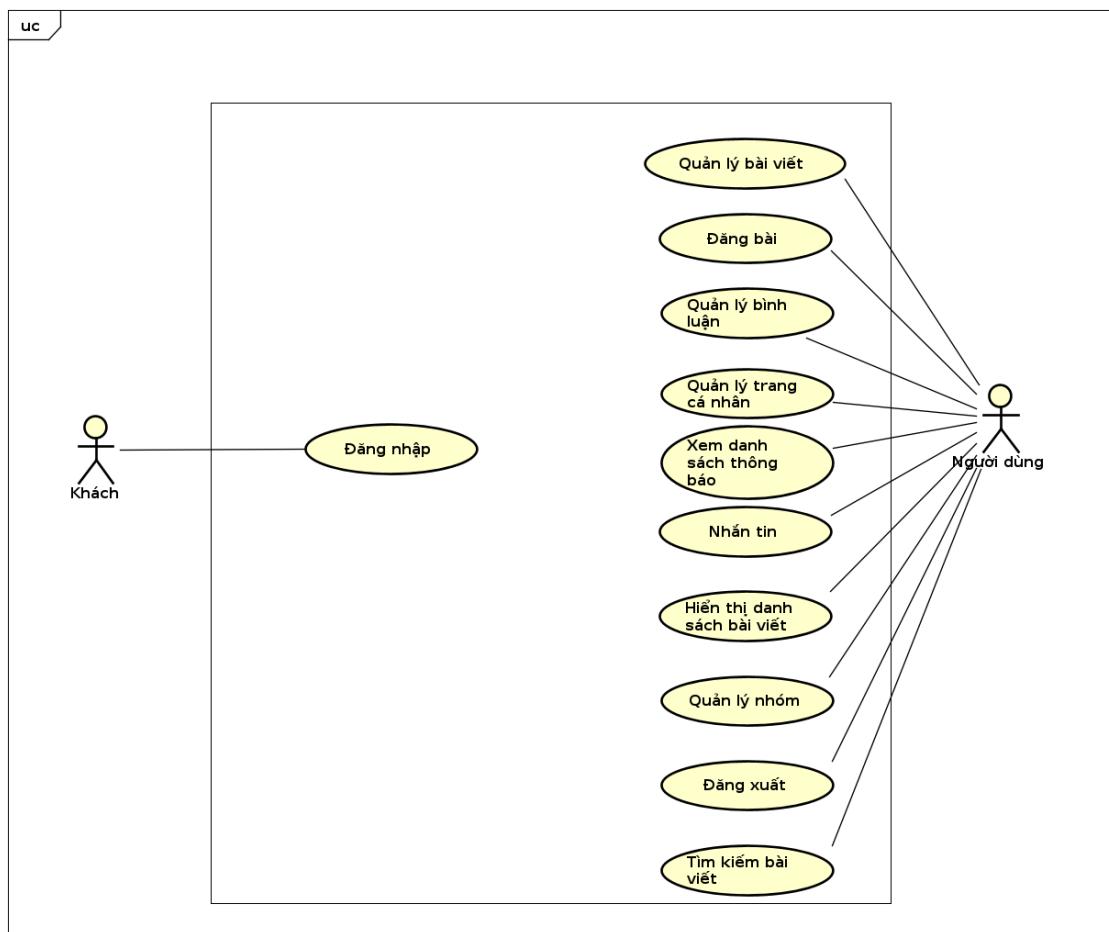
Hiểu được những nhu cầu đó, em sẽ xây dựng một mạng xã hội nơi mà chúng ta có thể trao đổi thông tin một cách dễ dàng. Qua những phân tích, em sẽ xây dựng một trang mạng xã hội với các chức năng chính như sau: đăng bài, tìm kiếm bài viết, bình luận, bày tỏ cảm xúc, quản lý các nhóm, quản lý thông báo, quản lý trang cá nhân, nhắn tin.

Sau khi phân tích các chức năng chính. Sau đây, em sẽ phân tích tổng quan chức

năng bằng các biểu đồ usecase và đặc tả các chức năng đó. Tất cả các chức năng đều sẽ được mô tả thông qua hình vẽ và bảng biểu, kèm theo đó là những lời giải thích và phân tích rõ ràng và bảng biểu và hình vẽ đó.

2.2 Tổng quan chức năng

2.2.1 Biểu đồ use case tổng quát



Hình 2.1: Biểu đồ usecase Tổng quát

Trên đây là sơ đồ tổng quan của hệ thống. Có 4 tác nhân chính là:

- Khách: Người dùng chưa đăng nhập
- Người dùng: Những user đã đăng nhập

a, Biểu đồ usecase tìm kiếm bài viết.

- Bảng đặc tả usecase tìm kiếm bài viết.

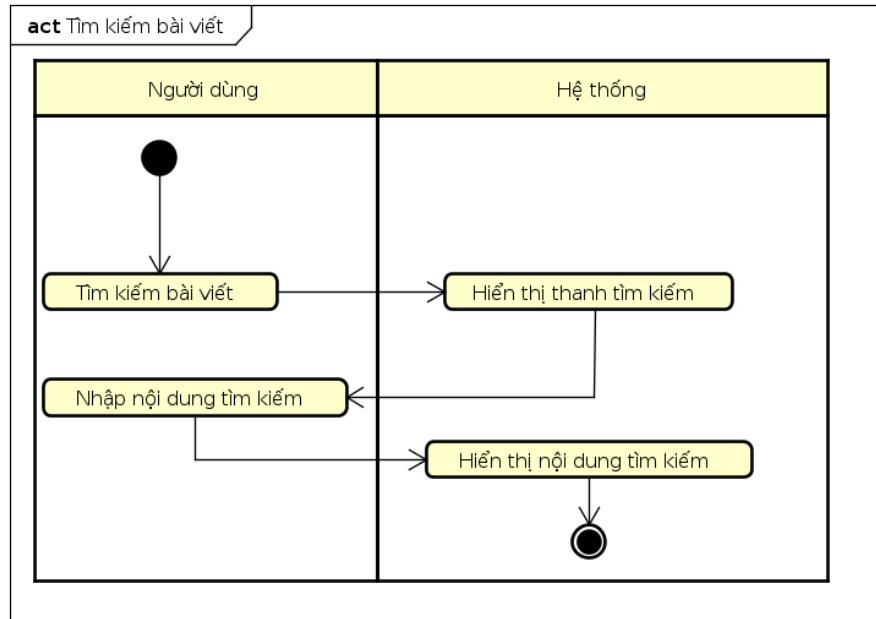
Tên usecase	Tìm kiếm bài viết.		
Mô tả	Kịch bản tìm kiếm bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn tìm kiếm bài viết.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Click chọn chức năng tìm kiếm bài viết.
	2	Hệ thống	Hiển thị thanh tìm kiếm.
	3	Người dùng	Nhập nội dung tìm kiếm và chọn tìm kiếm.
	4	Hệ thống	Hiển thị nội dung tìm kiếm.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.1: Bảng đặc tả usecase thích bài viết.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

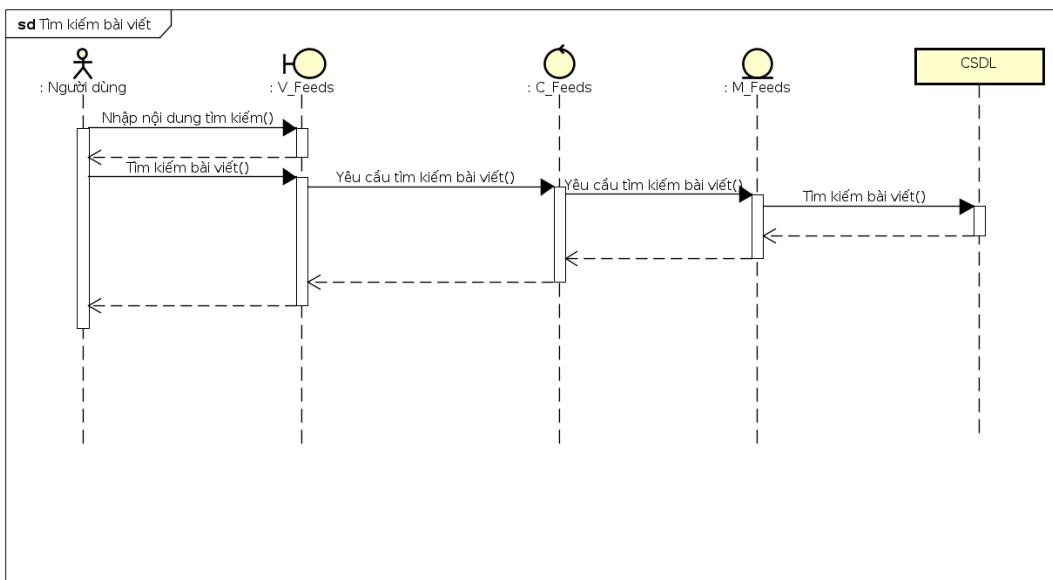
- Biểu đồ hoạt động Tìm kiếm bài viết.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng Tìm kiếm bài viết.



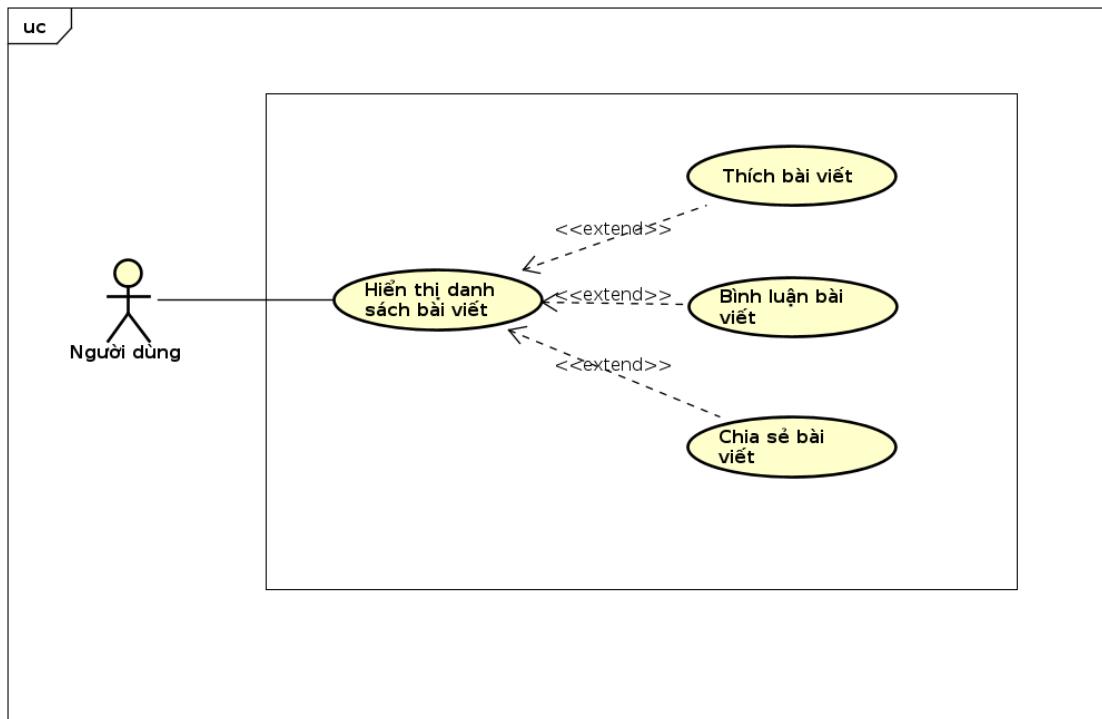
Hình 2.2: Biểu đồ hoạt động Tìm kiếm bài viết

- Biểu đồ tuần tự tìm kiếm bài viết.



Hình 2.3: Biểu đồ tuần tự tìm kiếm bài viết

2.2.2 Biểu đồ use case phân rã hiển thị danh sách bài viết



Hình 2.4: Biểu đồ usecase hiển thị danh sách bài viết

Hình 2.4 là usecase phân rã "hiển thị danh sách bài viết". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Hiển thị danh sách bài viết" được phân rã thành các usecase nhỏ hơn bao gồm: "Thích bài viết", "Bình luận bài viết", "Chia sẻ bài viết", "Hiển thị danh sách bình luận".

a, Biểu đồ usecase thích bài viết.

- Bảng đặc tả usecase thích bài viết.

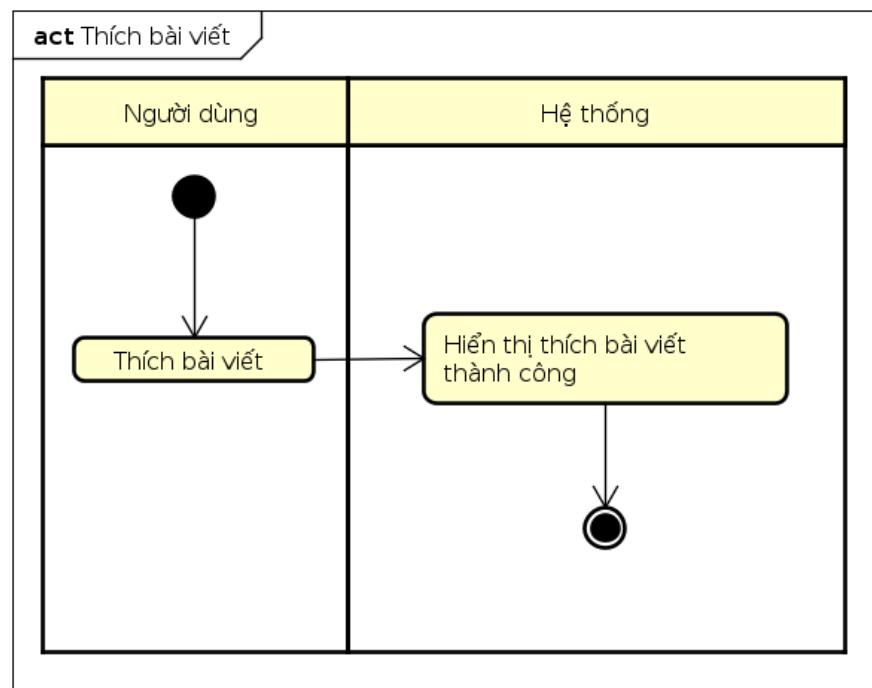
Tên usecase	Thích bài viết.		
Mô tả	Kịch bản thích bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn thích bài viết.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Click chọn chức năng thích bài viết.
	2	Hệ thống	Hiển thị trạng thái đã thích bài viết.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.2: Bảng đặc tả usecase thích bài viết.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

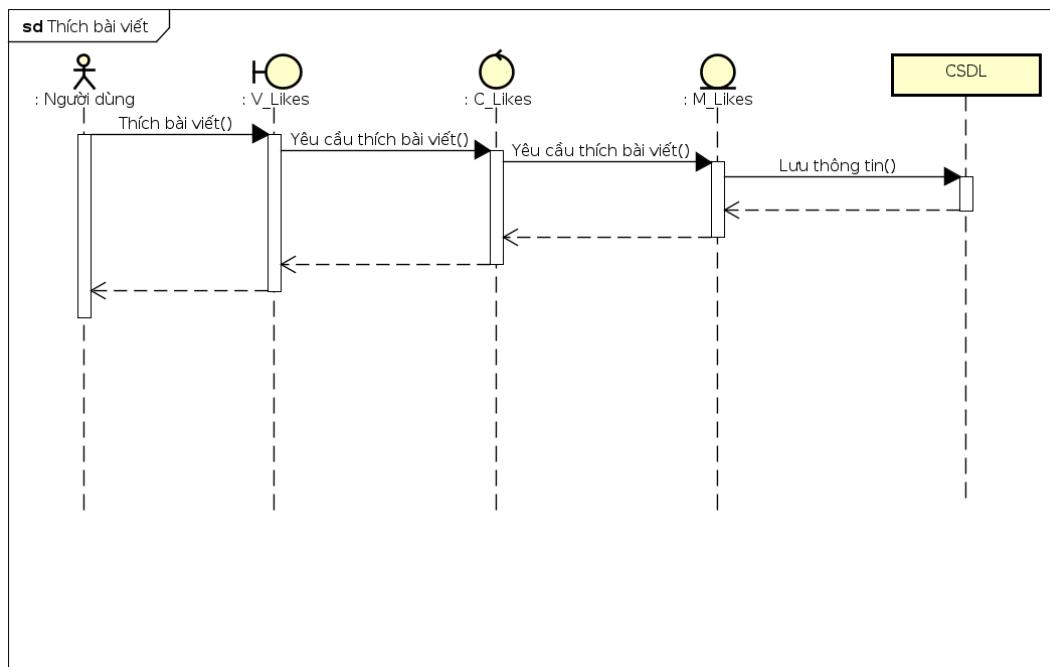
- Biểu đồ hoạt động thích bài viết.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng thích bài viết.



Hình 2.5: Biểu đồ hoạt động thích bài viết

- Biểu đồ tuần tự thích bài viết.



Hình 2.6: Biểu đồ tuần tự thích bài viết

b, Biểu đồ usecase bình luận bài viết.

- Bảng đặc tả usecase hiển thị bình luận bài viết.

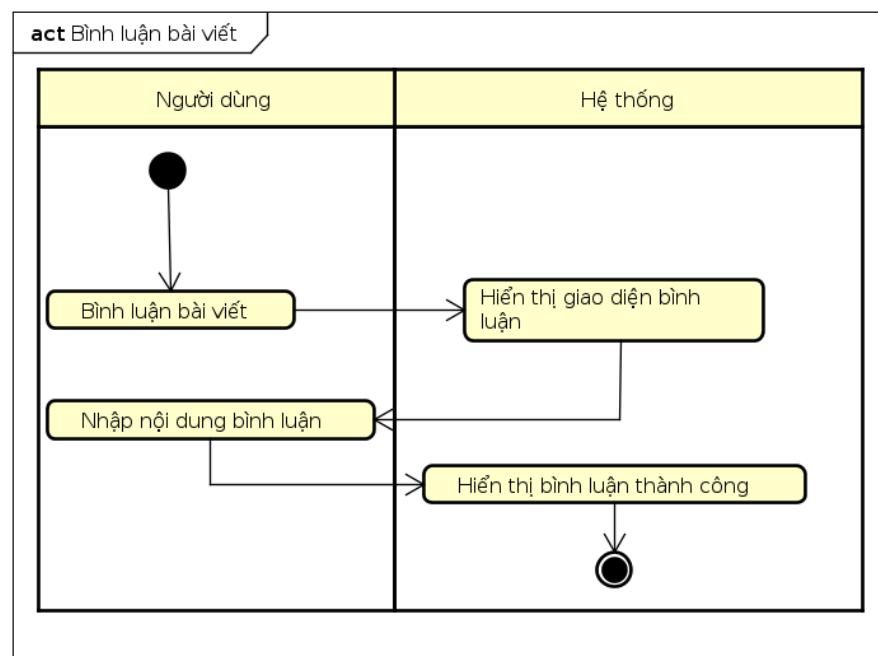
Tên usecase	Bình luận bài viết.		
Mô tả	Kịch bản bình luận bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn bình luận bài viết.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Click chọn chức năng bình luận bài viết.
	2	Người dùng	Nhập bình luận.
	3	Người dùng	Đăng bình luận.
	4	Hệ thống	Hiển thị bình luận thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.3: Bảng đặc tả usecase bình luận bài viết.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

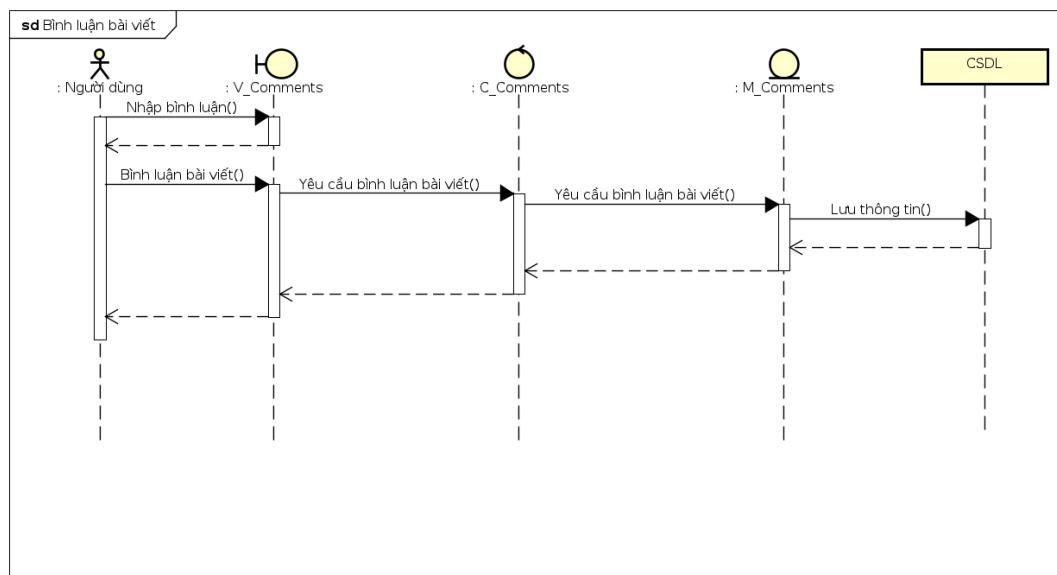
- Biểu đồ hoạt động bình luận bài viết.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng bình luận bài viết.



Hình 2.7: Biểu đồ hoạt động bình luận bài viết

- Biểu đồ tuần tự bình luận bài viết.



Hình 2.8: Biểu đồ tuần tự bình luận bài viết

c, Biểu đồ usecase chia sẻ bài viết.

- Bảng đặc tả usecase chia sẻ bài viết.

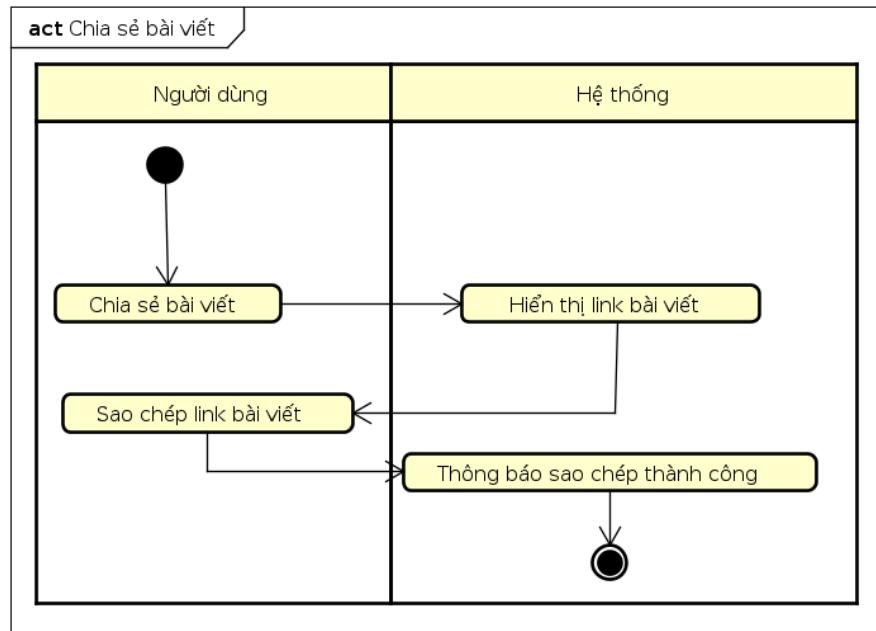
Tên usecase	Chia sẻ bài viết.		
Mô tả	Kịch bản chia sẻ bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn chia sẻ bài viết.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Click chọn chức năng chia sẻ bài viết.
	2	Hệ thống	Hiển thị link bài viết.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.4: Bảng đặc tả usecase chia sẻ bài viết.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

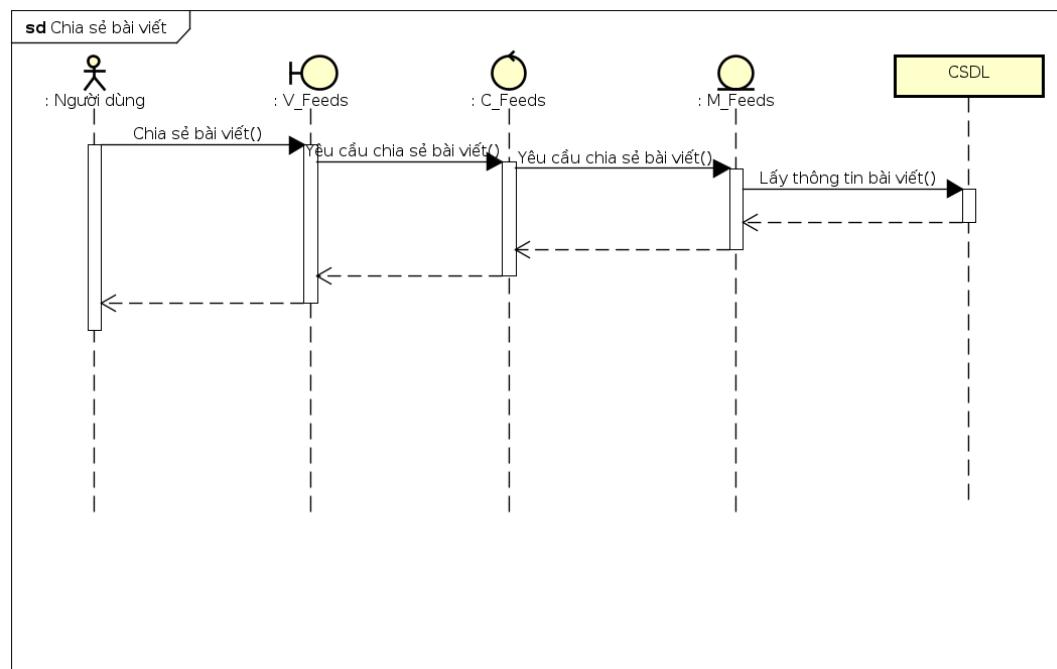
- Biểu đồ hoạt động chia sẻ bài viết

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng chia sẻ bài viết.



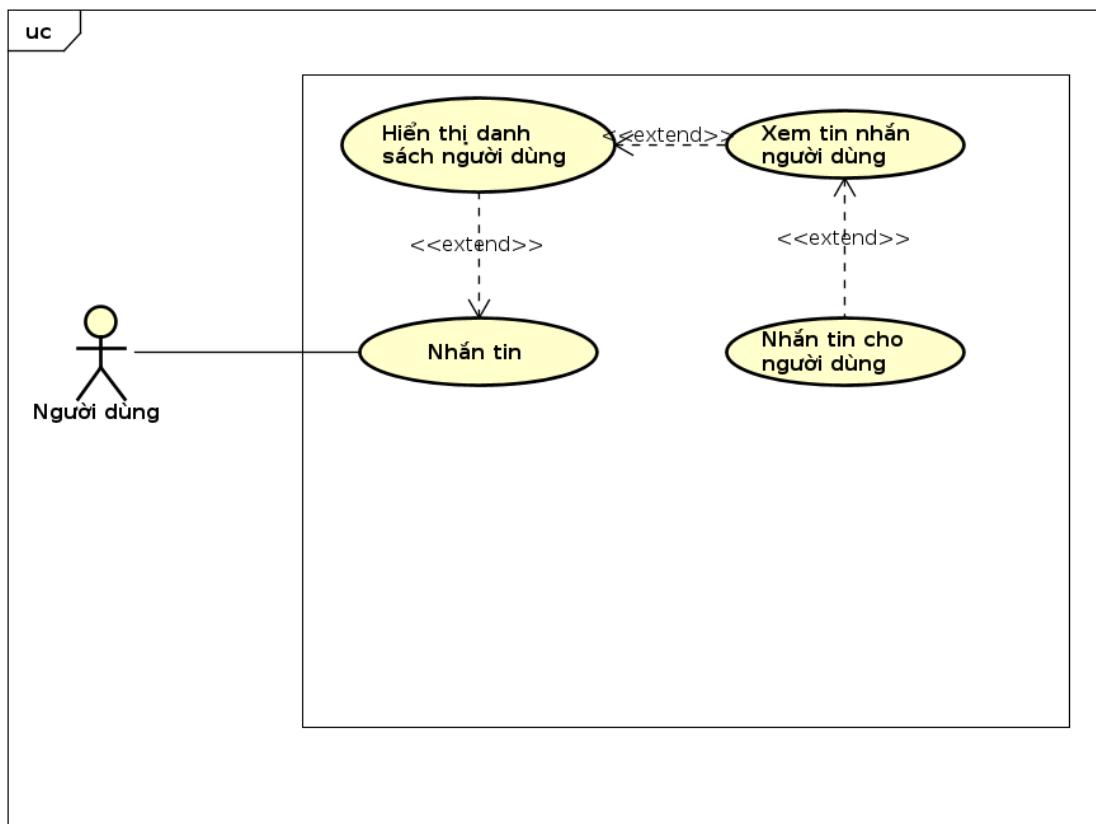
Hình 2.9: Biểu đồ hoạt động chia sẻ bài viết

- Biểu đồ tuần tự chia sẻ bài viết.



Hình 2.10: Biểu đồ tuần tự chia sẻ bài viết

2.2.3 Biểu đồ use case phân rã nhắn tin

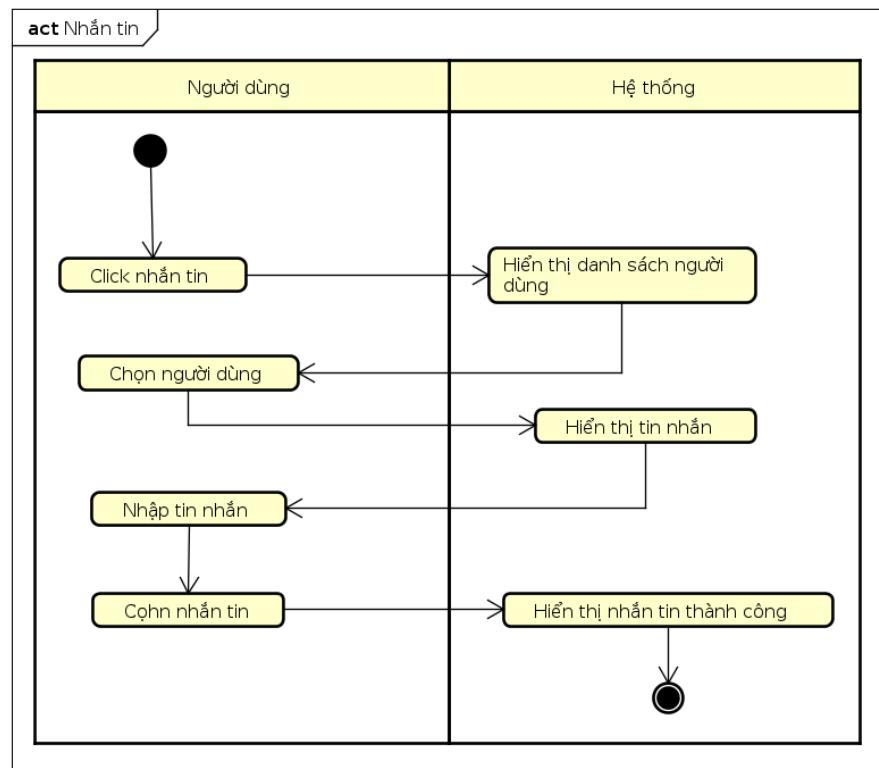


Hình 2.11: Biểu đồ usecase nhắn tin

Hình 2.11 là usecase phân rã "Nhắn tin". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Nhắn tin" được phân rã thành "Hiển thị danh sách người dùng". Usecase "Hiển thị danh sách người dùng" được phân rã thành "Xem tin nhắn người dùng". Usecase "Xem tin nhắn người dùng" được phân rã thành "Nhắn tin cho người dùng".

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng nhắn tin.



Hình 2.12: Biểu đồ hoạt động nhắn tin

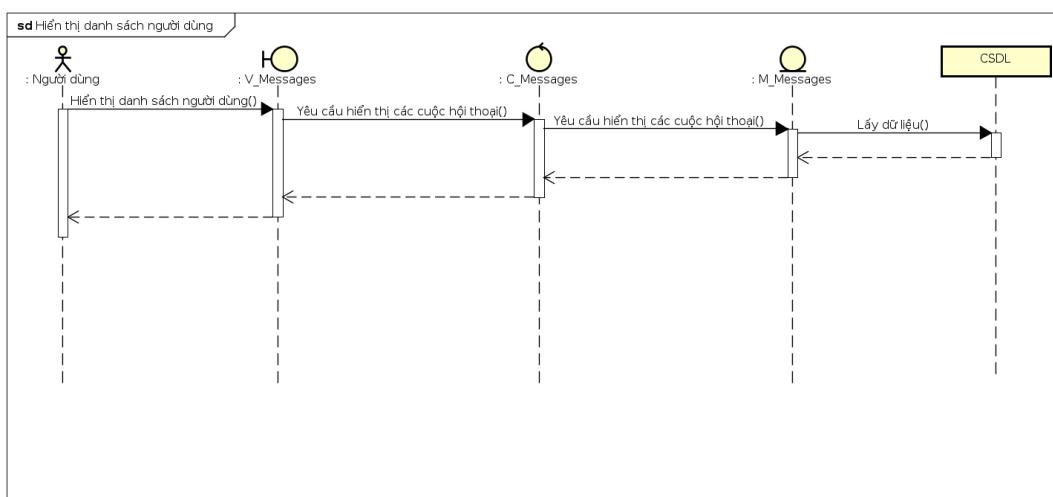
a, Biểu đồ usecase hiển thị danh sách người dùng.

- Bảng đặc tả usecase hiển thị danh sách người dùng.

Tên usecase	Hiển thị danh sách người dùng.		
Mô tả	Kịch bản hiển thị danh sách người dùng.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn nhắm tin.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Click chọn chức năng nhắm tin.
	2	Hệ thống	Hiển thị danh sách người dùng.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.5: Bảng đặc tả usecase hiển thị danh sách người dùng

- Biểu đồ tuần tự hiển thị danh sách người dùng.



Hình 2.13: Biểu đồ tuần tự hiển thị danh sách người dùng

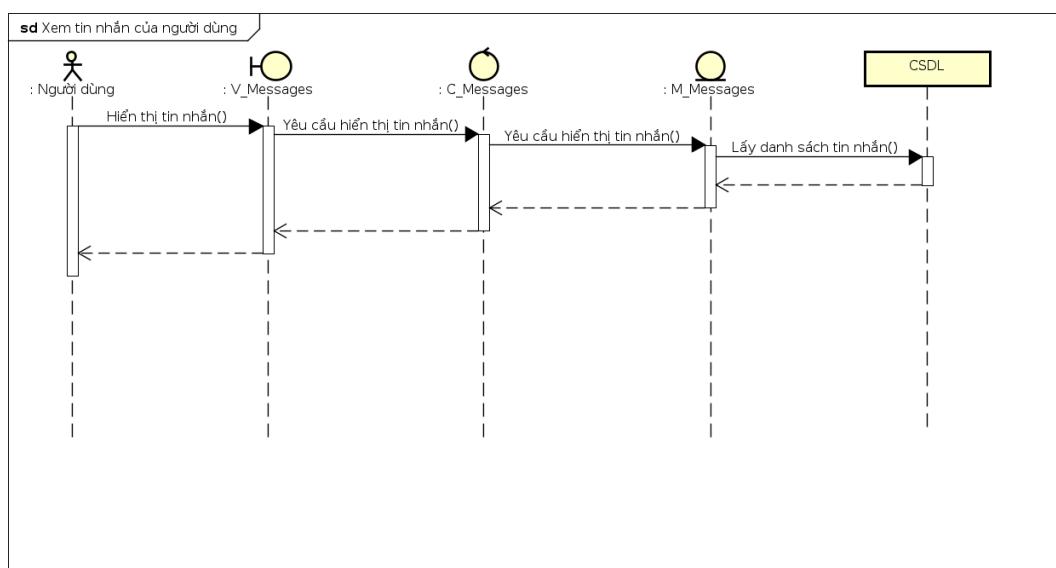
b, Biểu đồ usecase xem tin nhắn người dùng.

- Bảng đặc tả usecase xem tin nhắn người dùng.

Tên usecase	Xem tin nhắn người dùng.		
Mô tả	Kịch bản xem tin nhắn người dùng.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn người dùng trong danh sách người dùng.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Click chọn người dùng trong danh sách người dùng.
	2	Hệ thống	Hiển thị tin nhắn.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.6: Bảng đặc tả usecase xem tin nhắn người dùng.

- Biểu đồ tuần tự xem tin nhắn người dùng.



Hình 2.14: Biểu đồ tuần tự xem tin nhắn người dùng

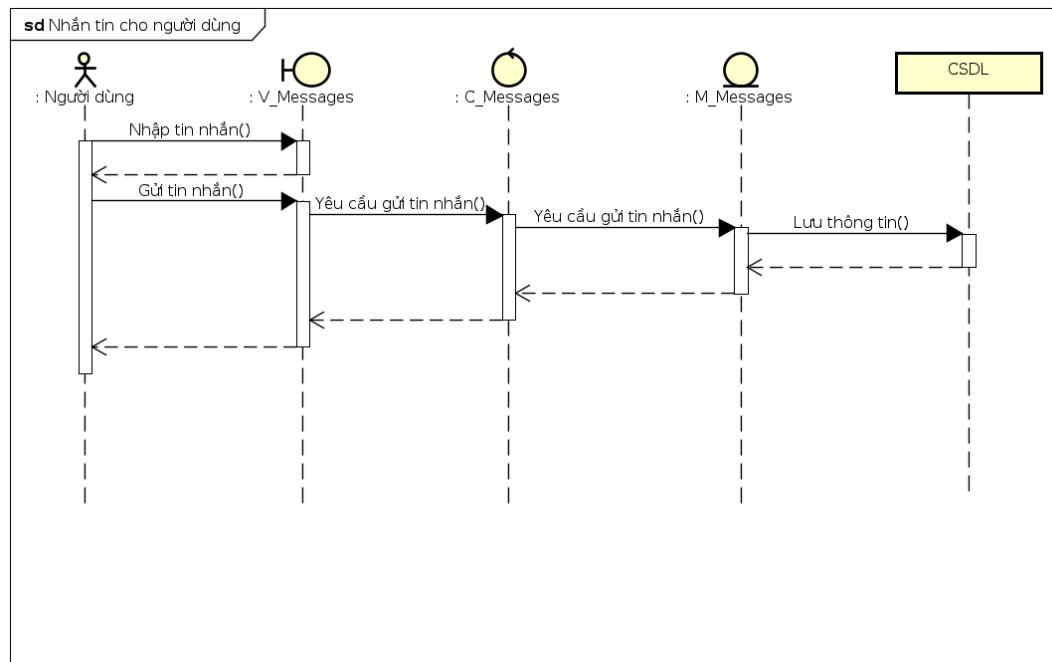
c, Biểu đồ usecase nhắn tin cho người dùng.

- Bảng đặc tả usecase nhắn tin cho người dùng.

Tên usecase	Nhắn tin cho người dùng.		
Mô tả	Kịch bản nhắn tin cho người dùng.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng đang mở cuộc hội thoại nhắn tin.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Nhập tin nhắn.
	2	Người dùng	Gửi tin nhắn.
	2	Hệ thống	Hiển thị tin nhắn gửi thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

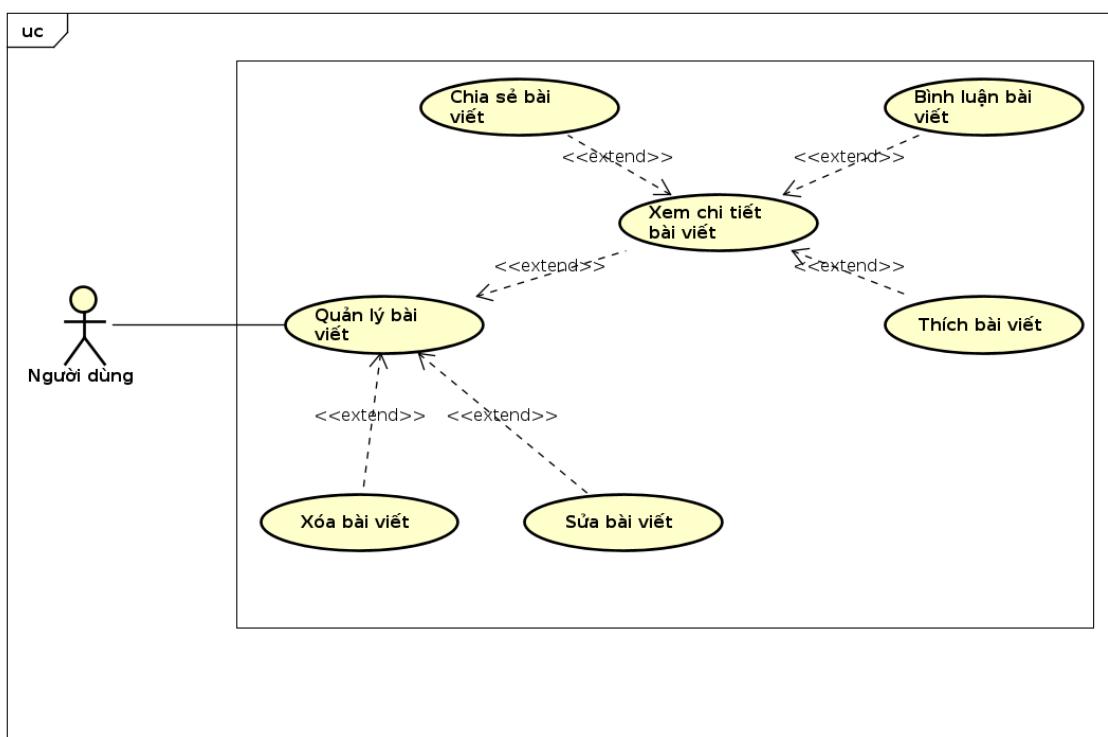
Bảng 2.7: Bảng đặc tả usecase nhắn tin cho người dùng.

- Biểu đồ tuần tự nhẫn tin cho người dùng.



Hình 2.15: Biểu đồ tuần tự nhẫn tin cho người dùng

2.2.4 Biểu đồ use case phân rã quản lý bài viết



Hình 2.16: Biểu đồ usecase quản lý bài viết

Hình 2.16 là usecase phân rã "Quản lý bài viết". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Quản lý bài viết" được phân rã thành "Xóa bài viết", "Sửa bài viết", "Xem chi tiết bài viết". Usecase "Xem chi tiết bài viết" được phân rã thành "Chia sẻ bài viết", "Bình luận về bài viết", "Thích bài viết".

a, Biểu đồ usecase xóa bài viết.

- Bảng đặc tả usecase xóa bài viết.

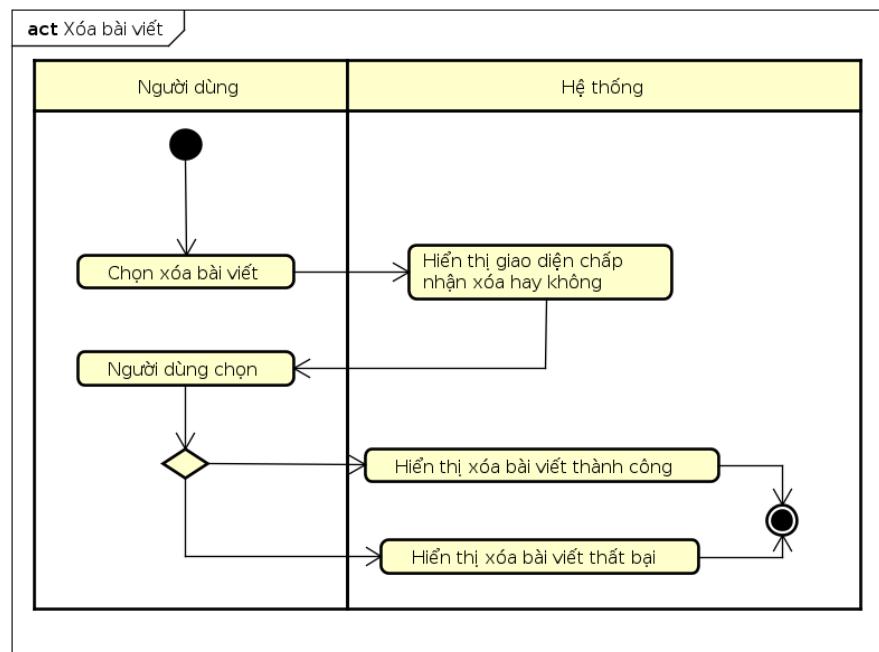
Tên usecase	Xóa bài viết.		
Mô tả	Kịch bản xóa bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng đang xem danh sách bài viết.		
Điều kiện tiên quyết	Người dùng phải là người tạo bài viết.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	xóa bài viết.
	2	Hệ thống	Hiển thị người dùng có chắc chắn muốn xóa bài viết không.
	3	Người dùng	Chọn Có.
	4	Hệ thống	Hiển thị xóa bài viết thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
	3a	Người dùng	Chọn không.
	4a	Hệ thống	Hiển thị xóa bài viết thất bại.
Hậu điều kiện	Không có.		

Bảng 2.8: Bảng đặc tả usecase xóa bài viết

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

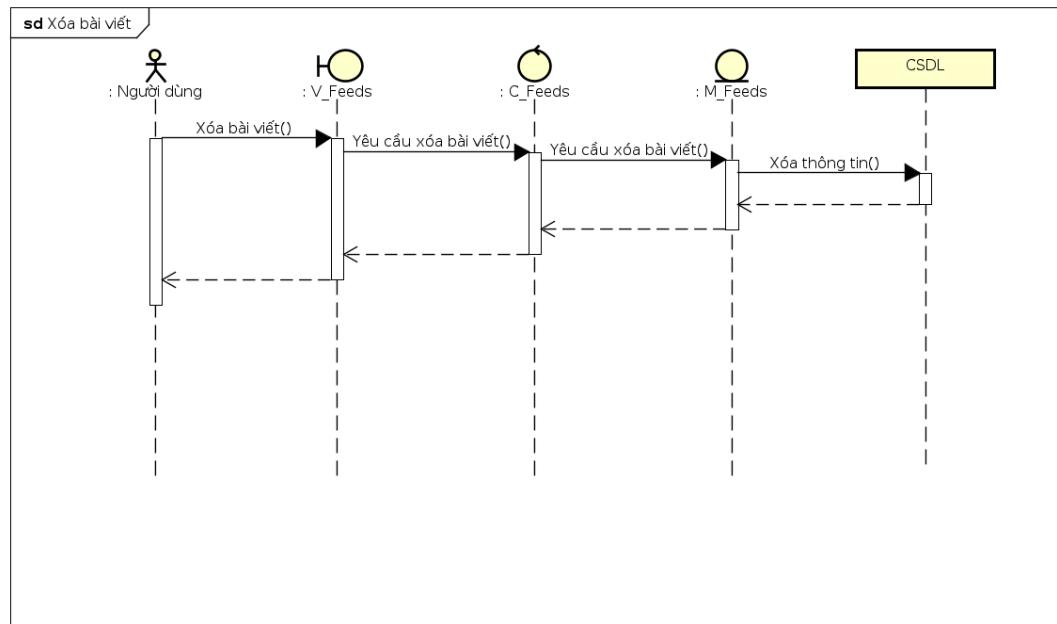
- Biểu đồ hoạt động xóa bài viết.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng xóa bài viết.



Hình 2.17: Biểu đồ hoạt động xóa bài viết

- Biểu đồ tuần tự xóa bài viết



Hình 2.18: Biểu đồ tuần tự xóa bài viết

b, Biểu đồ usecase sửa bài viết.

- Bảng đặc tả usecase sửa bài viết.

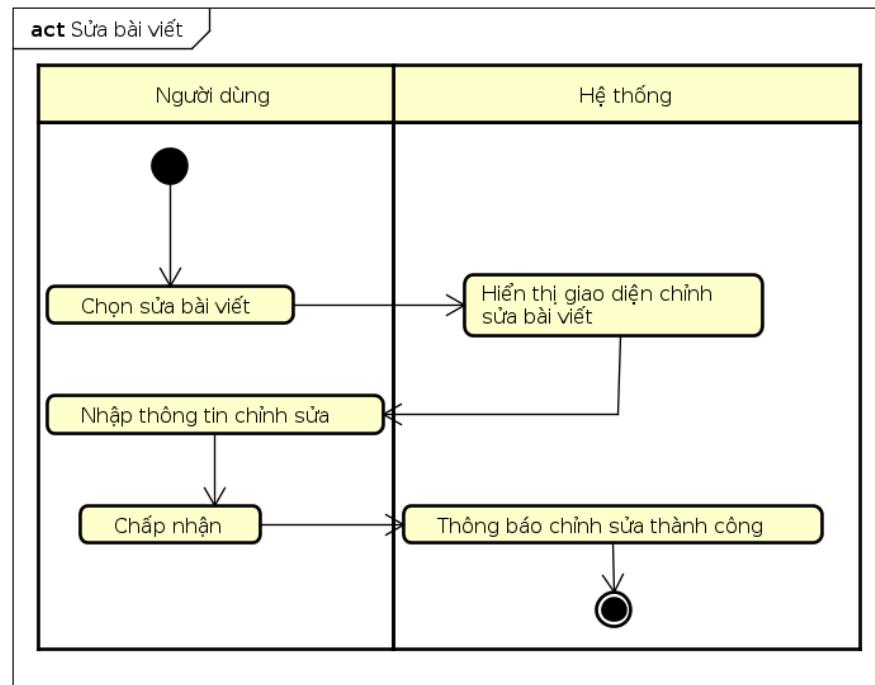
Tên usecase	Sửa bài viết.		
Mô tả	Kịch bản sửa bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng đang xem danh sách bài viết.		
Điều kiện tiên quyết	Người dùng phải là người tạo bài viết.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	sửa bài viết.
	2	Hệ thống	Hiển thị giao diện chỉnh sửa bài viết.
	3	Người dùng	Chỉnh sửa và chọn lưu bài viết.
	4	Hệ thống	Hiển thị sửa bài viết thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.9: Bảng đặc tả usecase sửa bài viết.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

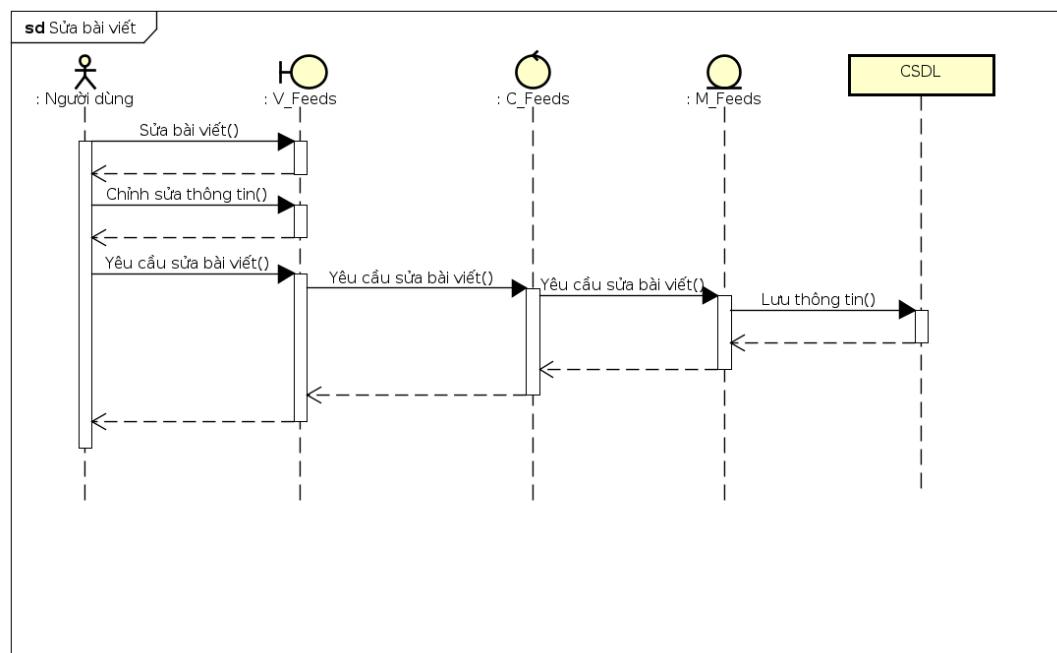
- Biểu đồ hoạt động sửa bài viết.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng sửa bài viết.



Hình 2.19: Biểu đồ hoạt động sửa bài viết

- Biểu đồ tuần tự sửa bài viết.



Hình 2.20: Biểu đồ tuần tự sửa bài viết

c, Biểu đồ usecase xem chi tiết bài viết.

- Bảng đặc tả usecase xem chi tiết bài viết.

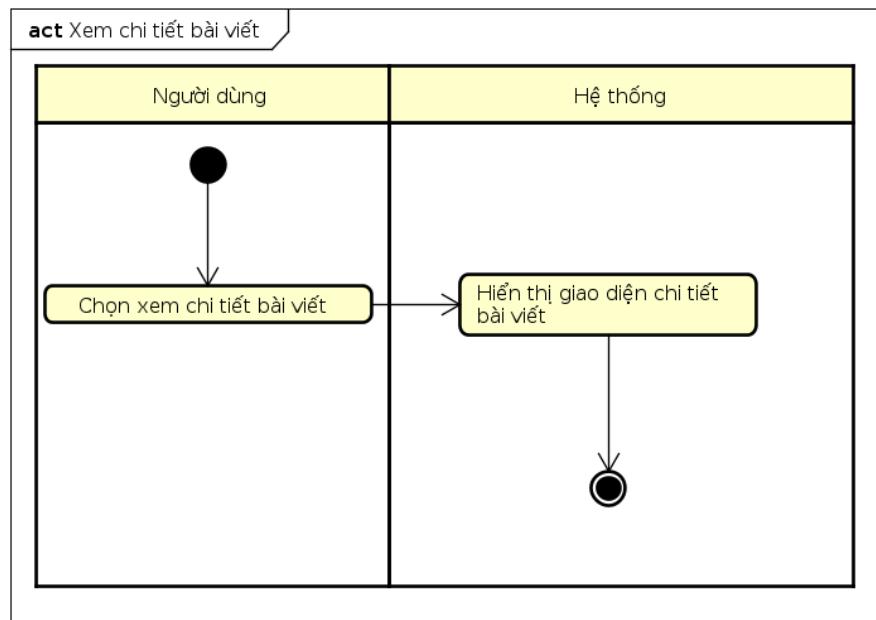
Tên usecase	Xem chi tiết bài viết.		
Mô tả	Kịch bản xem chi tiết bài viết.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng đang xem danh sách bài viết.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn xem chi tiết bài viết.
	2	Hệ thống	Hiển thị giao diện chi tiết bài viết.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.10: Bảng đặc tả usecase xem chi tiết bài viết.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

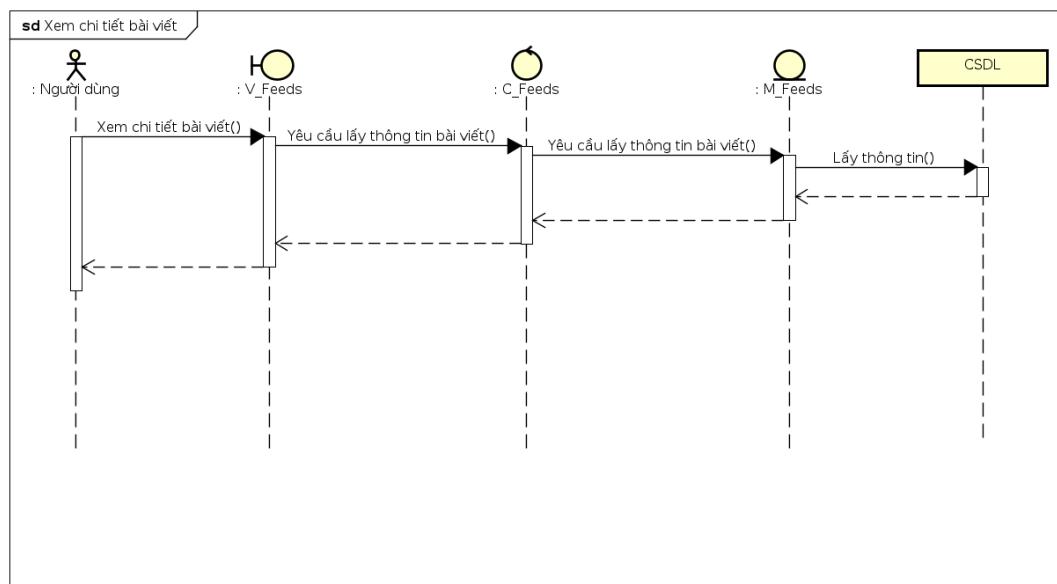
- Biểu đồ hoạt động xem chi tiết bài viết.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng xem chi tiết bài viết.



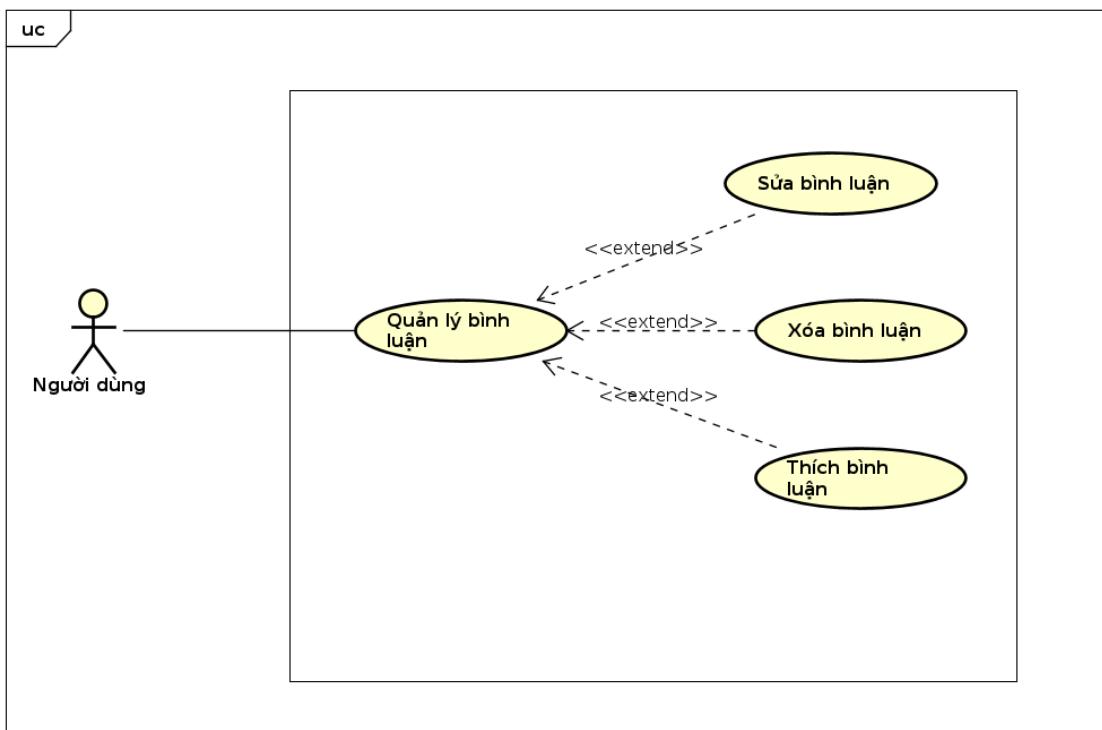
Hình 2.21: Biểu đồ hoạt động xem chi tiết bài viết

- Biểu đồ tuần tự xem chi tiết bài viết.



Hình 2.22: Biểu đồ tuần tự xem chi tiết bài viết

2.2.5 Biểu đồ use case phân rã quản lý bình luận



Hình 2.23: Biểu đồ usecase quản lý bình luận

Hình 2.23 là usecase phân rã "Quản lý bình luận". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Quản lý bình luận" được phân rã thành "Xóa bình luận", "Sửa bình luận".

a, Biểu đồ usecase sửa bình luận.

- Bảng đặc tả usecase sửa bình luận.

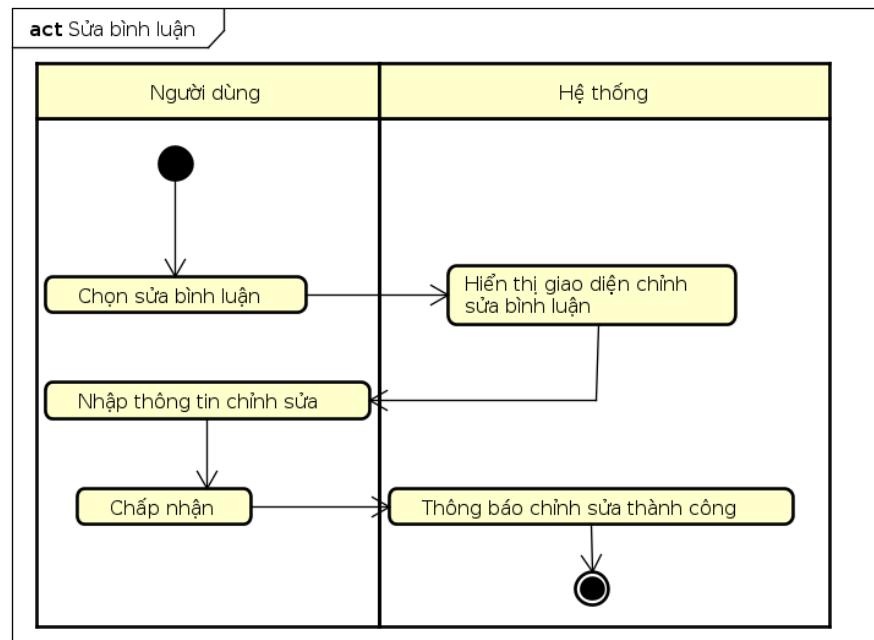
Tên usecase	Sửa bình luận.
Mô tả	Kịch bản sửa bình luận.
Tác nhân sử dụng	Người dùng.
Sự kiện kích hoạt	Người dùng đang xem danh sách bình luận.
Điều kiện tiên quyết	Bình luận phải là bình luận của người dùng.

	STT	Thực hiện bởi	Hành động
Luồng sự kiện chính	1	Người dùng	Chọn sửa bình luận.
	2	Hệ thống	Hiển thị giao diện sửa bình luận.
	3	Người dùng	Chỉnh sửa bình luận.
	4	Hệ thống	Hiển thị sửa bình luận thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.11: Bảng đặc tả usecase sửa bình luận

- Biểu đồ hoạt động sửa bình luận.

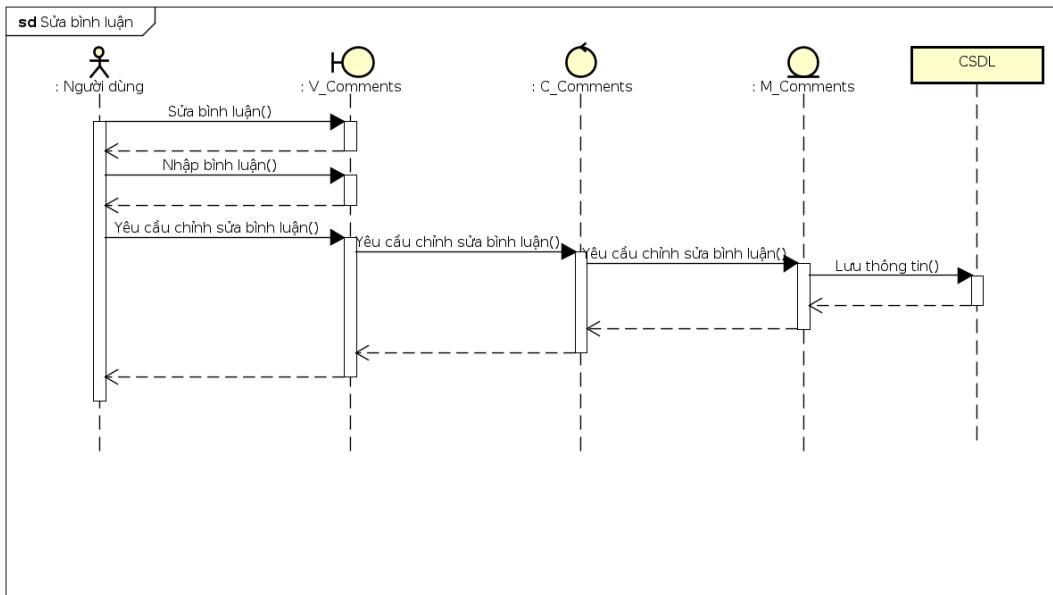
Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng sửa bình luận.



Hình 2.24: Biểu đồ hoạt động sửa bình luận

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

- Biểu đồ tuần tự sửa bình luận.



Hình 2.25: Biểu đồ tuần tự sửa bình luận

b, Biểu đồ usecase xóa bình luận.

- Bảng đặc tả usecase xóa bình luận.

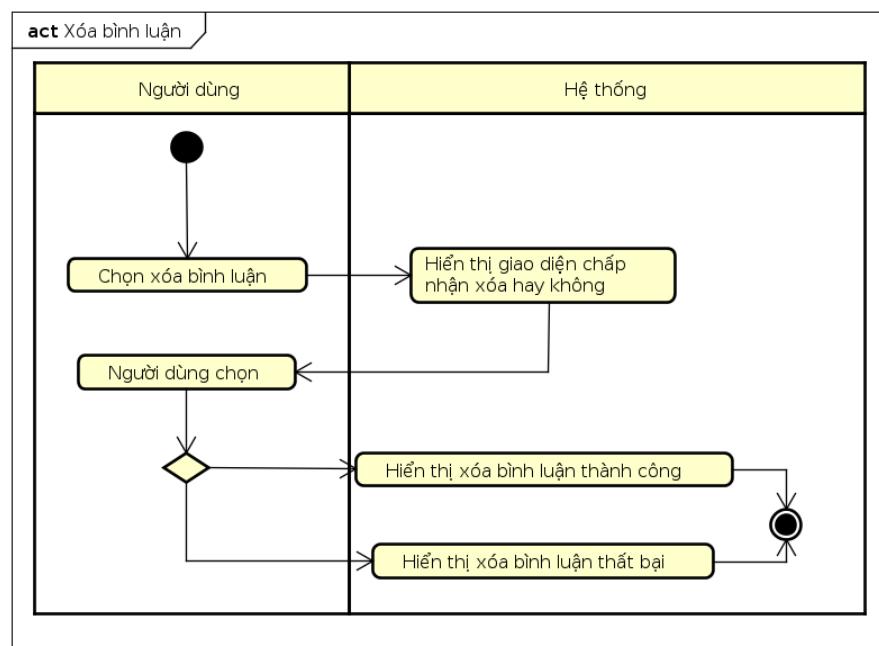
Tên usecase	Xóa bình luận.		
Mô tả	Kịch bản xóa bình luận.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng đang xem danh sách bình luận.		
Điều kiện tiên quyết	Bình luận phải là bình luận của người dùng.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn xóa bình luận.
	2	Hệ thống	Hiển thị thông báo người dùng muốn xóa bình luận không.
	3	Người dùng	Chọn có.
	4	Hệ thống	Hiển thị xóa bình luận thành công.

	STT	Thực hiện bởi	Hành động
Luồng sự kiện phụ	3a	Người dùng	Chọn không.
	4a	Hệ thống	Hiển thị xóa bình luận thất bại.
Hậu điều kiện	Không có.		

Bảng 2.12: Bảng đặc tả usecase xóa bình luận.

- Biểu đồ hoạt động xóa bình luận.

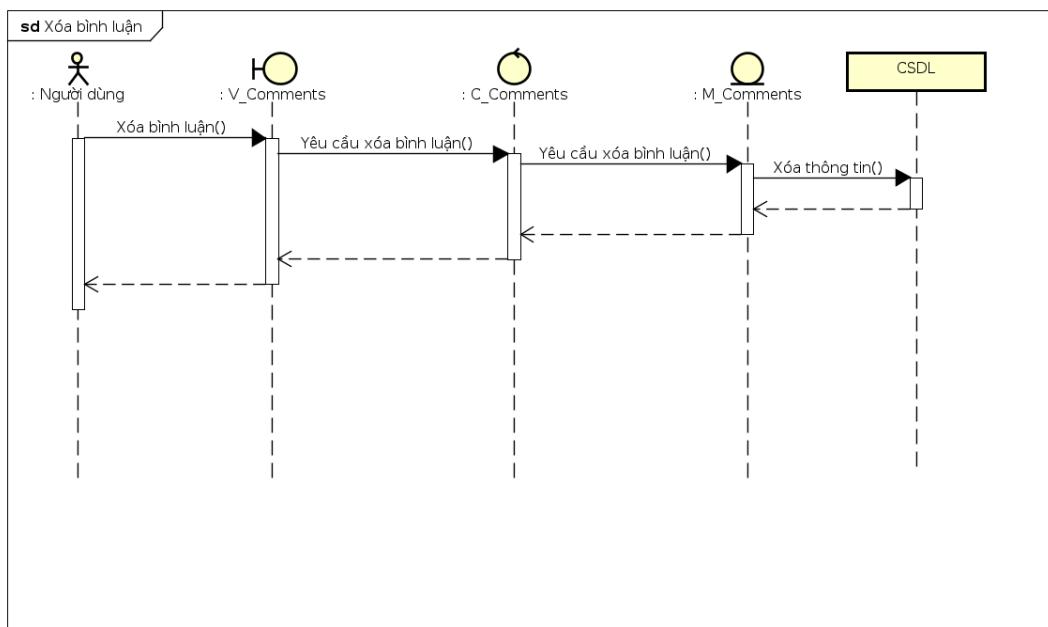
Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng xóa bình luận.



Hình 2.26: Biểu đồ hoạt động xóa bình luận

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

- Biểu đồ tuần tự xóa bình luận.



Hình 2.27: Biểu đồ tuần tự xóa bình luận

c, Biểu đồ usecase thích bình luận.

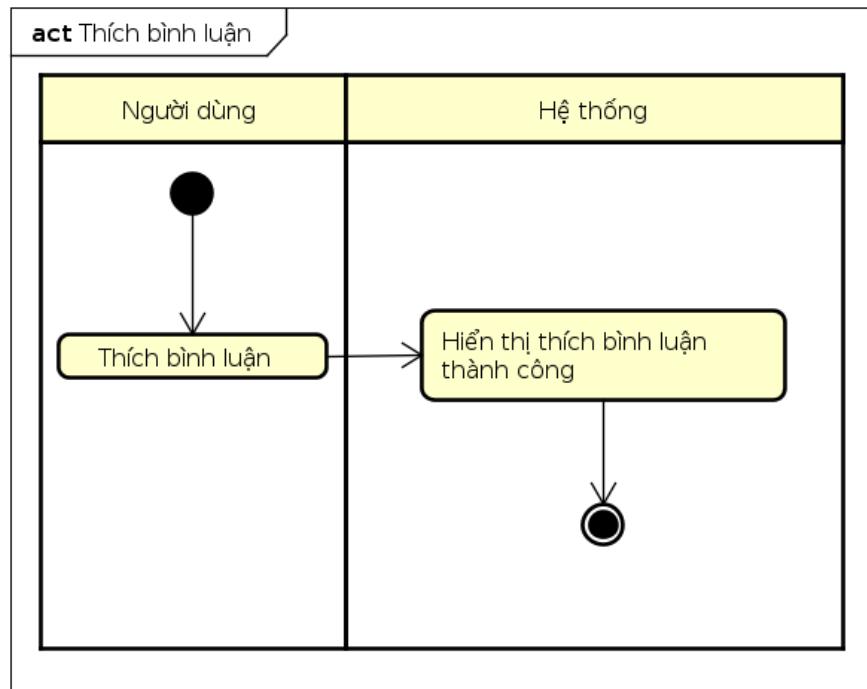
- Bảng đặc tả usecase thích bình luận.

Tên usecase	Thích bình luận.		
Mô tả	Kịch bản thích bình luận.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng đang xem danh sách bình luận.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn thích bình luận.
	2	Hệ thống	Hiển thị thích bình luận thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.13: Bảng đặc tả usecase thích bình luận.

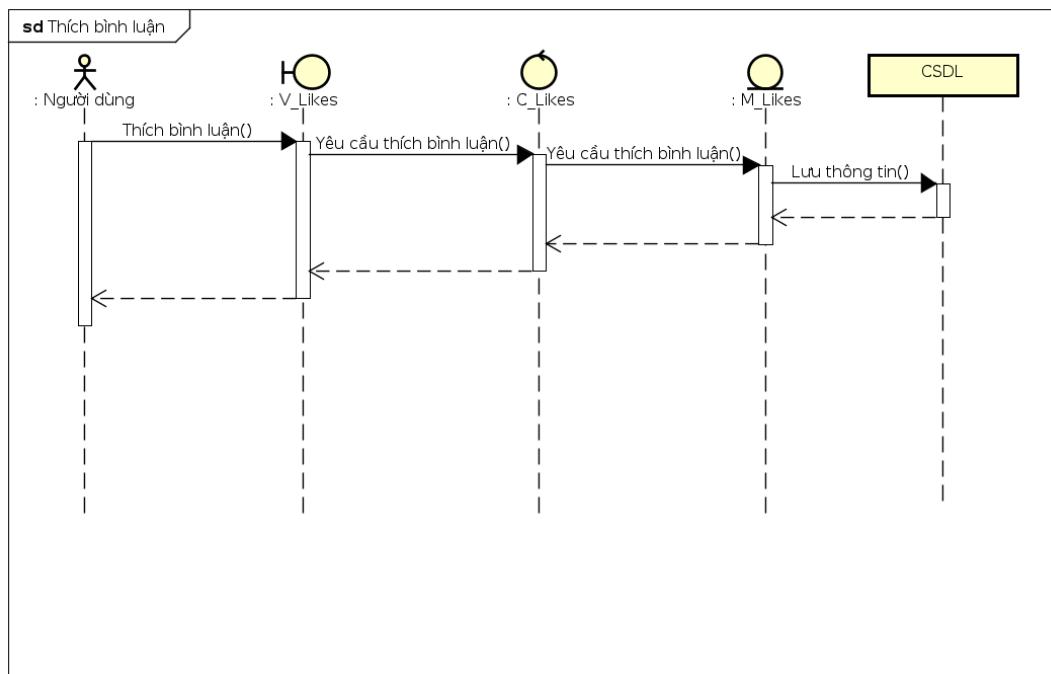
- Biểu đồ hoạt động thích bình luận.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng thích bình luận.



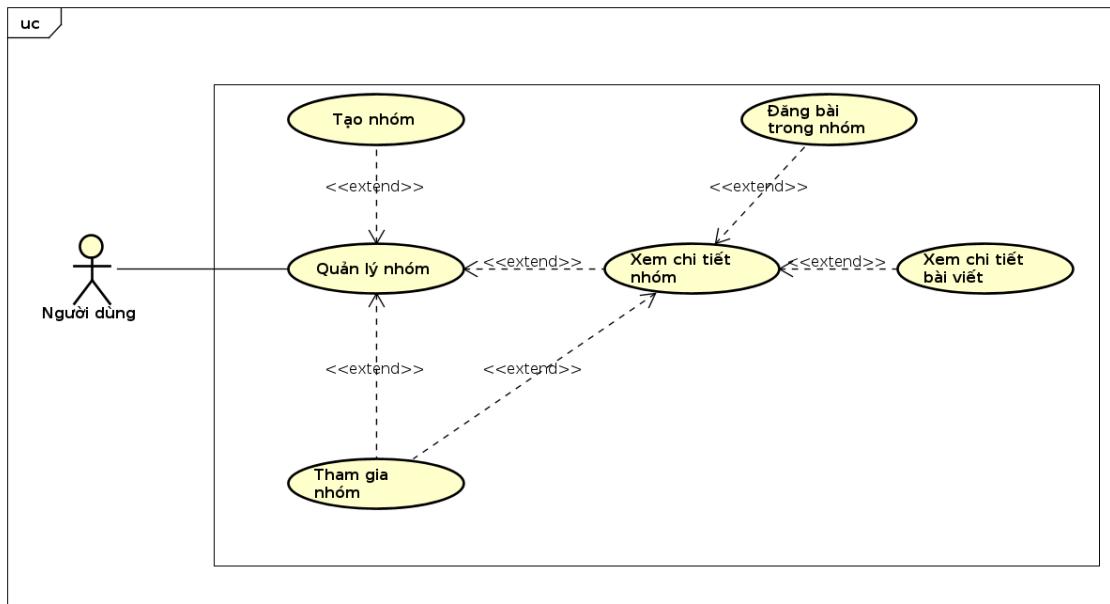
Hình 2.28: Biểu đồ hoạt động thích bình luận

- Biểu đồ tuần tự thích bình luận.



Hình 2.29: Biểu đồ tuần tự thích bình luận

2.2.6 Biểu đồ use case phân rã quản lý nhóm



Hình 2.30: Biểu đồ usecase quản lý nhóm

Hình 2.30 là usecase phân rã "Quản lý nhóm". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Quản lý nhóm" được phân rã thành "Tạo nhóm", "Tham gia nhóm", "Xem chi tiết nhóm". Usecase "Xem chi tiết nhóm" được phân rã thành "Đăng bài trong nhóm" và "Xem chi tiết bài viết".

a, Biểu đồ usecase tạo nhóm.

- Bảng đặc tả usecase tạo nhóm.

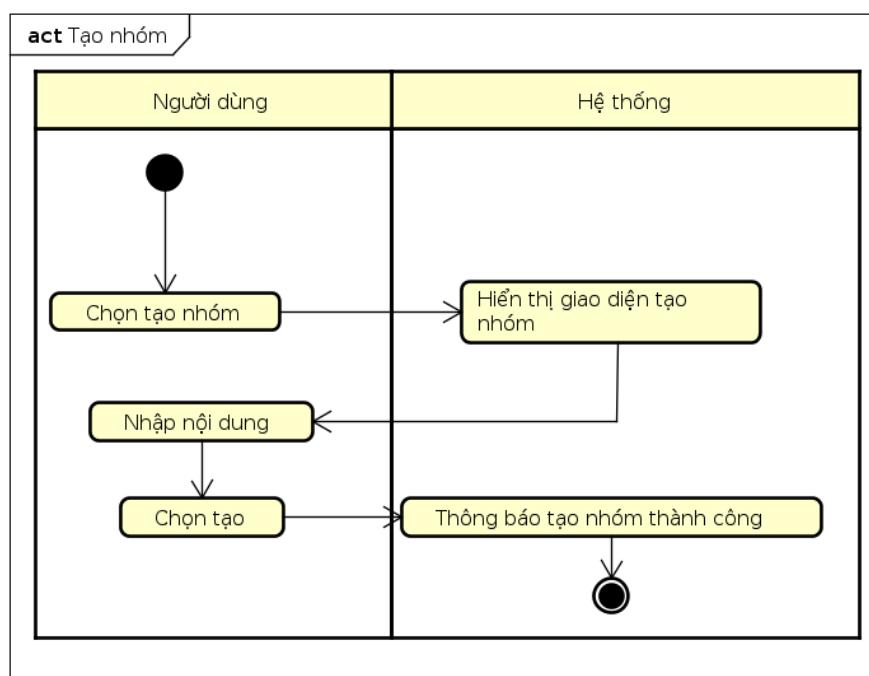
Tên usecase	Tạo nhóm.
Mô tả	Kịch bản tạo nhóm.
Tác nhân sử dụng	Người dùng.
Sự kiện kích hoạt	Người dùng chọn tạo nhóm.
Điều kiện tiên quyết	Không có.

	STT	Thực hiện bởi	Hành động
Luồng sự kiện chính	1	Người dùng	Chọn tạo nhóm.
	2	Hệ thống	Hiển thị giao diện tạo nhóm.
	3	Người dùng	Nhập thông tin và chọn tạo.
	4	Hệ thống	Hiển thị tạo nhóm thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	
Hậu điều kiện	Không có.		

Bảng 2.14: Bảng đặc tả usecase tạo nhóm.

- Biểu đồ hoạt động tạo nhóm.

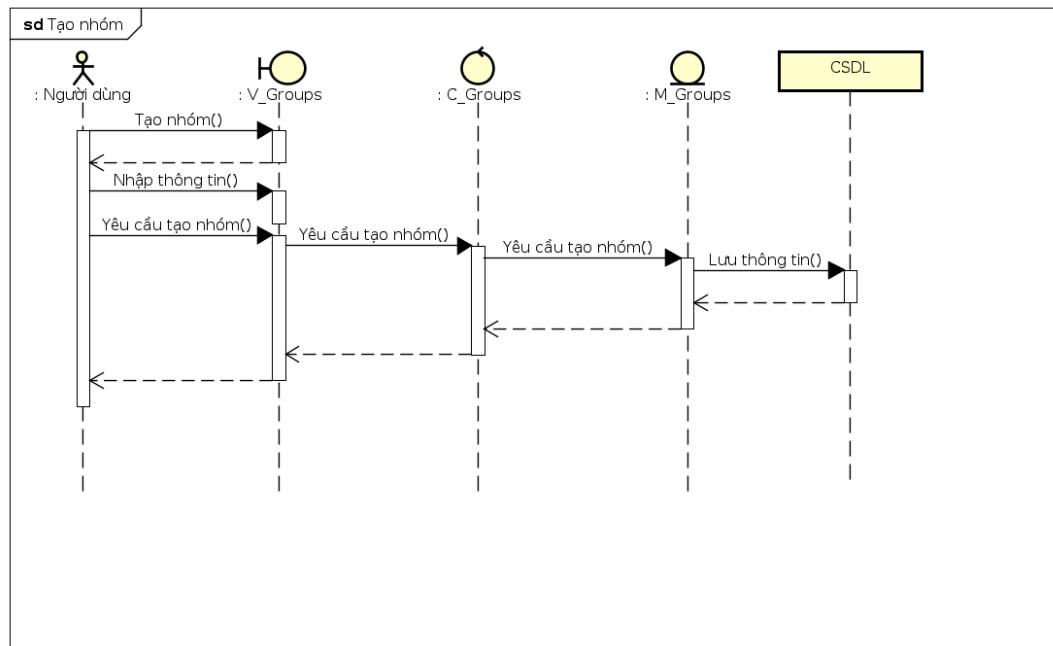
Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng tạo nhóm.



Hình 2.31: Biểu đồ hoạt động tạo nhóm

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

- Biểu đồ tuần tự tạo nhóm.



Hình 2.32: Biểu đồ tuần tự tạo nhóm

b, Biểu đồ usecase tham gia nhóm.

- Bảng đặc tả usecase tham gia nhóm.

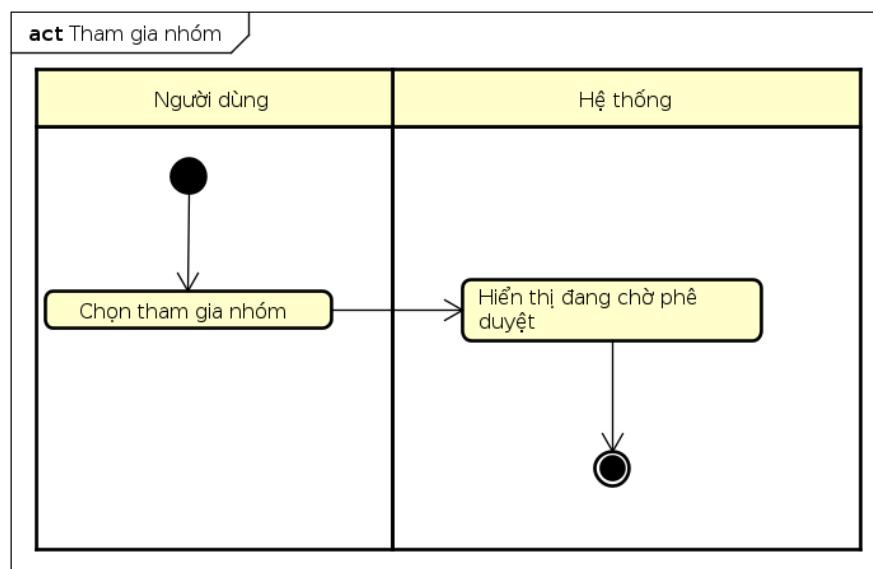
Tên usecase	Tham gia nhóm.		
Mô tả	Kịch bản tham gia nhóm.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng xem danh sách nhóm hoặc xem chi tiết nhóm.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn tham gia nhóm.
	2	Hệ thống	Hiển thị thông báo tham gia nhóm thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động

Hậu điều kiện	Không có.
---------------	-----------

Bảng 2.15: Bảng đặc tả usecase tham gia nhóm.

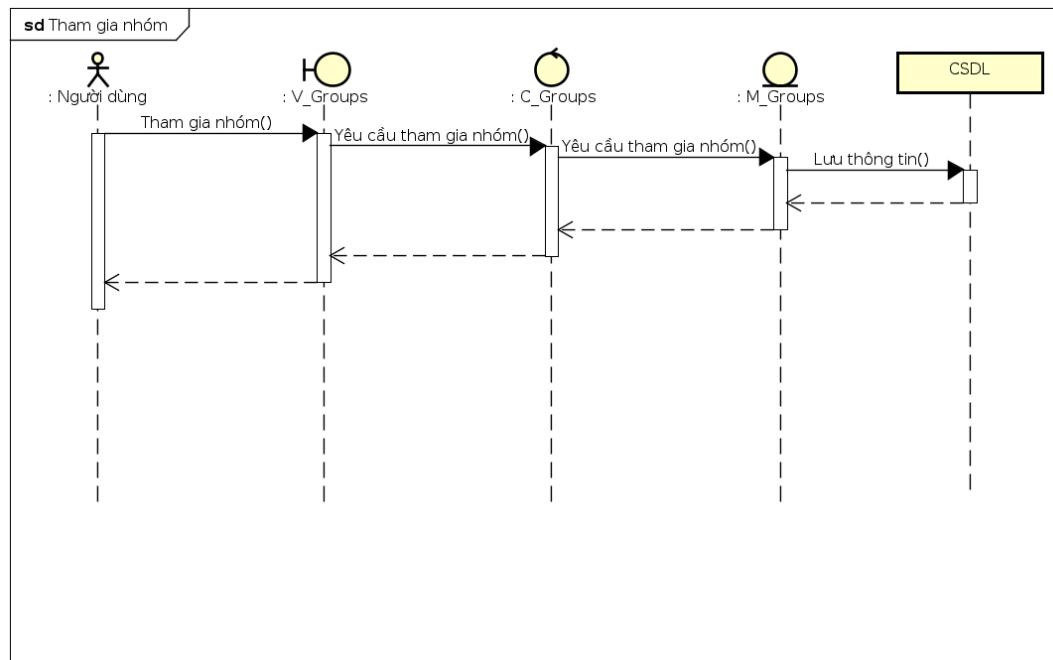
- Biểu đồ hoạt động tham gia nhóm.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng tham gia nhóm.



Hình 2.33: Biểu đồ hoạt động tham gia nhóm

- Biểu đồ tuần tự tham gia nhóm.



Hình 2.34: Biểu đồ tuần tự tham gia nhóm

c, Biểu đồ usecase xem chi tiết nhóm.

- Bảng đặc tả usecase xem chi tiết nhóm.

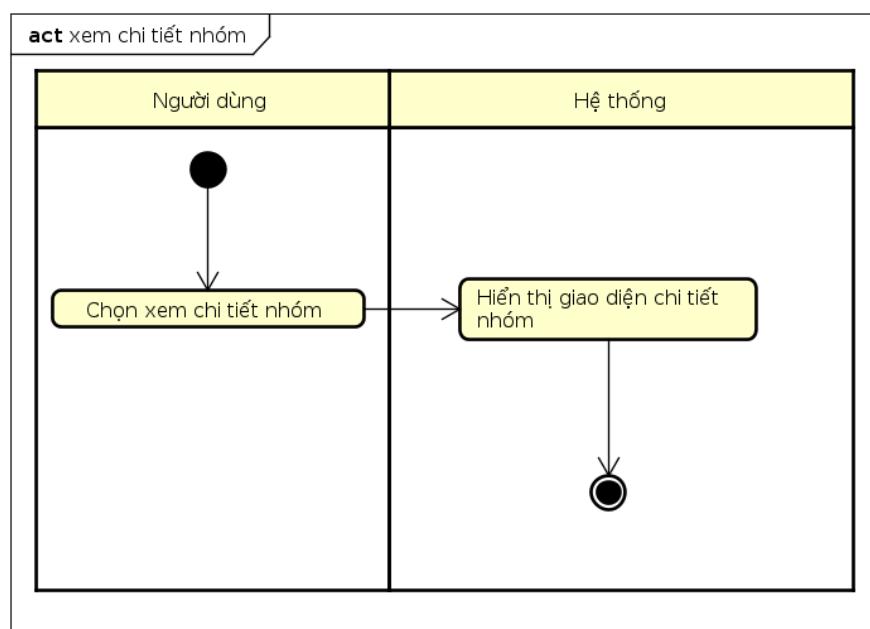
Tên usecase	Xem chi tiết nhóm.		
Mô tả	Kịch bản xem chi tiết nhóm.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng xem danh sách nhóm.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn xem chi tiết nhóm.
	2	Hệ thống	Hiển thị thông tin nhóm và danh sách các bài đăng trong nhóm.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động

Hậu điều kiện	Không có.
---------------	-----------

Bảng 2.16: Bảng đặc tả usecase xem chi tiết nhóm.

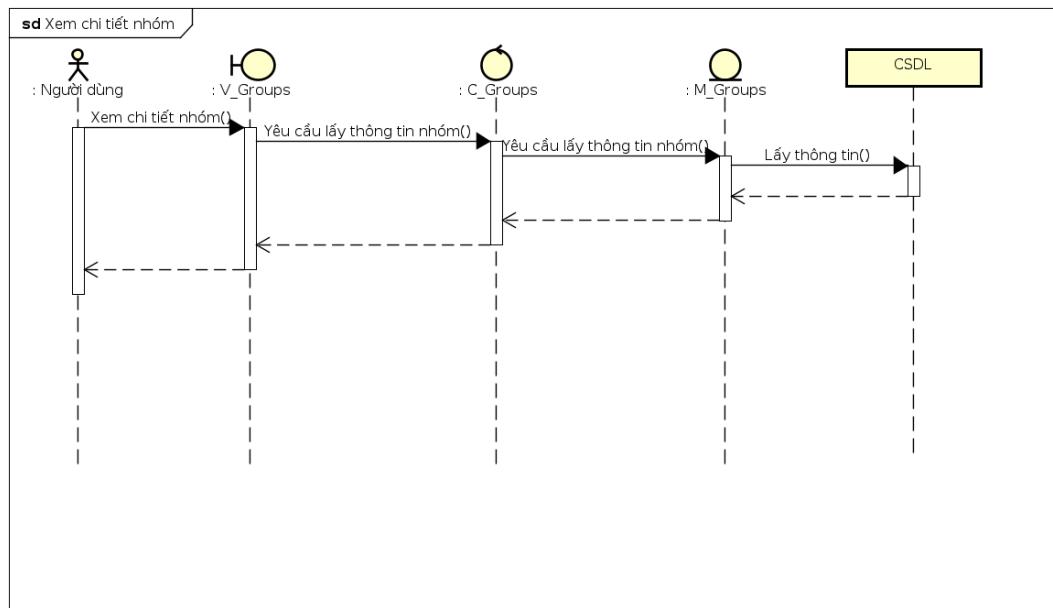
- Biểu đồ hoạt động xem chi tiết nhóm.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng xem chi tiết nhóm.



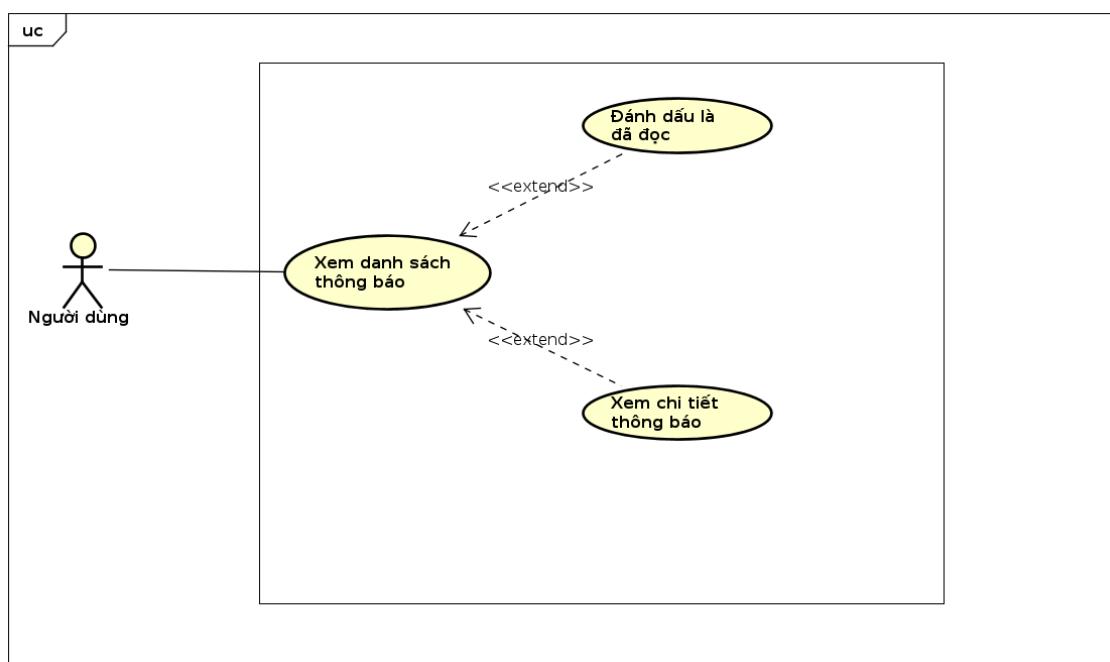
Hình 2.35: Biểu đồ hoạt động xem chi tiết nhóm

- Biểu đồ tuần tự xem chi tiết nhóm.



Hình 2.36: Biểu đồ tuần tự xem chi tiết nhóm

2.2.7 Biểu đồ use case phân rã quản lý thông báo



Hình 2.37: Biểu đồ usecase quản lý thông báo

Hình 2.37 là usecase phân rã "Quản lý thông báo". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Quản lý thông báo" được phân rã thành "Xem danh sách thông báo". Usecase "Xem danh sách thông báo" được phân rã thành "Đánh dấu là đã đọc" và "Xem chi tiết thông báo".

a, Biểu đồ usecase đánh dấu là đã đọc.

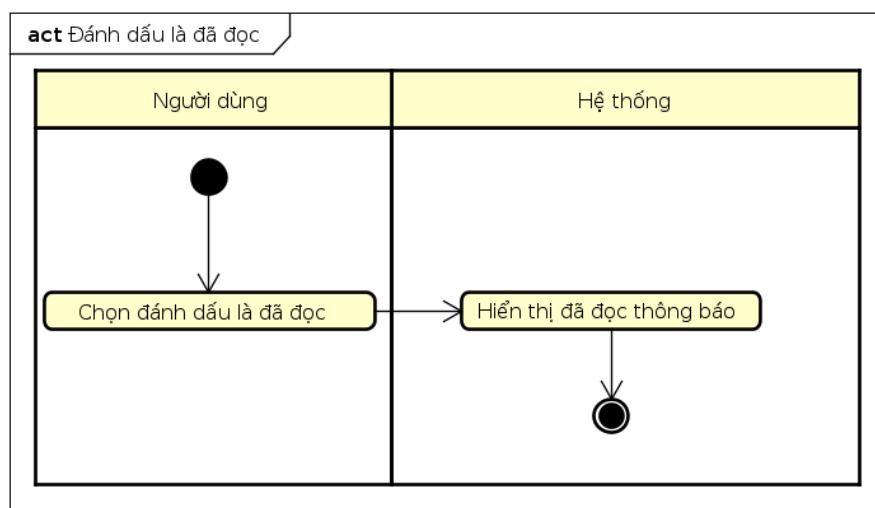
- Bảng đặc tả usecase đánh dấu là đã đọc.

Tên usecase	Đánh dấu là đã đọc.		
Mô tả	Kịch bản đánh dấu là đã đọc.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng xem danh sách thông báo.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn đánh dấu là đã đọc.
	2	Hệ thống	Hiển thị thông báo đã được đọc.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.17: Bảng đặc tả usecase đánh dấu là đã đọc.

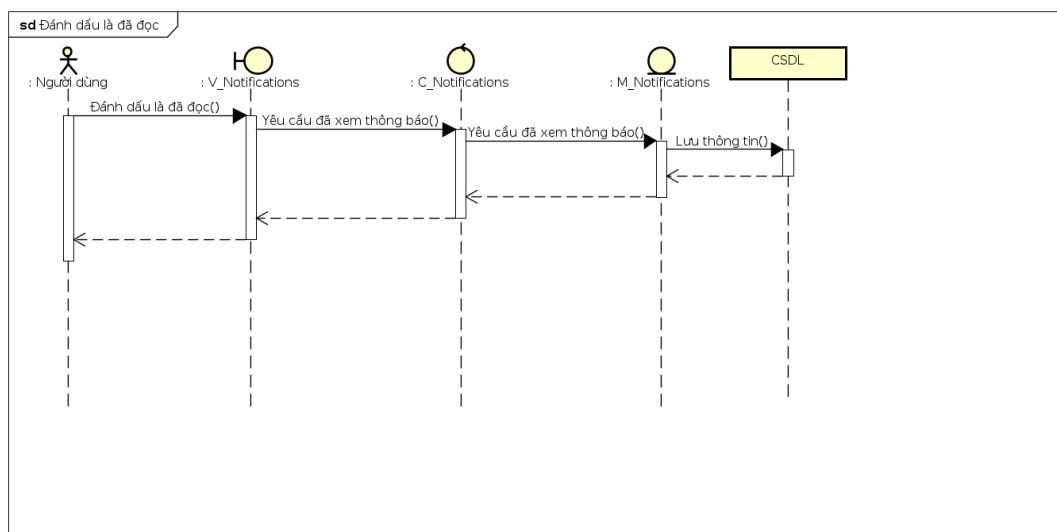
CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

- Biểu đồ hoạt động đánh dấu là đã đọc.
- Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng đánh dấu là đã đọc.



Hình 2.38: Biểu đồ hoạt động đánh dấu là đã đọc

- Biểu đồ tuần tự đánh dấu là đã đọc.



Hình 2.39: Biểu đồ tuần tự đánh dấu là đã đọc

b, Biểu đồ usecase xem chi tiết thông báo.

- Bảng đặc tả usecase xem chi tiết thông báo.

Tên usecase	Xem chi tiết thông báo.
Mô tả	Kịch bản xem chi tiết thông báo.

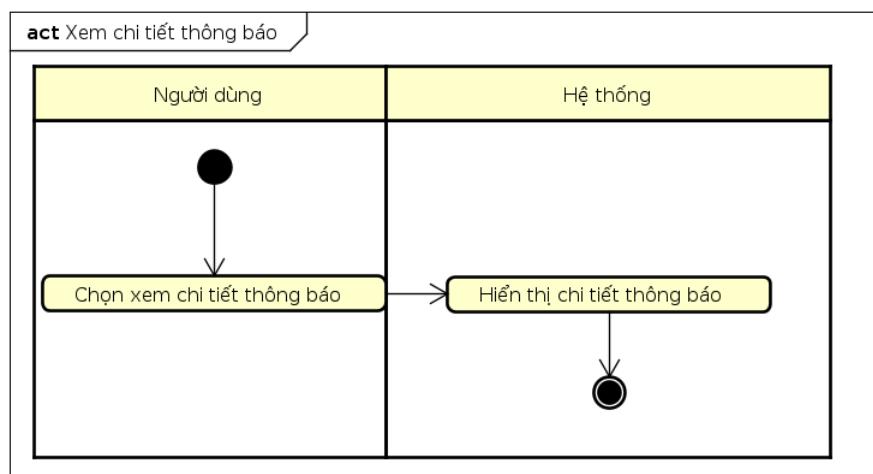
CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng xem chi tiết thông báo.		
Điều kiện tiên quyết	Không có.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn xem chi tiết thông báo.
	2	Hệ thống	Hiển thị nội dung thông báo.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.18: Bảng đặc tả usecase xem chi tiết thông báo.

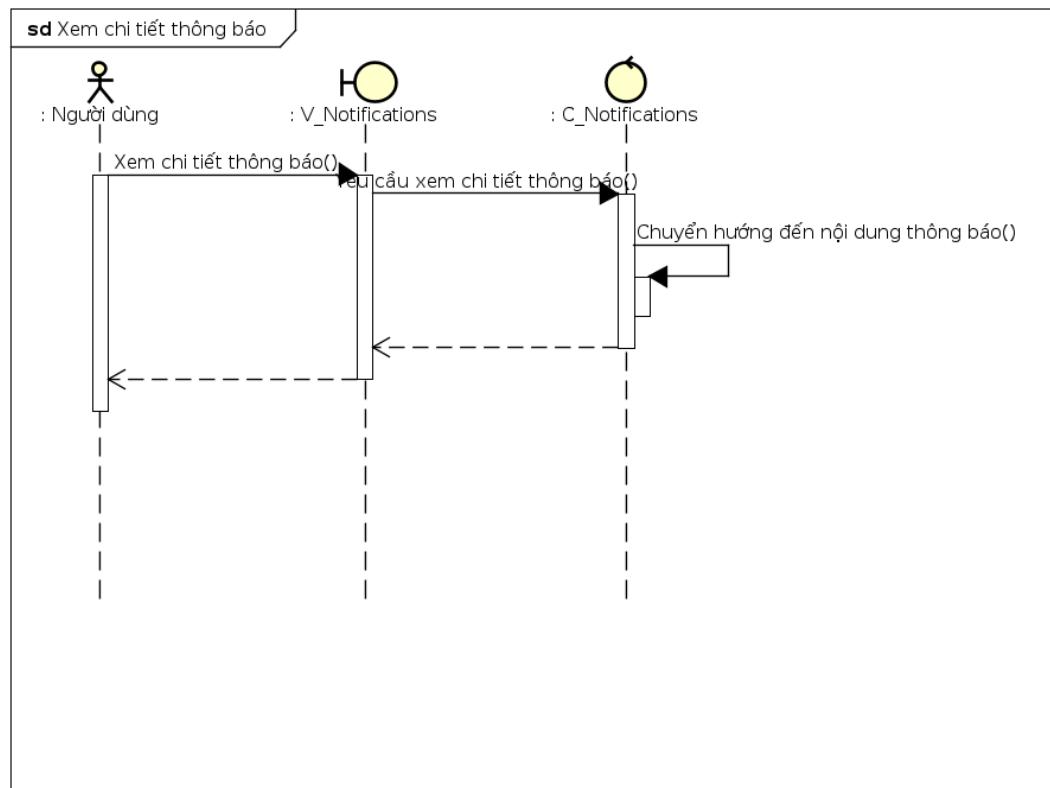
- Biểu đồ hoạt động xem chi tiết thông báo.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng xem chi tiết thông báo.



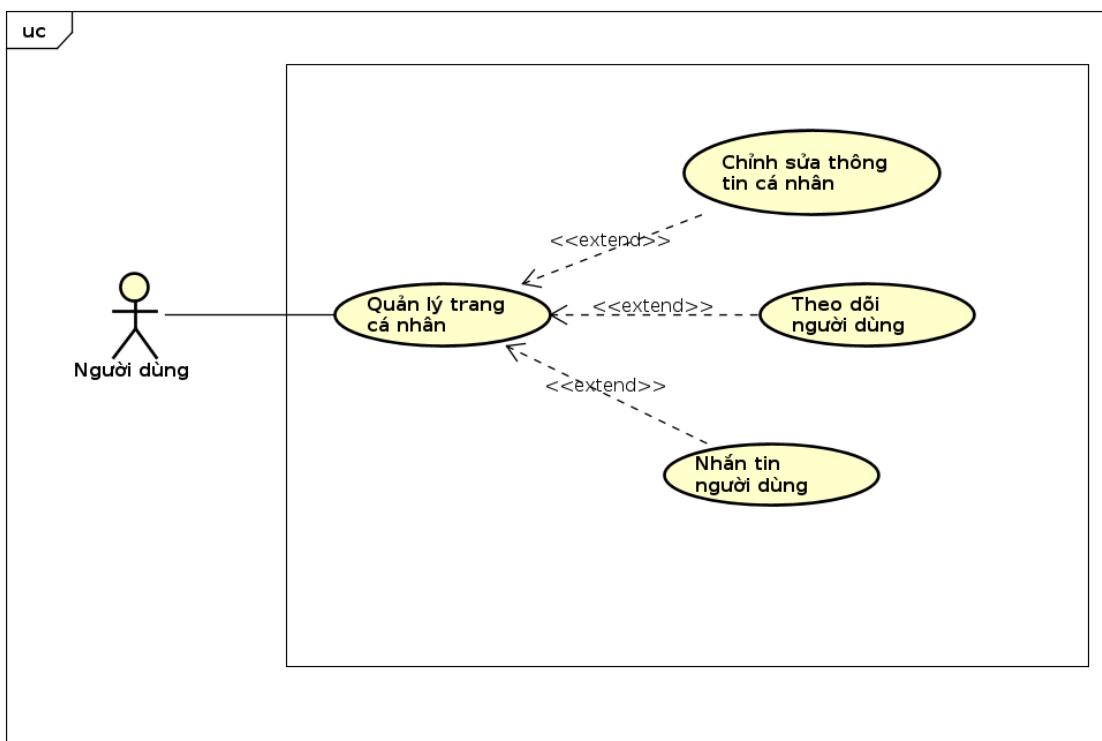
Hình 2.40: Biểu đồ hoạt động xem chi tiết thông báo

- Biểu đồ tuần tự xem chi tiết thông báo.



Hình 2.41: Biểu đồ tuần tự xem chi tiết thông báo

2.2.8 Biểu đồ use case phân rã quản lý trang cá nhân



Hình 2.42: Biểu đồ usecase quản lý trang cá nhân

Hình 2.42 là usecase phân rã "Quản lý trang cá nhân". Tác nhân bao gồm: Người dùng đã đăng nhập. Usecase "Quản lý trang cá nhân" được phân rã thành "Chỉnh sửa thông tin cá nhân", "Theo dõi người dùng", "Nhắn tin người dùng".

a, Biểu đồ usecase chỉnh sửa thông tin cá nhân.

- Bảng đặc tả usecase chỉnh sửa thông tin cá nhân.

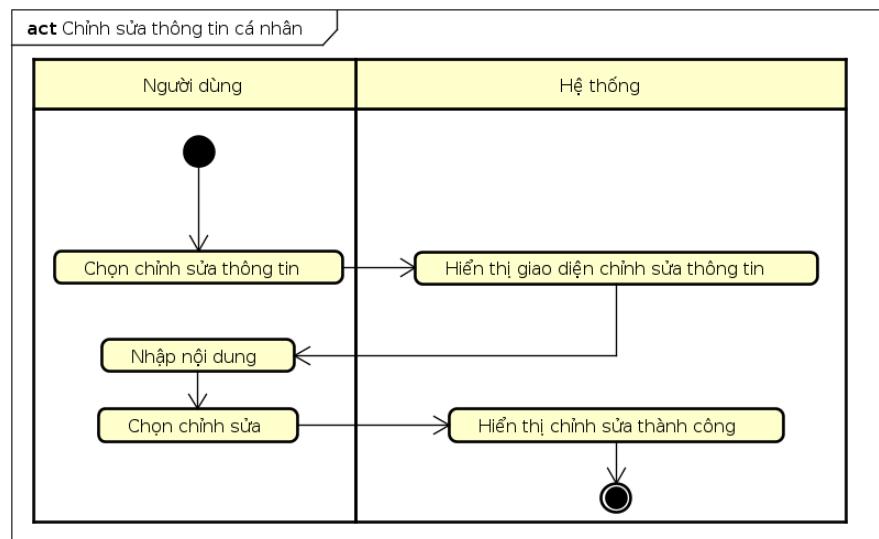
Tên usecase	Chỉnh sửa thông tin cá nhân.		
Mô tả	Kịch bản chỉnh sửa thông tin cá nhân.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng chọn chỉnh sửa thông tin cá nhân.		
Điều kiện tiên quyết	Phải là trang cá nhân của người dùng.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn chỉnh sửa thông tin cá nhân.
	2	Hệ thống	Hiển thị form chỉnh sửa thông tin cá nhân.
	3	Người dùng	Nhập thông tin chỉnh sửa.
	4	Hệ thống	Hiển thị chỉnh sửa thành công.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.19: Bảng đặc tả usecase chỉnh sửa thông tin cá nhân.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

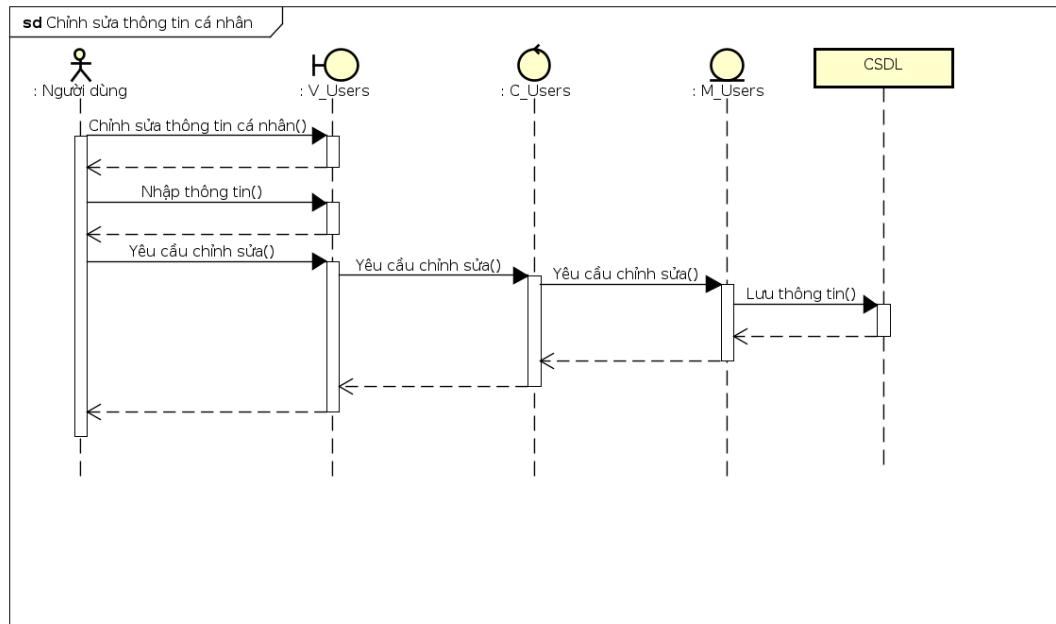
- Biểu đồ hoạt động chỉnh sửa thông tin cá nhân.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng chỉnh sửa thông tin cá nhân.



Hình 2.43: Biểu đồ hoạt động chỉnh sửa thông tin cá nhân

- Biểu đồ tuần tự chỉnh sửa thông tin cá nhân.



Hình 2.44: Biểu đồ tuần tự chỉnh sửa thông tin cá nhân

b, Biểu đồ usecase nhắm tin người dùng.

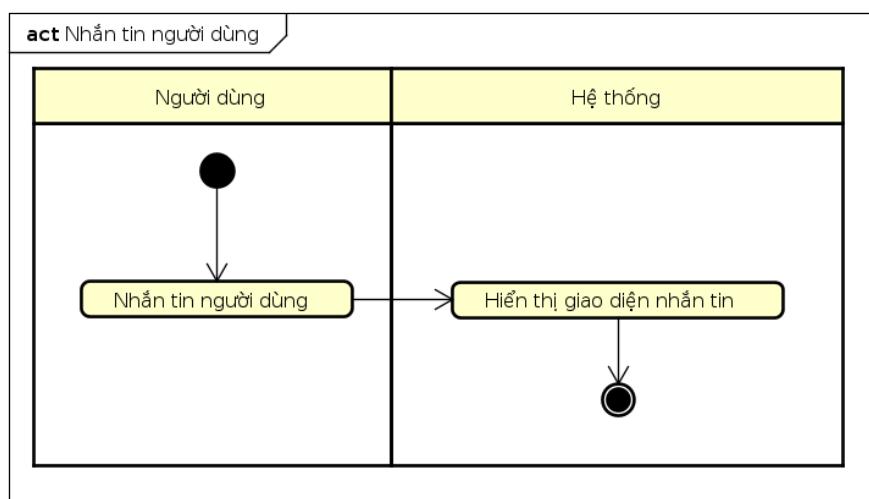
- Bảng đặc tả usecase nhắm tin người dùng.

Tên usecase	Nhắn tin người dùng.		
Mô tả	Kịch bản nhắn tin người dùng.		
Tác nhân sử dụng	Người dùng.		
Sự kiện kích hoạt	Người dùng xem trang cá nhân.		
Điều kiện tiên quyết	Không phải trang cá nhân của người dùng.		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Chọn nhắn tin.
	2	Hệ thống	Hiển thị giao diện nhắn tin.
Luồng sự kiện phụ	STT	Thực hiện bởi	Hành động
Hậu điều kiện	Không có.		

Bảng 2.20: Bảng đặc tả usecase nhắn tin.

- Biểu đồ hoạt động nhắn tin người dùng.

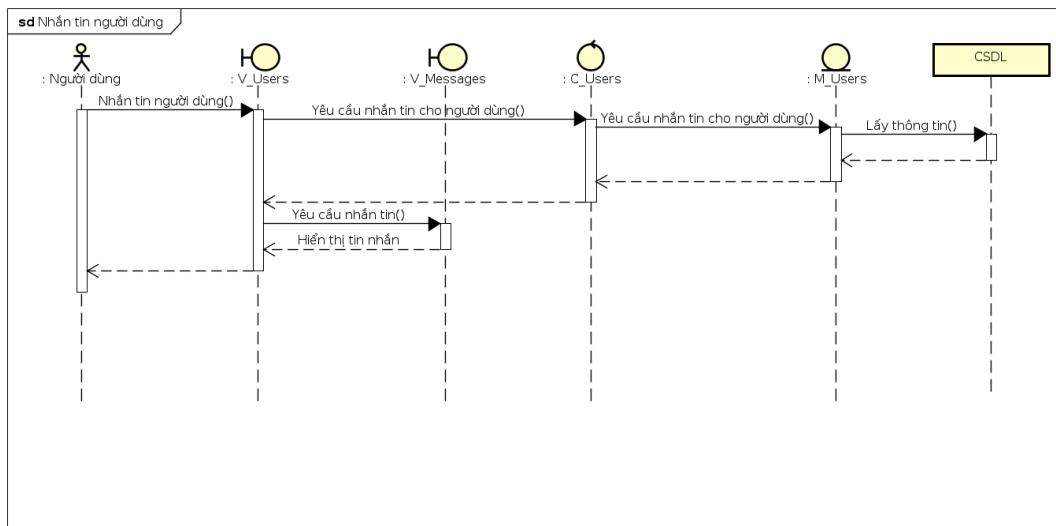
Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng nhắn tin người dùng.



Hình 2.45: Biểu đồ hoạt động nhắn tin người dùng

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

- Biểu đồ tuần tự nhắn tin người dùng.



Hình 2.46: Biểu đồ tuần tự nhắn tin người dùng

c, Biểu đồ usecase theo dõi người dùng.

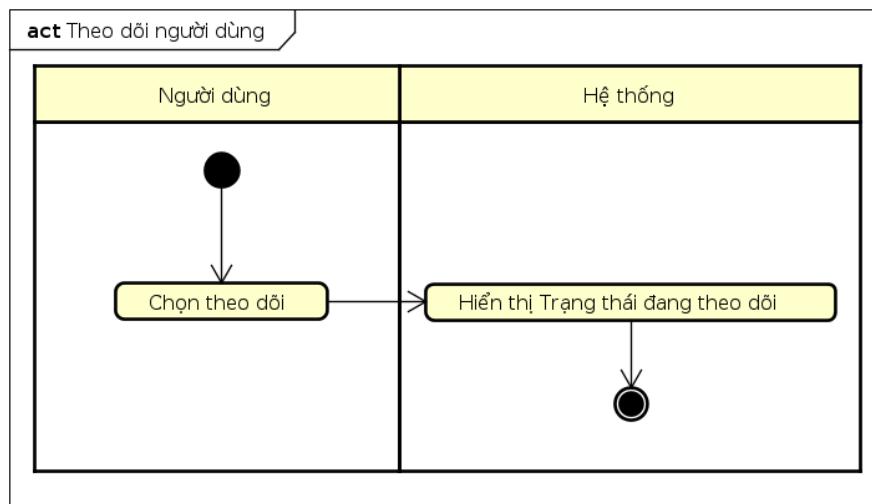
- Bảng đặc tả usecase theo dõi người dùng.

Tên usecase	Theo dõi người dùng.									
Mô tả	Kịch bản theo dõi người dùng.									
Tác nhân sử dụng	Người dùng.									
Sự kiện kích hoạt	Người dùng xem trang cá nhân.									
Điều kiện tiên quyết	Không phải trang cá nhân của người dùng.									
Luồng sự kiện chính	<table border="1"> <thead> <tr> <th>STT</th> <th>Thực hiện bởi</th> <th>Hành động</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Người dùng</td> <td>Chọn Theo dõi.</td> </tr> <tr> <td>2</td> <td>Hệ thống</td> <td>Hiển thị theo dõi thành công.</td> </tr> </tbody> </table>	STT	Thực hiện bởi	Hành động	1	Người dùng	Chọn Theo dõi.	2	Hệ thống	Hiển thị theo dõi thành công.
STT	Thực hiện bởi	Hành động								
1	Người dùng	Chọn Theo dõi.								
2	Hệ thống	Hiển thị theo dõi thành công.								
Luồng sự kiện phụ	<table border="1"> <thead> <tr> <th>STT</th> <th>Thực hiện bởi</th> <th>Hành động</th> </tr> </thead> </table>	STT	Thực hiện bởi	Hành động						
STT	Thực hiện bởi	Hành động								
Hậu điều kiện	Không có.									

Bảng 2.21: Bảng đặc tả usecase theo dõi người dùng.

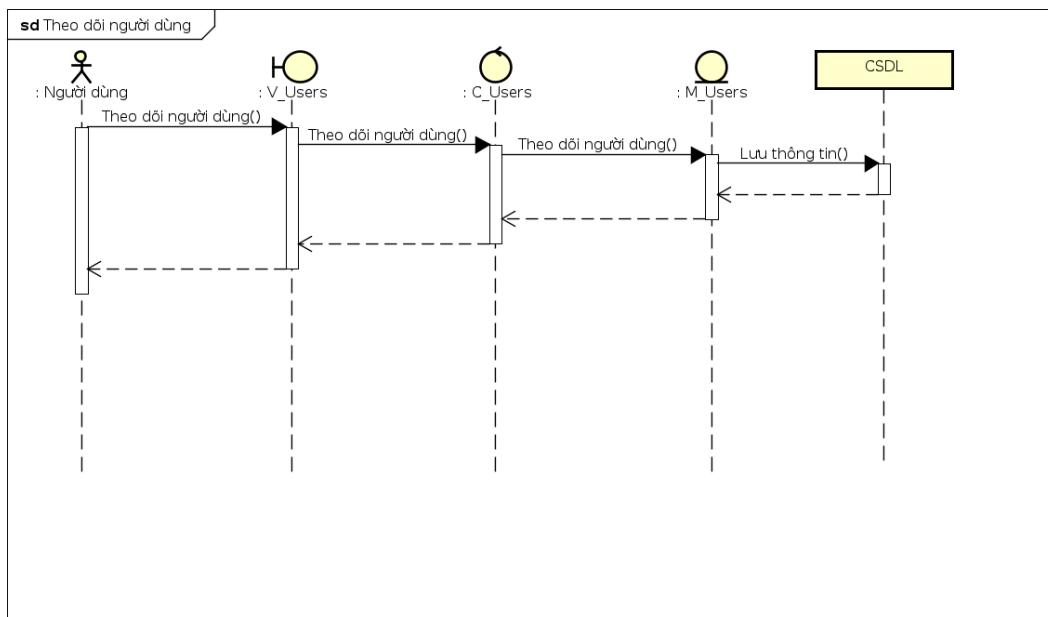
- Biểu đồ hoạt động theo dõi người dùng.

Hình vẽ dưới đây mô tả luồng hoạt động của hệ thống với chức năng theo dõi người dùng.



Hình 2.47: Biểu đồ hoạt động theo dõi người dùng

- Biểu đồ tuần tự theo dõi người dùng.



Hình 2.48: Biểu đồ tuần tự theo dõi người dùng

2.3 Yêu cầu phi chức năng

Những yêu cầu phi chức năng của hệ thống:

- Về giao diện người dùng: Trực quan, đơn giản, dễ sử dụng, làm quen, thao tác ổn định đem lại trải nghiệm tốt cho người dùng và cuối cùng có thể tương thích tốt, hoạt động mượt mà trên nhiều loại thiết bị khác nhau
- Về mặt hiệu năng: Đảm bảo khả năng nhanh và ổn định. Trong trường hợp có nhiều người dùng sử dụng hệ thống trong cùng một thời điểm, hệ thống phải có khả năng chịu tải tốt đáp ứng nhiều người dùng. Thời gian phản hồi của api phải nằm trong một mức nhất định có thể chấp nhận được.
- Về mặt bảo mật: Hệ thống phải có phân quyền và xác thực người dùng.

CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

Sau khi phân tích những chức năng chính của hệ thống ở chương 2, tiếp theo, em sẽ lựa chọn những công nghệ để xây dựng lên hệ thống của mình. Sau đây, em sẽ liệt kê ra những công nghệ mà em sẽ sử dụng.

3.1 VueJS

Vue.js [1] là một framework javascript mã nguồn mở được dùng để phát triển, xây dựng các giao diện web tương tác. Vue là một trong những framework nổi tiếng để phát triển web vì nó đơn giản và dễ sử dụng. Nó có thể dễ dàng tích hợp vào các hệ thống mà không gặp bất cứ sự cố nào. Thư viện của Vue chỉ tập trung vào lớp hiển thị, nó sử dụng rất đơn giản.

Ưu điểm của Vue là kích thước của nó rất là nhỏ., điều này giúp cho việc cài đặt rất nhanh. Ngoài ra, Vue.js còn sử dụng DOM ảo, việc này sẽ giúp tìm ra những phần tử cần cập nhật mà không phải kết xuất cập nhật lại toàn bộ lại DOM. Cách này giúp cải thiện hiệu của ứng dụng và giúp hiển thị trang nhanh hơn.

Tuy có nhiều ưu điểm như vậy nhưng Vue.js vẫn có nhược điểm, nhược điểm ở đây là cộng đồng của Vue vẫn còn bé vắn chưa lớn bằng những cộng đồng của một số framework nổi tiếng khác như React hay Angular.

3.2 ExpressJS

ExpressJS [2] đây là một framework mã nguồn mở miễn phí cho Node.js được sử dụng trong xây dựng và thiết kế các ứng dụng web với nhiều tính năng mạnh mẽ trên nền tảng web. Express chỉ yêu cầu ngôn ngữ lập trình Javascript, nó hỗ trợ các phương thức HTTP và middleware nên việc tạo ra một API trở nên đơn giản hơn, mạnh mẽ và dễ sử dụng hơn.

Trong đồ án này, em chọn sử dụng ExpressJS để làm Backend cho đồ án do tính đơn giản, dễ sử dụng và hiệu suất cao của nó. Ngoài ra, nó còn có một cộng đồng lớn để có thể trao đổi và hỗ trợ tích cực.

3.3 MongoDB

MongoDB [3] là một hệ quản trị cơ sở dữ liệu hướng tài liệu, nó là một dạng của hệ quản trị cơ sở dữ liệu NoSQL. MongoDB sẽ lưu trữ dữ liệu dưới dạng JSON nên mỗi bảng ghi dữ liệu sẽ có các kích cỡ khác nhau. Việc lưu trữ dữ liệu dưới dạng JSON cũng sẽ giúp việc truy vấn trở nên nhanh gọn hơn.

MongoDB là cơ sở dữ liệu kiểu NoSQL, nó không có các kiểu quan hệ giống như SQL cho nên điều này sẽ giúp MongoDB trở nên linh hoạt và thích ứng với

nhiều yêu cầu hơn nhưng bù lại các kiểu ràng buộc trong cơ sở dữ liệu cũng sẽ không được chặt chẽ.

Ngoài ra, MongoDB còn có một số ưu điểm khác như do không có tính ràng buộc, toàn vẹn nên hiệu suất lớn, tính sẵn sàng cao và dễ dàng mở rộng lưu trữ. Dữ liệu sẽ được ghi đệm trên RAM nên tốc độ đọc ghi sẽ rất cao.

3.4 Google Firebase

Google Firebase [4] là một nền tảng phát triển ứng dụng di động và web của Google, cung cấp một loạt các dịch vụ và công cụ giúp nhà phát triển xây dựng ứng dụng một cách dễ dàng. Điều này bao gồm cơ sở dữ liệu thời gian thực, xác thực người dùng, hosting đám mây, và nhiều tính năng khác.

Firebase giúp nhà phát triển quản lý đăng nhập và xác thực người dùng thông qua dịch vụ Authentication. Nó cũng cung cấp Realtime Database và Cloud Firestore, hai loại cơ sở dữ liệu NoSQL giúp đồng bộ dữ liệu trực tiếp giữa các thiết bị mà không cần làm mới trang.

Ngoài ra, Firebase còn cung cấp các dịch vụ như lưu trữ đám mây cho tệp đa phương tiện, thông báo đám mây, theo dõi hiệu suất ứng dụng, kiểm thử tự động, và phân tích lỗi. Điều này giúp nhà phát triển tập trung vào việc xây dựng và cải thiện ứng dụng mà không cần lo lắng nhiều về các tác vụ hạ tầng phức tạp.

3.5 Redis

Redis [5] là một hệ thống cơ sở dữ liệu key-value in-memory, nổi tiếng với hiệu suất cao và khả năng mở rộng. Được phát triển dưới dạng mã nguồn mở, Redis thường được sử dụng để lưu trữ dữ liệu tạm thời trong bộ nhớ RAM, giúp tăng cường hiệu suất đọc/ghi so với các cơ sở dữ liệu truyền thống dựa trên đĩa.

Một trong những đặc điểm quan trọng của Redis là khả năng lưu trữ các kiểu dữ liệu phức tạp như chuỗi, hash, danh sách, tập hợp và bản đồ bit. Điều này làm cho Redis trở thành một lựa chọn linh hoạt cho việc lưu trữ và truy xuất dữ liệu đa dạng.

Redis cũng hỗ trợ các tính năng như pub/sub (publisher/subscriber) cho việc gửi và nhận thông báo, đồng bộ hóa dữ liệu, và có thể tích hợp với nhiều ngôn ngữ lập trình khác nhau thông qua các thư viện và API khác nhau.

Với việc tập trung vào tốc độ và khả năng mở rộng, Redis thường được sử dụng trong các ứng dụng yêu cầu xử lý dữ liệu nhanh, như các hệ thống đếm số lượt xem, bảng xếp hạng, hay bất kỳ ứng dụng nào cần truy xuất dữ liệu từ bộ nhớ RAM một cách nhanh chóng.

3.6 Minio

Minio [6] là một hệ thống lưu trữ đối tượng mã nguồn mở, được thiết kế để cung cấp dịch vụ lưu trữ đám mây trên các máy chủ riêng hoặc trong môi trường đám mây công cộng. Minio giữ vững cấu trúc đơn giản và dễ triển khai, giúp người dùng tạo ra hệ thống lưu trữ đám mây cá nhân hoặc doanh nghiệp mà không phải dựa vào các dịch vụ lưu trữ lớn từ các nhà cung cấp đám mây khác.

Với Minio, người dùng có thể tạo ra các đối tượng lưu trữ linh hoạt, lưu trữ và quản lý dữ liệu như hình ảnh, video, và các tệp tin khác. Hệ thống này hỗ trợ giao thức đám mây phổ biến như Amazon S3, giúp tích hợp dễ dàng với các ứng dụng sử dụng giao thức này.

Minio cung cấp khả năng mở rộng ngang, cho phép người dùng mở rộng dung lượng lưu trữ và hiệu suất theo nhu cầu mà không làm giảm hiệu suất. Điều này làm cho Minio trở thành một giải pháp linh hoạt và tiết kiệm chi phí cho việc triển khai lưu trữ đám mây tại các doanh nghiệp và tổ chức.

3.7 Docker

Docker [7] là một nền tảng mã nguồn mở giúp đơn giản hóa quá trình triển khai ứng dụng và quản lý các môi trường phát triển. Docker sử dụng công nghệ containerization để đóng gói ứng dụng và tất cả các phụ thuộc của chúng vào một container duy nhất, đảm bảo rằng ứng dụng có thể chạy một cách nhất quán trên bất kỳ hệ thống nào đã cài đặt Docker.

Mỗi container Docker chứa môi trường thực thi độc lập, bao gồm cả hệ điều hành, thư viện, và các tài nguyên cần thiết, mà không ảnh hưởng đến hệ thống máy chủ chủ. Điều này tạo ra môi trường cô lập, giúp đảm bảo tính nhất quán và di động cho ứng dụng, đồng thời giảm tối thiểu các xung đột giữa các phần mềm chạy trên cùng một hệ thống.

Docker cung cấp một tập hợp các công cụ quản lý và xây dựng hình ảnh, giúp các nhà phát triển dễ dàng chia sẻ, triển khai và quản lý ứng dụng trên nhiều môi trường. Docker còn hỗ trợ các dịch vụ như Docker Compose để đơn giản hóa việc triển khai và quản lý nhiều container cùng một lúc.

Đối với cộng đồng phát triển và quản trị hệ thống, Docker đã trở thành một công cụ quan trọng trong việc tăng cường linh hoạt và tính di động của các ứng dụng và dịch vụ.

3.8 Microservices

Microservices [8] là một kiến trúc phần mềm phân tách ứng dụng thành các dịch vụ nhỏ, độc lập chức năng, chạy trên các quy trình và cơ sở hạ tầng riêng biệt.

Mỗi dịch vụ trong kiến trúc Microservices có thể được phát triển, triển khai, và mở rộng độc lập, giúp tối ưu hóa sự linh hoạt và khả năng mở rộng của hệ thống.

Kiến trúc Microservices giúp giải quyết các vấn đề liên quan đến quy mô và phát triển trong các ứng dụng monolithic bằng cách chia nhỏ chúng thành các thành phần quản lý riêng biệt. Các dịch vụ nhỏ này có thể được phát triển bằng ngôn ngữ lập trình và công nghệ phù hợp cho nhiệm vụ cụ thể của chúng, giảm bớt sự phụ thuộc và cải thiện khả năng quản lý.

Microservices cung cấp các lợi ích như tăng khả năng mở rộng, thuận tiện trong quản lý và bảo trì, cũng như khả năng triển khai liên tục. Tuy nhiên, cũng đồng điệu với những lợi ích này là các thách thức về quản lý độ phức tạp, giao tiếp giữa các dịch vụ, và việc duy trì tính nhất quán trong hệ thống phức tạp.

3.9 RabbitMQ

RabbitMQ [9] là một hệ thống message broker mã nguồn mở, được thiết kế để hỗ trợ việc truyền thông và gửi nhận thông điệp giữa các ứng dụng và dịch vụ khác nhau. Sử dụng mô hình hóa message queuing, RabbitMQ giúp tạo ra một hệ thống linh hoạt và có khả năng mở rộng để xử lý thông điệp trong môi trường phân tán.

RabbitMQ sử dụng giao thức AMQP để đảm bảo tính nhất quán và độ tin cậy trong quá trình truyền thông. Nó hỗ trợ nhiều tính năng như message acknowledgement, định tuyến linh hoạt, và quản lý độ ưu tiên, giúp tối ưu hóa quá trình trao đổi thông điệp giữa các thành phần khác nhau của một hệ thống. RabbitMQ thường được sử dụng trong kiến trúc phần mềm phân tán và các hệ thống microservices để giải quyết vấn đề gửi và nhận thông điệp hiệu quả.

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

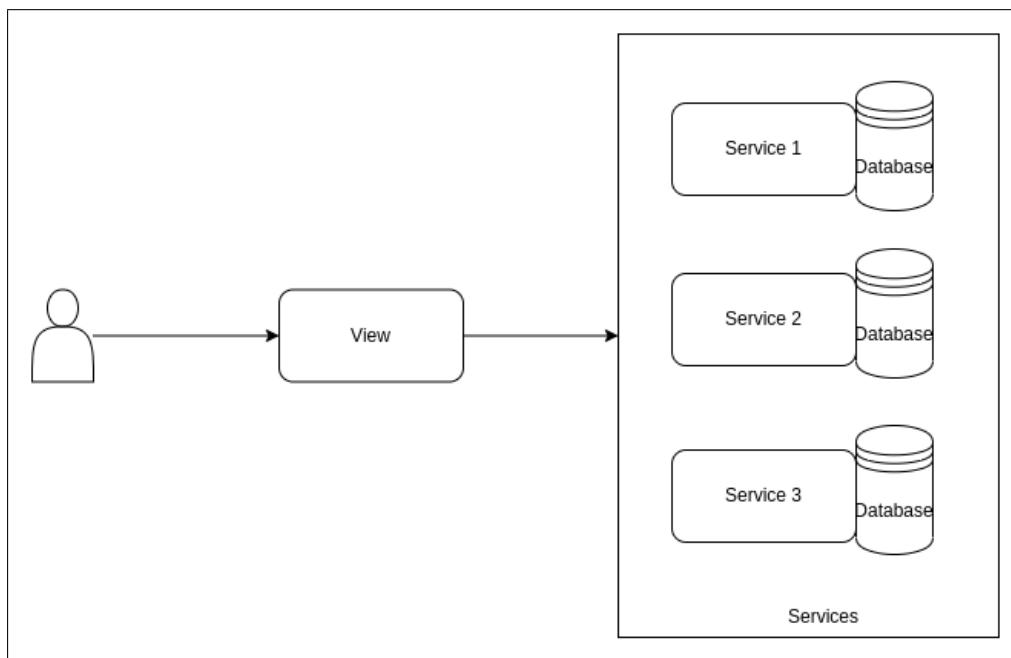
Tiếp theo, phần nội dung được trình bày sau đây là quá trình thiết kế, xây dựng và triển khai hệ thống lên môi trường thật. Kiến trúc em lựa chọn ở đây là kiến trúc microservices.

4.1 Thiết kế kiến trúc

4.1.1 Lựa chọn kiến trúc phần mềm

a. Mô hình microservices

Ngày xưa, chúng ta thường hay thiết kế và xây dựng kiến trúc theo nguyên khối (monolithic). Nhưng theo thời gian, các hệ thống càng ngày càng phát triển lớn và mở rộng thêm, việc thiết kế hệ thống theo kiến trúc nguyên khối làm ta khó mở rộng và việc phát triển sản phẩm trở nên khó khăn và tốn nhiều công sức hơn rất nhiều. Từ đó mà mô hình microservices ra đời. Với microservices, hệ thống có thể chia nhỏ ra thành nhiều service khác nhau, mỗi service đảm nhận một nghiệp vụ riêng. Như vậy việc phát triển phần mềm sẽ trở nên dễ dàng, nhanh và mở rộng dễ hơn.



Hình 4.1: Mô hình Microservices

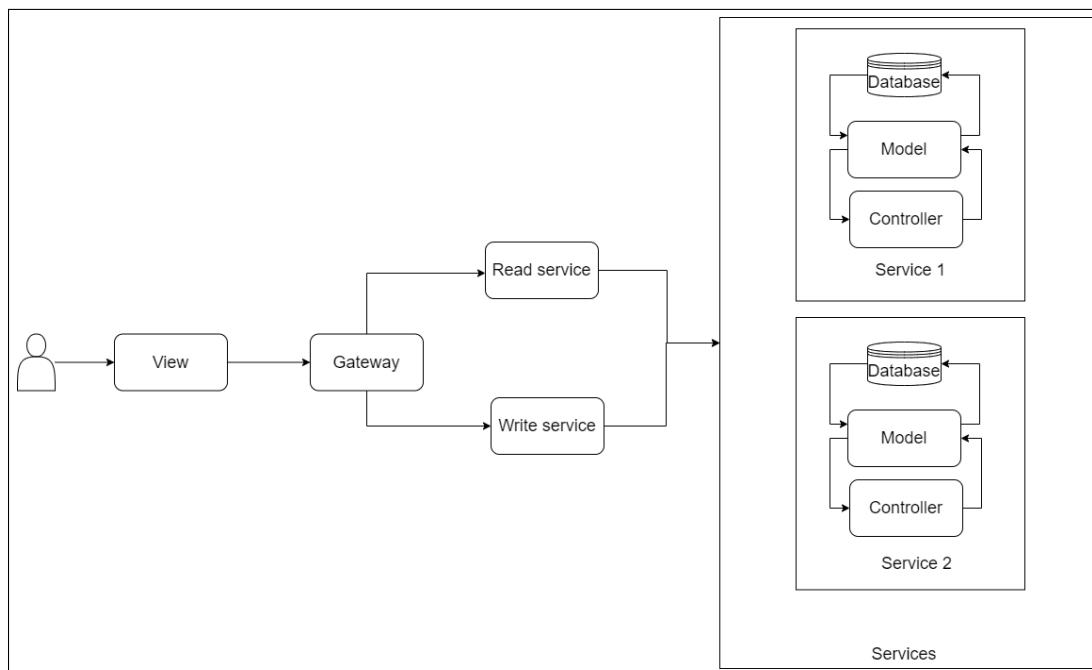
Các thành phần chính của kiến trúc này bao gồm:

- View: Đây là thành phần có tác dụng hiển thị giao diện của ứng dụng. Đây là nơi cho người dùng tương tác, nhận các yêu cầu từ người dùng và đưa xuống cho các service xử lý.

- Service: Đây là bộ phận tiếp nhận và xử lý các yêu cầu của người dùng được chuyển tới từ thành phần View. Bên trong Service thường sẽ có hai phần là Controller và Model:
 - Controller: Là nơi nhận những yêu cầu trước, là nơi xử lý những yêu cầu của người dùng. Nó là nơi liên kết giữa Model và View.
 - Model: Đây là thành phần lưu trữ dữ liệu và xử lý các logic. Model là nơi chứa các hàm, truy vấn trực tiếp tới cơ sở dữ liệu.

b, Mô hình microservices trong hệ thống

Dựa vào mô hình microservices, hệ thống của em được thiết kế như sau:



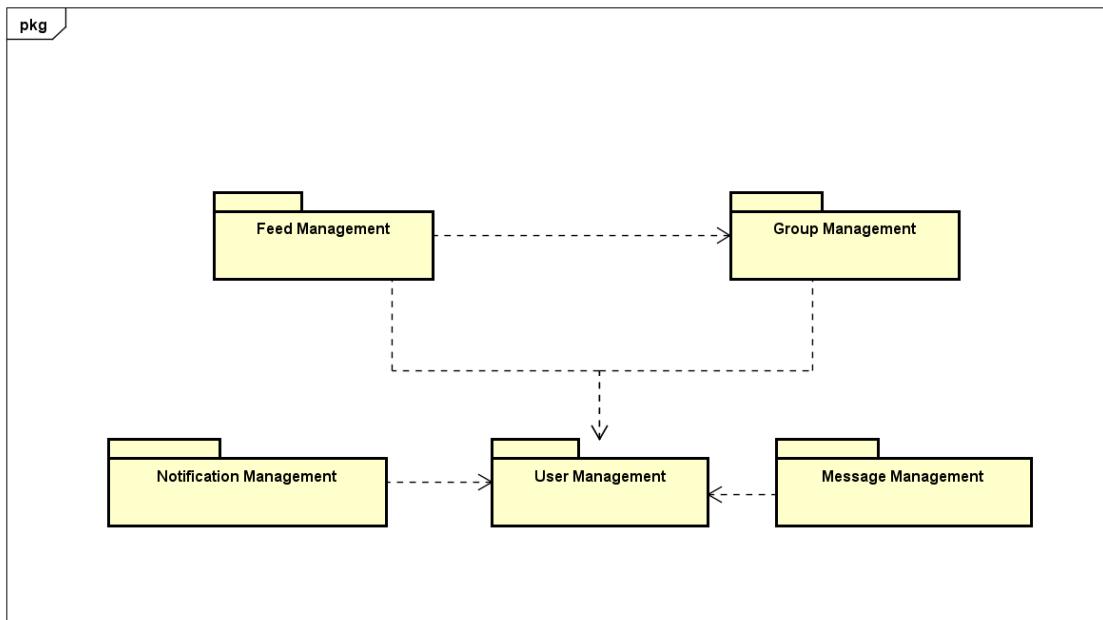
Hình 4.2: Mô hình Microservices trong hệ thống

Kiến trúc này cụ thể như sau:

- View: Hiển thị giao diện, tương tác, nhận yêu cầu từ người dùng gửi đến Gateway API.
- Gateway: Là nơi nhận yêu cầu từ người dùng, kiểm tra các quyền của người dùng.
- ReadApi: Là nơi mà các api GET sẽ đi qua.
- WriteApi: Là nơi mà những api POST, PUT, DELETE sẽ đi qua.
- Controller: Chứa các hàm thực thi.
- Model: Lưu trữ thông tin của các đối tượng. Nó như một đối tượng ánh xạ với các bảng trong cơ sở dữ liệu.

- User: Người dùng website.

c, Thiết kế tổng quan



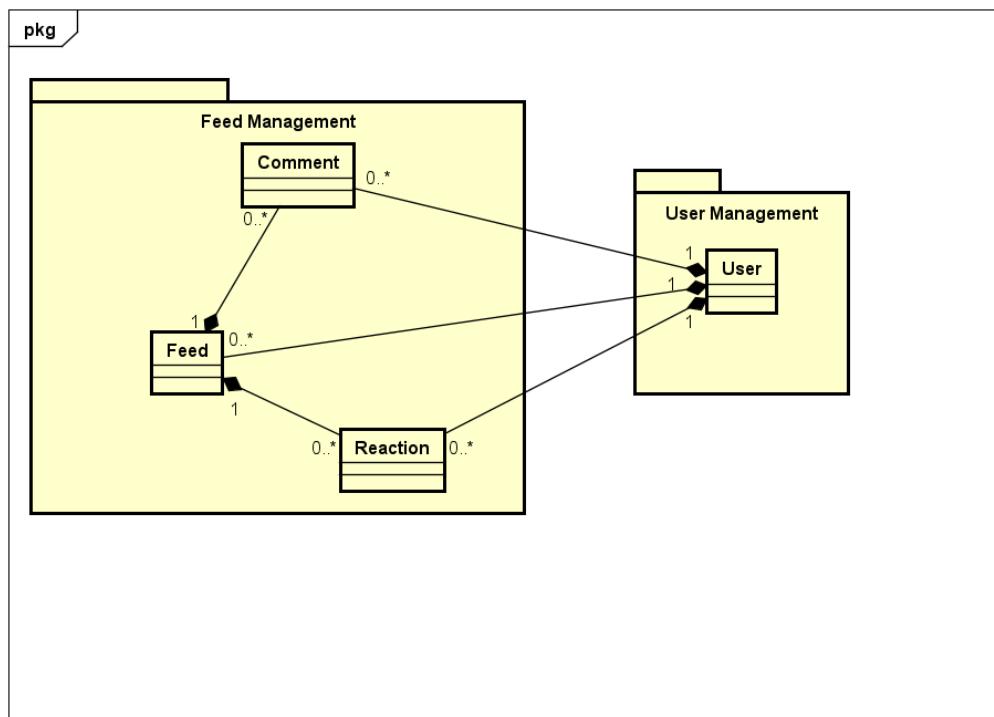
Hình 4.3: Biểu đồ gói của hệ thống

Hình 4.3 là sơ đồ gói của hệ thống. Chức năng của từng gói như sau:

- Feed: Chứa các lớp liên quan đến bài viết với các thao tác như tạo bài viết, chỉnh sửa bài viết, tìm kiếm,... Ngoài ra còn chứa các lớp liên quan đến bình luận với các thao tác chỉnh sửa bình luận, xóa bình luận và các lớp liên quan đến thích với các thao tác thích bài viết, thích bình luận.
- Group: Chứa các lớp liên quan đến nhóm với các thao tác như tạo nhóm, chỉnh sửa nhóm, tìm nhóm.
- Notification: Chứa các lớp liên quan đến thông báo với các thao tác như tìm kiếm, lọc thông báo, gửi thông báo.
- User: Chứa các lớp liên quan đến người dùng với các thao tác như lấy thông tin người dùng, chỉnh sửa thông tin người dùng...
- Message: Chứa các lớp liên quan đến nhắn tin với các thao tác như lấy danh sách cuộc hội thoại, lấy tin nhắn, gửi tin nhắn,..

4.1.2 Thiết kế chi tiết gói

a, Biểu đồ gói chi tiết Feed Management

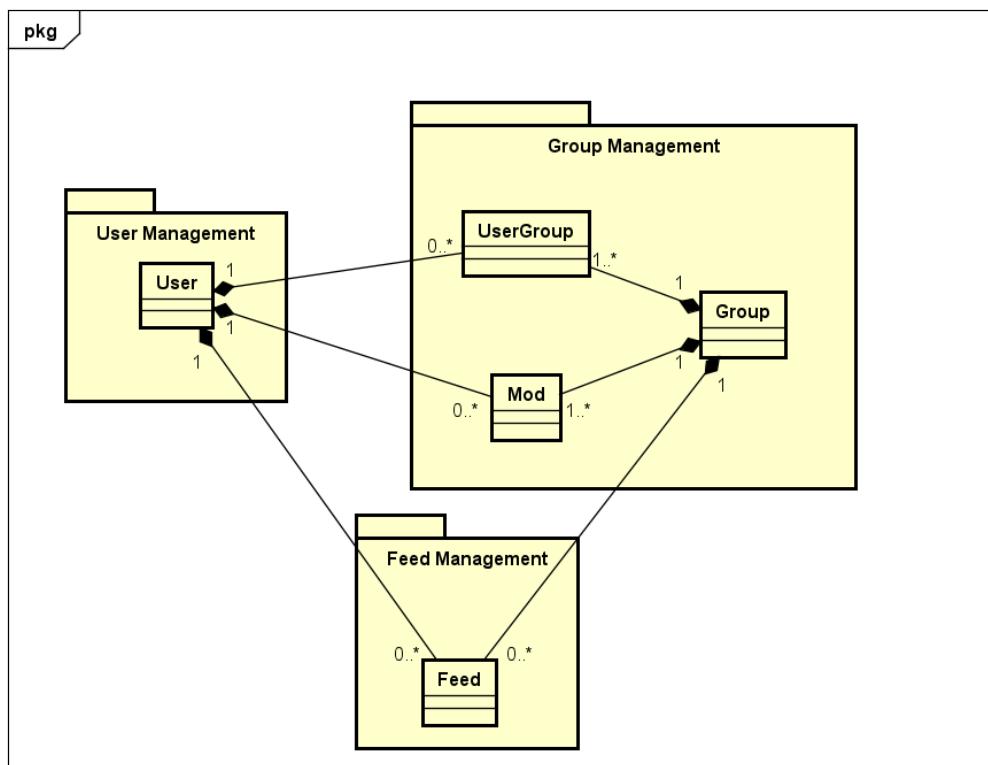


Hình 4.4: Biểu đồ gói chi tiết Feed Management

Hình 4.4 mô tả chi tiết gói Feed Management. Trong đó:

- Feed: Class chứa các thuộc tính và phương thức về các bài viết.
- Comment: Class chứa các thuộc tính và phương thức và bình luận.
- Reaction: Class chứa các thuộc tính và phương thức và cảm xúc với bài viết.
- User: Class chứa các thuộc tính và phương thức về người dùng.

b, Biểu đồ gói chi tiết Group Management

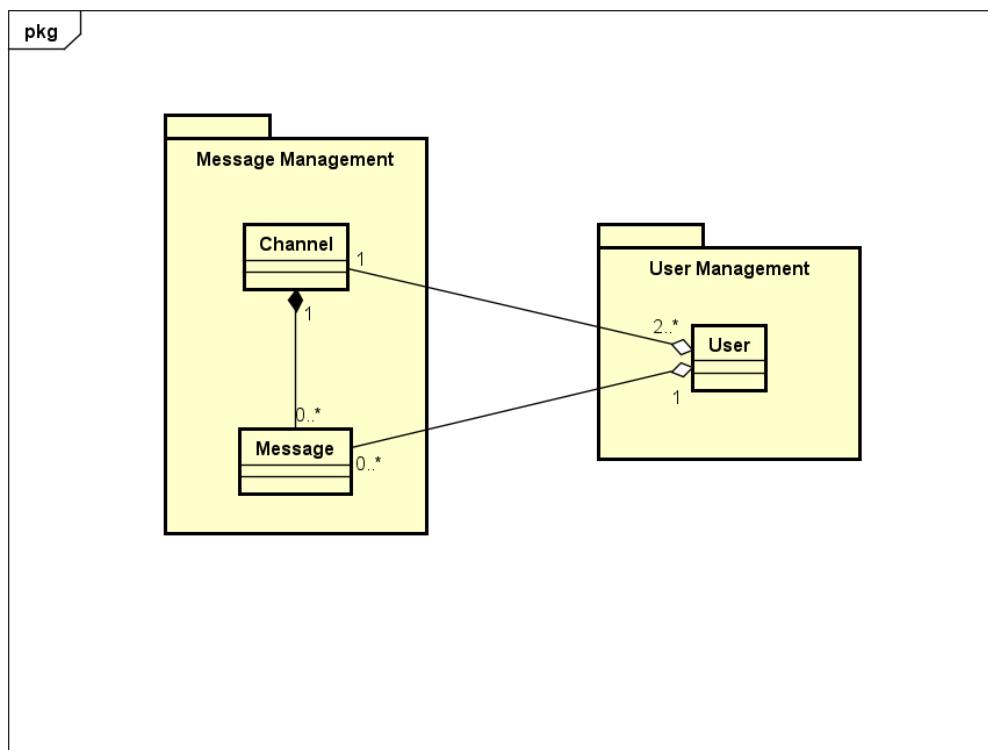


Hình 4.5: Biểu đồ gói chi tiết Group Management

Hình 4.5 mô tả chi tiết gói Group Management. Trong đó:

- Group: Class chứa các thuộc tính và phương thức về các nhóm.
 - UserGroup: Class chứa các thuộc tính và phương thức và những người dùng trong nhóm.
 - Mod: Class chứa các thuộc tính và phương thức về các quản trị viên của nhóm.
 - Feed: Class chứa các thuộc tính và phương thức về các bài viết.
 - User: Class chứa các thuộc tính và phương thức về người dùng.

c, Biểu đồ gói chi tiết Message Management

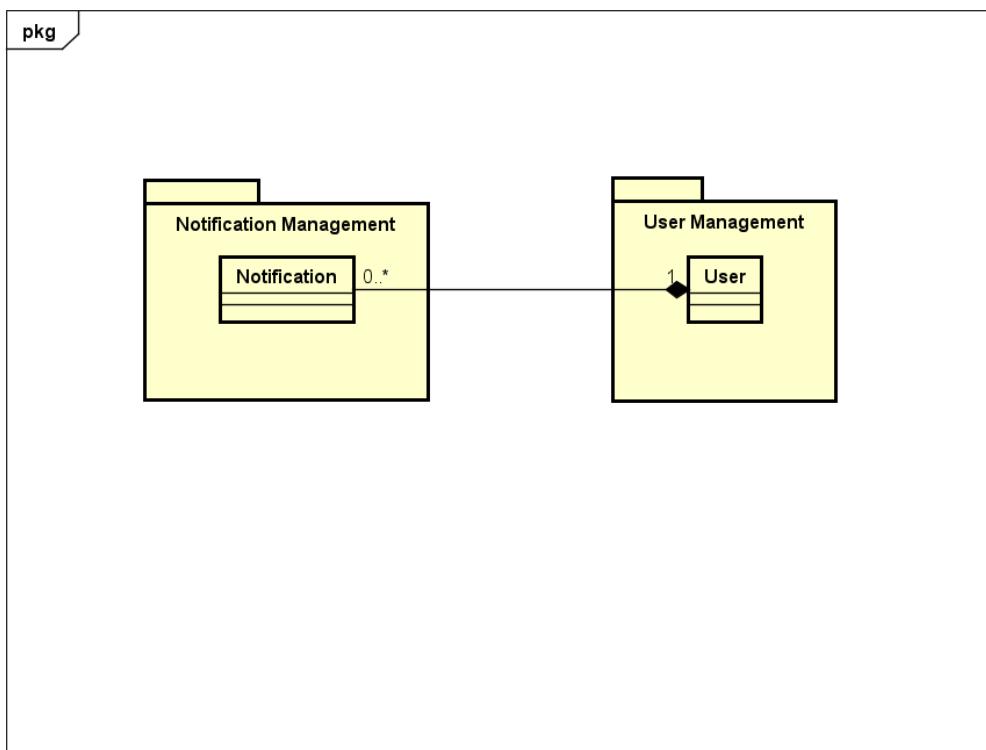


Hình 4.6: Biểu đồ gói chi tiết Message Management

Hình 4.6 mô tả chi tiết gói Message Management. Trong đó:

- **Channel:** Class chứa các thuộc tính và phương thức về các cuộc hội thoại.
- **Message:** Class chứa các thuộc tính và phương thức về tin nhắn người dùng.
- **User:** Class chứa các thuộc tính và phương thức về người dùng.

d, Biểu đồ gói chi tiết Notification Management

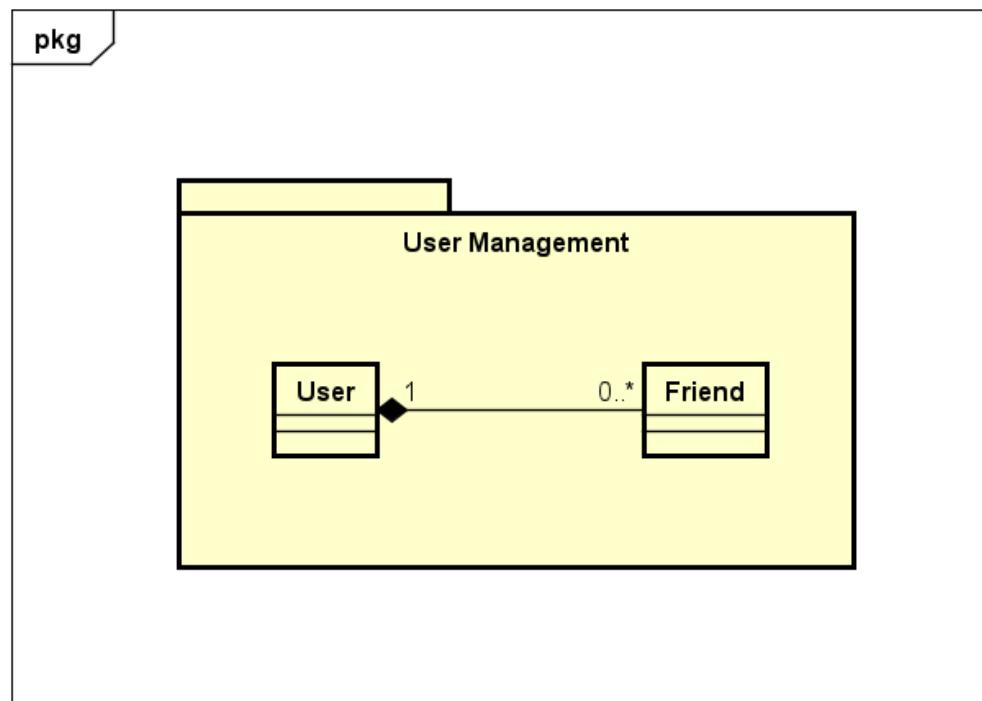


Hình 4.7: Biểu đồ gói chi tiết Notification Management

Hình 4.7 mô tả chi tiết gói Notification Management. Trong đó:

- **Notification:** Class chứa các thuộc tính và phương thức về các nội dung thông báo.
- **User:** Class chứa các thuộc tính và phương thức về người dùng.

e, Biểu đồ gói chi tiết User Management

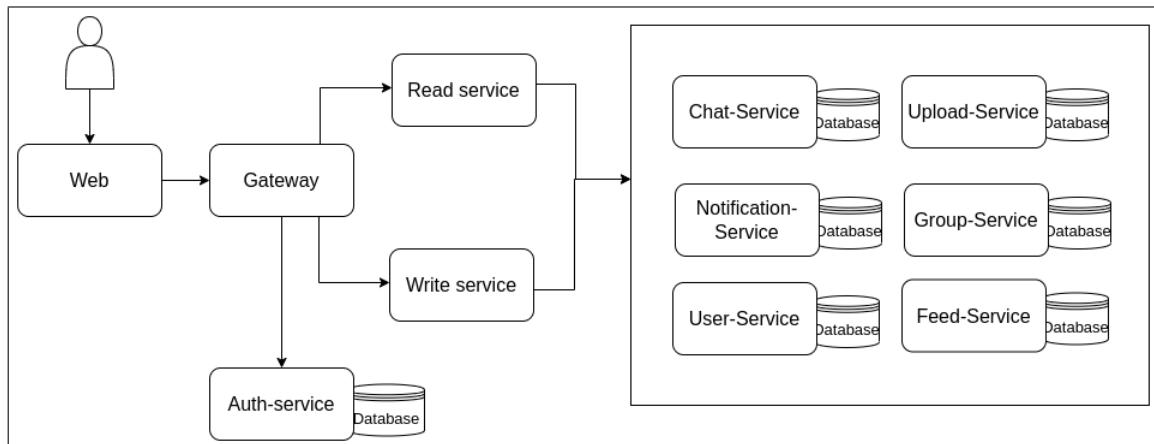


Hình 4.8: Biểu đồ gói chi tiết User Management

Hình 4.8 mô tả chi tiết gói User Management. Trong đó:

- Friend: Class chứa các thuộc tính và phương thức về các mối quan hệ bạn bè.
- User: Class chứa các thuộc tính và phương thức về người dùng.

4.1.3 Thiết kế chi tiết kiến trúc microservices cho hệ thống



Hình 4.9: Thiết kế chi tiết kiến trúc microservices cho hệ thống

Hình 5.1 là thiết kế chi tiết kiến trúc microservices cho hệ thống của em. Nó dựa trên các gói em đã phân tích bên trên để chia thành các service riêng biệt dựa theo

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

nghiệp vụ đã phân tích. Về kiến trúc microservices của em nó sẽ gồm các phần sau:

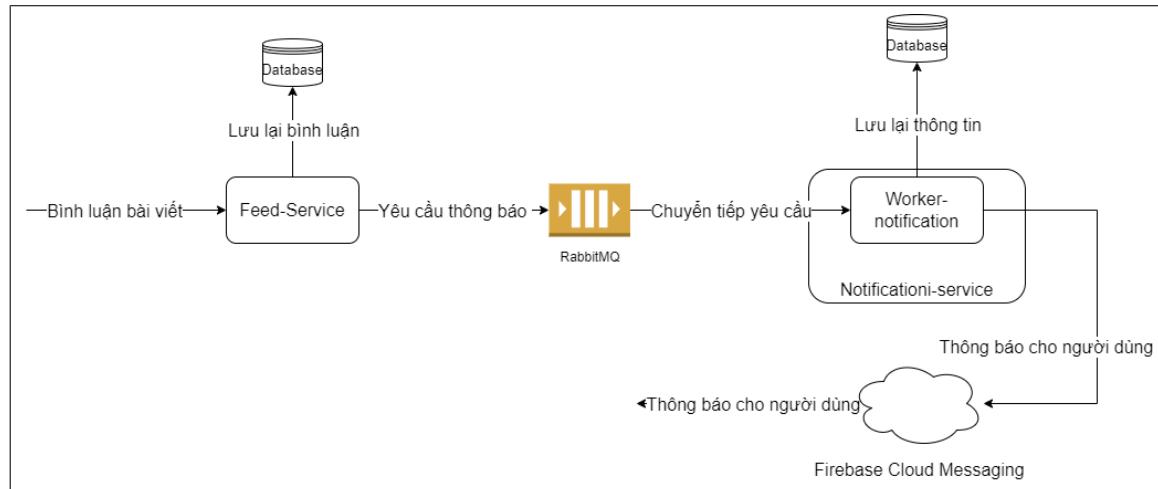
- Web: Đây là nơi người dùng thao tác thực hiện và gửi các yêu cầu đến cho server thực hiện.
- Gateway: Sẽ là nơi tiếp nhận các yêu cầu từ người dùng, sau đó điều hướng những yêu cầu đó đến các service thực hiện những yêu cầu đó. Đây cũng là nơi sử dụng bộ nhớ đệm để lưu lại kết quả trả về của các yêu cầu từ người dùng, giúp cho lần sau người dùng truy cập vào hệ thống với tốc độ nhanh hơn, tăng trải nghiệm của người dùng.
- Auth-service: đây là service dùng để đăng nhập và xác thực người dùng. Mỗi khi có bất cứ yêu cầu nào từ người dùng, những yêu cầu đó sẽ được Gateway chuyển tiếp qua service này để xác thực trước, nếu xác thực thành công sẽ cho yêu cầu thực hiện tiếp.
- Read-service: Đây là nơi các yêu cầu về đọc dữ liệu của người dùng sẽ đi qua. Nếu một yêu cầu muốn đọc dữ liệu từ nhiều service khác nhau, đây sẽ là nơi đóng gói lại tất cả dữ liệu và trả lại cho người dùng.
- Write-service: Đây là nơi các yêu cầu về ghi dữ liệu của người dùng sẽ đi qua. Nếu một yêu cầu của người dùng muốn ghi dữ liệu ở nhiều service khác nhau thì việc điều phối sẽ do service này thực hiện.
- Chat-service: Đây là nơi thực hiện những yêu cầu về nhắn tin.
- Upload-service: Đây là nơi thực hiện những yêu cầu về tải file lên.
- Notification-service: Đây là nơi thực hiện những yêu cầu về thông báo.
- Group-service: Đây là nơi thực hiện những yêu cầu về nhóm người dùng.
- User-service: Đây là nơi thực hiện những yêu cầu liên quan đến người dùng.
- Feed-service: Đây là nơi thực hiện những yêu cầu liên quan đến bài viết

Như có thể thấy ở trên hình, theo kiến trúc microservices này, mỗi service của em sẽ có một database riêng lưu trữ dữ liệu của riêng mình, riêng biệt không liên quan, không có quan hệ đến bất cứ service nào. Mọi dữ liệu muốn lấy từ service khác sẽ được tổng hợp thông qua Read service. Các service ở đây được xây dựng và tách biệt theo từng nghiệp vụ của từng service như đã được phân tích ở trên. Tuy vậy, việc xây dựng các mối quan hệ giữa các lớp hay với các bảng trong cơ sở dữ liệu quan hệ vẫn sẽ được thể hiện rõ. Những phần tiếp theo trong đồ án em sẽ thể hiện rõ hơn.

Các service bên trong kiến trúc này, em sẽ dùng API để giao tiếp giữa các service với nhau. Nhưng ngoài ra, với những trường hợp đặc biệt cần giao tiếp với nhau, em sẽ sử dụng hàng đợi để giao tiếp với nhau để tránh mất mát dữ liệu.

Sau đây em sẽ thể hiện một số luồng hoạt động bên trong kiến trúc microservices mà em xây dựng

a, Luồng bình luận và gửi thông báo cho người dùng

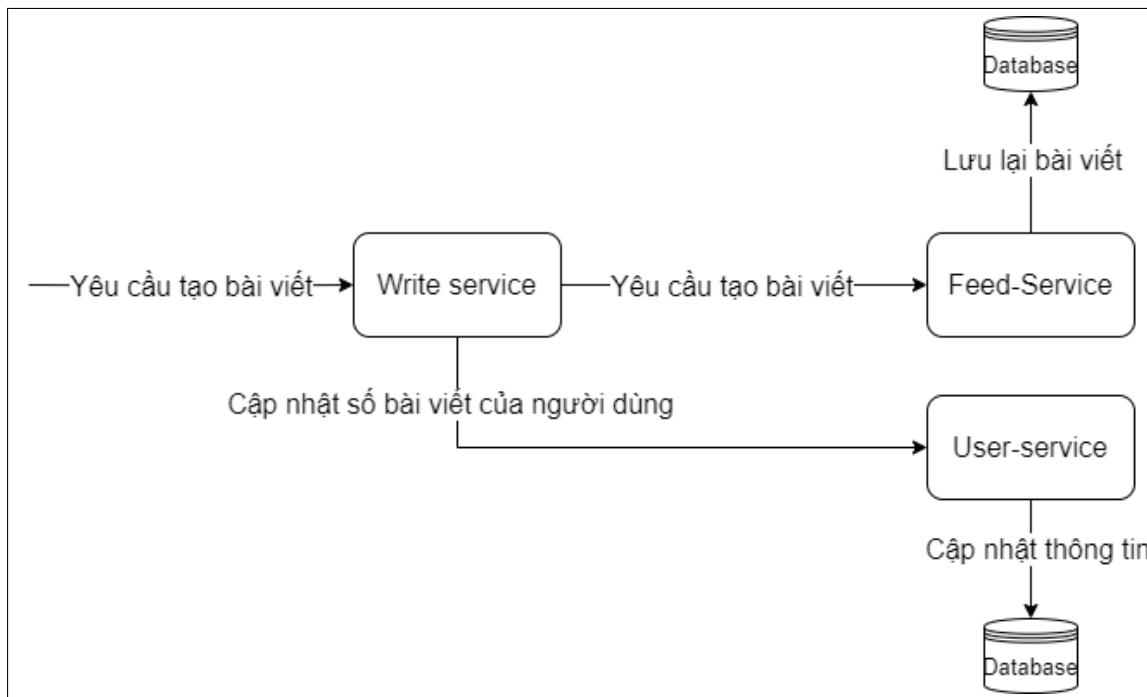


Hình 4.10: Luồng bình luận và gửi thông báo cho người dùng

Hình 4.10 thể hiện luồng bình luận và gửi thông báo cho người dùng trong hệ thống của em. Cụ thể luồng như sau:

- Đầu tiên, khi có một yêu cầu bình luận bài viết từ người dùng, yêu cầu đó sẽ được Feed service xử lý trước.
- Feed service sẽ lưu lại dữ liệu bình luận. Khi lưu dữ liệu thành công, feed service sẽ gửi một yêu cầu thông báo tới người dùng - người mà tạo ra bài viết đấy, thông qua hàng đợi, ở đây em dùng là RabbitMQ [9].
- RabbitMQ sẽ lưu lại dữ liệu tại hàng đợi, sau khi Feed service đẩy dữ liệu vào.
- Bên trong Notification service sẽ có một worker hoạt động bên trong. Worker này có tác dụng là lắng nghe từ hàng đợi, nếu trong hàng đợi có dữ liệu thì sẽ lấy ra và xử lý. Khi xử lý thành công sẽ xoá dữ liệu ở hàng đợi đó đi. Tại đây, worker đã lấy dữ liệu từ hàng đợi, sau đó xử lý dữ liệu, lưu dữ liệu và cuối cùng gửi lên FCM [10].
- Khi FCM (Firebase Cloud Messaging) nhận dữ liệu nó sẽ gửi trực tiếp thông báo đến thiết bị của người dùng và hiển thị lên giao diện cho người dùng.

b, Luồng tạo bài viết

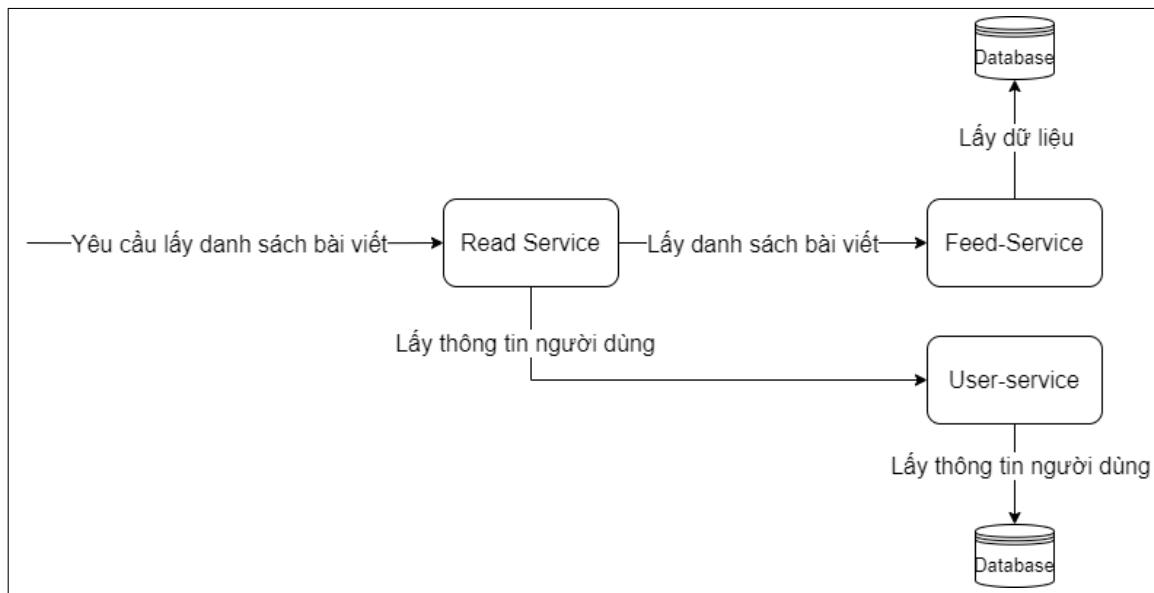


Hình 4.11: Luồng tạo bài viết

Hình 4.11 thể hiện luồng tạo bài viết cho người dùng trong hệ thống. Cụ thể luồng như sau:

- Đầu tiên khi có một yêu cầu tạo bài viết từ người dùng, yêu cầu đó sẽ được xử lý ở Write service trước. Service này sẽ thực hiện việc điều phối xử lý ở các service khác.
- Trước tiên Write service sẽ gửi yêu cầu đến Feed Service và yêu cầu lưu lại bài viết. Feed service nhận yêu cầu và lưu lại vào cơ sở dữ liệu. Sau đó, thông báo với Write service là đã lưu trữ dữ liệu thành công.
- Sau khi lưu trữ dữ liệu thành công vào Feed service. Write service sẽ cập nhật lại số lượng bài viết mà người dùng đã đăng ở trong User service.

c, Luồng lấy danh sách bài viết

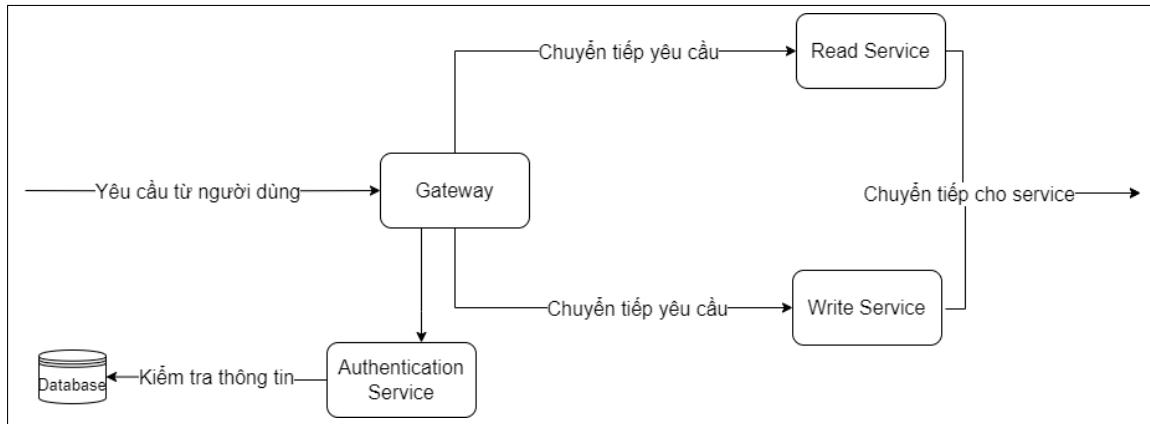


Hình 4.12: Luồng lấy danh sách bài viết

Hình 4.12 thể hiện luồng lấy danh sách bài viết ở trong hệ thống. Cụ thể luồng như sau:

- Đầu tiên có một yêu cầu lấy danh sách bài viết từ người dùng, yêu cầu đó sẽ được xử lý ở Read service trước. Service này sẽ thực hiện việc điều phối xử lý ở các service khác.
- Trước tiên Read Service này sẽ gửi yêu cầu đến Feed service và lấy danh sách bài viết từ Feed service ra.
- Sau đó với dữ liệu danh sách bài viết vừa lấy ra. Read service sẽ gọi vào User service và yêu cầu lấy những thông tin người dùng cần thiết.
- Cuối cùng, sau khi Read service lấy được những thông tin cần thiết. Nó sẽ định dạng và ánh xạ lại dữ liệu cuối cùng gửi lại về cho người dùng.

d, Luồng xác thực người dùng



Hình 4.13: Luồng xác thực người dùng

Hình 4.13 thể hiện luồng xác thực người dùng trong hệ thống. Với bất kỳ yêu cầu nào từ người dùng vào hệ thống đều sẽ bắt buộc phải xác thực, việc này để kiểm tra xem người dùng có được thực hiện yêu cầu này không. Sau đây là chi tiết về luồng này:

- Đầu tiên khi có yêu cầu từ người dùng vào hệ thống, Gateway sẽ nhận những yêu cầu từ người dùng.
- Khi nhận yêu cầu từ người dùng, đầu tiên Gateway sẽ xác thực người dùng, kiểm tra xem người dùng có được thực hiện yêu cầu này không, nó sẽ chuyển yêu cầu đến Authentication service.
- Khi Authentication service nhận được yêu cầu nó sẽ kiểm tra thông tin người dùng và gửi lại kết quả cho Gateway.
- Sau khi xác thực xong. Nếu xác thực thành công, Gateway sẽ chuyển tiếp yêu cầu cho các service bên dưới để thực hiện yêu cầu của người dùng. Nếu xác thực thất bại, nó sẽ gửi lại một thông báo lỗi cho người dùng.

e, Một số luồng khác

Sau đây là các luồng còn lại trong hệ thống. Nó được liệt kê dựa trên service điều hướng của nó và những service sẽ cần để lấy hoặc là ghi lại dữ liệu người dùng.

Tên luồng	Service trung gian	Service thực hiện logic
Bày tỏ cảm xúc	Write service	<ul style="list-style-type: none"> • Feed service: Lưu lại thông tin người thích. • Notification service: Thông báo có người bày tỏ cảm xúc.
Lấy danh sách bình luận	Read service	<ul style="list-style-type: none"> • Feed service: Lấy danh sách bình luận. • User service: Lấy danh sách người dùng bình luận.
Lấy thông tin chi tiết bài viết	Read service	<ul style="list-style-type: none"> • Feed service: Lấy thông tin bài viết. • User service: Lấy thông tin người dùng.
Xoá bài viết	Write service	<ul style="list-style-type: none"> • Feed service: Xoá dữ liệu bài viết. • User service: Cập nhật số lượng bài viết.
Xoá bình luận	Write service	<ul style="list-style-type: none"> • Feed service: Xoá dữ liệu bình luận.

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

Lấy thông tin chi tiết của người dùng	Read service	<ul style="list-style-type: none"> User service: Lấy thông tin người dùng.
Chỉnh sửa thông tin người dùng	Write service	<ul style="list-style-type: none"> User service: Cập nhật dữ liệu.
Theo dõi người dùng	Write service	<ul style="list-style-type: none"> User service: Lưu dữ liệu. Notification service: Thông báo có người theo dõi.
Lấy danh sách nhóm người dùng	Read service	<ul style="list-style-type: none"> Group service: Lấy danh sách các nhóm.
Lấy chi tiết nhóm người dùng	Read service	<ul style="list-style-type: none"> Group service: Lấy thông tin chi tiết nhóm.
Chỉnh sửa thông tin nhóm	Read service	<ul style="list-style-type: none"> Group service: Cập nhật thông tin nhóm.
Lấy danh sách tin nhắn	Read service	<ul style="list-style-type: none"> Chat service: Lấy danh sách tin nhắn.

Bảng 4.1: Bảng Mô tả một số luồng khác

Trên đó là mô tả về thiết kế hệ thống microservices của em và một số luồng cơ bản trong hệ thống microservices của mình. Nó đã thể hiện được tính tách biệt hoàn toàn và độc lập của hệ thống microservices.

4.2 Thiết kế chi tiết

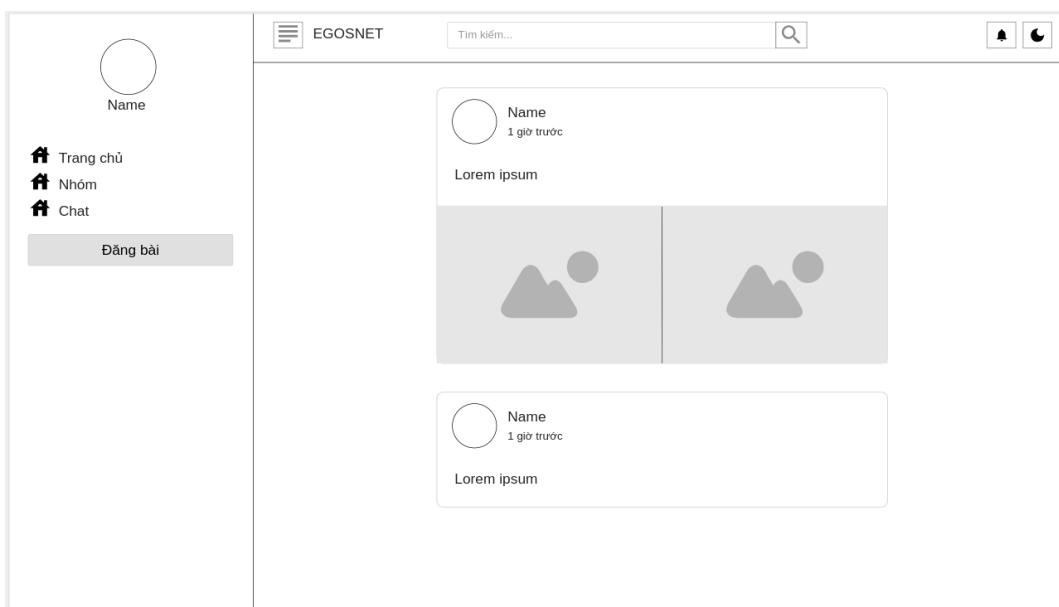
4.2.1 Thiết kế giao diện

a, Thông tin màn hình

- Độ phân giải màn hình: 1920×1080 pixel.
- Kích thước màn hình chuẩn: 23.8 inch.
- Các thành phần như button, textinput, image, line... được thống nhất trên toàn trang web.
- Màu sắc chủ đạo của trang web được thiết kế đồng nhất về tông màu, dễ quan sát và trực quan.
- Khoảng cách giữa các thành phần giao diện đồng đều, thống nhất trên toàn trang web.

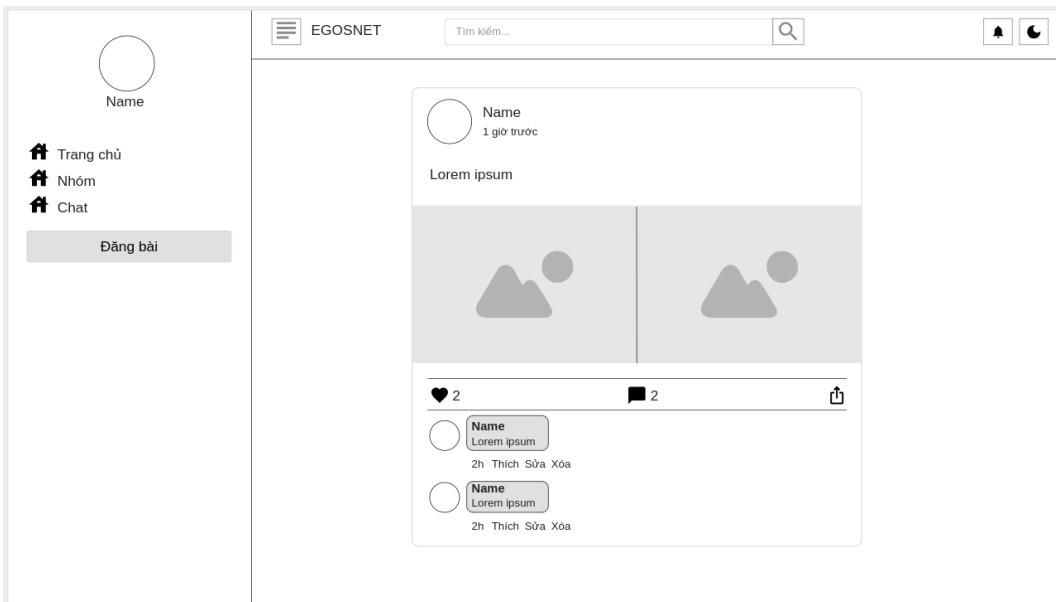
b, Hình ảnh thiết kế giao diện

Một số hình ảnh minh họa thiết kế giao diện

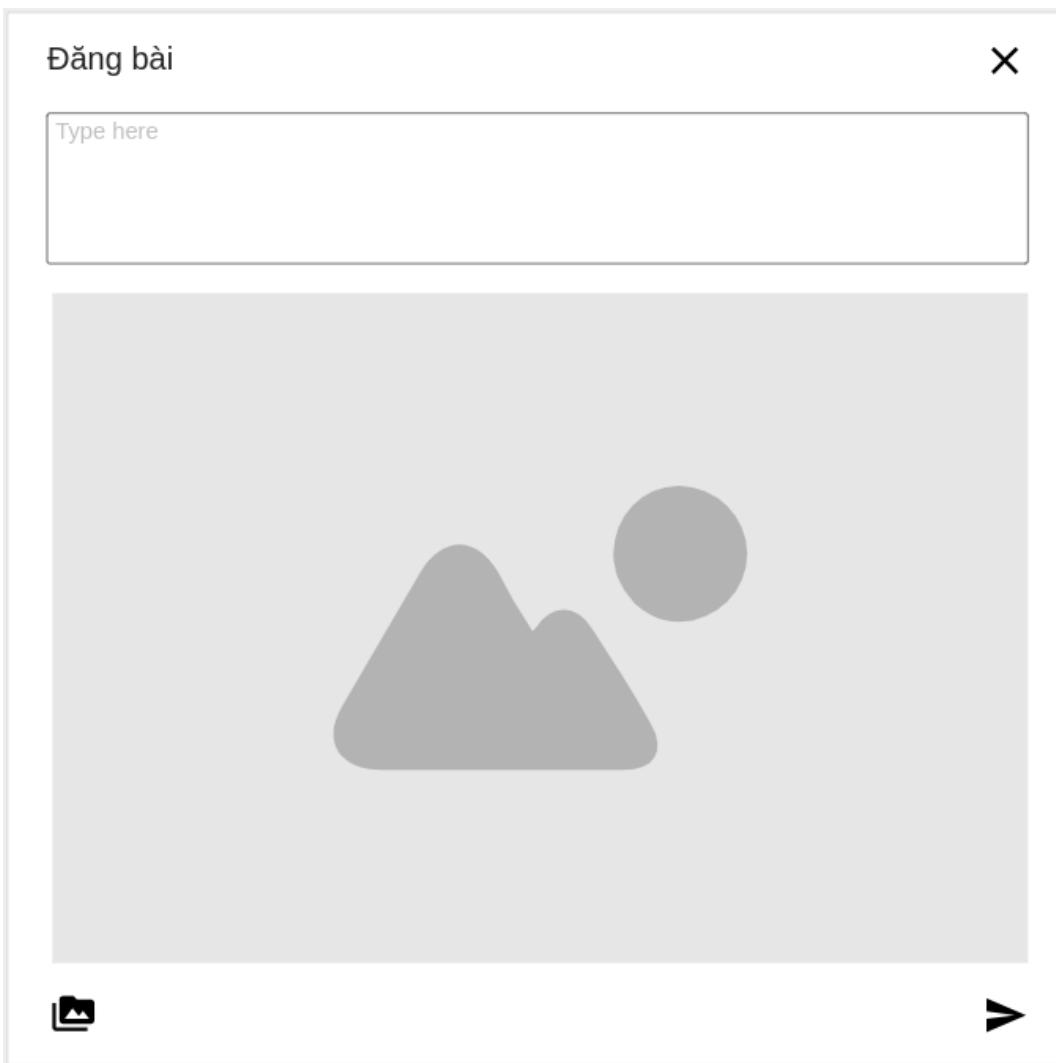


Hình 4.14: Thiết kế giao diện trang chủ

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

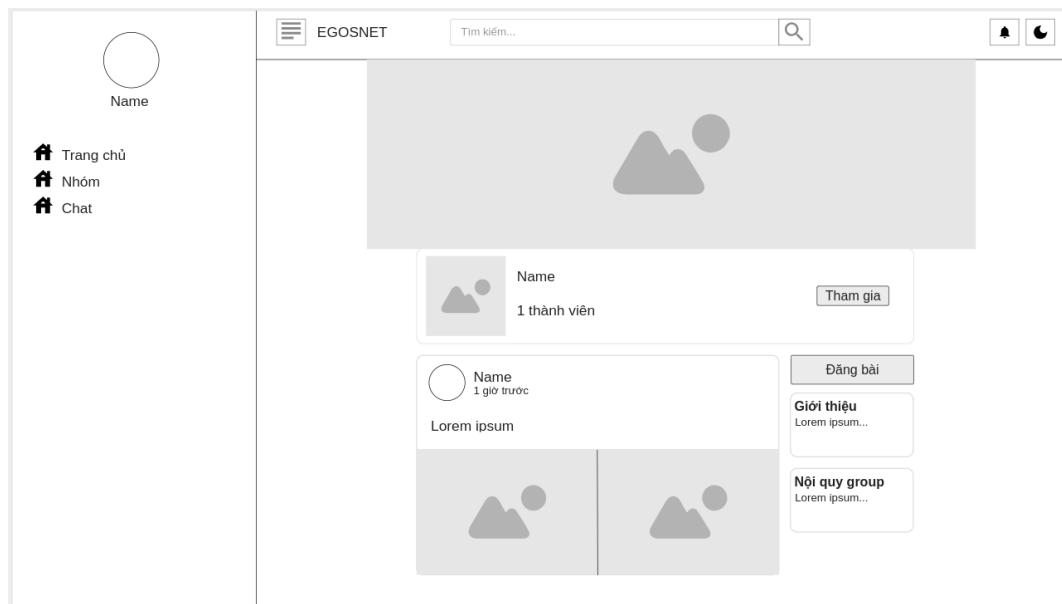


Hình 4.15: Thiết kế giao diện chi tiết bài viết

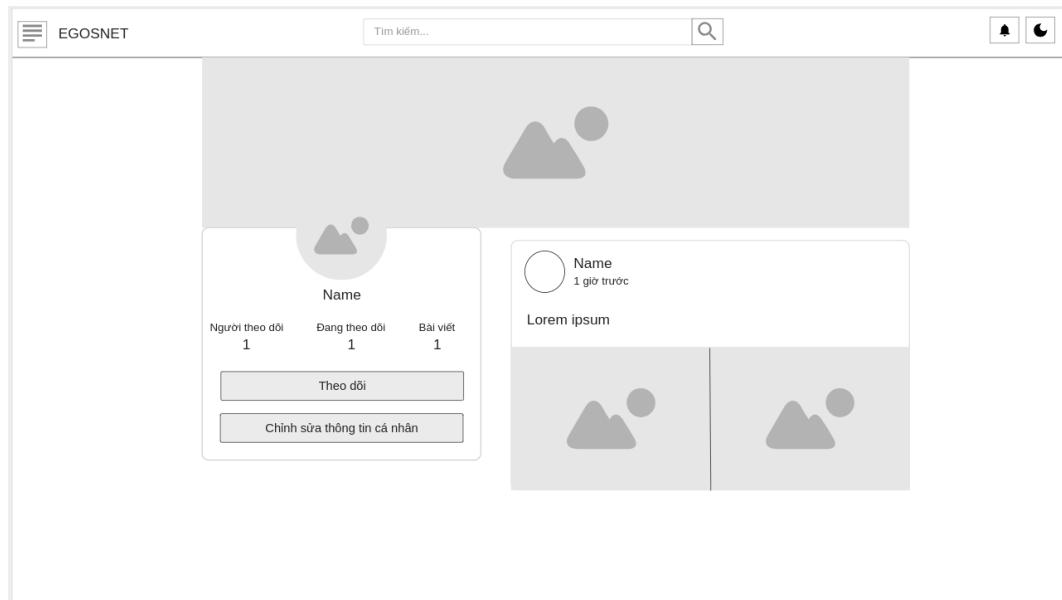


Hình 4.16: Thiết kế giao diện đăng bài

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

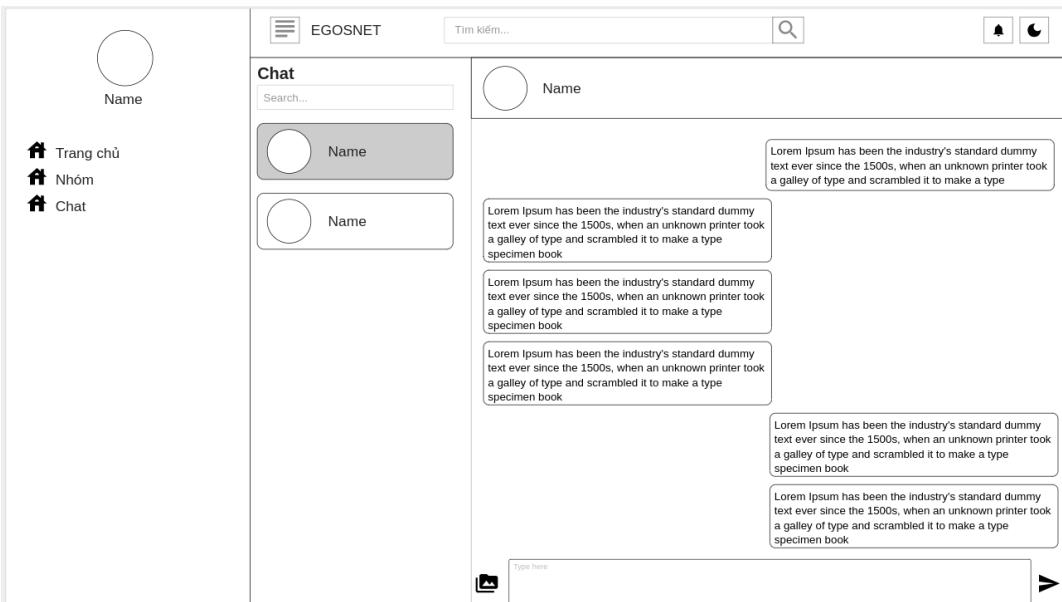


Hình 4.17: Thiết kế giao diện nhóm



Hình 4.18: Thiết kế giao diện trang cá nhân

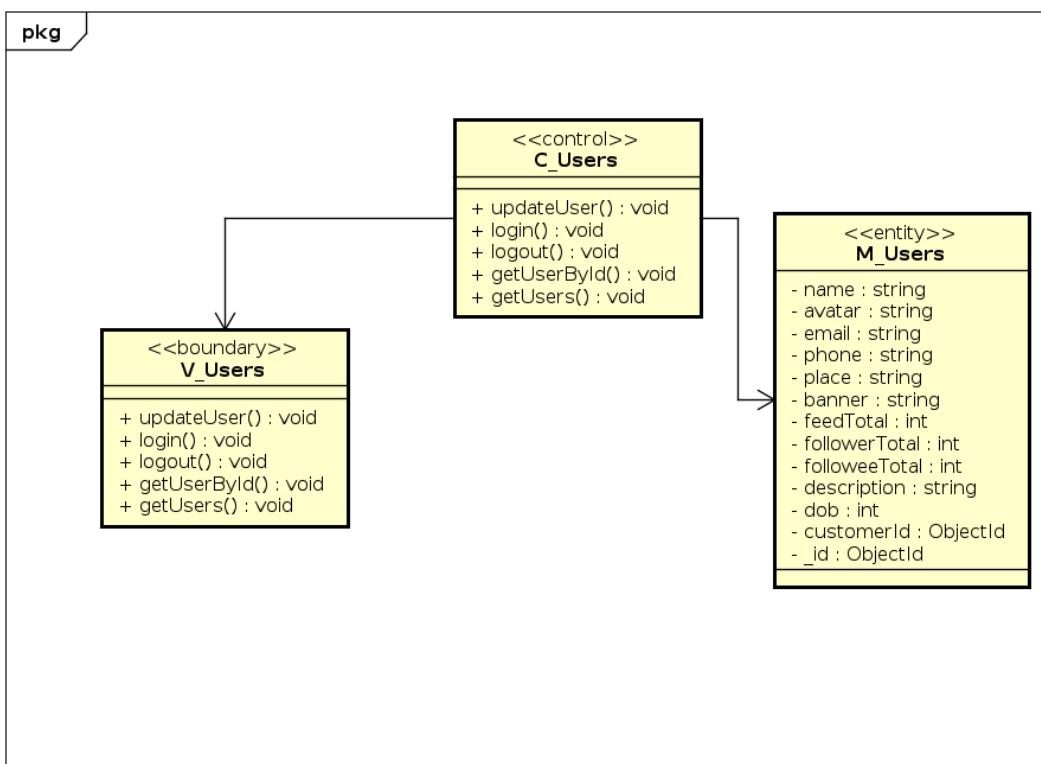
CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



Hình 4.19: Thiết kế giao diện nhắn tin

4.2.2 Thiết kế lớp

a, Chi tiết sơ đồ lớp User



Hình 4.20: Chi tiết sơ đồ lớp User

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- MUser là lớp chứa các thông tin tài khoản người dùng.

Tên thuộc tính	Mô tả
_id	ID người dùng
name	Tên người dùng
avatar	Ảnh đại diện
email	Email người dùng
phone	Số điện thoại người dùng
place	Địa chỉ người dùng
banner	Ảnh bìa
feedTotal	Tổng bài viết
followerTotal	Tổng số người theo dõi
followeeTotal	Tổng số người đang theo dõi
description	Mô tả
dob	Ngày sinh nhật
customerId	Id người dùng trong authen service

Bảng 4.2: Bảng đặc tả lớp MUser

- CUser là lớp thực thi các tác vụ liên quan đến tài khoản người dùng.

Tên phương thức	Mô tả
login	Đăng nhập và đăng ký người dùng
logout	Đăng xuất
updateUser	Cập nhật thông tin cá nhân
getUserById	Lấy thông tin chi tiết của người dùng cụ thể
getUsers	Lấy danh sách thông tin người dùng

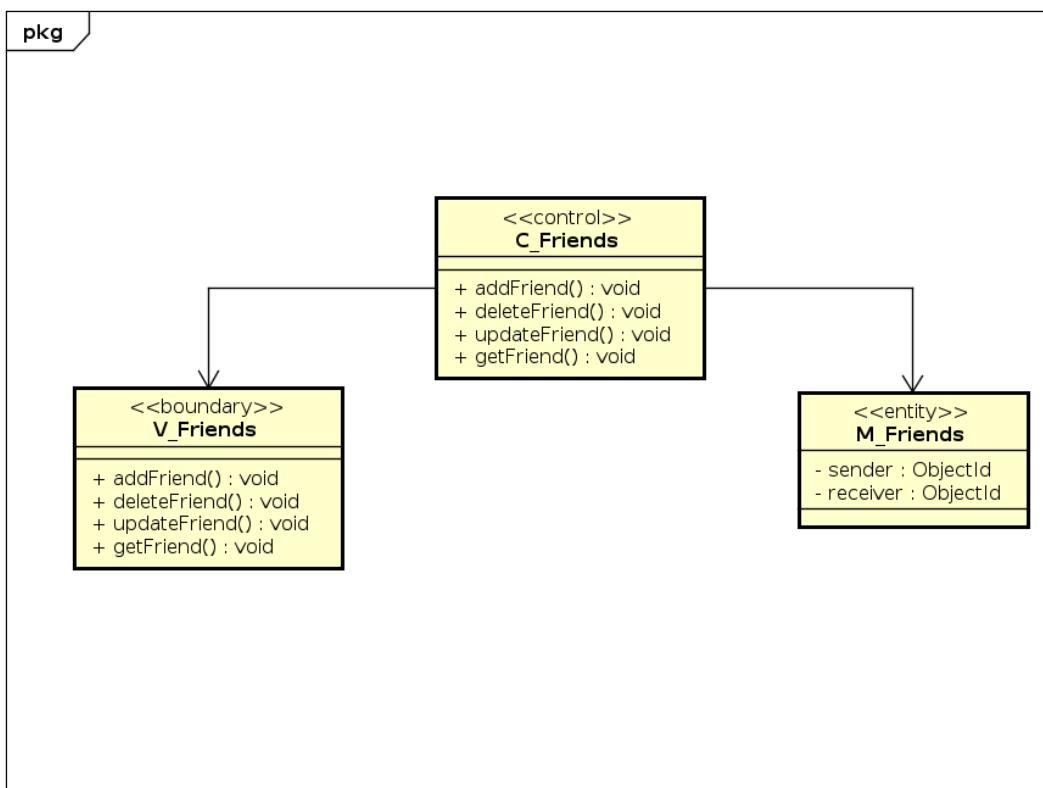
Bảng 4.3: Bảng đặc tả lớp CUser

- VUser là lớp cung cấp các tác vụ liên quan đến tài khoản người dùng.

Tên phương thức	Mô tả
login	Đăng nhập và đăng ký người dùng
logout	Đăng xuất
updateUser	Cập nhật thông tin cá nhân
getUserById	Lấy thông tin chi tiết của người dùng cụ thể
getUsers	Lấy danh sách thông tin người dùng

Bảng 4.4: Bảng đặc tả lớp VUser

b, Chi tiết sơ đồ lớp Friend



Hình 4.21: Chi tiết sơ đồ lớp Friend

- MFriend là lớp chứa các thông tin về bạn bè.

Tên thuộc tính	Mô tả
_id	ID người dùng
sender	Người theo dõi
receiver	Người được theo dõi

Bảng 4.5: Bảng đặc tả lớp MFriend

- CFriend là lớp thực thi các tác vụ liên quan đến bạn bè.

Tên phương thức	Mô tả
addFriend	Theo dõi người dùng
deleteFriend	Bỏ theo dõi người dùng
getFollowers	Lấy danh sách người mà theo dõi người dùng
getFollowees	Lấy danh sách người mà người dùng đang theo dõi

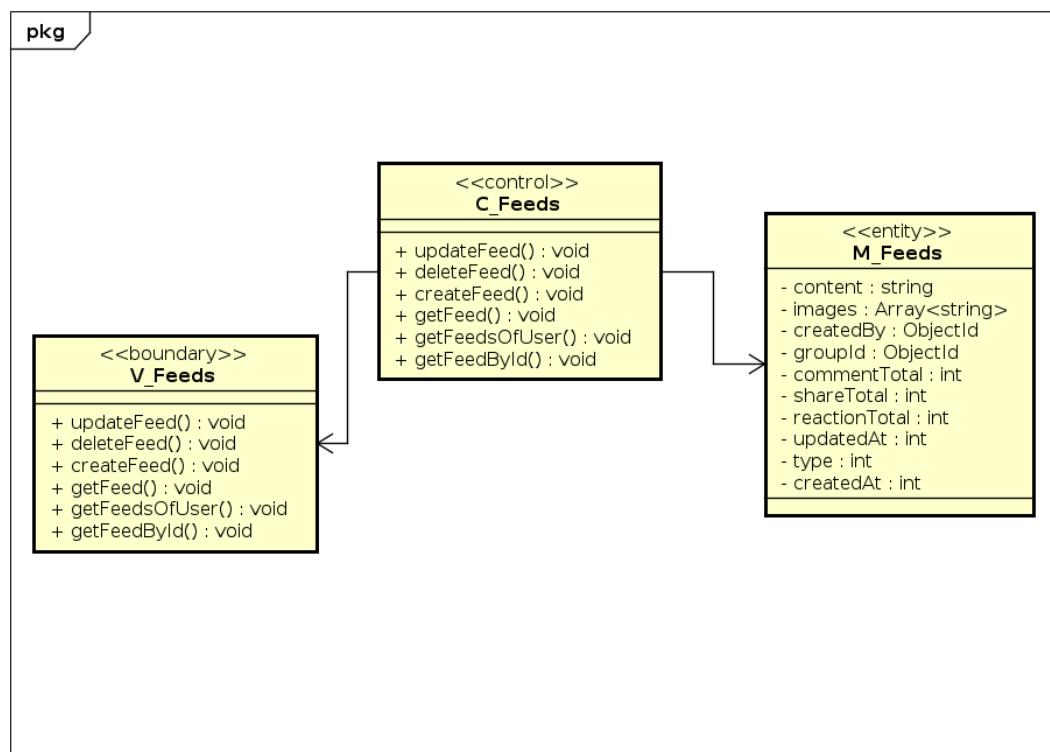
Bảng 4.6: Bảng đặc tả lớp CFriend

- VFriend là lớp cung cấp các tác vụ liên quan đến bạn bè.

Tên phương thức	Mô tả
addFriend	Theo dõi người dùng
deleteFriend	Bỏ theo dõi người dùng
getFollowers	Lấy danh sách người mà theo dõi người dùng
getFollowees	Lấy danh sách người mà người dùng đang theo dõi

Bảng 4.7: Bảng đặc tả lớp VFriend

c, Chi tiết sơ đồ lớp Feed



Hình 4.22: Chi tiết sơ đồ lớp Feed

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- MFeed là lớp chứa các thông tin về bài viết.

Tên thuộc tính	Mô tả
_id	ID bài viết
content	Nội dung bài viết
images	Ảnh bài viết
createdBy	Id người dùng tạo bài viết
groupId	Id nhóm mà bài viết được tạo
commentTotal	Tổng số bình luận
reactionTotal	Tổng số lượt thích
updatedAt	Ngày cập nhật
type	Loại bài viết
createdAt	Ngày tạo bài viết

Bảng 4.8: Bảng đặc tả lớp MFeed

- CFeed là lớp thực thi các tác vụ liên quan đến bài viết.

Tên phương thức	Mô tả
updateFeed	Cập nhật bài viết
deleteFeed	Xóa bài viết
createFeed	Tạo bài viết
getFeed	Lấy danh sách bài viết
getFeedsOfUser	Lấy danh sách bài viết của người dùng
getFeedById	Lấy chi tiết bài viết

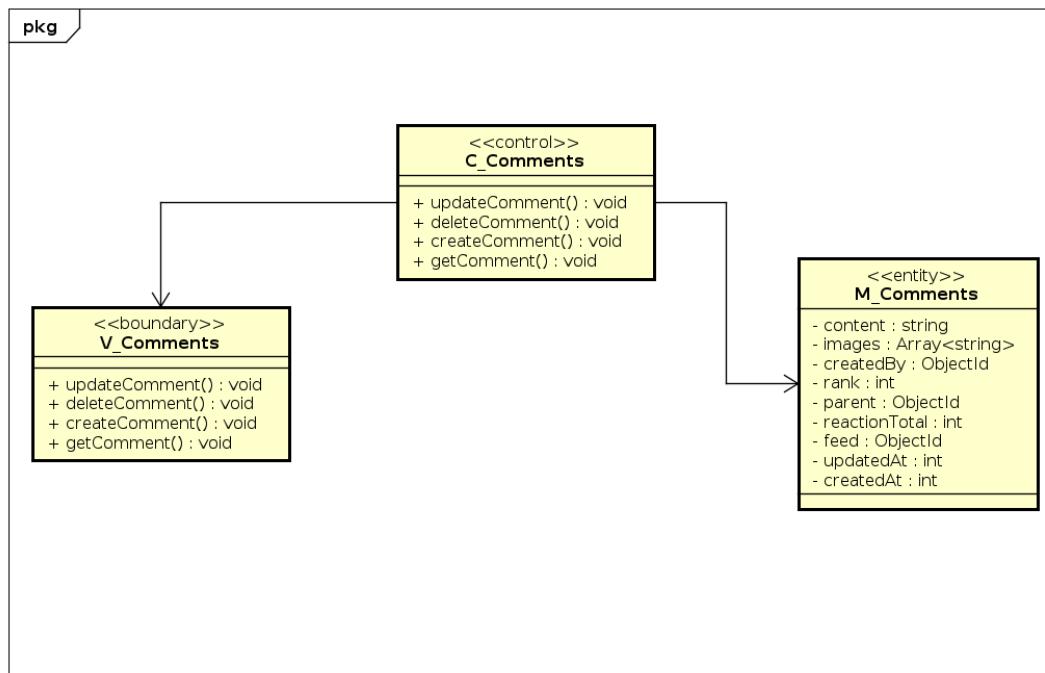
Bảng 4.9: Bảng đặc tả lớp CFeed

- VFeed là lớp cung cấp các tác vụ liên quan đến bài viết.

Tên phương thức	Mô tả
updateFeed	Cập nhật bài viết
deleteFeed	Xóa bài viết
createFeed	Tạo bài viết
getFeed	Lấy danh sách bài viết
getFeedsOfUser	Lấy danh sách bài viết của người dùng
getFeedById	Lấy chi tiết bài viết

Bảng 4.10: Bảng đặc tả lớp VFeed

d, Chi tiết sơ đồ lớp Comment



Hình 4.23: Chi tiết sơ đồ lớp Comment

- MComment là lớp chứa các thông tin về bình luận.

Tên thuộc tính	Mô tả
_id	ID bài viết
content	Nội dung bình luận
images	Ảnh bình luận
createdBy	Id người dùng tạo bình luận
parent	Id bình luận cha
reactionTotal	Tổng số lượt thích
updatedAt	Ngày cập nhật
createdAt	Ngày tạo bài viết

Bảng 4.11: Bảng đặc tả lớp MComment

- CComment là lớp thực thi các tác vụ liên quan đến bình luận.

Tên phương thức	Mô tả
updateComment	Cập nhật bình luận
deleteComment	Xóa bình luận
createComment	Tạo bình luận
getComment	Lấy danh sách bình luận

Bảng 4.12: Bảng đặc tả lớp CComment

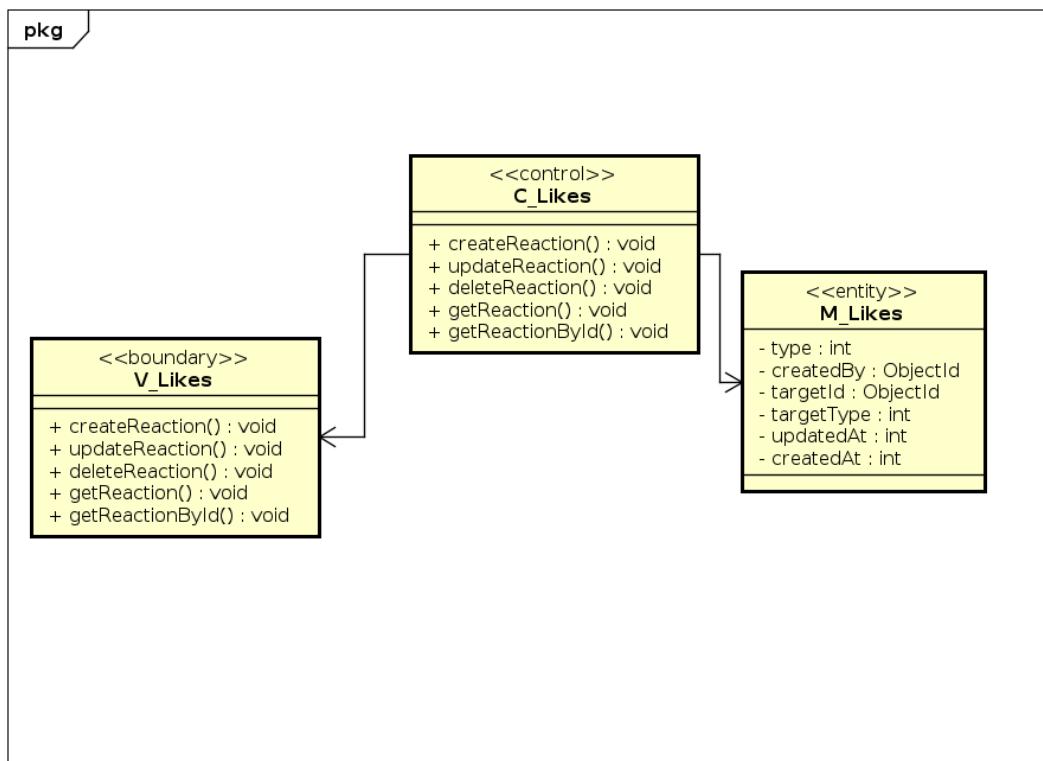
CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- VComment là lớp cung cấp các tác vụ liên quan đến bình luận.

Tên phương thức	Mô tả
updateComment	Cập nhật bình luận
deleteComment	Xóa bình luận
createComment	Tạo bình luận
getComment	Lấy danh sách bình luận

Bảng 4.13: Bảng đặc tả lớp VComment

e, Chi tiết sơ đồ lớp Reaction



Hình 4.24: Chi tiết sơ đồ lớp Reaction

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- MLike là lớp chứa các thông tin về thể hiện cảm xúc.

Tên thuộc tính	Mô tả
_id	ID cảm xúc
type	Loại cảm xúc
targetId	Đối tượng(bài viết hoặc bình luận) được thả cảm xúc
createdBy	Id người dùng thể hiện cảm xúc
targetType	Loại đối tượng được thể hiện cảm xúc
updatedAt	Ngày cập nhật
createdAt	Ngày tạo

Bảng 4.14: Bảng đặc tả lớp MLike

- CLike là lớp thực thi các tác vụ liên quan đến thể hiện cảm xúc.

Tên phương thức	Mô tả
createReaction	Thể hiện cảm xúc
updateReaction	Chỉnh sửa cảm xúc
deleteReaction	Xóa cảm xúc
getReactionById	Lấy chi tiết cảm xúc
getReaction	Lấy danh sách người dùng thể hiện cảm xúc

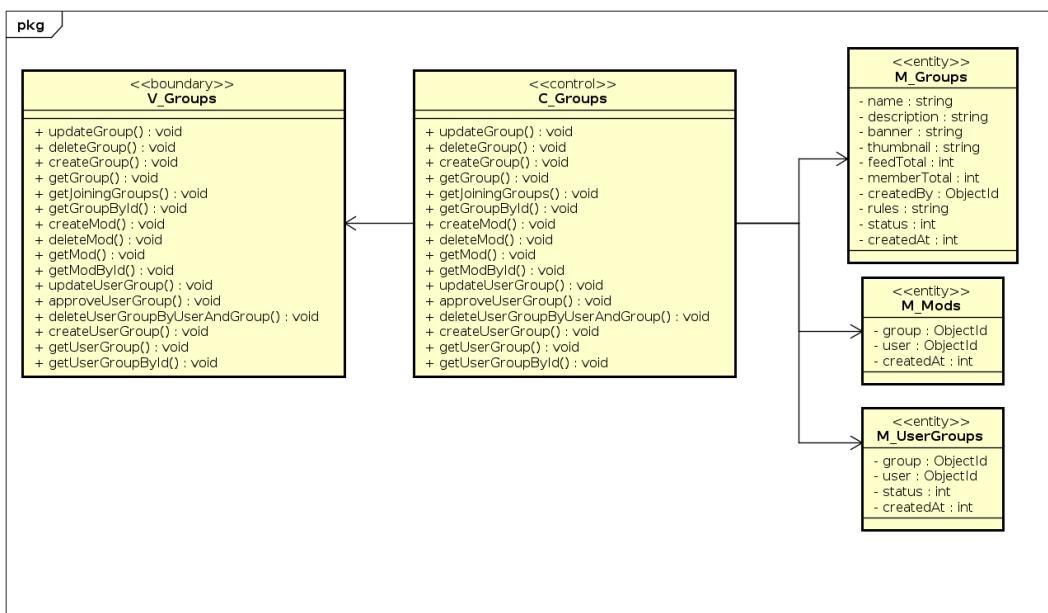
Bảng 4.15: Bảng đặc tả lớp CLike

- VLike là lớp cung cấp các tác vụ liên quan đến bình luận.

Tên phương thức	Mô tả
createReaction	Thể hiện cảm xúc
updateReaction	Chỉnh sửa cảm xúc
deleteReaction	Xóa cảm xúc
getReactionById	Lấy chi tiết cảm xúc
getReaction	Lấy danh sách người dùng thể hiện cảm xúc

Bảng 4.16: Bảng đặc tả lớp VLike

f, Chi tiết sơ đồ lớp Group



Hình 4.25: Chi tiết sơ đồ lớp Group

- MGroup là lớp chứa các thông tin về nhóm.

Tên thuộc tính	Mô tả
_id	ID nhóm
name	Tên nhóm
description	Mô tả nhóm
banner	Ảnh bìa
thumbnail	Ảnh nhóm
feedTotal	Tổng bài viết
memberTotal	Tổng thành viên
createdBy	Người dùng tạo nhóm
rules	Nội quy nhóm
status	Trạng thái của nhóm
createdAt	Ngày tạo nhóm

Bảng 4.17: Bảng đặc tả lớp MGroup

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- MMod là lớp chứa các thông tin về quản trị viên.

Tên thuộc tính	Mô tả
_id	ID mod
group	Id nhóm
user	Id người dùng
createdAt	Ngày tạo mod

Bảng 4.18: Bảng đặc tả lớp MMod

- MUserGroups là lớp chứa các thông tin về người tham gia nhóm.

Tên thuộc tính	Mô tả
_id	ID userGroup
group	Id nhóm
user	Id người dùng
status	Trạng thái
createdAt	Ngày tạo mod

Bảng 4.19: Bảng đặc tả lớp MUserGroups

- CGroup là lớp thực thi các tác vụ liên quan đến nhóm.

Tên phương thức	Mô tả
updateGroup	Chỉnh sửa thông tin nhóm
deleteGroup	Xóa nhóm
createGroup	Tạo nhóm
getGroup	Lấy danh sách nhóm
getJoiningGroups	Lấy danh sách nhóm mà người dùng đã tham gia
getGroupById	Lấy thông tin chi tiết của group
createMod	Thêm mod
deleteMod	Xóa mod
getMod	Lấy danh sách mod
getModById	Lấy thông tin chi tiết mod
approveUserGroup	Phê duyệt người dùng tham gia nhóm
deleteUserGroup	Kích người dùng khỏi nhóm
createUserGroup	Tham gia nhóm
getUserGroup	Lấy danh sách người dùng trong nhóm

Bảng 4.20: Bảng đặc tả lớp CGroup

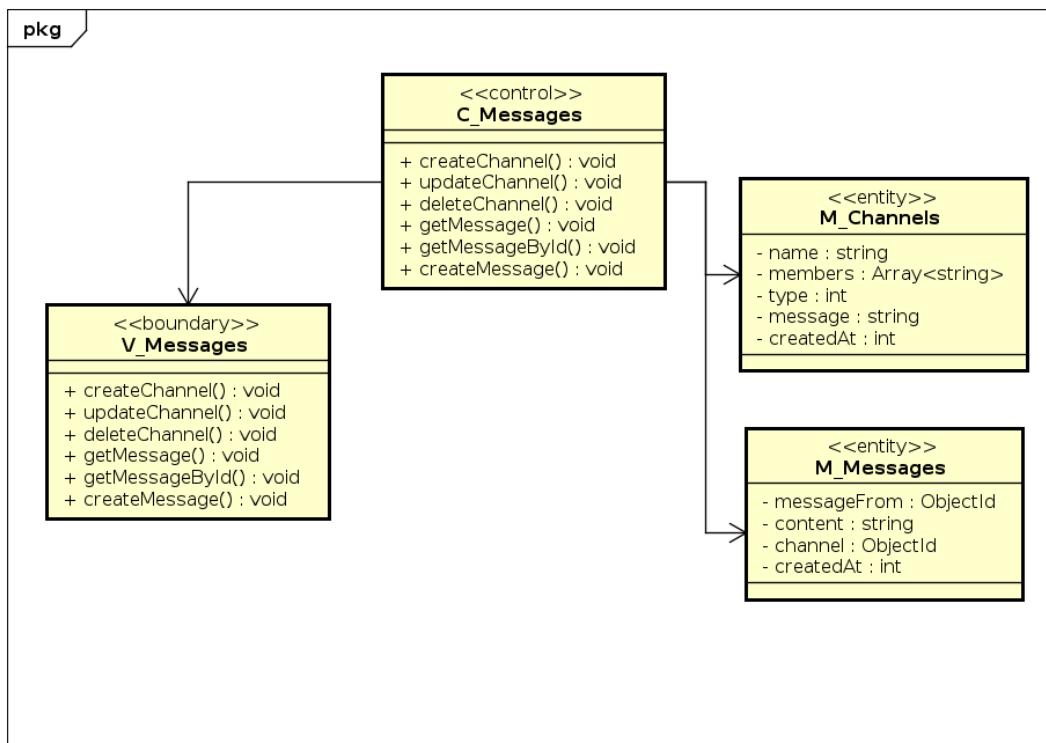
CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- VGroup là lớp cung cấp các tác vụ liên quan đến nhóm.

Tên phương thức	Mô tả
updateGroup	Chỉnh sửa thông tin nhóm
deleteGroup	Xóa nhóm
createGroup	Tạo nhóm
getGroup	Lấy danh sách nhóm
getJoiningGroups	Lấy danh sách nhóm mà người dùng đã tham gia
getGroupById	Lấy thông tin chi tiết của group
createMod	Thêm mod
deleteMod	Xóa mod
getMod	Lấy danh sách mod
getModById	Lấy thông tin chi tiết mod
approveUserGroup	Phê duyệt người dùng tham gia nhóm
deleteUserGroup	Kích người dùng khỏi nhóm
createUserGroup	Tham gia nhóm
getUserGroup	Lấy danh sách người dùng trong nhóm

Bảng 4.21: Bảng đặc tả lớp VGroup

g, Chi tiết sơ đồ lớp Message



Hình 4.26: Chi tiết sơ đồ lớp Message

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- MChannel là lớp chứa các thông tin về cuộc hội thoại.

Tên thuộc tính	Mô tả
_id	ID cuộc hội thoại
name	Tên cuộc hội thoại
members	Thành viên của cuộc hội thoại
type	Loại hội thoại
createdAt	Ngày tạo

Bảng 4.22: Bảng đặc tả lớp MChannel

- MMessage là lớp chứa các thông tin về tin nhắn.

Tên thuộc tính	Mô tả
_id	ID tin nhắn
messageFrom	Người gửi
content	nội dung tin nhắn
channel	Id cuộc hội thoại
createdAt	Ngày tạo

Bảng 4.23: Bảng đặc tả lớp MMessage

- CMessage là lớp thực thi các tác vụ liên quan đến tin nhắn.

Tên phương thức	Mô tả
createChannel	Tạo cuộc hội thoại
updateChannel	Chỉnh sửa thông tin cuộc hội thoại
deleteChannel	Xóa cuộc hội thoại
getMessage	Lấy danh sách tin nhắn
createMessage	Gửi tin nhắn
getMessageById	Lấy thông tin chi tiết tin nhắn

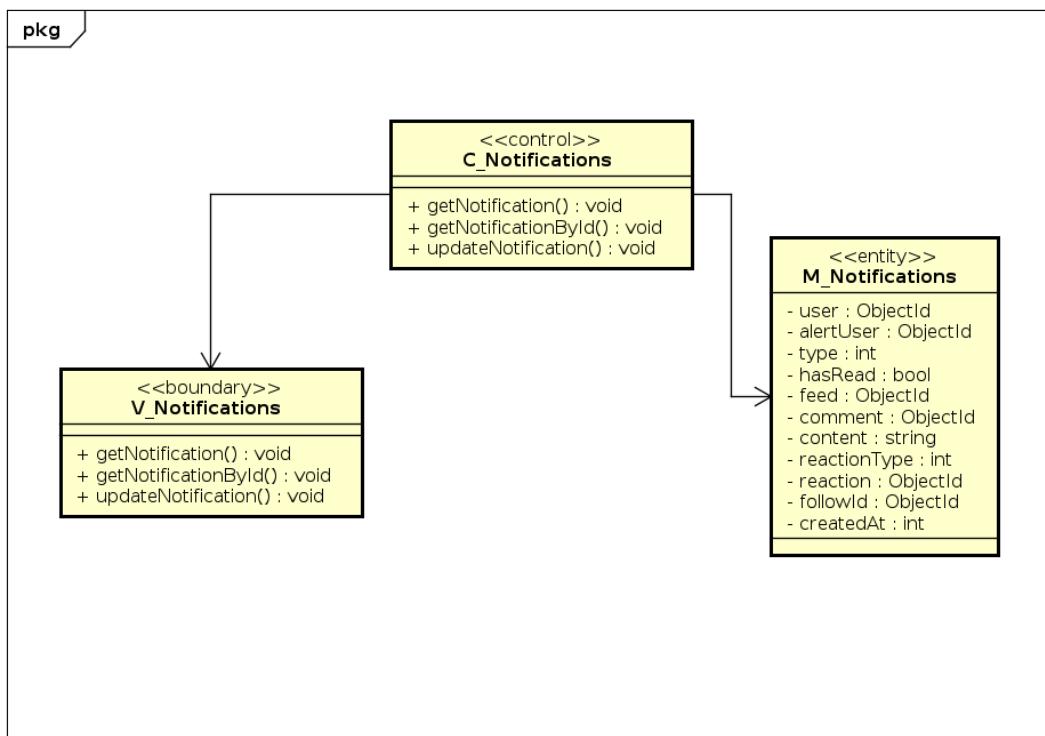
Bảng 4.24: Bảng đặc tả lớp CMessage

- VMMessage là lớp cung cấp các tác vụ liên quan đến tin nhắn.

Tên phương thức	Mô tả
createChannel	Tạo cuộc hội thoại
updateChannel	Chỉnh sửa thông tin cuộc hội thoại
deleteChannel	Xóa cuộc hội thoại
getMessage	Lấy danh sách tin nhắn
createMessage	Gửi tin nhắn
getMessageById	Lấy thông tin chi tiết tin nhắn

Bảng 4.25: Bảng đặc tả lớp VMMessage

h, Chi tiết sơ đồ lớp Notification



Hình 4.27: Chi tiết sơ đồ lớp Notification

- MNotification là lớp chứa các thông tin về thể hiện thông báo.

Tên thuộc tính	Mô tả
_id	ID thông báo
user	ID người dùng
alertUser	ID người dùng được thông báo
type	Loại thông báo
hasRead	Kiểm tra xem thông báo đã đọc hay chưa
feed	ID của bài viết
comment	ID của bình luận
content	Nội dung của thông báo
reactionType	Loại cảm xúc
reaction	ID của cảm xúc
followId	ID của người dùng theo dõi
createdAt	Ngày tạo

Bảng 4.26: Bảng đặc tả lớp MNotification

- CNotification là lớp thực thi các tác vụ liên quan đến thông báo.

Tên phương thức	Mô tả
getNotification	Lấy danh sách thông báo
getNotificationById	Lấy thông tin chi tiết thông báo
updateNotification	Cập nhật thông tin thông báo

Bảng 4.27: Bảng đặc tả lớp CNotification

- VNotification là lớp cung cấp các tác vụ liên quan đến thông báo.

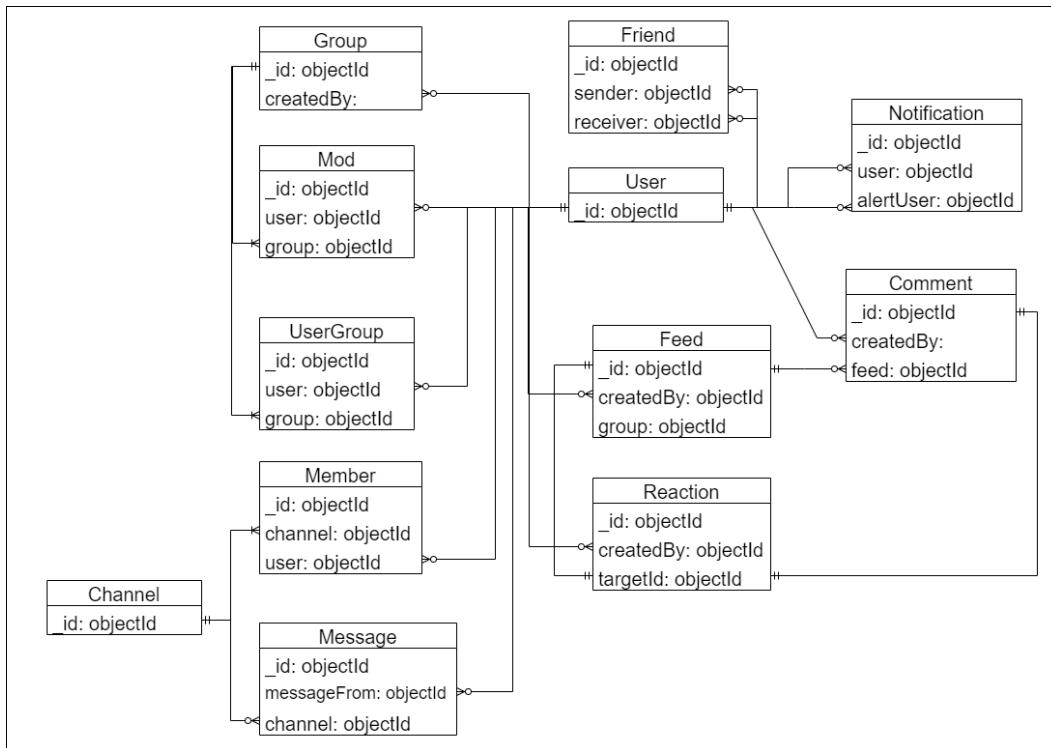
Tên phương thức	Mô tả
getNotification	Lấy danh sách thông báo
getNotificationById	Lấy thông tin chi tiết thông báo
updateNotification	Cập nhật thông tin thông báo

Bảng 4.28: Bảng đặc tả lớp VNotification

4.2.3 Thiết kế cơ sở dữ liệu

a, Sơ đồ quan hệ giữa các bảng dữ liệu

Hình vẽ dưới đây là sơ đồ thể hiện mối quan hệ của các bảng dữ liệu với nhau. Hình vẽ dưới đây bao gồm các bảng và ở mỗi bảng chỉ bao gồm một số trường có sự liên kết trực tiếp với bảng khác. Sau khi phân tích vẽ ra sơ đồ quan hệ giữa các bảng dữ liệu, em đã lựa chọn hệ quản trị cơ sở dữ liệu MongoDB cho đồ án này. Phần chi tiết các trường của bảng dữ liệu em sẽ trình bày ở bên dưới.



Hình 4.28: Sơ đồ thực thể liên kết

b, Chi tiết các bảng dữ liệu

- Bảng User

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID người dùng
2	name	String		Tên người dùng
3	avatar	String		Ảnh đại diện người dùng
4	email	String	x	Email người dùng
5	phone	String	x	Số điện thoại người dùng
6	isLocked	Number		Trạng thái khóa tài khoản
7	place	String	x	Địa chỉ người dùng
8	banner	String	x	Ảnh bìa của người dùng
10	feedTotal	Number		Tổng bài viết
11	followerTotal	Number		Tổng số người theo dõi
12	followeeTotal	String		Tổng số người đang theo dõi người dùng
13	description	String	x	Mô tả
14	dob	Number	x	Ngày sinh nhật
15	createdAt	Number		Ngày tạo

Bảng 4.29: Bảng User

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- Bảng Friend

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID người dùng
2	sender	ObjectId		Người theo dõi
3	receiver	ObjectId		Người được theo dõi

Bảng 4.30: Bảng Friend

- Bảng Feed

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID bài viết
2	content	String	x	Nội dung bài viết
3	images	Array<String>	x	Ảnh bài viết
4	createdBy	ObjectId		Id người dùng tạo bài viết
5	groupId	ObjectId	x	Id nhóm mà bài viết được tạo
6	commentTotal	Number		Tổng số bình luận
7	reactionTotal	Number		Tổng số lượt thích
8	updatedAt	Number	x	Ngày cập nhật
9	type	Number		Loại bài viết
10	createdAt	Number		Ngày tạo bài viết

Bảng 4.31: Bảng Feed

- Bảng Comment

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID bài viết
2	content	String	x	Nội dung bình luận
3	images	Array<String>	x	Ảnh bình luận
4	createdBy	ObjectId		Id người dùng tạo bình luận
5	parent	ObjectId	x	Id bình luận cha
6	reactionTotal	Number		Tổng số lượt thích
7	updatedAt	Number		Ngày cập nhật
8	createdAt	Number	x	Ngày tạo bài viết

Bảng 4.32: Bảng Comment

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- Bảng Reaction

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID cảm xúc
2	type	Number		Loại cảm xúc
3	targetId	ObjectId		Đối tượng(bài viết hoặc bình luận) được thả cảm xúc
4	createdBy	ObjectId		Id người dùng thể hiện cảm xúc
5	targetType	Number		Loại đối tượng được thể hiện cảm xúc
6	updatedAt	Number	x	Ngày cập nhật
7	createdAt	Number		Ngày tạo

Bảng 4.33: Bảng Reaction

- Bảng Group

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID nhóm
2	name	String		Tên nhóm
3	description	String		Mô tả nhóm
4	banner	String	x	Ảnh bìa
5	thumbnail	String	x	Ảnh nhóm
6	feedTotal	Number		Tổng bài viết
7	memberTotal	Number		Tổng thành viên
8	createdBy	ObjectId		Người dùng tạo nhóm
9	rules	String	x	Nội quy nhóm
10	status	Number		Trạng thái của nhóm
11	createdAt	Number		Ngày tạo nhóm

Bảng 4.34: Bảng Group

- Bảng Mod

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID mod
2	group	ObjectId		Id nhóm
3	user	ObjectId		Id người dùng
4	createdAt	Number		Ngày tạo mod

Bảng 4.35: Bảng Mod

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- Bảng UserGroup

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID userGroup
2	group	ObjectId		Id nhóm
3	user	ObjectId		Id người dùng
4	status	Number		Trạng thái
5	createdAt	Number		Ngày tạo mod

Bảng 4.36: Bảng UserGroup

- Bảng Channel

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID cuộc hội thoại
2	name	String	x	Tên cuộc hội thoại
3	type	Number		Loại hội thoại
4	createdAt	Number		Ngày tạo

Bảng 4.37: Bảng Channel

- Bảng Member

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID member
2	channel	ObjectId	x	Id cuộc hội thoại
3	user	ObjectId		Id người dùng

Bảng 4.38: Bảng Member

- Bảng Message

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID tin nhắn
2	messageFrom	ObjectId		Người gửi
3	content	String		nội dung tin nhắn
4	channel	ObjectId		Id cuộc hội thoại
5	createdAt	Number		Ngày tạo

Bảng 4.39: Bảng Message

CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

- Bảng Notification

STT	Tên thuộc tính	Kiểu dữ liệu	NULL	Mô tả
1	_id	ObjectId		ID thông báo
2	user	ObjectId		ID người dùng
3	alertUser	ObjectId		ID người dùng được thông báo
4	type	Number		Loại thông báo
5	hasRead	Number		Kiểm tra xem thông báo đã đọc hay chưa
5	feed	ObjectId	x	ID của bài viết
6	comment	ObjectId	x	ID của bình luận
7	content	String		Nội dung của thông báo
8	reactionType	String	x	Loại cảm xúc
9	reaction	ObjectId	x	ID của cảm xúc
10	followId	ObjectId	x	ID của người dùng theo dõi
11	createdAt	Number		Ngày tạo

Bảng 4.40: Bảng Notification

4.3 Xây dựng ứng dụng

4.3.1 Thư viện và công cụ sử dụng

Mục đích	Công cụ	Địa chỉ URL
IDE lập trình	WebStorm v2023.1.3	https://www.jetbrains.com/webstorm/
Vue.js Framework lập trình website	Vue.js Framework v3.3.4	https://vuejs.org/
CSS framework	Windicss v3.5.6	https://windicss.org/
Framework NodeJS lập trình server	Express v4.18.2	https://expressjs.com/
Hệ quản trị cơ sở dữ liệu	MongoDB v7.0	https://www.mongodb.com/
ORM cho NodeJS với MongoDB	Mongoose v7.4.0	https://www.mongodb.com/
Công cụ kiểm thử API	Postman	https://www.postman.com/
Deploy	Docker	https://www.docker.com/
Nhà cung cấp dịch vụ Cloud Hosting	Google Cloud	https://cloud.google.com/

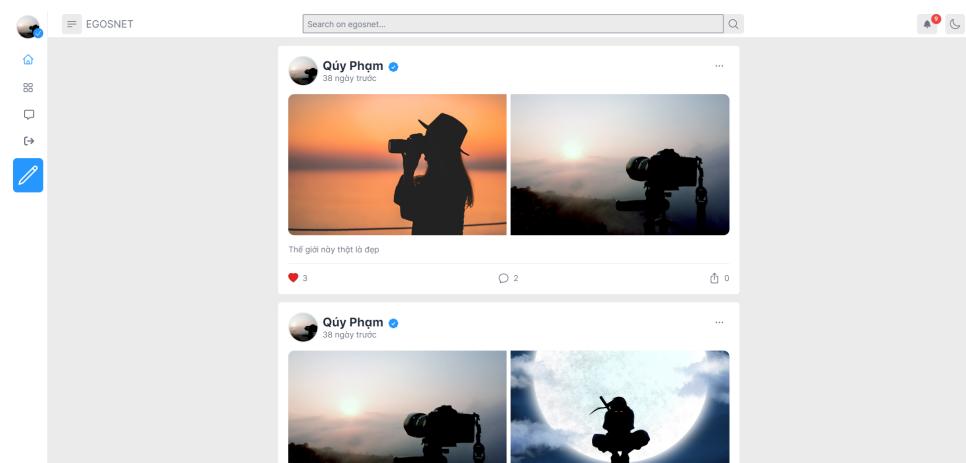
Bảng 4.41: Danh sách thư viện và công cụ sử dụng

4.3.2 Kết quả đạt được

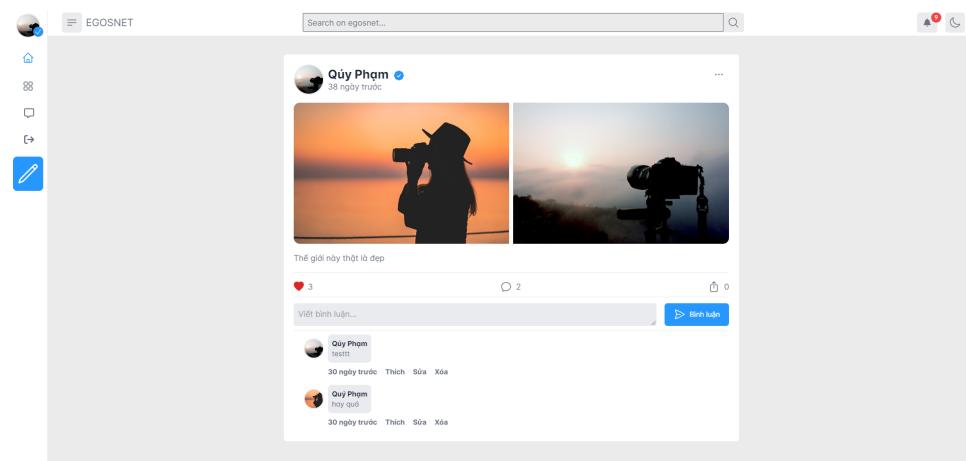
Một số hình ảnh minh họa các chức năng chính:



Hình 4.29: Giao diện màn hình đăng nhập cho người dùng cuối

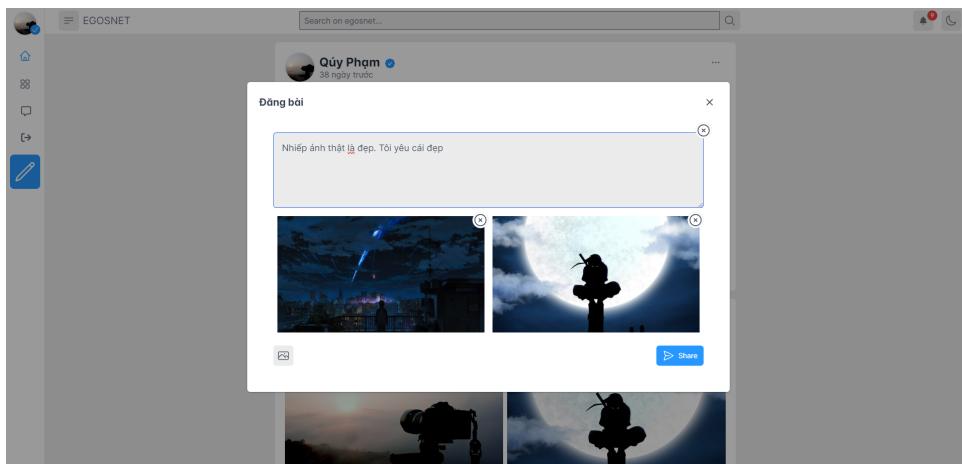


Hình 4.30: Giao diện màn hình trang chủ

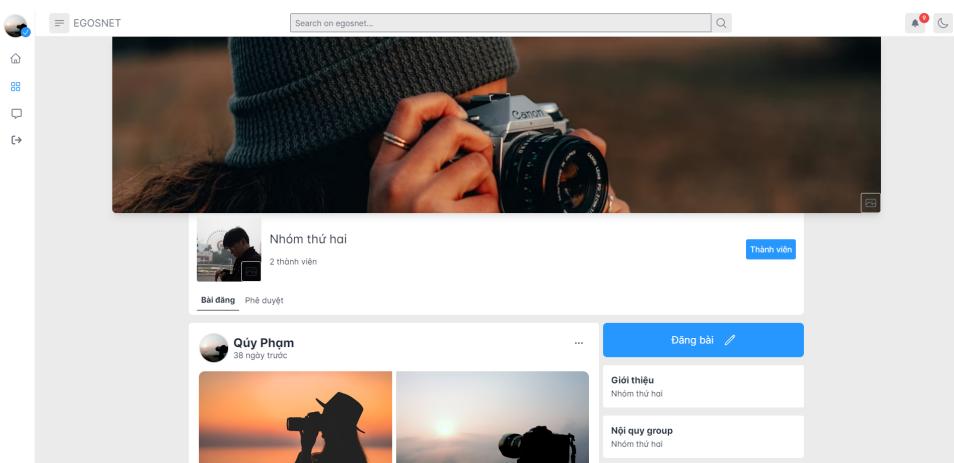


Hình 4.31: Giao diện màn hình chi tiết bài viết

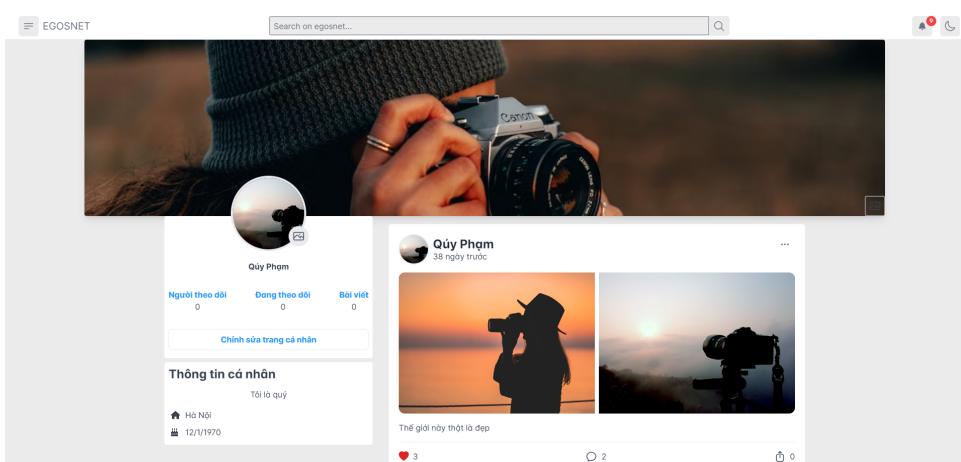
CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



Hình 4.32: Giao diện màn hình tạo bài viết

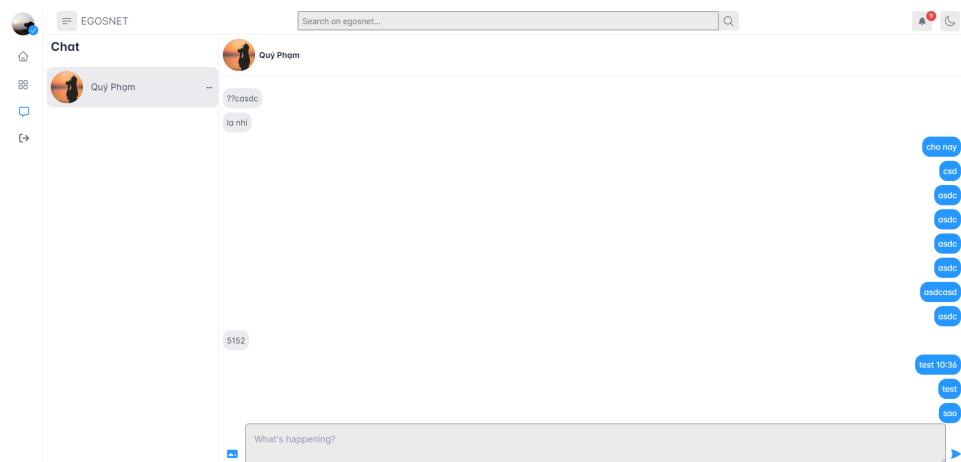


Hình 4.33: Giao diện màn hình nhóm

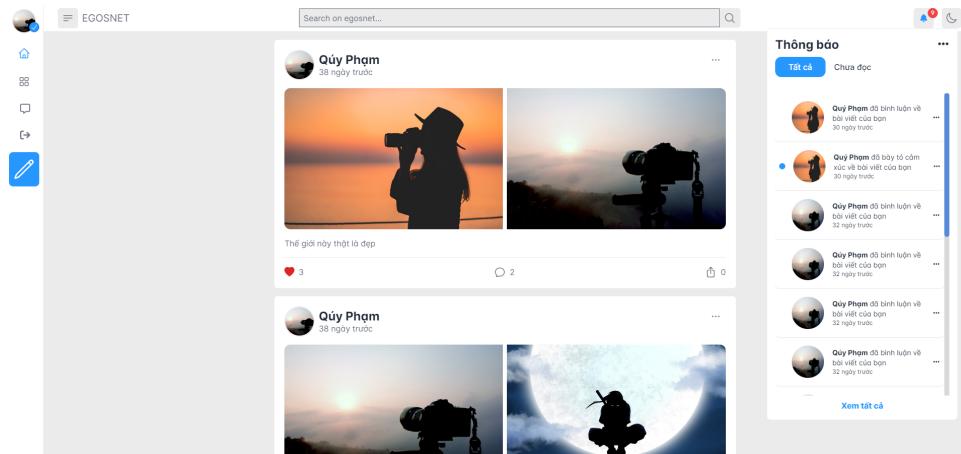


Hình 4.34: Giao diện màn hình trang cá nhân

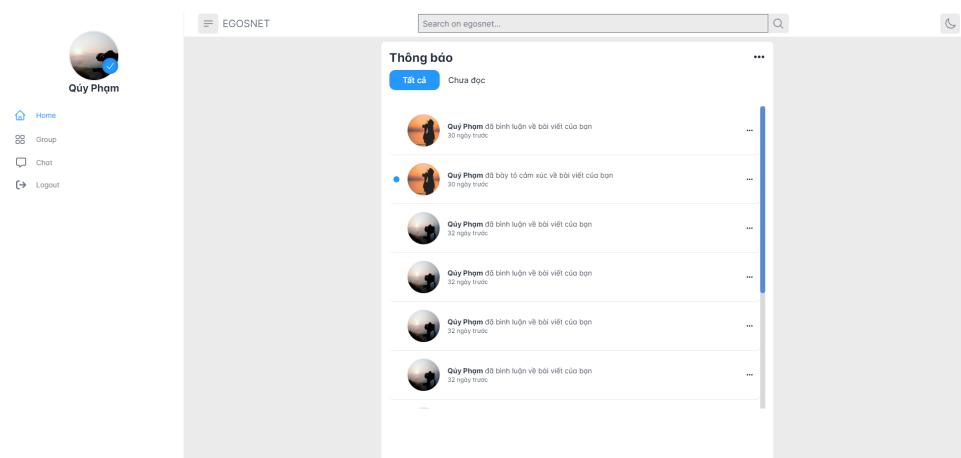
CHƯƠNG 4. THIẾT KẾ, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG



Hình 4.35: Giao diện màn hình nhắn tin



Hình 4.36: Giao diện màn hình xem thông báo ở trang chủ



Hình 4.37: Giao diện màn hình xem thông báo ở trang chi tiết

4.4 Kiểm thử

4.4.1 Kiểm thử chức năng đăng bài

Sử dụng kỹ thuật kiểm thử phân vùng tương đương cho chức năng này. Với điều kiện người dùng đã đăng nhập thành công vào hệ thống, ta có những trường cần kiểm thử:

- Nội dung: Bắt buộc, độ dài ký tự không giới hạn.
- Hình ảnh: Không bắt buộc, dung lượng tối đa 20MB.

Từ các trường đã được liệt kê và các yêu cầu tương ứng, xác định các vùng hợp lệ và chưa hợp lệ như sau:

Tên trường	Hợp lệ	Chưa hợp lệ
Nội dung	<ul style="list-style-type: none"> • Không có. 	<ul style="list-style-type: none"> • Không nhập nội dung.
Hình ảnh	<ul style="list-style-type: none"> • Ảnh dung lượng không quá 20MB. 	<ul style="list-style-type: none"> • Ảnh dung lượng lớn hơn 20MB.

Bảng 4.42: Bảng kiểm thử chức năng đăng bài

4.4.2 Kiểm thử chức năng bình luận

Sử dụng kỹ thuật kiểm thử phân vùng tương đương cho chức năng này. Với điều kiện người dùng đã đăng nhập thành công vào hệ thống, ta có những trường cần kiểm thử:

- Nội dung: Bắt buộc, độ dài ký tự không giới hạn.

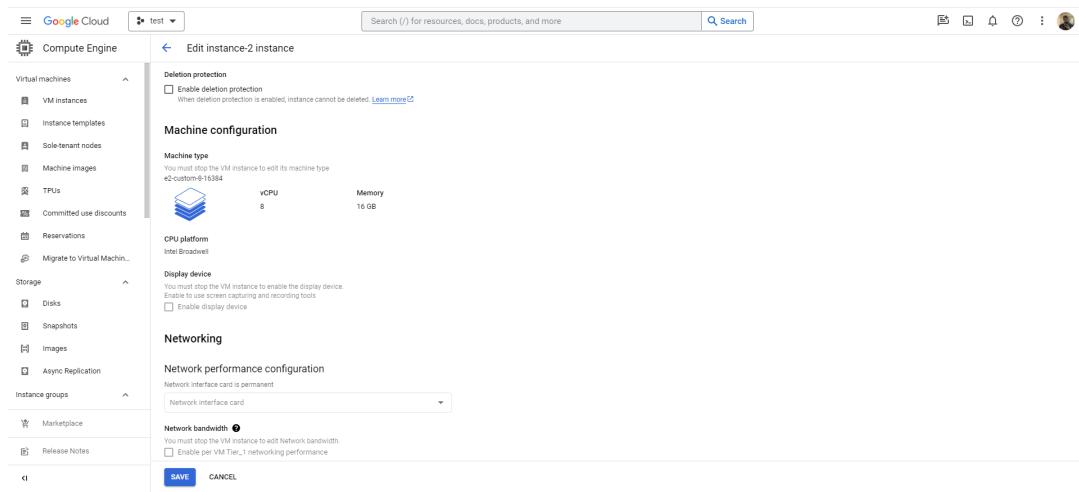
Từ các trường đã được liệt kê và các yêu cầu tương ứng, xác định các vùng hợp lệ và chưa hợp lệ như sau:

Tên trường	Hợp lệ	Chưa hợp lệ
Nội dung	<ul style="list-style-type: none"> • Không có. 	<ul style="list-style-type: none"> • Không nhập nội dung.

Bảng 4.43: Bảng kiểm thử chức năng bình luận

4.5 Triển khai

4.5.1 Tạo server



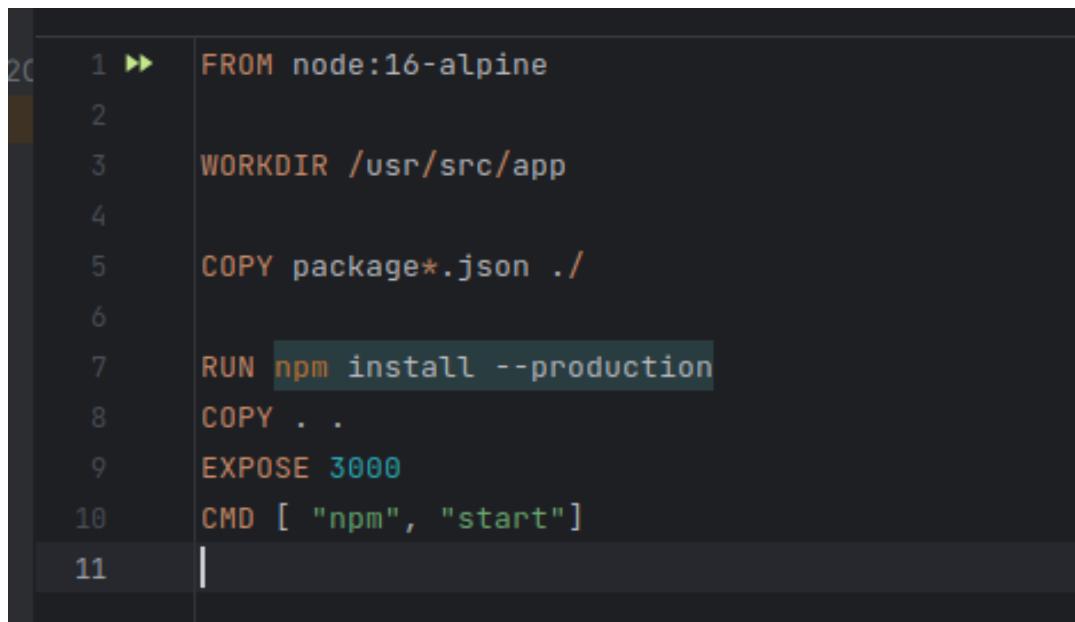
Hình 4.38: Thông tin tổng quan của server

Trong đồ án lần này, em sử dụng một máy chủ được cung cấp bởi dịch vụ của Google Cloud. Thông tin chi tiết như sau:

- Địa điểm đặt máy chủ: Asia southeast
- Địa chỉ IPv4: 34.143.240.38
- vCPU/s: 8
- RAM: 16GB
- Bộ nhớ: 80GB
- Hệ điều hành: Ubuntu 22.04 x64

4.5.2 Tạo các file Docker và docker-compose

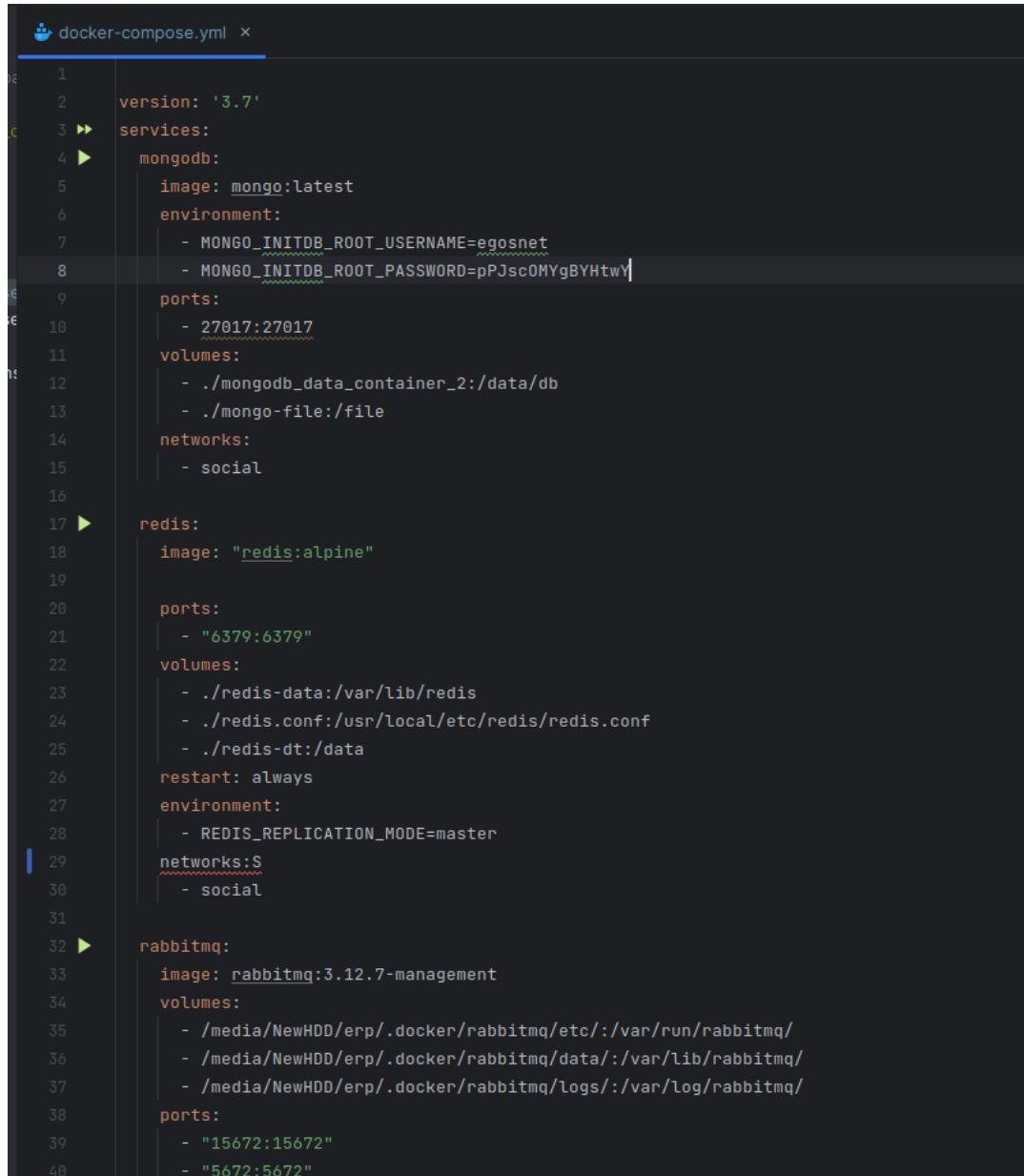
Dockerfile:



```
1 ► FROM node:16-alpine
2
3 WORKDIR /usr/src/app
4
5 COPY package*.json .
6
7 RUN npm install --production
8 COPY . .
9 EXPOSE 3000
10 CMD [ "npm", "start"]
11 |
```

Hình 4.39: Dockerfile

Docker compose file:



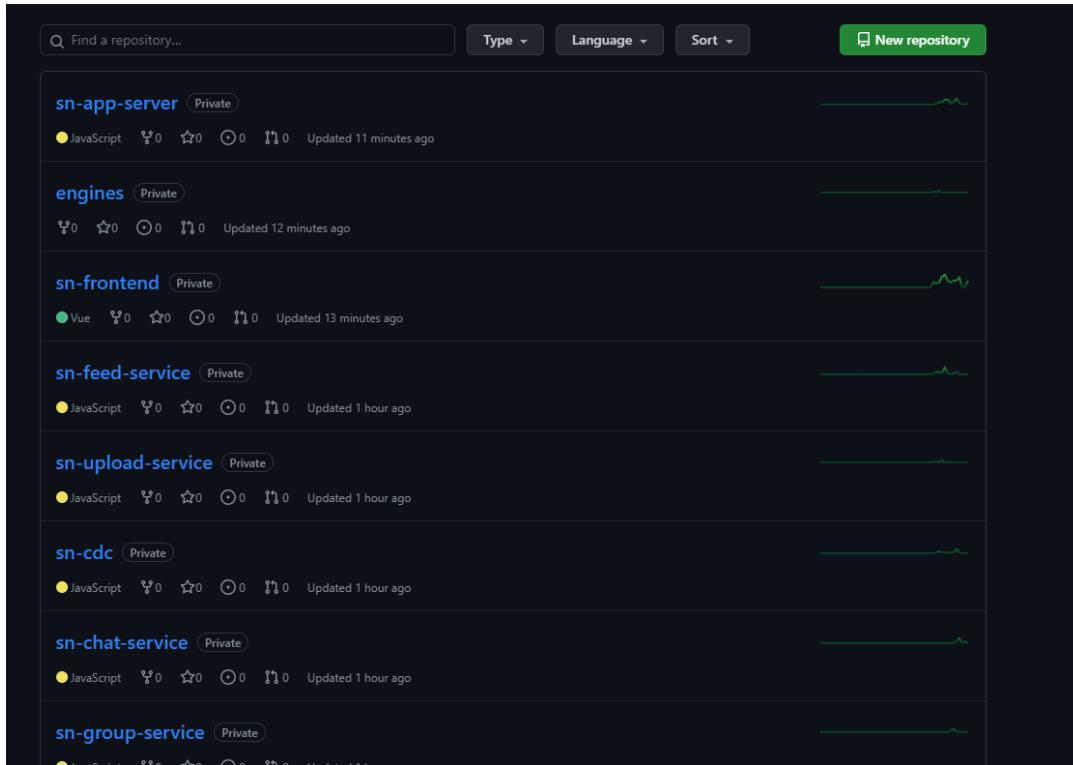
```

1  version: '3.7'
2
3  services:
4    mongodb:
5      image: mongo:latest
6      environment:
7        - MONGO_INITDB_ROOT_USERNAME=egosnet
8        - MONGO_INITDB_ROOT_PASSWORD=pPJsc0MYgBYHtwY
9      ports:
10        - 27017:27017
11      volumes:
12        - ./mongodb_data_container_2:/data/db
13        - ./mongo-file:/file
14      networks:
15        - social
16
17  redis:
18    image: "redis:alpine"
19
20    ports:
21      - "6379:6379"
22    volumes:
23      - ./redis-data:/var/lib/redis
24      - ./redis.conf:/usr/local/etc/redis/redis.conf
25      - ./redis-dt:/data
26    restart: always
27    environment:
28      - REDIS_REPLICATION_MODE=master
29    networks:
30      - social
31
32  rabbitmq:
33    image: rabbitmq:3.12.7-management
34    volumes:
35      - /media/NewHDD/erp/.docker/rabbitmq/etc:/var/run/rabbitmq/
36      - /media/NewHDD/erp/.docker/rabbitmq/data:/var/lib/rabbitmq/
37      - /media/NewHDD/erp/.docker/rabbitmq/logs:/var/log/rabbitmq/
38    ports:
39      - "15672:15672"
40      - "5672:5672"

```

Hình 4.40: Docker compose file

4.5.3 Lưu trữ source code trên Github



Hình 4.41: Lưu trữ source code trên github

4.5.4 Build mã nguồn trên server

Đầu tiên clone mã nguồn từ github về. Sau đó sử dụng "docker build" để build thành các ảnh tương ứng. Cuối cùng dùng "docker compose up -d" để khởi chạy toàn bộ container được cấu hình bên trong file docker-compose.yaml.

4.5.5 Trỏ tên miền

DNS management for **egosnet.click**

Review, add, and edit DNS records. Edits will go into effect once saved.

Type	Name	Content	Proxy status	TTL	Actions
A	egosnet.click	34.143.240.38	Proxied	Auto	Edit

Hình 4.42: Cấu hình DNS

Trên đây là cấu hình DNS. Em đã sử dụng tên miền egosnet.click là tên miền chính thức của website.

Nội dung vừa trình bày ở trên là tất cả quá trình em thiết kế, xây dựng và triển khai hệ thống. Tiếp theo trong chương 5, em sẽ trình bày một số giải pháp em đã áp dụng vào đồ án.

CHƯƠNG 5. CÁC GIẢI PHÁP VÀ ĐÓNG GÓP NỔI BẬT

5.1 Thiết kế hệ thống theo kiến trúc microservices

5.1.1 Đặt vấn đề

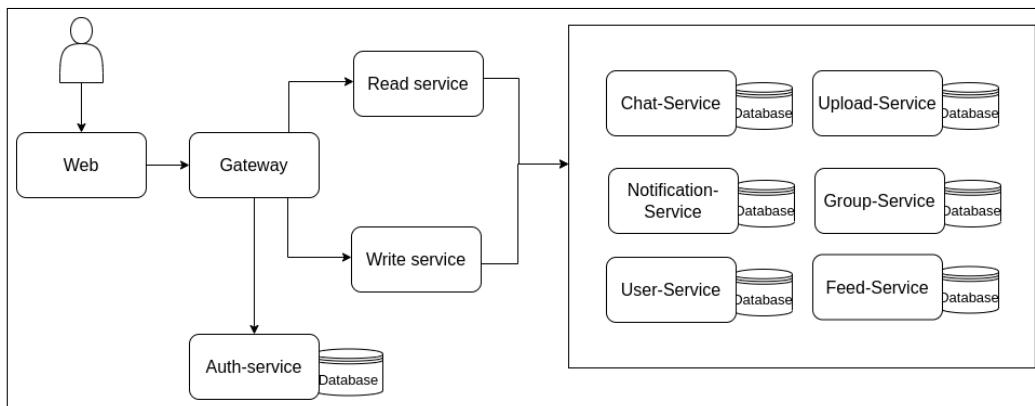
Trong quá trình làm ĐATN, em đã học và tìm hiểu rất nhiều các kiến trúc thiết kế hệ thống. Việc chọn một kiến trúc hệ thống ngay từ ban đầu là rất quan trọng đối với một hệ thống. Nó giúp cho việc vận hành, bảo trì hay mở rộng hệ thống sau này trở nên dễ dàng hơn. Trong đó, có hai kiến trúc mà em thấy được sử dụng nhiều hơn cả đó là kiến trúc nguyên khối (Monolithic) và kiến trúc dịch vụ nhỏ (microservices).

Đối với kiến trúc nguyên khối, đây là một kiến trúc mà việc triển khai, xây dựng và đóng gói thành một khối. Kiến trúc nguyên khối này được sử dụng cực kì phổ biến bởi vì ở giai đoạn đầu trong khi triển khai chúng hoạt động vô cùng tốt và việc triển khai nó lên rất nhanh. Nhưng ở các giai đoạn sau, khi hệ thống ngày càng mở rộng và lớn hơn thì kiến trúc này lại bộc lộ rất nhiều nhược điểm. Khi ứng dụng ngày càng được phát triển, càng nhiều tính năng mới được thêm vào, dữ liệu tăng, logic phức tạp hơn và rất nhiều thứ khác cần thêm nữa, khi đó ứng dụng đơn giản từ đầu sẽ trở nên cồng kềnh hơn rất nhiều, nếu muốn bảo trì ứng dụng này sẽ tốn rất nhiều công sức, chỉ một chỉnh sửa nhỏ, sẽ phải tham chiếu đến nhiều chỗ sử dụng nó và xem xét ảnh hưởng của nó lên toàn hệ thống. Và như vậy việc chỉnh sửa, thêm tính năng mới trở nên khó hơn và tốn nhiều thời gian hơn. Ngoài ra, khả năng mở rộng của cả hệ thống sẽ gặp khó khăn rất nhiều vì nó chỉ có thể mở rộng theo chiều dọc. Và cuối cùng việc sử dụng kiến trúc này phải sử dụng chung công nghệ nên rất khó để thay đổi hay áp dụng công nghệ mới.

Đối với kiến trúc dịch vụ nhỏ, đây là một kiểu kiến trúc xây dựng hệ thống mà hệ thống này được tổng hợp từ nhiều services nhỏ và độc lập có thể chạy riêng biệt, phát triển và triển khai độc lập với các service khác. Như đã nêu ở trên, với kiến trúc microservices này, ta hoàn toàn có thể khắc phục được những nhược điểm của kiến trúc nguyên khối mà ta vừa nêu ra ở trên. Các service ở trong kiến trúc này sẽ giao tiếp với nhau thông qua API, khả năng mở rộng của hệ thống cũng dễ hơn nhưng bù lại lại gặp khó khăn trong việc triển khai và quản lý các service. Em xin trình bày cụ thể cách mà em áp dụng microservices vào trong đồ án này ở phần giải pháp bên dưới.

5.1.2 Thiết kế giải pháp

Dựa trên những chức năng mà em đã phân tích từ phần khảo sát, phân tích nghiệp vụ. Em đã thiết kế lên hệ thống microservices như sau:



Hình 5.1: Cấu trúc hệ thống

Hình vẽ 5.1 mô tả cấu trúc hệ thống mà em đã thiết kế. Ở cấu trúc bên trên, em đã thiết kế hệ thống của em chia thành nhiều service khác nhau và mỗi service chứa các nghiệp vụ và chức năng riêng biệt nhau, hoạt động riêng biệt nhau, các service sẽ trao đổi thông tin với nhau thông qua API, cụ thể các service sẽ có các chức năng và nghiệp vụ như sau:

- **Gateway:** Đây là service dùng để điều hướng các API, các yêu cầu tới từ người dùng và trả lại kết quả cho người dùng.
- **Auth-service:** Đây là service dùng để kiểm tra xác thực thông tin của người dùng. Khi người dùng muốn gọi API bất kỳ nào trong hệ thống, hệ thống sẽ gọi qua đây service trước để xác thực người dùng.
- **Read service:** Đây là service dùng để gọi các API GET trong hệ thống. Khi một API của người dùng phải gọi lấy nhiều thông tin từ service khác nhau thì read service là nơi tổng hợp lại tất cả các dữ liệu và trả lại cho người dùng.
- **Write service:** Cũng giống như read service nhưng khác ở chỗ đây là service dùng để gọi các API POST, PUT, DELETE trong hệ thống.
- **Chat service:** Đây là service thực hiện các nghiệp vụ của chức năng nhắn tin, lưu trữ những thông tin về các cuộc hội thoại.
- **Upload service:** Đây là service thực hiện các nghiệp vụ tải file, lưu trữ file lên hệ thống.
- **Notification service:** Đây là service thực hiện các nghiệp vụ về thông báo. Tại đây cũng có một worker giúp nhận thông báo từ các service khác qua queue

và thực hiện đẩy thông báo về cho người dùng.

- Group service: Đây là service thực hiện các nghiệp vụ của chức năng nhóm người dùng, lưu trữ những thông tin của các nhóm người dùng.
- User service: Đây là service thực hiện các nghiệp vụ của chức năng người dùng, lưu trữ những thông tin của người dùng.
- Feed service: Đây là service thực hiện các nghiệp vụ của chức năng bài viết, lưu trữ những thông tin về bài viết, bình luận và bày tỏ cảm xúc của người dùng.

Ngoài ra, ở trong cấu trúc hệ thống trên, Các service sẽ giao tiếp với nhau thông qua API và như định nghĩa về kiến trúc microservices mỗi service sẽ lưu trữ dữ liệu vào một cơ sở dữ liệu riêng biệt và hoàn toàn không liên quan đến nhau.

5.1.3 Kết quả đạt được

Em đã áp dụng những kiến thức mình học và nghiên cứu được vào đồ án và xây dựng, thiết kế thành công hệ thống theo kiến trúc microservices. Nhìn vào kiến trúc hệ thống bên trên, việc mở rộng hệ thống trở lên dễ dàng hơn rất nhiều và mình hoàn toàn có thể mở rộng theo từng chức năng riêng biệt phụ thuộc vào lưu lượng truy cập của người dùng vào từng chức năng.

5.2 Chức năng thông báo

5.2.1 Tổng quan vấn đề

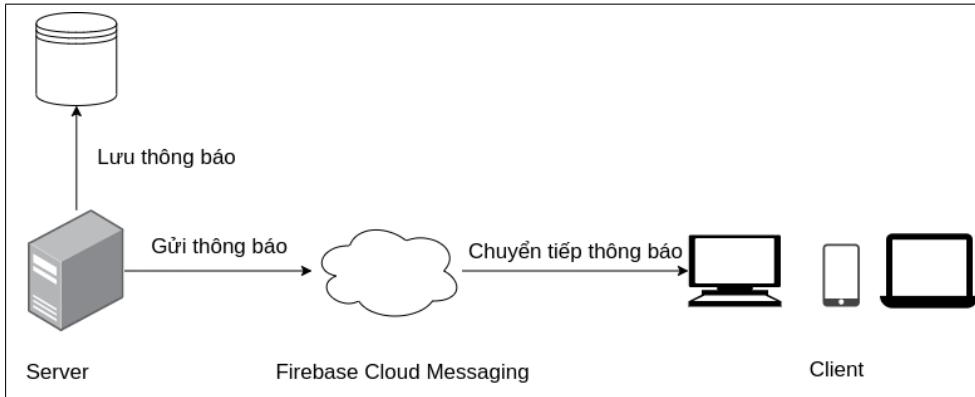
Qua quá trình khảo sát, tìm hiểu và nghiên cứu hành vi người dùng, nhận thấy trên mạng xã hội hàng vi tương tác của người dùng là rất quan trọng nó giúp tăng tương tác trên mạng xã hội và giúp người dùng giao tiếp với nhau dễ dàng hơn. Việc gửi thông báo sẽ giúp cho người dùng cập nhật các thông tin cần thiết ngay lập tức, nó giúp cho người dùng gần như là không bỏ lỡ bất kỳ thông tin quan trọng nào xảy ra xung quanh người dùng. Do đó, trong sản phẩm đồ án này, em đã xây dựng một chức năng gửi thông báo cho người dùng thời gian thực giúp tăng khả năng tương tác của người dùng với mạng xã hội.

5.2.2 Thiết kế giải pháp

Đối với những bài viết liên quan đến người dùng, khi có một người dùng tương tác với bài viết, hệ thống sẽ gửi thông báo đến người dùng có liên quan đến hành vi tương tác đó.

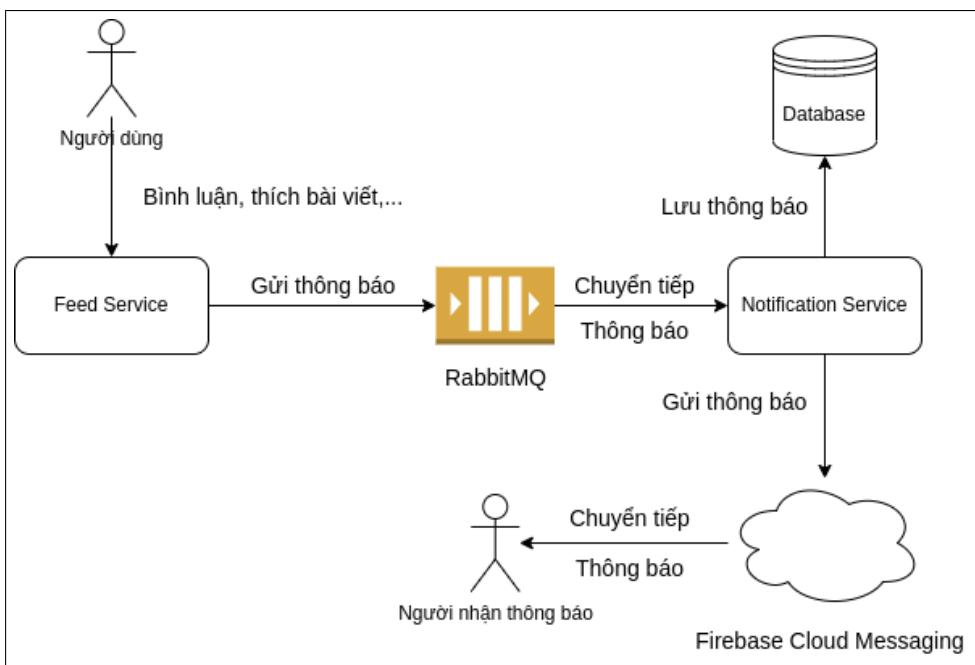
Để có thể gửi thông báo cho người dùng, em lựa chọn sử dụng dịch vụ Firebase Cloud Messaging(FCM) [10]. Đây là một dịch vụ thông báo đám mây miễn phí được cung cấp bởi Google Firebase. Sử dụng FCM, em có thể gửi các thông điệp

nhanh chóng, an toàn tới các thiết bị sử dụng hệ thống của em. Việc tích hợp dịch vụ này vào hệ thống cũng dễ dàng thông qua các API đơn giản. Ngoài ra nó còn có một vài thống kê và tương tác người dùng với thông báo.



Hình 5.2: Mô hình hoạt động phổ biến của FCM

Hình 5.2 là hình vẽ mô tả mô hình hoạt động phổ biến, cơ bản nhất của Dịch vụ FCM. Có thể hiểu đơn giản cách hoạt động của nó như sau. Hệ thống máy chủ của mình sau khi lưu trữ thông báo sẽ đẩy thông báo lên cho dịch vụ FCM kèm theo thông tin Client muốn gửi đến. Sau khi FCM nhận được thông tin sẽ chuyển tiếp thông báo tới cho những người dùng được khai báo. Dựa trên mô hình trên, em đã thiết kế và sử dụng dịch vụ FCM cho hệ thống như sau.



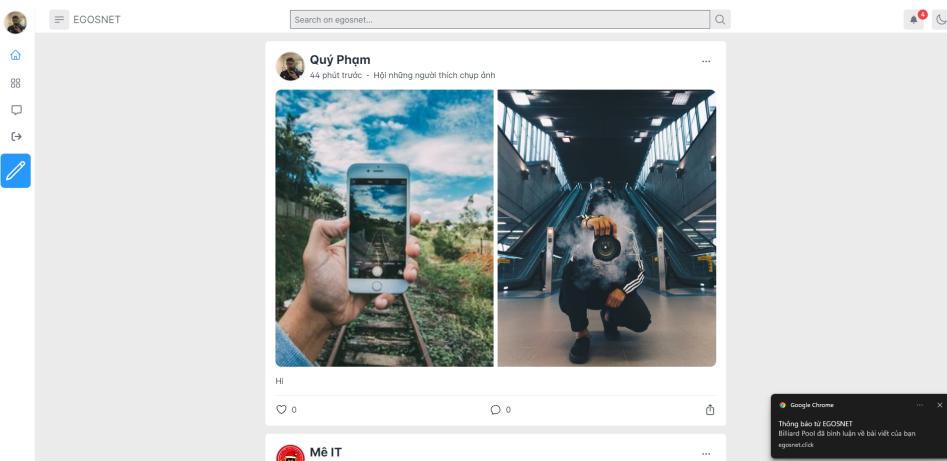
Hình 5.3: Luồng hoạt động của chức năng thông báo

Hình 5.5 là hình vẽ mô tả luồng hoạt động của chức năng thông báo đã áp dụng dịch vụ FCM vào trong hệ thống và sử dụng Hàng đợi, tại đây em sử dụng

RabbitMQ để tránh mất mát dữ liệu và tốc độ gửi đi của nó cũng nhanh. Cụ thể như sau:

- Với mỗi người dùng sau khi đăng nhập vào trong hệ thống đều có thể sử dụng tính năng bình luận và bày tỏ cảm xúc đối với bài viết. Khi người dùng sử dụng tính năng bình luận và bày tỏ cảm xúc thành công, hệ thống sẽ lưu lại dữ liệu đồng thời sẽ tạo ra một yêu cầu thông báo tới người tạo bài viết rằng có người đã bình luận hoặc bày tỏ cảm xúc đối với bài viết của người dùng.
- Khi tạo xong yêu cầu, yêu cầu đó sẽ được gửi vào hàng đợi kèm theo dữ liệu cần thiết để gửi thông báo. Hàng đợi sau đó sẽ lưu lại và chuyển tiếp dữ liệu đến cho Notification Service để xử lý dữ liệu.
- Notification service sẽ luôn lắng nghe từ hàng đợi, khi có một yêu cầu mới đến nó sẽ nhận và xử lý yêu cầu đó. Khi xử lý xong nó sẽ xóa đi yêu cầu bên trong hàng đợi và tiếp tục xử lý những yêu cầu đến sau. Đây cũng chính là nguyên lý hoạt động của hàng đợi.
- Sau khi nhận yêu cầu từ hàng đợi, Notification Service sẽ xử lý dữ liệu nhận được và lưu trữ nó vào cơ sở dữ liệu. Sau khi lưu trữ thành công, nó sẽ gửi dữ liệu lên cho FCM kèm theo những thông tin những thiết bị sẽ được nhận thông báo.
- Với mỗi yêu cầu được gửi lên cho FCM, FCM sẽ gửi thông báo đến cho lần lượt các thiết bị khớp với những thông tin mà hệ thống đã gửi lên cho FCM.
- Cuối cùng, ở trên website sẽ có một chương trình chạy ngầm trên trình duyệt. Nó sẽ lắng nghe sự kiện đẩy thông báo từ FCM [10]. Sau khi nhận được dữ liệu từ FCM nó sẽ hiển thị thông báo và cập nhật giao diện mới cho người dùng.

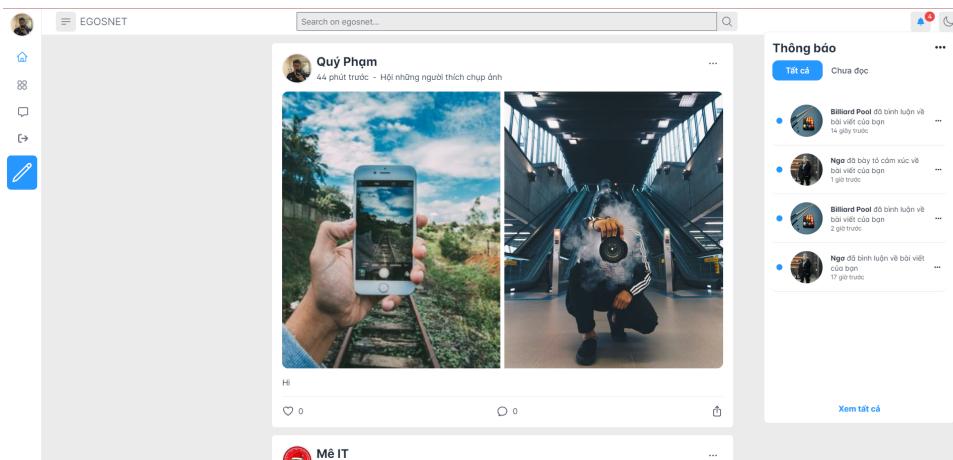
5.2.3 Kết quả đạt được



Hình 5.4: Kết quả đạt được của chức năng thông báo

Trên đây là kết quả đạt được của chức năng thông báo. Sau khi có người dùng khác bình luận vào bài viết, thông báo đã được đẩy về và thông báo cho chủ bài viết. Ngoài ra khi người dùng khác bày tỏ cảm xúc hoặc theo dõi, thông báo cũng sẽ được gửi về cho người dùng.

Độ trễ của việc gửi thông báo khá là thấp và được cập nhật gần như là thời gian thực cho người dùng. Dưới đây là danh sách thông báo, người dùng có thể ấn vào để xem chi tiết thông báo.



Hình 5.5: Danh sách thông báo

CHƯƠNG 6. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

6.1 Kết luận

Sau một thời gian tìm hiểu, khảo sát, nghiên cứu và nghiêm túc làm việc, áp dụng những kiến thức đã học và tìm hiểu, cuối cùng đồ án của em đã hoàn thành là xây dựng mạng xã hội egosnet. Website cho phép người dùng có thể tương tác, trao đổi thông tin, đăng tải những thông tin lên mạng xã hội này.

Tuy vậy, bên cạnh những tính năng mà em đã phát triển được, sản phẩm vẫn còn các hạn chế như sau:

- Chưa kiểm duyệt được nội dung người dùng.
- Chưa có thuật toán để phân hóa nội dung theo người dùng.

Qua quá trình thực hiện đồ án này, em đã tích lũy được rất nhiều kiến thức và kinh nghiệm cho bản thân. Tự hoàn thiện một trang web, tự xây dựng và thiết kế một hệ thống, lập trình cả Front-end, Back-end, cách tổ chức mã nguồn, phân tích nghiệp vụ và thiết kế hệ thống cuối cùng là triển khai lên một máy chủ thực tế. Ngoài ra, em còn được cải thiện thêm rất nhiều về kỹ năng mềm, quản lý thời gian, quản lý dự án và có thêm nhiều kinh nghiệm, trải nghiệm thực tế.

6.2 Hướng phát triển

Sau khi hoàn thiện trang web của mình, việc liên tục cập nhật các nội dung mới là việc đầu tiên và em sẽ phát triển, khắc phục và hoàn thiện một số nhược điểm nêu ở trên như sau:

- Cần có sự can thiệp xác thực, kiểm duyệt những nội dung nhạy cảm được đăng tải trên mạng xã hội. Có thể áp dụng là sẽ xây dựng thêm các vai trò như Quản trị viên giúp kiểm duyệt nội dung hoặc áp dụng học máy vào kiểm duyệt.
- Bổ sung thuật toán cá nhân hóa người dùng, giúp đề xuất các bài viết một cách đa dạng hơn.
- Tăng cường bảo mật về dữ liệu người dùng hơn.

TÀI LIỆU THAM KHẢO

- [1] B. Nelson, *Getting to Know Vue.js: Learn to Build Single Page Applications in Vue from Scratch*. Brett Nelson, 2018.
- [2] A. Mardan, *Express.js Guide: The Comprehensive Book on Express.js*. Azat Mardan, 2014.
- [3] K. Banker, D. Garrett, P. Bakkum, and S. Verch, *MongoDB in action: covers MongoDB version 3.0*. Simon and Schuster, 2016.
- [4] L. Moroney and L. Moroney, “An introduction to firebase,” *The Definitive Guide to Firebase: Build Android Apps on Google’s Mobile Platform*. pp. 1–24, 2017.
- [5] D. J. L. Carlson, *Redis in Action*. Manning, 2013.
- [6] D. of Minio, Available: <https://min.io/docs/minio/container/index.html>. (visited on 01/06/2024).
- [7] N. Poulton, *Docker Deep Dive*. Nigel Poulton, 2017.
- [8] S. Newman, *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, 2015.
- [9] G. M. Roy, *RabbitMQ in Depth*. Manning, 2017.
- [10] L. Moroney and L. Moroney, “Firebase cloud messaging,” *The Definitive Guide to Firebase: Build Android Apps on Google’s Mobile Platform*, pp. 163–188, 2017.

PHỤ LỤC