

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

ĐỒ ÁN TỐT NGHIỆP

Phần mềm nhắn tin, gọi điện video trực tuyến

VŨ QUÝ ĐẠT

dat.vq176082@sis.hust.edu.vn

Ngành kỹ thuật phần mềm
Chương trình đào tạo quốc tế SIE

Giảng viên hướng dẫn: PGS.TS Trần Đình Khang

Chữ ký GVHD

Khoa: Khoa học máy tính

Trường: Công nghệ thông tin và Truyền thông

HÀ NỘI, 03/2023

LỜI CẢM ƠN

Đồ án tốt nghiệp là minh chứng thành quả học tập tích luỹ của em trong suốt 5 năm học tập tại Trường Đại học bách khoa Hà Nội. Ngoài sự nỗ lực của bản thân, đồ án tốt nghiệp của em sẽ không thể hoàn thành tốt, chỉn chu nếu không có sự chỉ bảo và hướng dẫn tận tình của thầy, cô trong suốt thời gian qua.

Trước hết em xin bày tỏ lòng biết ơn sâu sắc đến thầy hướng dẫn là PGS.TS Trần Đình Khang người luôn đồng hành, đưa ra lời khuyên giúp em có thể hoàn thiện đồ án. Em xin chân thành cảm ơn các thầy, cô Trường Công nghệ Thông tin và Truyền thông - ĐH Bách khoa Hà Nội đã truyền đạt những kiến thức quý báu của mình cho em cũng như các bạn sinh viên khác, có thể nói đây là hành trang, là những kiến thức nền tảng quý báu trước khi bước ra ngoài xã hội.

Cuối cùng em xin gửi lời cảm ơn tới gia đình bạn bè luôn động viên, tin tưởng em trong thời gian qua. Trong quá trình làm đồ án, do thời gian có hạn và kinh nghiệm làm các ứng dụng thực tế của em còn ít nên không thể tránh khỏi các sai sót, em mong nhận được sự đóng góp của các thầy cô, bạn bè để đồ án của em có thể hoàn thiện hơn.

TÓM TẮT NỘI DUNG ĐỒ ÁN

Được truyền cảm hứng từ các ứng dụng nhắn tin trực tuyến hiện đang có như Messenger, Zalo, Telegram cùng với việc công ty em đang làm mong muốn có một ứng dụng nhắn tin dành riêng cho tập đoàn, em muốn tạo một ứng dụng nhắn tin cho cá nhân mình và những đồng nghiệp trong công ty cùng sử dụng. Do đó em đánh giá ứng dụng này sẽ mang đến nhiều lợi ích thiết thực, chính vì vậy em quyết định lựa chọn đề tài "Phần mềm nhắn tin, gọi điện video trực tuyến" xây dựng trên nền tảng iOS.

Phần mềm nhắn tin, gọi điện video trực tuyến được xây dựng với mục tiêu là phần mềm mã nguồn mở, nhằm mục đích giúp cho các doanh nghiệp, các tổ chức không tin cậy việc trao đổi thông tin liên quan đến doanh nghiệp, tổ chức trên phần mềm của bên thứ ba, có thể tự khai thác mã nguồn và triển khai một ứng dụng nhắn tin, gọi điện video trực tuyến riêng cho cá nhân tổ chức, doanh nghiệp. Người sử dụng ứng dụng có thể nhìn thấy những người dùng khác, tạo một phòng trò chuyện với người dùng họ muốn. Một phòng trò chuyện có thể có nhiều thành viên ở trong đó, người dùng có thể gửi ảnh, video trong phòng trò chuyện. Tính năng gọi điện video trực tuyến khả dụng trong những phòng trò chuyện chỉ có 2 thành viên. Ngoài ra mỗi người dùng có một trang cá nhân cho riêng bản thân, giúp cho mọi người dùng đều có khả năng nắm rõ thông tin của nhau như chức vụ, vị trí, tuổi tác,... Người dùng có thể cập nhật thông tin trên trang cá nhân của bản thân cũng như ảnh đại diện của họ.

MỤC LỤC

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI.....	1
1.1 Lý do chọn đề tài	1
1.2 Mục tiêu và phạm vi đề tài.....	1
1.3 Bố cục đồ án	1
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 Phân rã use case Nhắn tin	6
2.2.3 Phân rã use case Chính sửa thông tin cá nhân	6
2.2.4 Phân rã use case Quản lý nhóm chat	7
2.3 Quy trình nghiệp vụ	7
2.3.1 Nghiệp vụ Đăng ký	7
2.3.2 Nghiệp vụ Đăng nhập.....	8
2.3.3 Nghiệp vụ Thay đổi mật khẩu	9
2.3.4 Nghiệp vụ Chính sửa thông tin cá nhân được công khai	10
2.3.5 Nghiệp vụ Thêm thành viên vào nhóm.....	10
2.3.6 Nghiệp vụ Xoá thành viên khỏi nhóm chat	11
2.3.7 Nghiệp vụ Gửi tin nhắn ảnh và video	12
2.3.8 Nghiệp vụ Gửi tin nhắn văn bản	13
2.3.9 Nghiệp vụ Đăng xuất.....	13
2.4 Đặc tả ca chức năng	14
2.4.1 Đặc cả use case Đăng ký.....	14
2.4.2 Đặc cả use case Đăng nhập	16

2.4.3 Đặc cả use case Đăng xuất	17
2.4.4 Đặc cả use case Gửi tin nhắn.....	18
2.4.5 Đặc cả use case Thêm người dùng vào phòng chat	19
2.4.6 Đặc cả use case Xoá người dùng khỏi phòng chat.....	20
2.5 Các yêu cầu phi chức năng.....	20
2.5.1 Yêu cầu về bảo mật	20
2.5.2 Yêu cầu về hiệu năng.....	20
2.5.3 Yêu cầu về giao diện.....	20
CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG.....	22
3.1 Ngôn ngữ lập trình Swift	22
3.2 Vapor	22
3.3 WebRTC.....	23
3.4 WebSocket	23
3.5 Hệ quản trị cơ sở dữ liệu PostgreSQL	23
3.6 MongoDB	24
3.7 Docker.....	24
CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG.....	26
4.1 Thiết kế kiến trúc.....	26
4.1.1 Lựa chọn kiến trúc phần mềm	26
4.1.2 Thiết kế tổng quan.....	30
4.1.3 Thiết kế chi tiết gói Messaging	32
4.1.4 Thiết kế chi tiết gói User Access.....	33
4.1.5 Thiết kế chi tiết gói Controllers	34
4.1.6 Thiết kế chi tiết gói Models.....	35
4.2 Thiết kế chi tiết.....	36
4.2.1 Thiết kế giao diện	36

4.2.2 Thiết kế lớp	36
4.2.3 Biểu đồ tuần tự tính năng gọi điện video	39
4.2.4 Thiết kế cơ sở dữ liệu	41
4.3 Xây dựng ứng dụng.....	44
4.3.1 Thư viện và công cụ sử dụng	44
4.3.2 Kết quả đạt được	45
4.3.3 Minh họa các chức năng chính	45
4.4 Kiểm thử.....	50
4.5 Triển khai	52
CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	57
5.1 Kết luận	57
5.2 Hướng phát triển của đồ án	57
TÀI LIỆU THAM KHẢO.....	58

DANH MỤC HÌNH VẼ

Hình 2.1	Sơ đồ use case tổng quan	5
Hình 2.2	Phân rã use case Nhắn tin	6
Hình 2.3	Phân rã use case Chính sửa thông tin cá nhân	6
Hình 2.4	Phân rã use case Quản lý nhóm chat	7
Hình 2.5	Sơ đồ hoạt động quy trình nghiệp vụ Đăng ký	7
Hình 2.6	Sơ đồ hoạt động quy trình nghiệp vụ Đăng nhập	8
Hình 2.7	Sơ đồ hoạt động quy trình nghiệp vụ Thay đổi mật khẩu	9
Hình 2.8	Sơ đồ hoạt động quy trình nghiệp vụ Chính sửa thông tin cá nhân được công khai	10
Hình 2.9	Sơ đồ hoạt động quy trình nghiệp vụ Thêm thành viên vào nhóm	10
Hình 2.10	Sơ đồ hoạt động quy trình nghiệp vụ Xoá thành viên khỏi nhóm chat	11
Hình 2.11	Sơ đồ hoạt động quy trình nghiệp vụ Gửi tin nhắn ảnh và video	12
Hình 2.12	Sơ đồ hoạt động quy trình nghiệp vụ Gửi tin nhắn văn bản . .	13
Hình 2.13	Sơ đồ hoạt động quy trình nghiệp vụ Đăng xuất	13
Hình 4.1	Kiến trúc tổng quan của hệ thống	26
Hình 4.2	Luồng hoạt động của BackEnd	27
Hình 4.3	Luồng hoạt động của ứng dụng trên iOS theo mô hình MVC .	29
Hình 4.4	Thiết kế gói tổng quan	30
Hình 4.5	Thiết kế chi tiết gói Messaging	32
Hình 4.6	Thiết kế chi tiết gói User Access	33
Hình 4.7	Thiết kế chi tiết gói Controllers	34
Hình 4.8	Thiết kế chi tiết gói Models	35
Hình 4.9	Thiết kế chi tiết lớp ChatBoxViewController	37
Hình 4.10	Thiết kế chi tiết lớp ManagingChatBoxViewController . . .	38
Hình 4.11	Biểu đồ tuần tự tính năng gọi điện video	40
Hình 4.12	Sơ đồ thiết kế cơ sở dữ liệu	41
Hình 4.13	Màn hình đăng ký	46
Hình 4.14	Màn hình đăng nhập	46
Hình 4.15	Màn hình thay đổi avatar	47
Hình 4.16	Màn hình xem thông tin bản thân	47
Hình 4.17	Màn hình quản lý thành viên chat box	48
Hình 4.18	Màn hình xoá thành viên khỏi chat box	48

Hình 4.19 Màn hình danh sách người dùng	49
Hình 4.20 Màn hình danh sách chat box	49
Hình 4.21 Màn hình nhắn tin	50
Hình 4.22 Container thông báo đang chạy PostgreSQL	54
Hình 4.23 Container thông báo đang chạy MongoDB	54

DANH MỤC BẢNG BIỂU

Bảng 2.1	Đặc tả usecase Đăng Ký	14
Bảng 2.2	Bảng dữ liệu đầu vào usecase Đăng ký	15
Bảng 2.3	Đặc tả usecase Đăng Nhập	16
Bảng 2.4	Bảng dữ liệu đầu vào usecase Đăng nhập	17
Bảng 2.5	Đặc tả usecase Đăng xuất	17
Bảng 2.6	Đặc tả usecase Gửi tin nhắn	18
Bảng 2.7	Bảng dữ liệu đầu vào usecase Gửi tin nhắn	18
Bảng 2.8	Đặc tả usecase Thêm người dùng vào phòng chat	19
Bảng 2.9	Đặc tả usecase Xoá người dùng khỏi phòng chat	20
Bảng 4.1	Bảng dữ liệu User	42
Bảng 4.2	Bảng dữ liệu Chatbox	43
Bảng 4.3	Bảng dữ liệu User	43
Bảng 4.4	Bảng dữ liệu members_of_chatbox	44
Bảng 4.5	Bảng danh sách thư viện và công cụ sử dụng	45
Bảng 4.6	Thông tin kết quả đạt được	45
Bảng 4.7	Kết quả kiểm thử	52

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)
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 Lý do chọn đề tài

Được truyền cảm hứng từ các ứng dụng nhắn tin trực tuyến hiện đang có như Messenger, Zalo cùng với việc công ty em đang làm mong muốn có một ứng dụng nhắn tin dành riêng cho tập đoàn, do đó em muốn tự mình xây dựng một ứng dụng nhắn tin cho những đồng nghiệp trong công ty cùng sử dụng. Em đánh giá ứng dụng này sẽ mang đến nhiều lợi ích thiết thực, chính vì vậy em quyết định lựa chọn đề tài "Phần mềm nhắn tin, gọi điện video trực tuyến".

Nhằm tìm hiểu quá trình phát triển cũng như có thể tự xây dựng một ứng dụng nhắn tin và gọi điện video trực tuyến trong khuôn khổ phù hợp với sinh viên, giảng viên hướng dẫn và em đã thống nhất quyết định lựa chọn đề tài xây dựng "Phần mềm nhắn tin, gọi điện video trực tuyến".

1.2 Mục tiêu và phạm vi đề tài

Mục tiêu và phạm vi của đồ án là thiết kế và xây dựng một ứng dụng nhắn tin, gọi điện video trực tuyến trên nền tảng iOS, có thể đáp ứng được các chức năng cơ bản của việc trao đổi thông tin hàng ngày giữa các nhân sự, cá nhân trong một tổ chức, doanh nghiệp như tạo tài khoản, chỉnh sửa thông tin cá nhân, nhắn tin văn bản, gửi tin nhắn ảnh và video, xem thông tin của người dùng khác, gọi điện video, tạo một phòng trò chuyện chỉ bao gồm những thành viên trong 1 dự án cụ thể, xoá thành viên khỏi nhóm chat, xoá nhóm chat.

1.3 Bố cục đồ án

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

Chương 2 trình bày về khảo sát và phân tích yêu cầu, phạm vi của đề tài. Trong chương này trình bày về ưu và nhược điểm của các ứng dụng đã thực hiện khảo sát. Từ đó đưa ra các chức năng cần phát triển cho ứng dụng của mình và các đối tượng sử dụng ứng dụng, nêu ra các chức năng của ứng dụng, trình bày tổng quan, phân tích làm rõ các quy trình nghiệp vụ, đặc tả chi tiết các usecase, các yêu cầu phi chức năng. Mỗi chức năng sẽ được mô tả thông qua biểu đồ usecase phân rã, quy trình nghiệp vụ, đặc tả chi tiết cho từng use case.

Chương 3 trình bày về các công nghệ, cơ sở lý thuyết của các công nghệ, ưu nhược điểm của nó, cách áp dụng nó vào đồ án này như thế nào và vì sao lại lựa chọn nó.

Chương 4 trình bày chi tiết về cách phát triển và triển khai ứng dụng, từ việc thiết kế tổng quan đến việc đi sâu vào thiết kế chi tiết từng gói từng lớp, thiết kế cơ

sở dữ liệu, thiết kế giao diện,.. Thêm vào đó chương này cũng sẽ trình bày các thư viện và công cụ sử dụng trong quá trình phát triển ứng dụng. Sau cùng là minh họa một số kết quả đạt được sau khi xây dựng thành công ứng dụng nhắn tin, gọi điện video trực tuyến và tiến hành kiểm thử.

Chương 5 đưa ra kết quả mà đồ án đạt được. Các ưu điểm và nhược điểm của ứng dụng. Chỉ ra được hướng phát triển của hệ thống trong tương lai.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

2.1 Khảo sát hiện trạng

Để có cái nhìn khách quan cũng như có trải nghiệm thực tế để xây dựng một ứng dụng nhắn tin, gọi điện video trực tuyến trên hệ điều hành iOS em đã khảo sát qua một số ứng dụng: Telegram, Messenger, Zalo.

Sau khi trải nghiệm các ứng dụng thì em nhận thấy nhìn chung tất cả các ứng dụng đều có giao diện rất thân thiện với người dùng. Về Telegram thông qua khảo sát em thấy ứng dụng này hướng đến sự linh hoạt, bảo mật, khả năng tìm kiếm lịch sử tin nhắn và file chia sẻ linh động, nó thu hút được khoảng 300 triệu người đăng ký sử dụng, tuy nhiên em nhận thấy vì tính năng của nó rất nhiều nên giao diện tương đối phức tạp. Các tính năng của Telegram rất hữu ích, nó là một công cụ nhắn tin có giao diện mượt mà, tốc độ load ảnh và tin nhắn nhanh do ứng dụng đã lưu những dữ liệu đã lấy về từ máy chủ vào bộ nhớ đệm. Tính năng ghim những tin nhắn quan trọng giúp người dùng có thể theo dõi tốt lịch sử tin nhắn đã bị lỡ.

Khác với Telegram thì Messenger tập trung hơn tới việc đơn giản hóa về giao diện, nhưng việc có nhiều bài báo viết về việc Messenger thu thập thông tin người dùng là một điểm trừ khiến cho người dùng không tin tưởng sử dụng để trao đổi công việc hay những nội dung quan trọng trên phần mềm này. Những file được gửi trong các tin nhắn khó có thể tìm lại, vì em đã sử dụng và thấy hiện tại phần mềm chỉ lưu trữ các file em đã gửi cho người khác trong vòng 1 năm trở lại. Ngoài ra người dùng messenger thường xuyên bị gửi những tin nhắn rác làm phiền.

Về Zalo, do phần mềm mang lại trải nghiệm khá tuyệt vời cho em. Nó có giao diện đơn giản, nhiều tính năng hữu ích được đơn giản hóa về giao diện như cập nhật thời tiết, cập nhật các thông tin báo chí, ghim các lịch biểu công việc, tuy nhiên giao diện zalo đôi khi bị giật mà không mượt mà trên các dòng máy đời thấp. Zalo cũng không tự động lưu lại lịch sử tin nhắn của người dùng mà yêu cầu người dùng chủ động vào cài đặt và chọn sao lưu, nhưng cũng chỉ lưu được tin nhắn văn bản ở bản miễn phí, dẫn đến tình trạng người dùng thường xuyên bị mất dữ liệu cũ. Zalo còn có hạn chế với tài khoản miễn phí là người dùng chỉ được phản hồi 40 hội thoại mỗi tháng từ người lạ.

Thông qua việc khảo sát 3 ứng dụng trên em đã quyết định phát triển ứng dụng "Ứng dụng nhắn tin, gọi điện video trực tuyến" trên nền tảng iOS, giúp cho những tổ chức, doanh nghiệp có thể tự triển khai một phần mềm nhắn tin của riêng họ, từ những tính năng cơ bản, họ có thể phát triển thêm những tính năng mới phù hợp

với tổ chức, doanh nghiệp của họ. Về cơ bản ứng dụng cho phép người dùng nhắn tin, gọi điện video trực tuyến, gửi ảnh và video, cập nhật thông tin cá nhân, và quản lý phòng chat.

2.2 Tổng quan chức năng

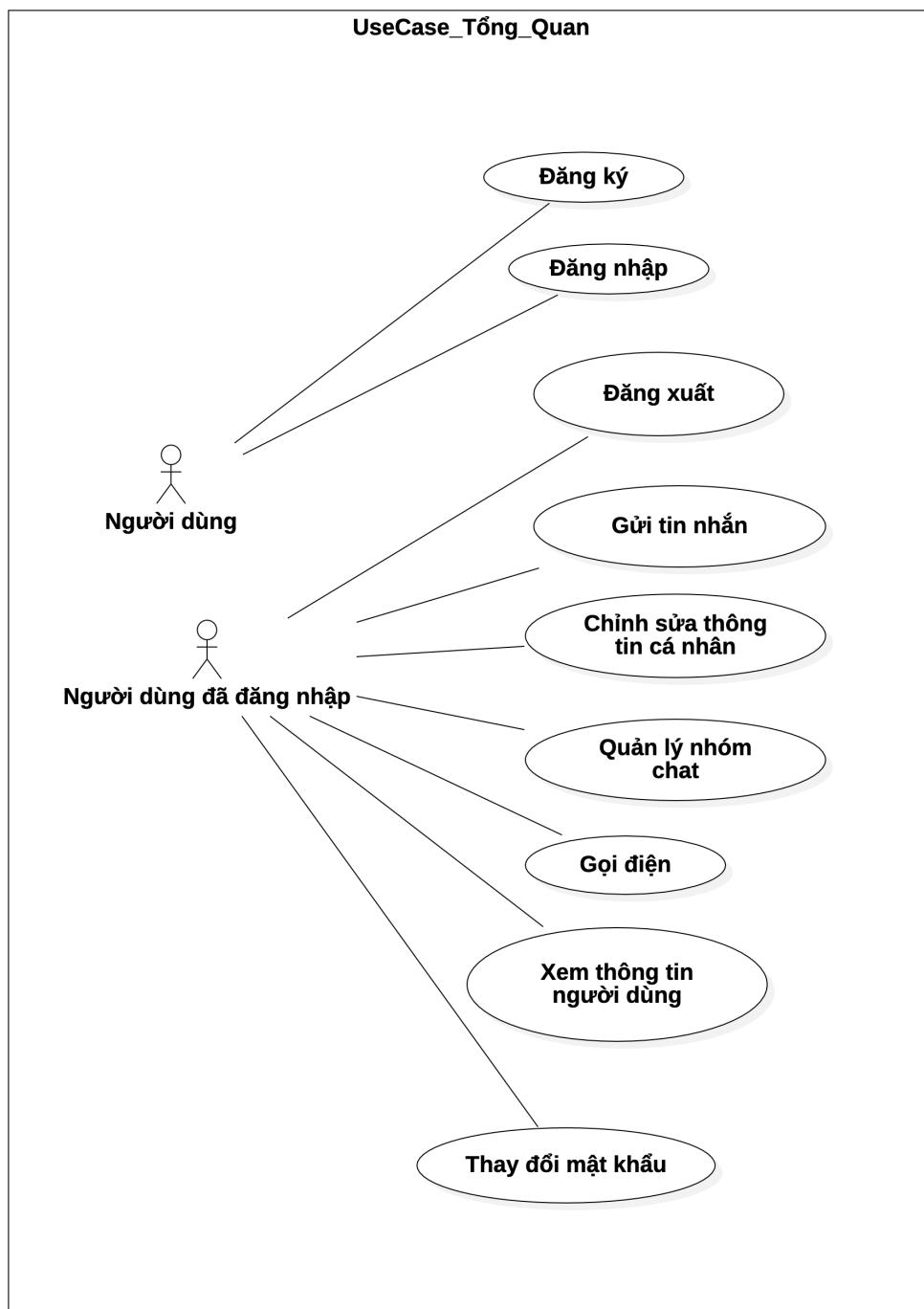
2.2.1 Biểu đồ use case tổng quát

Hệ thống bao gồm 2 tác nhân:

Người dùng: đối tượng chưa có tài khoản trên hệ thống.

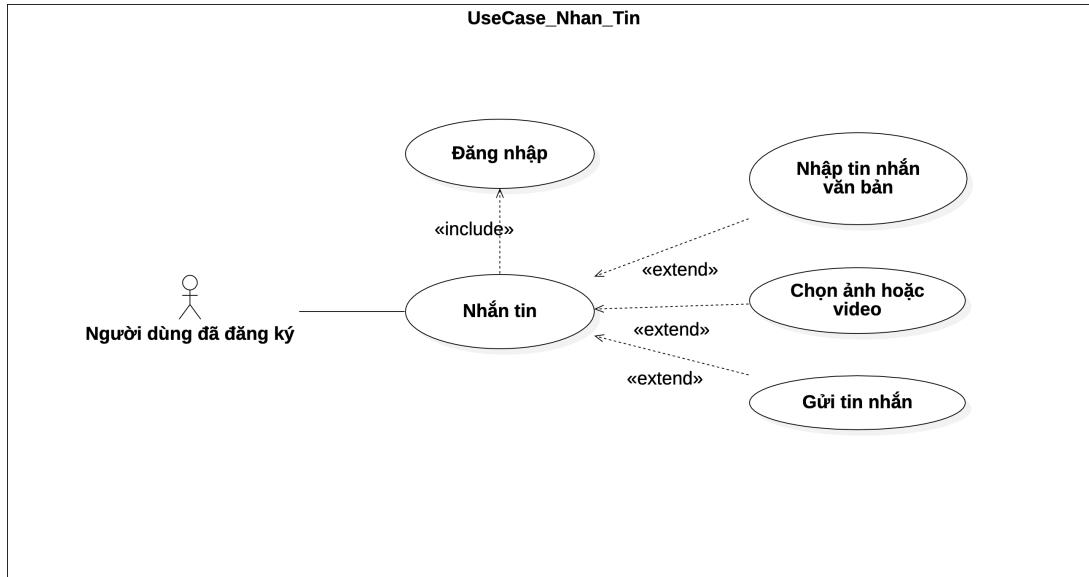
Người dùng đã đăng nhập: đối tượng đã có tài khoản và đăng nhập thành công vào hệ thống.

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU



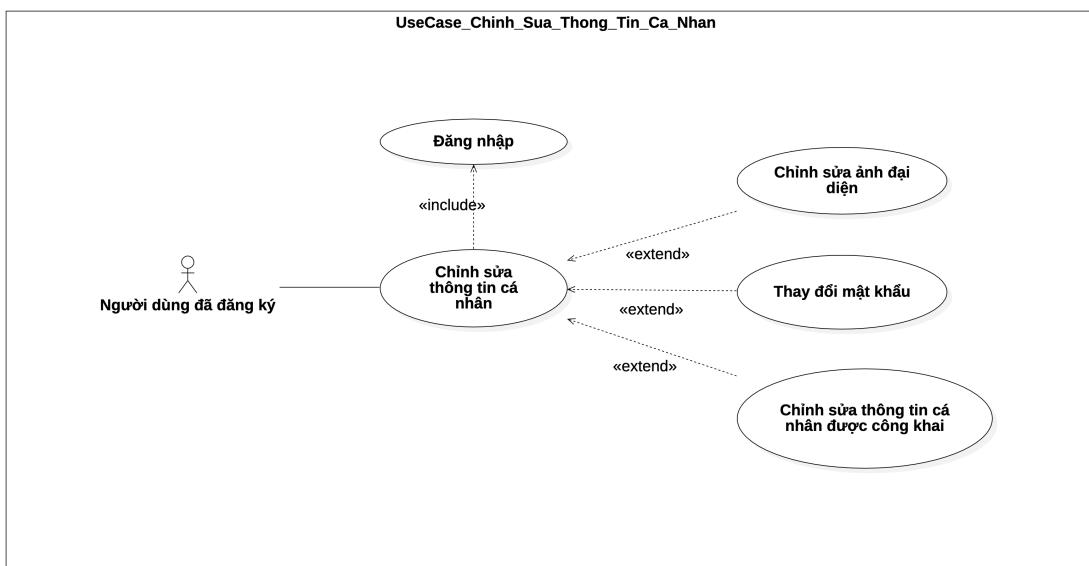
Hình 2.1: Sơ đồ use case tổng quan

2.2.2 Phân rã use case Nhắn tin



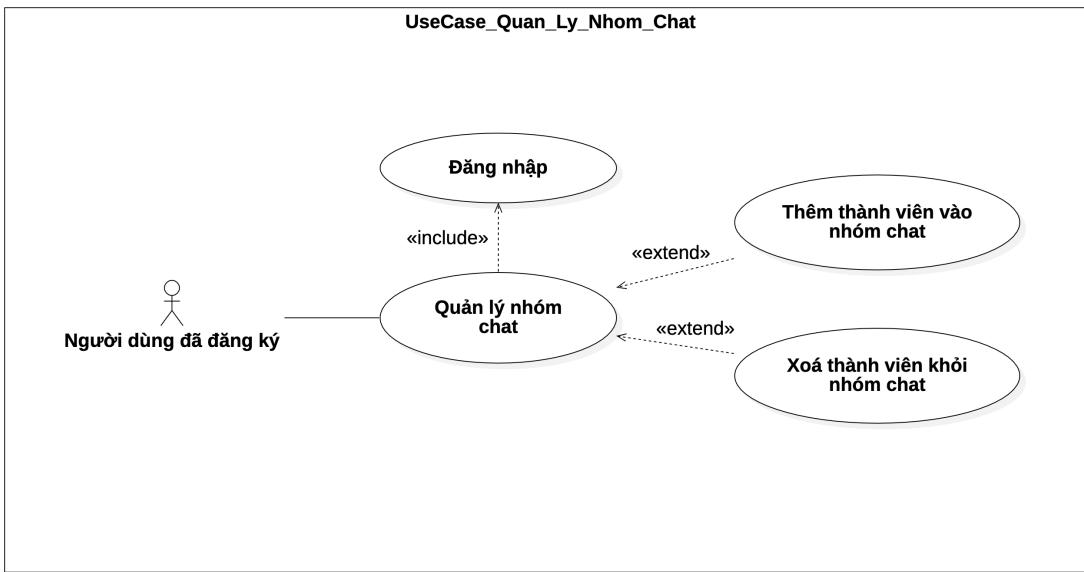
Hình 2.2: Phân rã use case Nhắn tin

2.2.3 Phân rã use case Chính sửa thông tin cá nhân



Hình 2.3: Phân rã use case Chính sửa thông tin cá nhân

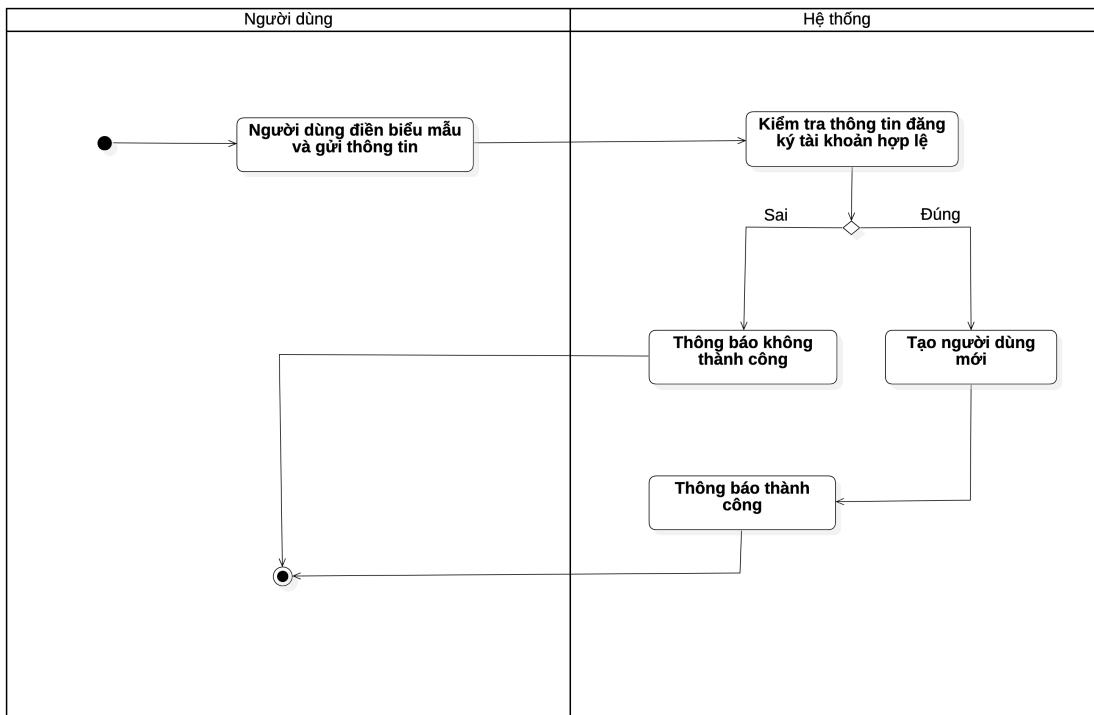
2.2.4 Phân rã use case Quản lý nhóm chat



Hình 2.4: Phân rã use case Quản lý nhóm chat

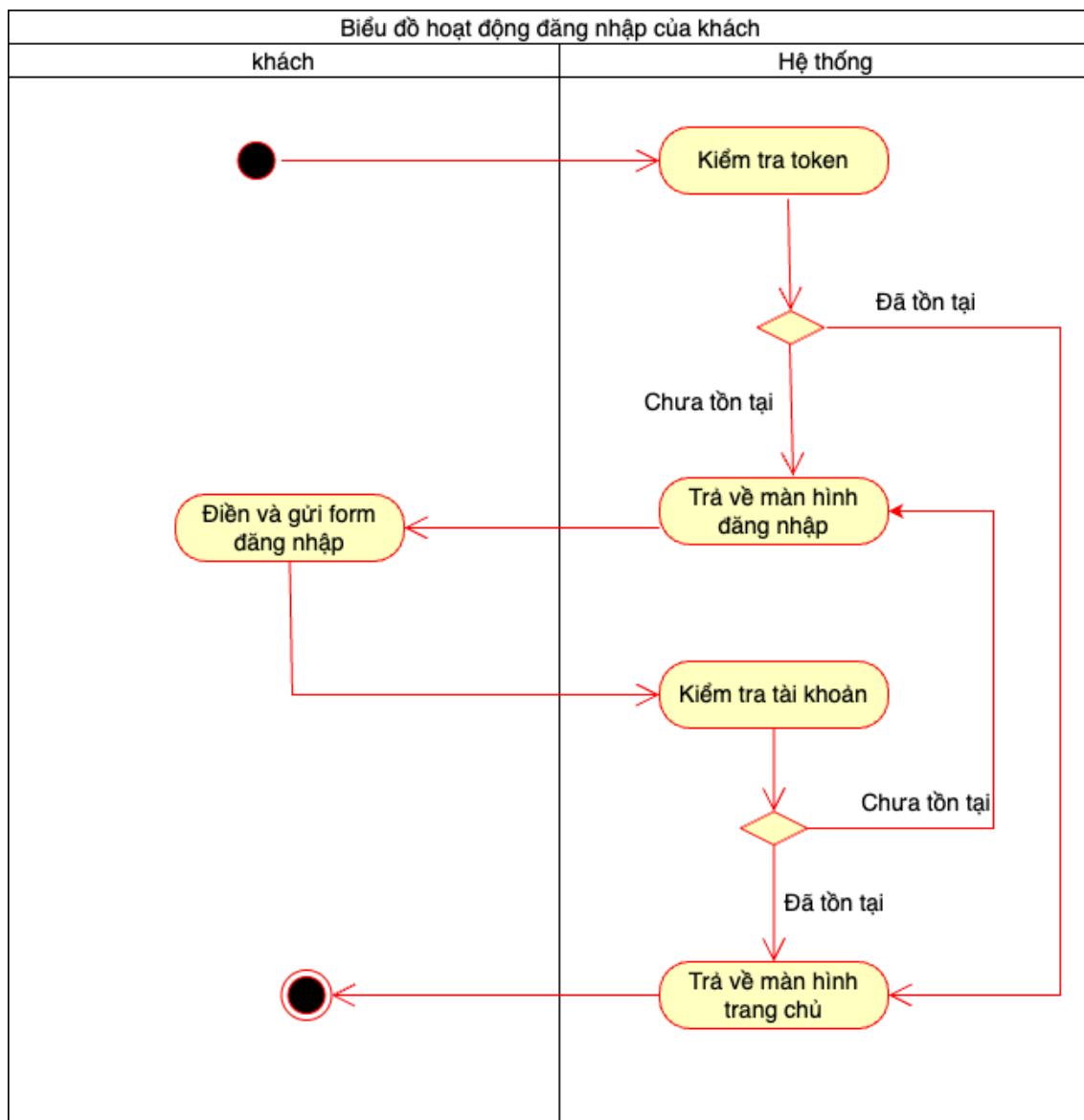
2.3 Quy trình nghiệp vụ

2.3.1 Nghiệp vụ Đăng ký



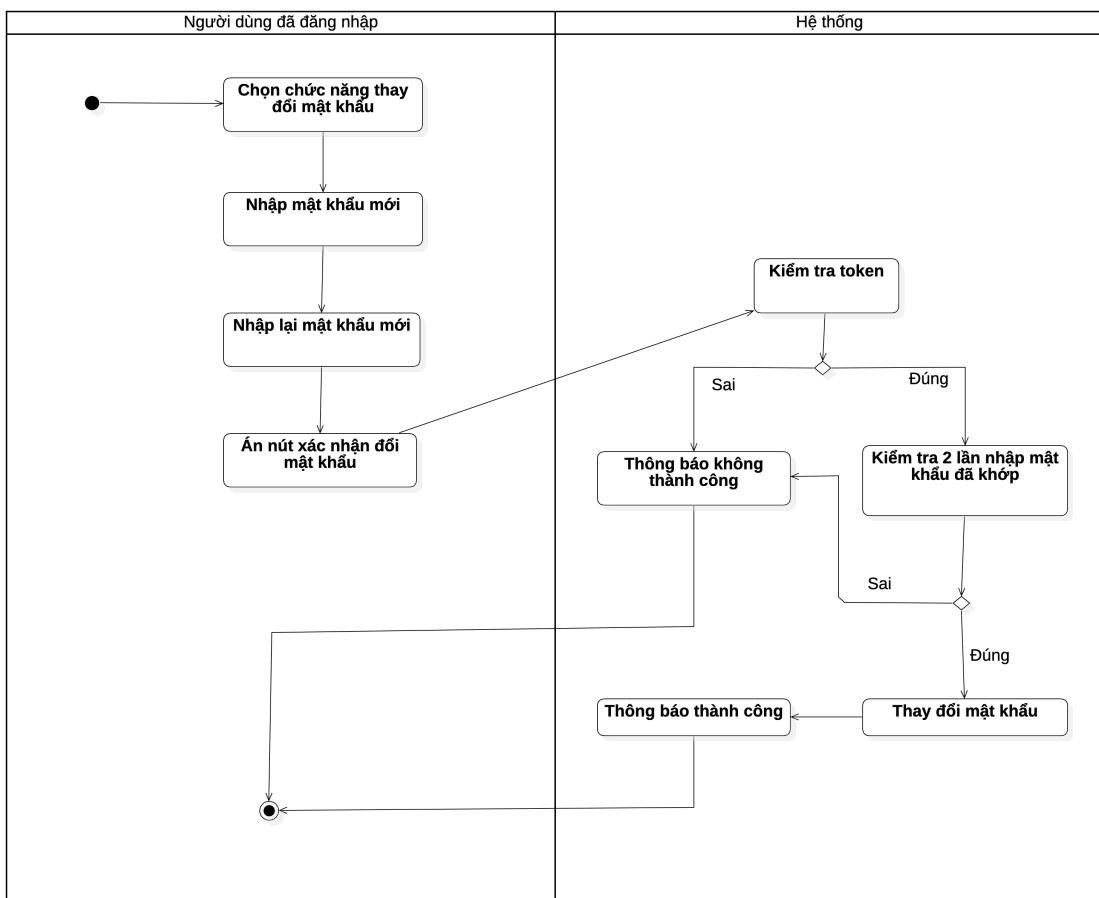
Hình 2.5: Sơ đồ hoạt động quy trình nghiệp vụ Đăng ký

2.3.2 Nghiệp vụ Đăng nhập



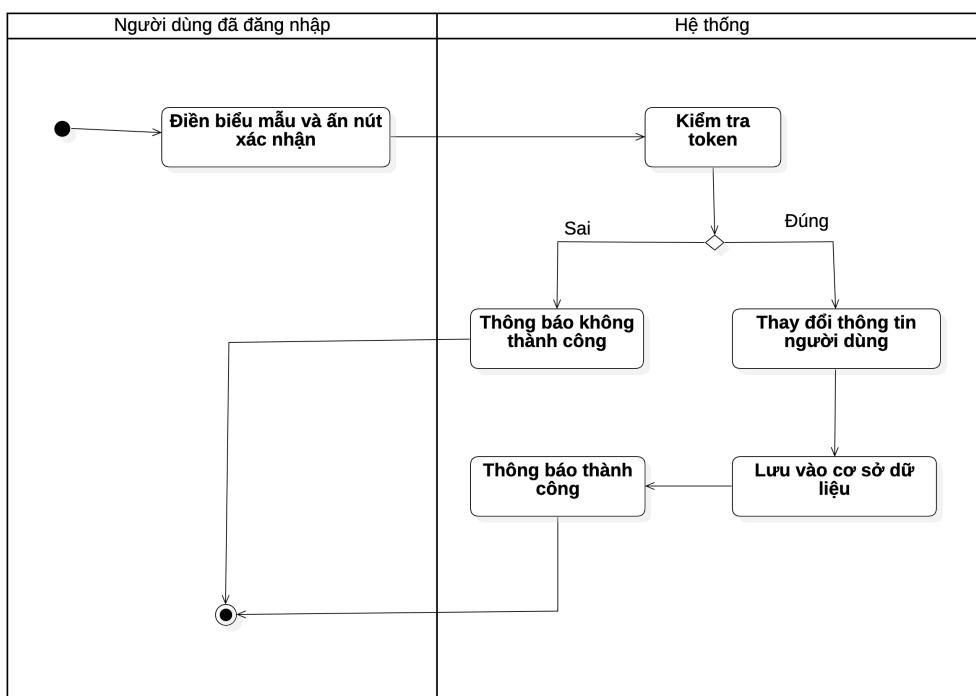
Hình 2.6: Sơ đồ hoạt động quy trình nghiệp vụ Đăng nhập

2.3.3 Nghiệp vụ Thay đổi mật khẩu



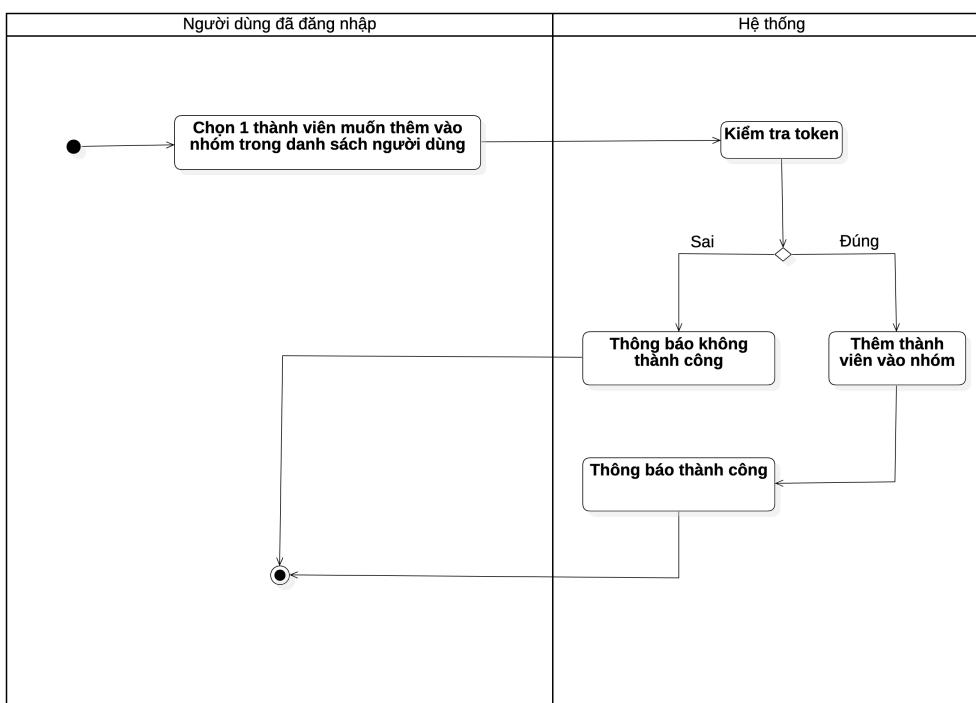
Hình 2.7: Sơ đồ hoạt động quy trình nghiệp vụ Thay đổi mật khẩu

2.3.4 Nghiệp vụ Chính sửa thông tin cá nhân được công khai



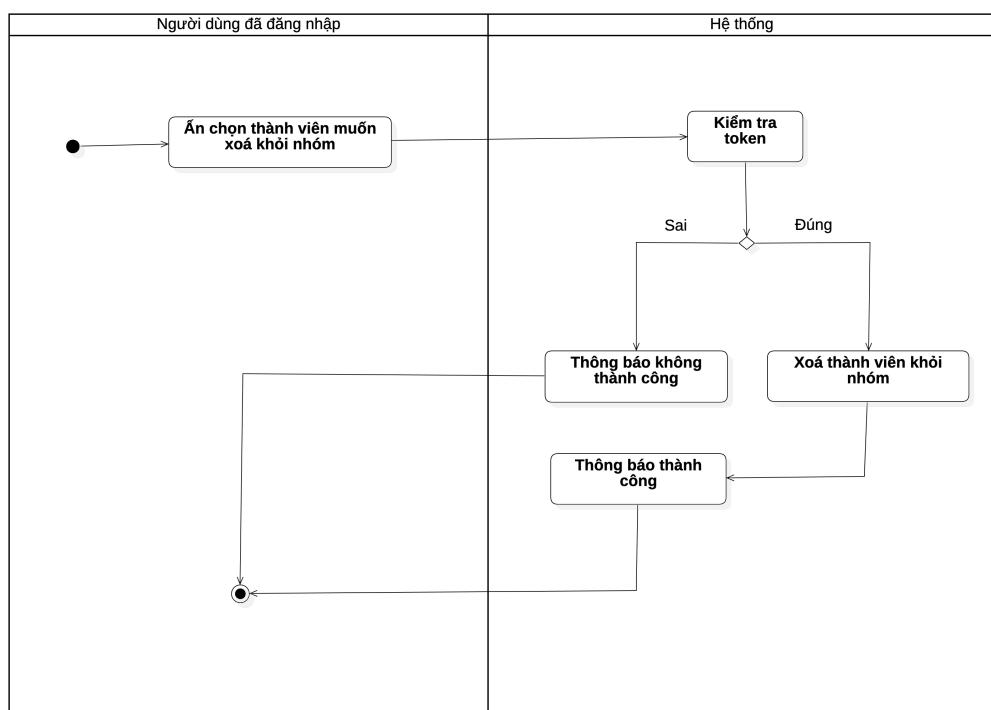
Hình 2.8: Sơ đồ hoạt động quy trình nghiệp vụ Chính sửa thông tin cá nhân được công khai

2.3.5 Nghiệp vụ Thêm thành viên vào nhóm



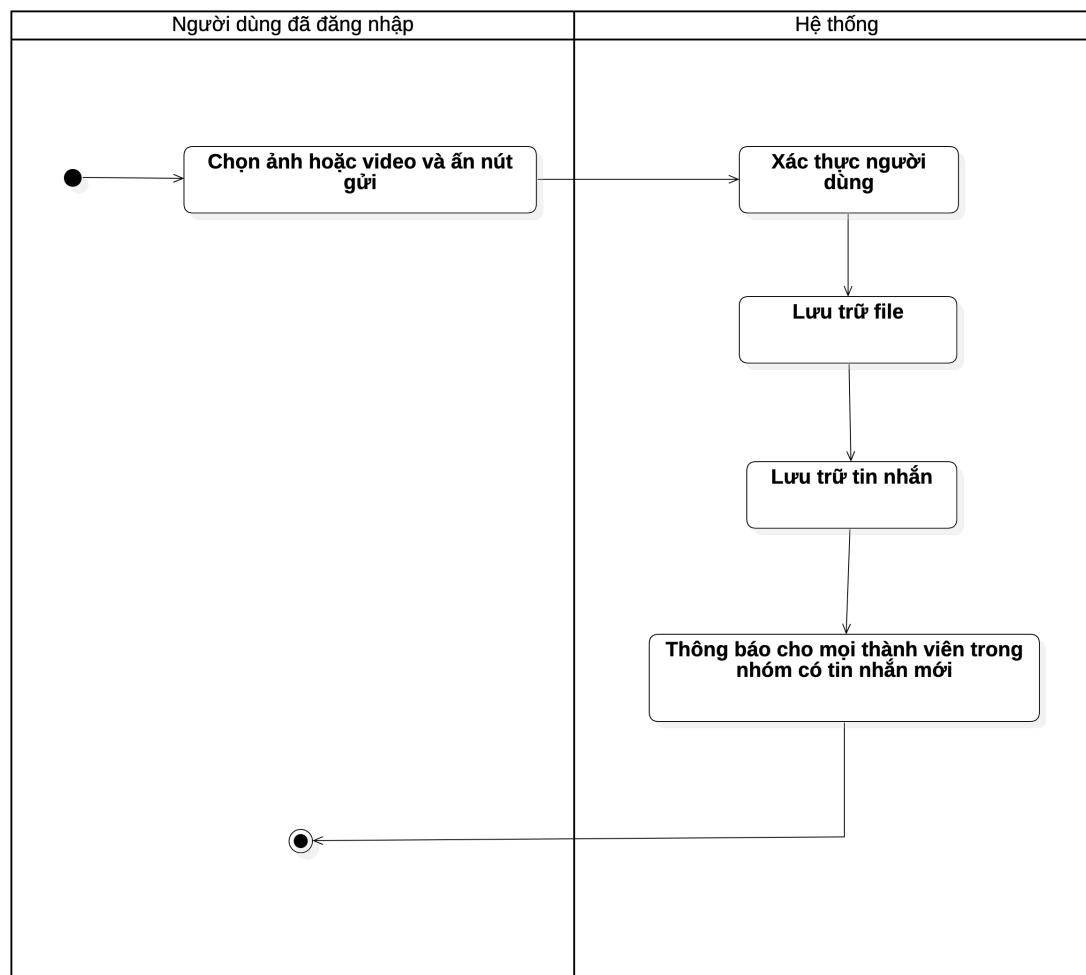
Hình 2.9: Sơ đồ hoạt động quy trình nghiệp vụ Thêm thành viên vào nhóm

2.3.6 Nghiệp vụ Xoá thành viên khỏi nhóm chat



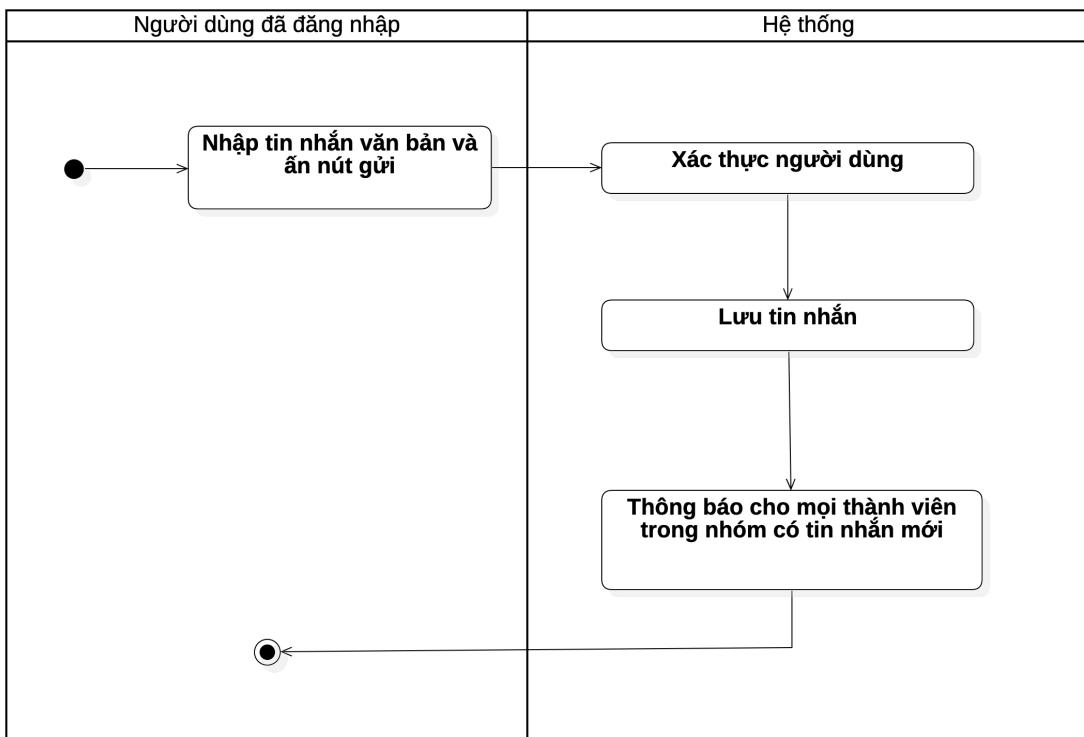
Hình 2.10: Sơ đồ hoạt động quy trình nghiệp vụ Xoá thành viên khỏi nhóm chat

2.3.7 Nghiệp vụ Gửi tin nhắn ảnh và video



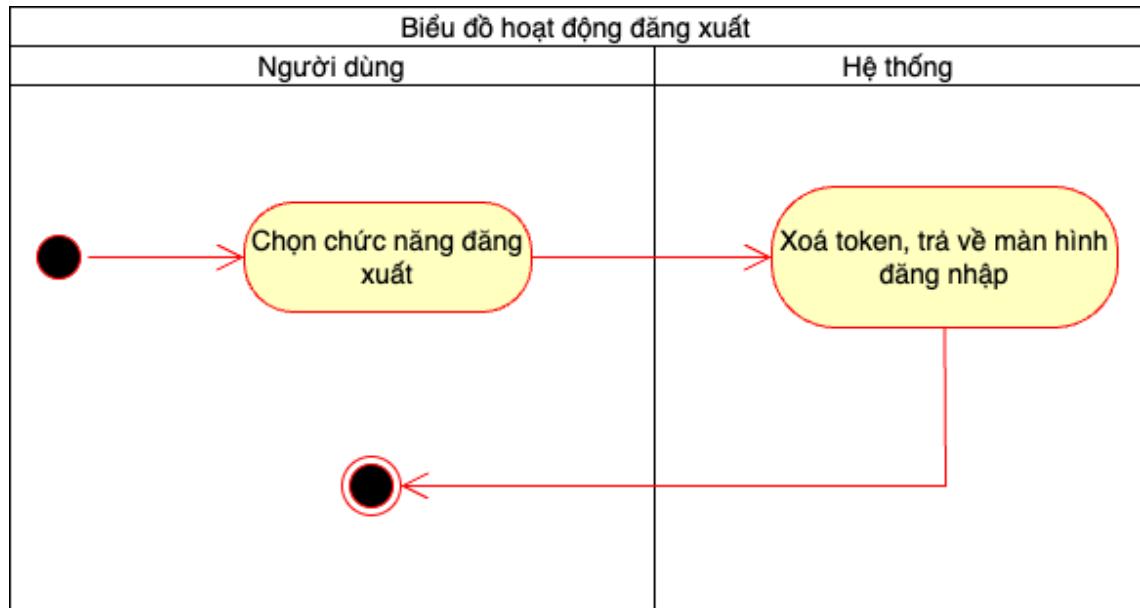
Hình 2.11: Sơ đồ hoạt động quy trình nghiệp vụ Gửi tin nhắn ảnh và video

2.3.8 Nghiệp vụ Gửi tin nhắn văn bản



Hình 2.12: Sơ đồ hoạt động quy trình nghiệp vụ Gửi tin nhắn văn bản

2.3.9 Nghiệp vụ Đăng xuất



Hình 2.13: Sơ đồ hoạt động quy trình nghiệp vụ Đăng xuất

2.4 Đặc tả ca chức năng

2.4.1 Đặc tả use case Đăng ký

Mã usecase	UC03	Tên Use case	Đăng ký
Mô tả	Cho phép người dùng đăng ký tài khoản trên hệ thống		
Tác nhân	Người dùng		
Tiền đề kiện	Không		
Sự kiện kích hoạt	Người dùng click vào button “Đăng ký”		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Điền thông tin đăng ký
	2	Người dùng	Gửi thông tin đăng ký
	3	Hệ thống	Kiểm tra dữ liệu đầu vào
	4	Hệ thống	Kiểm tra tên tài khoản xem đã tồn tại hay chưa
	5	Hệ thống	Lưu tài khoản người dùng
	6	Hệ thống	Hiển thị màn hình đăng nhập
Luồng sự kiện thay thế	STT	Thực hiện	Hành động
	3a	Hệ thống	Thông báo lỗi: Yêu cầu người dùng kiểm tra lại thông tin tài khoản, mật khẩu và mật khẩu xác nhận đã nhập
	4a	Hệ thống	Thông báo lỗi: Đã có người dùng đăng ký tên tài khoản này
Hậu điều kiện	Hiển thị màn hình đăng nhập		

Bảng 2.1: Đặc tả usecase Đăng Ký

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

Danh sách các trường dữ liệu đầu vào form đăng ký:

Thứ tự	Trường dữ liệu	Mô tả	Bắt buộc	Yêu cầu hợp lệ	Ví dụ
1	name	Tên của người dùng	Có		Vũ Quý Đạt
2	username	Tên đăng nhập của người dùng	Có	Không có ký tự khoảng trắng	VuQuyDatK62
3	password	Mật khẩu	Có	Không có ký tự khoảng trắng	DatVQ.176082@sis.hust.edu.vn
4	confirm pass-word	Mật khẩu xác nhận	Có	Không có ký tự khoảng trắng và phải trùng với Mật khẩu	DatVQ.176082@sis.hust.edu.vn
5	gender	giới tính của người dùng	Không	Hệ thống sẽ đưa 3 lựa chọn để người dùng chọn	Nam
6	bio	Thông tin giới thiệu bản thân của người dùng	Không		Tôi sinh năm 1999 đang làm việc tại công ty phát triển game trên iOS
7	birth	Ngày tháng năm sinh của người dùng	Không	Định dạng ngày/tháng/năm	02/09/1999
8	phone	Số điện thoại của người dùng	Không	Chỉ nhập số và không có ký tự khoảng trắng	0899081299
9	email	Email của người dùng	Không		datvq.176082@sis.hust.edu.vn
10	country	Quốc gia của người dùng	Không		VN, Viet Nam

Bảng 2.2: Bảng dữ liệu đầu vào usecase Đăng ký

2.4.2 Đặc tả use case Đăng nhập

Mã usecase	UC01	Tên Use case	Đăng nhập
Mô tả	Cho phép đăng nhập vào hệ thống và sử dụng các chức năng		
Tác nhân	Người dùng		
Tiền điều kiện	Người dùng đã có tài khoản được đăng ký thành công		
Sự kiện kích hoạt	Người dùng click vào button “Đăng nhập”		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Hệ thống	Kiểm tra token đã tồn tại hay chưa
	2	Người dùng	Điền thông tin đăng nhập
	3	Người dùng	Gửi yêu cầu đăng nhập
	4	Hệ thống	Hiển thị màn hình chờ và kiểm tra dữ liệu đầu vào
	5	Hệ thống	Kiểm tra xem tài khoản đã tồn tại hay chưa
	6	Hệ thống	Sinh token và lưu token
	7	Hệ thống	Ẩn màn hình chờ và hiển thị màn hình trang chủ (màn hình danh sách người dùng)
Luồng sự kiện thay thế	STT	Thực hiện	Hành động
	1a	Hệ thống	Hiển thị màn hình trang chủ (màn hình danh sách người dùng)
	4a	Hệ thống	Thông báo lỗi: Yêu cầu người dùng kiểm tra lại thông tin tài khoản và mật khẩu đã nhập. Ẩn màn hình chờ và hiển thị màn hình đăng nhập
	5a	Hệ thống	Thông báo lỗi: Đăng nhập không thành công. Ẩn màn hình chờ và hiển thị màn hình đăng nhập
Hậu điều kiện	Hiển thị màn hình trang chủ		

Bảng 2.3: Đặc tả usecase Đăng Nhập

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

Danh sách các trường dữ liệu đầu vào form đăng nhập:

Thứ tự	Trường dữ liệu	Mô tả	Bắt buộc	Yêu cầu hợp lệ	Ví dụ
1	username	Tên đăng nhập của người dùng	Có	Không có ký tự khoảng trắng	VuQuyDatK62
2	password	Mật khẩu	Có	Không có ký tự khoảng trắng	DatVQ.176082

Bảng 2.4: Bảng dữ liệu đầu vào usecase Đăng nhập

2.4.3 Đặc tả use case Đăng xuất

Mã usecase	UC02	Tên Use case	Đăng xuất
Mô tả	Cho phép người dùng đăng xuất khỏi hệ thống		
Tác nhân	Người dùng		
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống		
Sự kiện kích hoạt	Người dùng click vào button “Đăng xuất”		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Click vào button “Đăng xuất”
	2	Hệ thống	Xoá token, hiển thị màn hình đăng nhập
Luồng sự kiện thay thế	Không		
Hậu điều kiện	Đăng xuất người dùng khỏi hệ thống, hiển thị màn hình đăng nhập		

Bảng 2.5: Đặc tả usecase Đăng xuất

2.4.4 Đặc tả use case Gửi tin nhắn

Mã usecase	UC04	Tên Use case	Gửi tin nhắn
Mô tả	Gửi tin nhắn văn bản, tin nhắn ảnh, tin nhắn video		
Tác nhân	Người dùng		
Tiền điều kiện	Người dùng đã đăng nhập, tồn tại chat box với người muốn gửi tin nhắn		
Sự kiện kích hoạt	Người dùng click vào button “Gửi”		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Nhập văn bản hoặc chọn ảnh, video
	2	Người dùng	Click vào button “Gửi”
	3	Hệ thống	Kiểm tra dữ liệu đầu vào
	4	Hệ thống	Lưu tin nhắn
	5	Hệ thống	Hiển thị tin nhắn mới trên màn hình nhắn tin
Luồng sự kiện thay thế	Không		
Hậu điều kiện	Đồng bộ dữ liệu tin nhắn và hiển thị		

Bảng 2.6: Đặc tả usecase Gửi tin nhắn

Danh sách các trường dữ liệu đầu vào form Gửi tin nhắn:

Thứ tự	Trường dữ liệu	Mô tả	Bắt buộc	Yêu cầu hợp lệ	Ví dụ
1	content	Tin nhắn văn bản mà người dùng nhập vào	Có		“Đây là nội dung của tin nhắn văn bản”
2	data	Ảnh hoặc video mà người dùng đã chọn	Không		

Bảng 2.7: Bảng dữ liệu đầu vào usecase Gửi tin nhắn

2.4.5 ĐẶC CẢ USE CASE Thêm người dùng vào phòng chat

Mã usecase	UC05	Tên Use case	Thêm người dùng vào phòng chat
Mô tả	Cho phép thêm người dùng khác vào phòng chat đang tồn tại		
Tác nhân	Người dùng		
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống		
Sự kiện kích hoạt	Người dùng click vào button “Thêm”		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Chọn danh sách người dùng muốn thêm
	2	Người dùng	Click vào button “Thêm”
	3	Hệ thống	Thêm người dùng vào nhóm
Luồng sự kiện thay thế	Không		
Hậu điều kiện	Đồng bộ dữ liệu phòng chat và hiển thị		

Bảng 2.8: Đặc tả usecase Thêm người dùng vào phòng chat

2.4.6 Đặc tả use case Xoá người dùng khỏi phòng chat

Mã usecase	UC06	Tên Use case	Xoá người dùng khỏi phòng chat
Mô tả	Xoá thành viên nhóm khỏi phòng chat đang tồn tại		
Tác nhân	Người dùng		
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống, là thành viên phòng chat		
Sự kiện kích hoạt	Người dùng click thành viên và chọn “Xoá”		
Luồng sự kiện chính	STT	Thực hiện	Hành động
	1	Người dùng	Chọn danh sách người dùng muốn thêm
	2	Người dùng	Click vào button “Thêm”
	3	Hệ thống	Thêm người dùng vào nhóm
Luồng sự kiện thay thế	Không		
Hậu điều kiện	Đồng bộ dữ liệu phòng chat và hiển thị		

Bảng 2.9: Đặc tả usecase Xoá người dùng khỏi phòng chat

2.5 Các yêu cầu phi chức năng

2.5.1 Yêu cầu về bảo mật

Mật khẩu của người dùng phải được mã hoá bởi hàm băm. Các request lên server phải được đính kèm token. Tin nhắn người dùng trước khi được gửi và lưu trữ trên cơ sở dữ liệu phải được mã hoá.

Khi triển khai trên hệ thống thực tế phải sử dụng giao thức HTTPS để giao tiếp giữa server và client cho phép trao đổi thông tin một cách an toàn trên internet.

2.5.2 Yêu cầu về hiệu năng

Thời gian phản hồi chấp nhận được. Đối với các api gửi và nhận ảnh hoặc video, thời gian phản hồi trung bình cần phải nhỏ hơn kích thước của dữ liệu tính bằng MB chia cho 1(MB/1s) được coi là chấp nhận được.

2.5.3 Yêu cầu về giao diện

Ngôn ngữ sử dụng là Tiếng Việt, giao diện thân thiện với người dùng. Hỗ trợ giao diện sáng và tối.

Bố cục của ứng dụng: phải được thiết kế phù hợp với nhiều loại màn hình của

CHƯƠNG 2. KHẢO SÁT VÀ PHÂN TÍCH YÊU CẦU

các phiên bản iPhone. Tỷ lệ, kích thước và vị trí những phần tử hiển thị phải được sắp xếp hợp lý.

Màu sắc: màu sắc giữa các thành phần của ứng dụng phải có sự thống nhất, sử dụng 2 màu chính là trắng và xanh nước biển.

Thiết kế nút bấm: nút bấm sử dụng các icon tượng hình. Các icon được cung cấp bởi Apple. Tất cả các icon đều tuân theo chuẩn Material Design cả về màu sắc lẫn kích thước.

Thứ tự, cấu trúc các màn hình: các màn hình phải được sắp xếp hợp lý, chỉ nên cần tối đa 3 đến 4 chạm (số lần tương tác trên màn hình giao diện) để người dùng thực hiện được mục đích của mình.

Với những màn hình hiển thị dữ liệu từ server trả về, công việc đồng bộ dữ liệu từ server phải được đưa vào luồng phụ, tránh cho giao diện bị giật và không phản hồi lại thao tác người dùng khi sử dụng.

CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG

3.1 Ngôn ngữ lập trình Swift

Swift là một ngôn ngữ lập trình trực hướng đối tượng dành cho iOS, iPadOS, macOS, tvOS và watchOS. Là kết quả của nghiên cứu mới nhất về ngôn ngữ lập trình, kết hợp với kinh nghiệm hàng chục năm xây dựng nền tảng của Apple và cộng đồng mã nguồn mở. Phiên bản Swift mới nhất tính đến hiện tại là 5.7[1].

Swift có các ưu điểm là cú pháp ngắn gọn nhưng vẫn diễn đạt rõ ràng, tốc độ xử lý nhanh. Swift được thiết kế giúp lập trình viên loại bỏ những lỗi lập trình phổ biến bằng cách áp dụng các mẫu lập trình hiện đại, loại bỏ toàn bộ các lớp mã không an toàn:

- Các biến luôn được khởi tạo trước khi sử dụng.
- Các kiểu dữ liệu như mảng và số nguyên luôn được kiểm tra xem có bị tràn không.
- Bộ nhớ được quản lý tự động và thực thi quyền truy cập độc quyền vào bộ bảo vệ bộ nhớ để chống lại nhiều lỗi lập trình.

Một trong những lợi thế lớn của Swift trong phát triển ứng dụng di động là khả năng tương tác tốt giữa Swift - Objective C, thậm chí có thể dùng cả 2 ngôn ngữ Swift và Objective C trong cùng một dự án, trình biên dịch XCode vẫn chạy bình thường. Ngoài ra Swift được thiết kế để thực thi trên đa nền tảng từ BackEnd đến FrontEnd. Điều này giúp cho những lập trình viên đã quen với ngôn ngữ Swift có thể xây dựng các ứng dụng full-stack.

Với tất cả những ưu điểm trên và kinh nghiệm phát triển phần mềm bằng Swift sẵn có của bản thân, em lựa chọn Swift là ngôn ngữ lập trình chính để xây dựng phần mềm trên hệ điều hành iOS cho đề tài này.

3.2 Vapor

Vapor là một web framework được viết bằng ngôn ngữ Swift, cung cấp cho nhà phát triển khả năng xây dựng websites, xây dựng back-ends cho ứng dụng iOS, APIs và HTTP servers bằng Swift, có thể cài đặt Vapor trên các hệ điều hành MacOS và Ubuntu[2].

Những ưu điểm khi sử dụng Vapor để xây dựng back-end:

- Vapor được viết bằng ngôn ngữ Swift nên cú pháp ngắn gọn và dễ hiểu, tốc độ xử lý nhanh.

- Từ phiên bản Swift 5.7 được Apple hỗ trợ cú pháp bắt đồng bộ nên Vapor càng trở nên mạnh mẽ khi phát triển back-end.
- Tương thích tốt với hệ quản trị cơ sở dữ liệu PostgreSQL, MySQL và MongoDB.
- Tích hợp ORM framework, giúp câu truy vấn cơ sở dữ liệu ngắn gọn, dễ đọc và dễ hiểu hơn.

Với nhiều ưu điểm thừa kế từ Swift và tài liệu dành cho nhà phát triển từ Vapor dễ tìm kiếm, dễ hiểu, em lựa chọn Vapor để xây dựng phần back-end.

3.3 WebRTC

WebRTC là một tập hợp các hàm lập trình dùng cho việc liên lạc thời gian thực bằng video, âm thanh cũng như các loại dữ liệu khác. WebRTC có thể giúp 2 client gọi điện video ngay trong trình duyệt mà không cần đăng ký tài khoản, cũng không cần cài thêm plugin gì phức tạp, ngoài ra chúng còn được xài để phát triển game chơi trực tiếp trong trình duyệt và rất nhiều loại ứng dụng khác[3].

Em sử dụng WebRTC để phát triển tính năng gọi điện video thời gian thực trên phần mềm iOS của mình.

3.4 WebSocket

WebSocket là công nghệ hỗ trợ giao tiếp hai chiều giữa client và server bằng cách sử dụng một TCP socket để tạo một kết nối hiệu quả và ít tốn kém. Mặc dù được thiết kế để chuyên sử dụng cho các ứng dụng web, nhà phát triển vẫn có thể đưa chúng vào bất kỳ loại ứng dụng nào[4].

WebSocket cung cấp khả năng giao tiếp hai chiều hiệu quả, header gọn nhẹ nên sử dụng ít lưu lượng hơn giao thức HTTP truyền thống, có độ trễ thấp và dễ xử lý lỗi. Vì vậy, em lựa chọn webSocket để xây dựng tính năng nhắn tin thời gian thực của đề tài.

3.5 Hệ quản trị cơ sở dữ liệu PostgreSQL

PostgreSQL là một hệ quản trị cơ sở dữ liệu mã nguồn mở được ưa chuộng hàng đầu, được phát triển, phân phối và hỗ trợ bởi phòng khoa học máy tính tại đại học California. PostgreSQL được thiết kế để chạy trên các nền tảng tương tự UNIX. Tuy nhiên, PostgreSQL sau đó cũng được điều chỉnh linh động để có thể chạy được trên nhiều nền tảng khác nhau như Mac OS X, Solaris và Windows[5].

PostgreSQL không yêu cầu quá nhiều công tác bảo trì vì có tính ổn định cao. Do đó, nếu phát triển các ứng dụng dựa trên PostgreSQL, chi phí sở sỡ thấp hơn so với các hệ thống quản trị dữ liệu khác.

PostgreSQL được đánh giá là một hệ quản trị cơ sở dữ liệu có tốc độ cao, ổn định, dễ dùng, có khả năng thay đổi mô hình phù hợp với điều kiện công việc, độ bảo mật cao, đa tính năng, có khả năng mở rộng và mạnh mẽ.

Về nguyên tắc PostgreSQL hoạt động dựa trên mô hình client-server. Cốt lõi của PostgreSQL là máy chủ PostgreSQL, xử lý tất cả các hướng dẫn cơ sở dữ liệu hoặc các lệnh. Máy chủ PostgreSQL có sẵn như một chương trình riêng biệt để sử dụng trong môi trường client-server. PostgreSQL hoạt động cùng một số chương trình tiện ích hỗ trợ quản trị cơ sở dữ liệu PostgreSQL. Các lệnh được gửi đến PostgreSQL server thông qua máy khách được cài đặt trên máy tính.

Với những ưu điểm vượt trội, sự kết hợp của ngôn ngữ lập trình Swift với tính ổn định của PostgreSQL trên nền tảng MacOS, em lựa chọn PostgreSQL để xây dựng back-end.

3.6 MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng. MongoDB là một hệ quản trị cơ sở dữ liệu hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh[6]. Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ như PostgreSQL sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng. So với RDBMS thì trong MongoDB collection ứng với bảng, còn document sẽ ứng với các hàng của bảng, MongoDB sẽ dùng các document thay cho hàng trong RDBMS. Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định. Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB.

Do vậy em sử dụng MongoDB để xây dựng tính năng lưu trữ các dữ liệu ảnh, video trên back-end.

3.7 Docker

[7] nền tảng phần mềm cho phép dựng, kiểm thử và triển khai ứng dụng một cách nhanh chóng. Có thể coi Docker như một máy ảo cho phép cài đặt môi trường, cấu hình hệ thống, mọi thứ cần thiết để có thể chạy chương trình[7].

Có 2 khái niệm chính trong Docker:

- Image: Là một ảnh đóng gói của môi trường, chứa tất cả các thành phần (tệp, thư viện, phụ thuộc, cấu hình,...) để có thể khởi chạy ứng dụng.
- Container: Là một thực thể của Image (được tạo ra khi chạy Image). Các con-

tainer hoạt động độc lập nhau và với hệ điều hành chủ. Có thể khởi tạo hay gỡ bỏ một container rất dễ dàng.

Những ưu điểm khi sử dụng Docker làm môi trường cài đặt server:

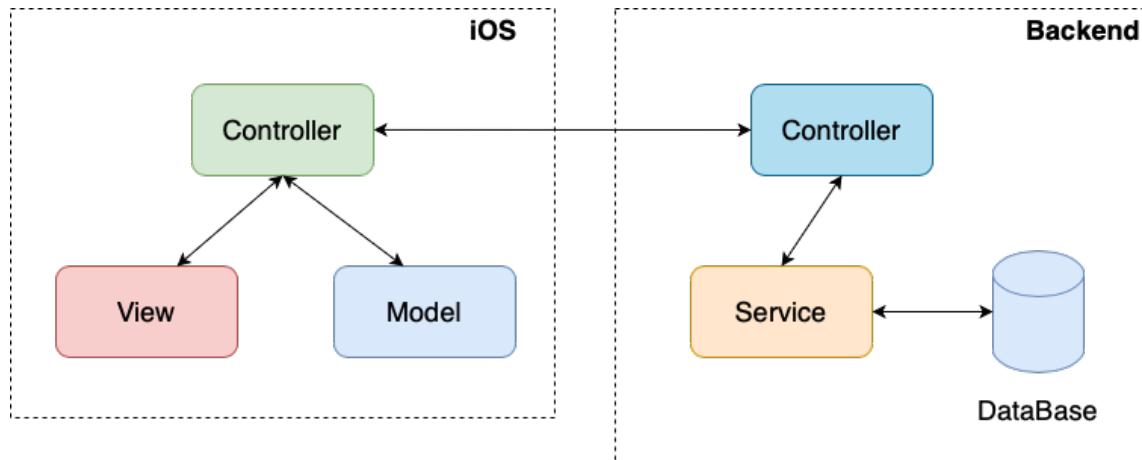
- Tính đồng nhất: Khi một ứng dụng được cài đặt hoặc phát triển trên nhiều thiết bị khác nhau sẽ không bị sai khác về mặt môi trường.
- Khả năng chia sẻ: Nhà phát triển có thể dễ dàng tìm thấy các Image được chia sẻ bởi cộng đồng trên Docker Hub. Điều này giúp giảm bớt thời gian tạo Image so với cách thông thường, tăng tốc độ phát triển phần mềm.

Trong quá trình xây dựng đồ án của mình, em dùng Docker thiết lập môi trường và chạy các ứng dụng back-end: Vapor, PostgreSQL, MongoDB.

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG

4.1 Thiết kế kiến trúc

4.1.1 Lựa chọn kiến trúc phần mềm



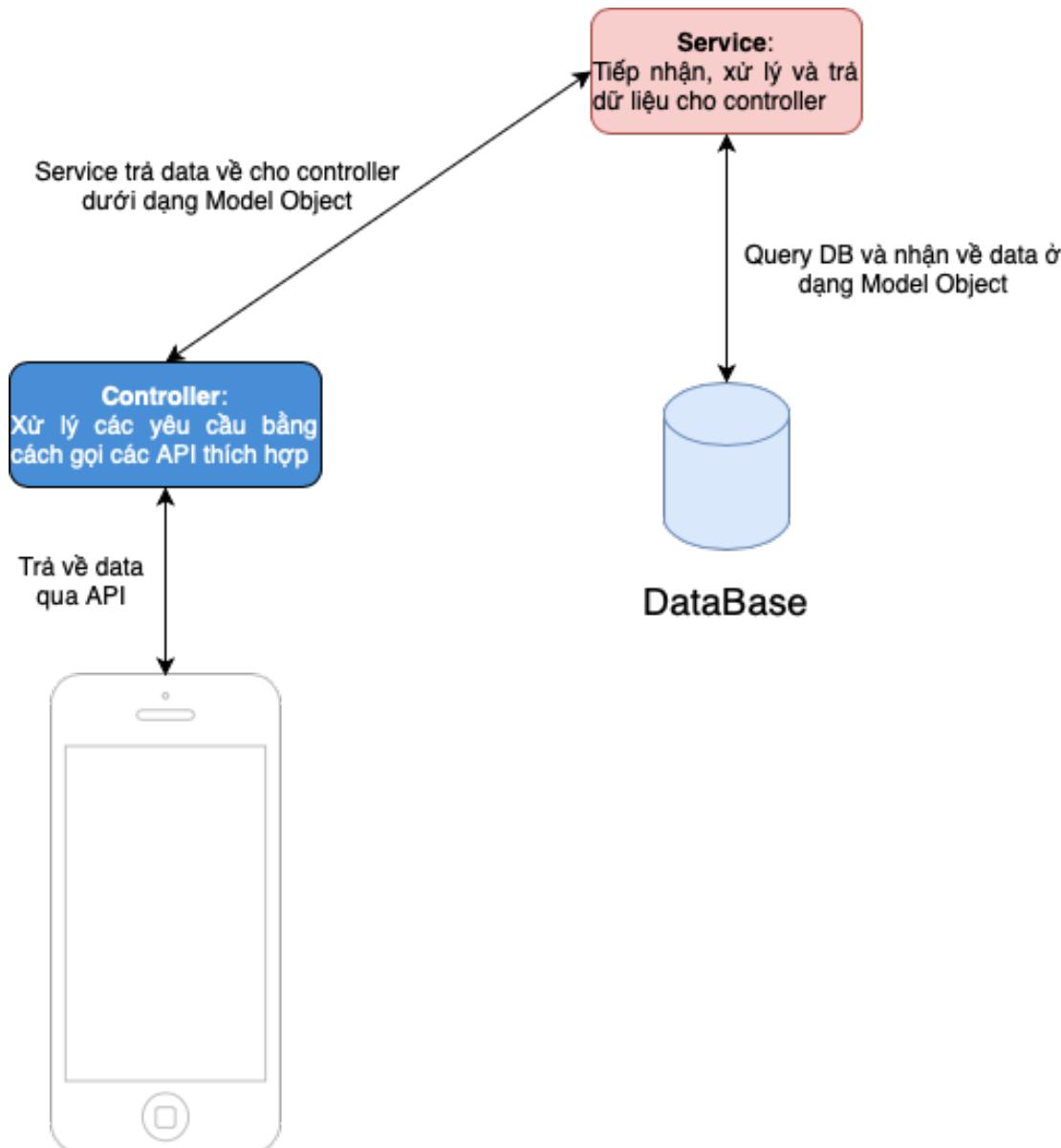
Hình 4.1: Kiến trúc tổng quan của hệ thống

Phần BackEnd của hệ thống được xây dựng bằng Vapor-Swift, dựa trên mô hình kiến trúc hướng sự kiện. Phần iOS được xây dựng dựa trên kiến trúc MVC.

Ưu điểm của việc sử dụng kiến trúc hướng sự kiện được sử dụng trong back-end:

- Có khả năng chịu tải cao hơn.
- Các thành phần trong hệ thống không còn phụ thuộc lẫn nhau.
- Khả năng mở rộng tốt.
- Phát triển backend nhanh, đơn giản, dễ lập trình, dễ bảo trì.
- Dễ dàng kiểm tra: dễ dàng hơn trong việc kiểm tra, rà soát lỗi do các controller được phân tách một cách độc lập.

Luồng dữ liệu của mô hình: Người dùng gửi yêu cầu bằng các thao tác trên giao diện, Controller trên iOS client sẽ kiểm tra dữ liệu từ Model và yêu cầu Model đồng bộ dữ liệu với Server, Controller trên Server sẽ nhận được request và giao yêu cầu đến Service tương ứng. Service truy cập vào Database truy vấn và trả về dữ liệu là các Model Object cho Controller. Controller trên server nhận được Model Object sẽ trả về cho Client thông qua lời gọi API. Model trên iOS client Nhận được dữ liệu sẽ gọi hàm CallBack đến Controller trên iOS Client, sau đó Controller cập nhật giao diện.



Hình 4.2: Luồng hoạt động của BackEnd

Mô hình Model-View-Controller (MVC) là một mẫu kiến trúc phân tách một ứng dụng thành ba thành phần logic chính Model, View và Controller. Do đó viết tắt MVC. Mỗi thành phần kiến trúc được xây dựng để xử lý khía cạnh phát triển cụ thể của một ứng dụng. MVC tách lớp logic nghiệp vụ và lớp hiển thị ra riêng biệt. Một số ưu điểm của MCV:

- Bảo trì code dễ dàng, dễ dàng mở rộng và phát triển.
- Việc phát triển các thành phần khác nhau có thể được thực hiện song song.
- Giúp nhà phát triển tránh sự phức tạp bằng cách chia ứng dụng thành ba đơn vị Model, View và Controller.
- Cung cấp hỗ trợ tốt nhất cho phát triển theo hướng thử nghiệm.

MVC là một mô hình phổ biến và dễ triển khai trong thiết kế xây dựng phần mềm. Ứng dụng được xây dựng dựa trên mô hình MVC được chia làm 3 phần: Model, View, Controller. Mỗi thành phần là độc lập với nhau và có vai trò riêng:

View:

- View là một phần của ứng dụng đại diện cho việc trình bày dữ liệu.
- View được tạo bởi các dữ liệu mà chúng ta lấy từ dữ liệu trong model. Một view yêu cầu model cung cấp đầy đủ dữ liệu để nó hiển thị đầu ra cho người dùng.
- View chính là nơi chứa những giao diện như một nút bấm, vùng nhập dữ liệu từ bàn phím, menu, hình ảnh, nó đảm nhiệm nhiệm vụ hiển thị dữ liệu và giúp người dùng tương tác với hệ thống.

Controller:

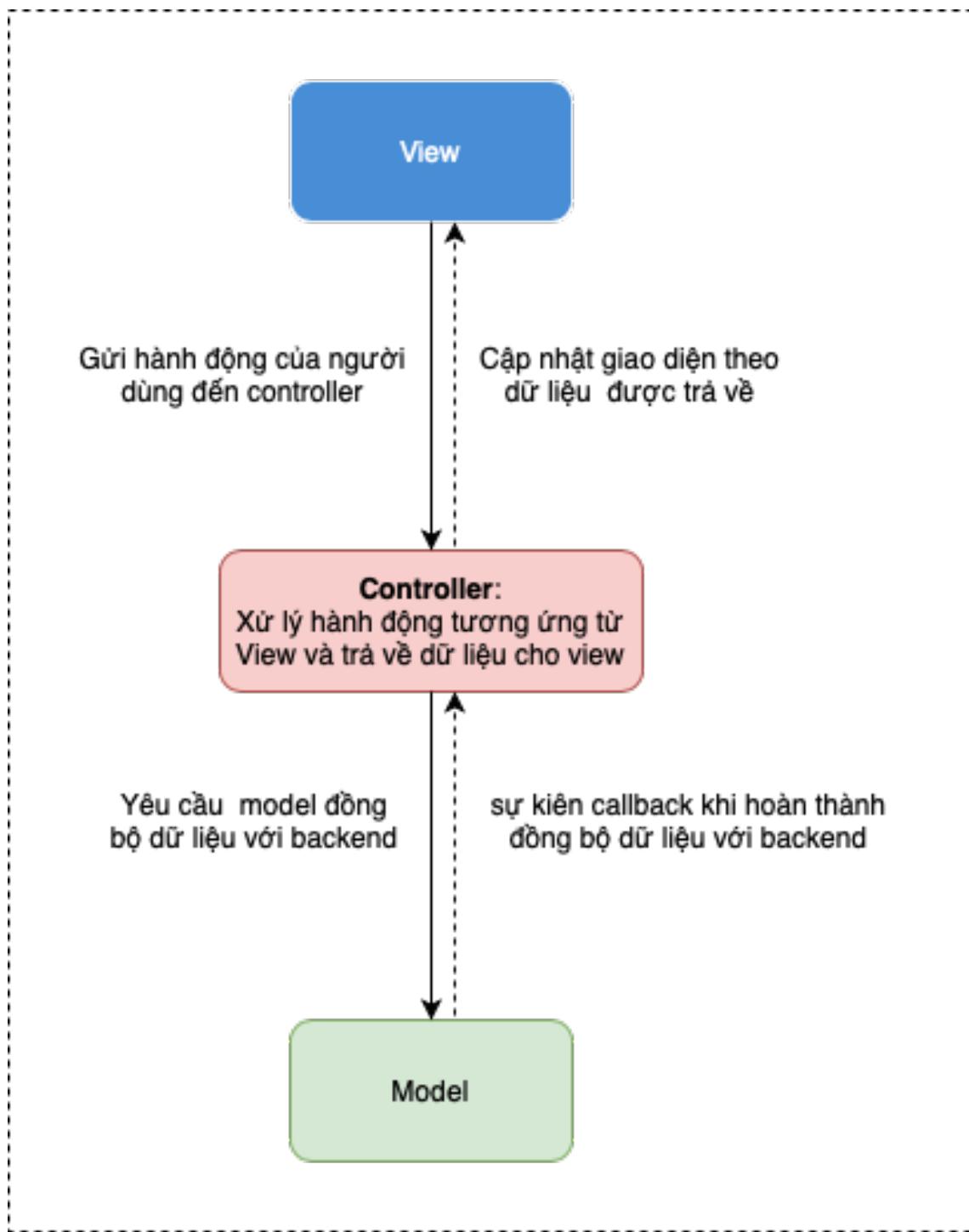
- Controller là một phần của ứng dụng xử lý tương tác của người dùng. Trên iOS, Controller sẽ nhận sự kiện kích hoạt từ View khi người dùng thao tác trên giao diện.
- Controller bao gồm những lớp và hàm xử lý nghiệp vụ logic giúp lấy đúng dữ liệu thông tin cần thiết nhờ các nghiệp vụ lớp Model cung cấp và hiển thị dữ liệu đó ra cho người dùng nhờ lớp View.
- Controller gửi các lệnh đến model để làm thay đổi trạng thái của nó. Controller cũng gửi các lệnh đến view liên quan của nó để thay đổi cách hiển thị của view.

Model:

- Thành phần model lưu trữ dữ liệu và logic liên quan của nó. Bao gồm các hàm xử lý các tác vụ như truy vấn, thêm, sửa hoặc xóa dữ liệu. Nó thao tác với dữ liệu trên Server thông qua API hoặc dữ liệu lưu trên thiết bị sau đó sử dụng nó để hiển thị dữ liệu.

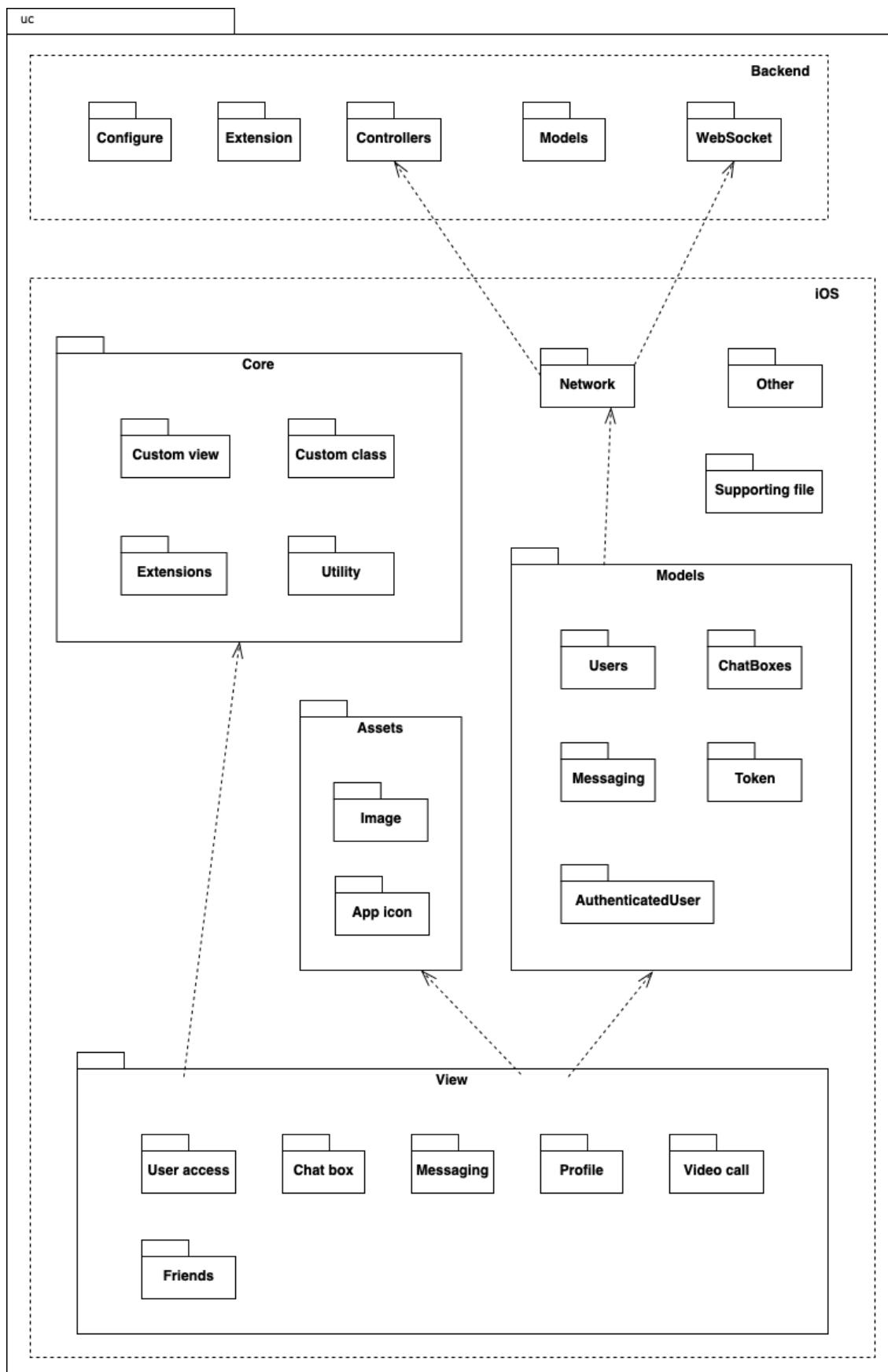
Sự tương tác giữa các thành phần trong mô hình MVC:

- Controller tương tác qua lại với View.
- Controller tương tác qua lại với Model.
- Model và View không có sự tương tác với nhau trực tiếp mà nó tương tác với nhau thông qua Controller.



Hình 4.3: Luồng hoạt động của ứng dụng trên iOS theo mô hình MVC

4.1.2 Thiết kế tổng quan



Hình 4.4: Thiết kế gói tổng quan

Hệ thống được chia làm 2 thành phần chính: back-end và iOS Client. Thành phần back-end bao gồm 5 gói chính: (i) Configure, (ii) Extension, (iii) Controllers, (iv) Models, (v) WebSocket:

Gói Configure: Gói chứa các cấu hình database, cấu hình liên quan đến mạng và cấu hình của server.

Gói Extension: Gói chứa các khai báo mở rộng phương thức cho các class, các kiểu dữ liệu có sẵn.

Gói Controllers: Gói controllers khai báo các class xử lý logic, điều phối lời gọi API.

Gói WebSocket: Gói chứa cái khai báo phục vụ cho việc quản lý Socket cho nhiều người dùng.

Thành phần iOS Client bao gồm 7 gói chính: (i) Core, (ii) Models, (iii) Network, (iv) Assets, (v) View, (vi) Other, (vii) Supporting file:

Gói Core: Gói chứa các class có thể tái sử dụng trong ứng dụng, như điều khiển phản hồi rung của thiết bị, màn hình loading. Ngoài ra gói chứa các khai báo mở rộng phương thức cho các class, các kiểu dữ liệu có sẵn.

Gói Models: Gói đại diện cho thành phần model trong kiến trúc MVC, cung cấp dữ liệu và khả năng lưu trữ dữ liệu vào bộ nhớ cho ứng dụng.

Gói NetWork: Gói chứa các class có chức năng giao tiếp với server.

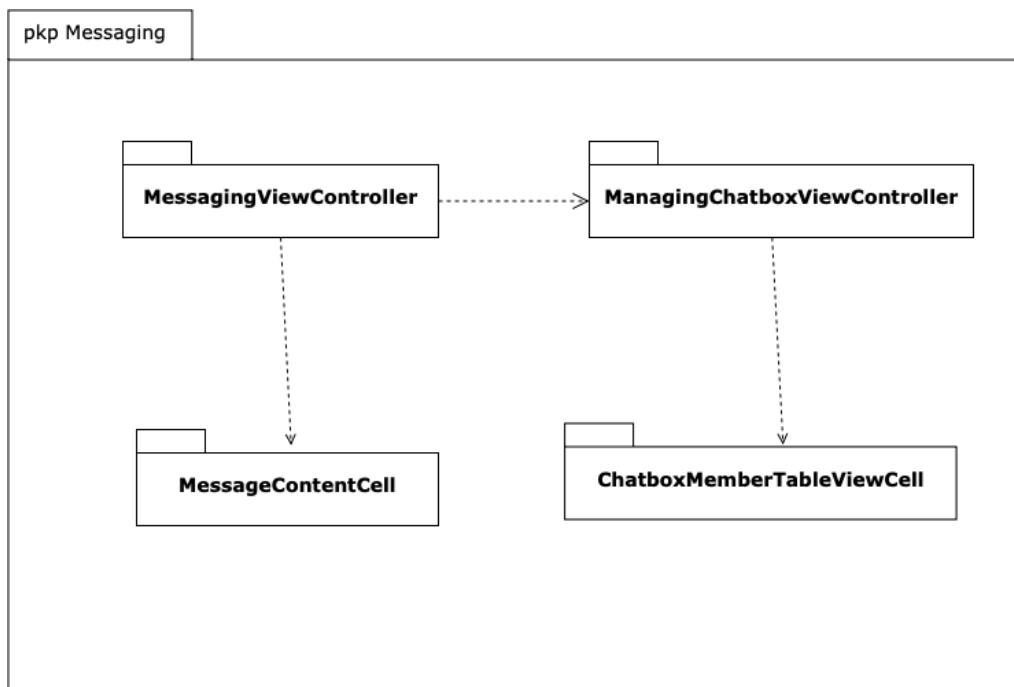
Gói Assets: Gói chứa dữ liệu media như ảnh của các button, video, ảnh logo của ứng dụng.

Gói View: Gói đại diện cho thành phần View trong kiến trúc MVC, gói chứa các gói thành phần, mỗi gói thành phần bao gồm View và controller trong nó. Gói có nhiệm vụ hiển thị dữ liệu, tương tác trực tiếp với người dùng cũng như xử lý nghiệp vụ logic.

Gói Other: Chứa các thành phần mà trình biên dịch tự động tạo ra hỗ trợ lập trình viên trong quá trình phát triển phần mềm, như kiểm thử tự động, khai báo quyền truy cập trên thiết bị, vòng đời của phần mềm,..

Gói Supporting file: Chứa các thư viện ngoài, khai báo dữ liệu mẫu,..

4.1.3 Thiết kế chi tiết gói Messaging

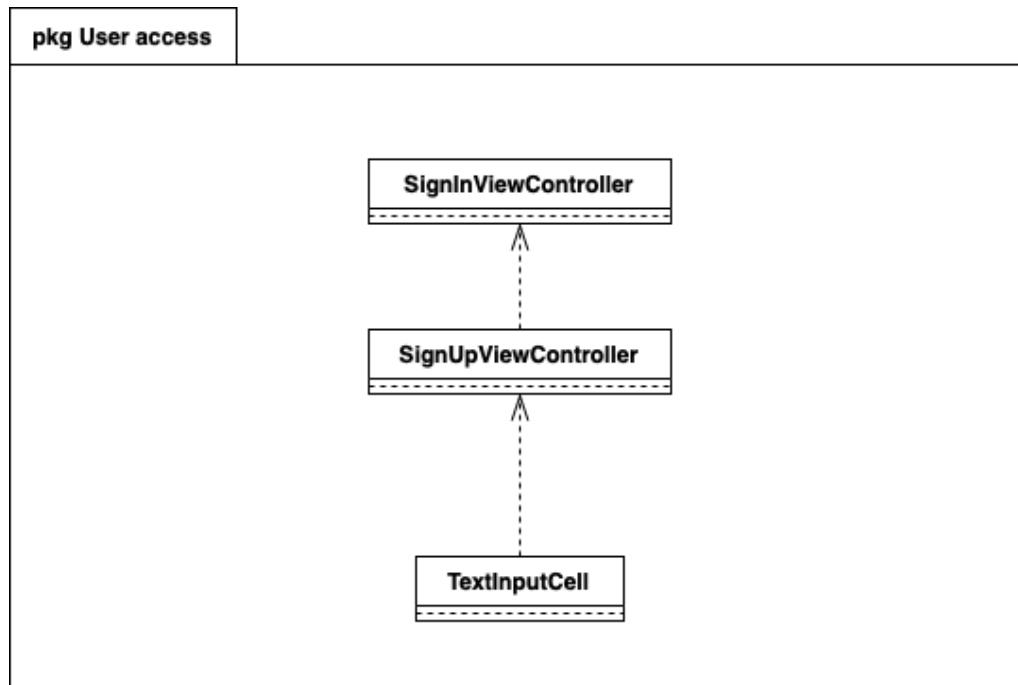


Hình 4.5: Thiết kế chi tiết gói Messaging

Hình 4.5 mô tả chi tiết gói Messaging. Bao gồm các lớp : (i) MessagingViewController, (ii) ManagingChatboxViewController, (iii) MessageContentCell, (iv) ChatBoxMemberTableViewCell. Trong đó từng lớp có nhiệm vụ sau:

- MessagingViewController: Controller xử lý các nghiệp vụ logic gửi tin nhắn và đồng bộ tin nhắn.
- ManagingChatboxViewController: Controller xử lý các nghiệp vụ logic thêm thành viên và xoá thành viên khỏi nhóm.
- MessageContentCell: View hiển thị tin nhắn văn bản hoặc ảnh.
- ChatBoxMemberTableViewCell: View hiển thị người dùng.

4.1.4 Thiết kế chi tiết gói User Access

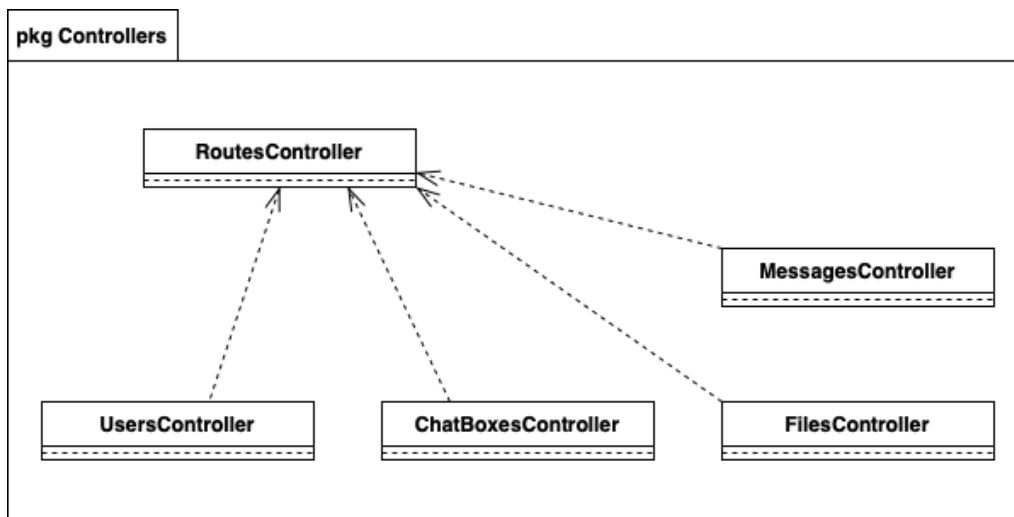


Hình 4.6: Thiết kế chi tiết gói User Access

Hình 4.6 mô tả chi tiết gói User Access. Bao gồm các lớp : (i) SignInViewController, (ii) SignUpViewController, (iii) TextInputCell. Trong đó từng lớp có nhiệm vụ sau:

- SignInViewController: Controller xử lý các nghiệp vụ logic màn hình đăng nhập.
- SignInViewController.xib: Nơi thiết kế giao diện màn hình đăng nhập.
- SignUpViewController: Controller xử lý các nghiệp vụ logic màn hình đăng ký.
- SignUpViewController.xib: Nơi thiết kế giao diện màn hình đăng ký.
- TextInputCell: Controller xử lý logic hiển thị cho vùng nhập dữ liệu.
- TextInputCell.xib: Đối tượng View cho phép nhập dữ liệu từ bàn phím.

4.1.5 Thiết kế chi tiết gói Controllers

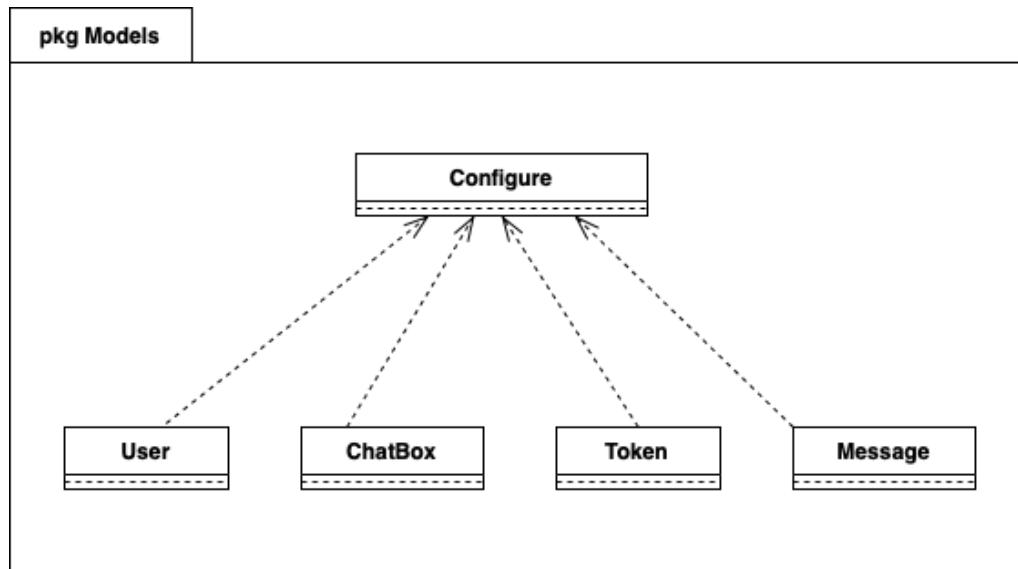


Hình 4.7: Thiết kế chi tiết gói Controllers

Hình 4.7 mô tả chi tiết gói Controllers. Bao gồm các class: (i) RoutesController, (ii) UsersController, (iii) ChatBoxesController, (iv) MessagesController. Trong đó nhiệm vụ của từng class như sau:

- RoutesController: Nơi khai báo class điều phối các Controller. Khi có một yêu cầu từ Client gọi đến, nó sẽ dựa vào đường dẫn và dữ liệu gửi tới để điều phối công việc cho Controller thích hợp.
- UsersController: Nơi khai báo class xử lý các nghiệp vụ logic liên quan đến bảng users.
- ChatBoxesController: Nơi khai báo class xử lý các nghiệp vụ logic liên quan đến bảng chatBoxes.
- MessagesController: Nơi khai báo class xử lý các nghiệp vụ logic liên quan đến bảng messages.

4.1.6 Thiết kế chi tiết gói Models



Hình 4.8: Thiết kế chi tiết gói Models

Hình 4.7 mô tả chi tiết gói Models. Bao gồm các struct: (i) User, (ii) ChatBox, (iii) MappingChatBoxPivot, (iv) Message. Trong đó nhiệm vụ của từng class như sau:

- User: Nơi khai báo các trường trong bảng users, mỗi đối tượng của struct này thể hiện 1 người dùng.
- ChatBox: Nơi khai báo các trường trong bảng chatBoxes, mỗi đối tượng của struct này thể hiện 1 phòng trò chuyện (1 chat box).
- MappingChatBoxPivot: Nơi khai báo mối liên hệ giữa 2 bảng users và chatBoxes. Nó thể hiện quan hệ 1 - nhiều giữa bảng users và chatBoxes.
- Message: Nơi khai báo các trường trong bảng messages, mỗi đối tượng của struct này thể hiện 1 tin nhắn đã được gửi đi và lưu thành công.

4.2 Thiết kế chi tiết

4.2.1 Thiết kế giao diện

Đối với việc xây dựng, phát triển ứng dụng trên di động, thiết kế giao diện là bước quan trọng. Kết quả của công đoạn thiết kế giao diện quyết định đến tần suất sử dụng phần mềm, hiệu năng tương tác của người dùng và dựa trên một số yếu tố sau:

Bố cục của ứng dụng: phải được thiết kế phù hợp với nhiều loại màn hình của các phiên bản iPhone. Tỷ lệ, kích thước và vị trí những phần tử hiển thị phải được sắp xếp hợp lý.

Màu sắc: màu sắc giữa các thành phần của ứng dụng phải có sự thống nhất, sử dụng 2 màu chính là trắng và xanh nước biển.

Thiết kế nút bấm: nút bấm sử dụng các icon tượng hình. Các icon được cung cấp bởi Apple. Tất cả các icon đều tuân theo chuẩn Material Design cả về màu sắc lẫn kích thước.

Thứ tự, cấu trúc các màn hình: các màn hình phải được sắp xếp hợp lý, chỉ nên cần tối đa 3 đến 4 chạm (số lần tương tác trên màn hình giao diện) để người dùng thực hiện được mục đích của mình.

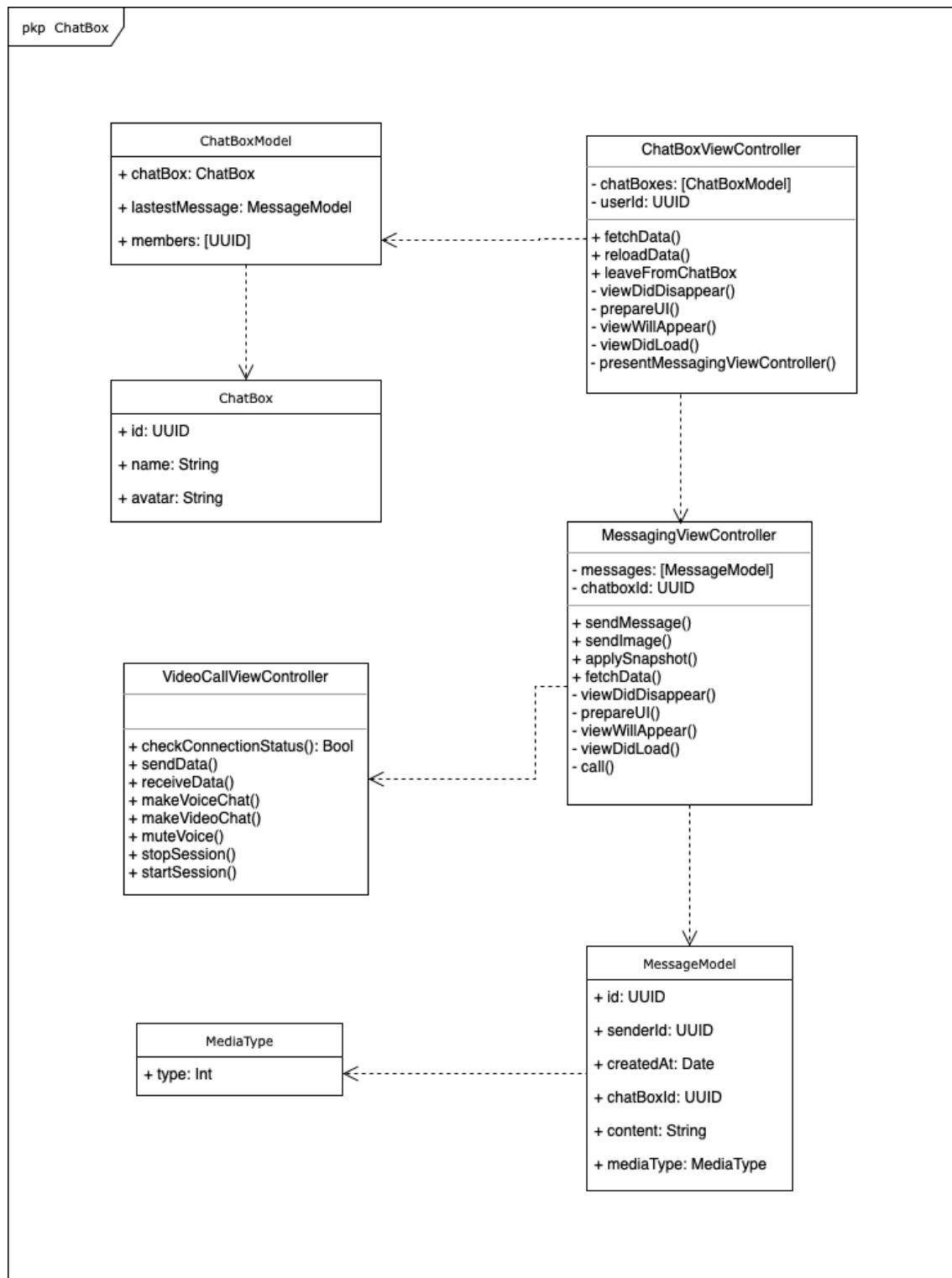
Với những màn hình hiển thị dữ liệu từ server trả về, công việc đồng bộ dữ liệu từ server phải được đưa vào luồng phụ, tránh cho giao diện bị giật và không phản hồi lại thao tác người dùng khi sử dụng.

4.2.2 Thiết kế lớp

a, Thiết kế chi tiết lớp ChatBoxViewController

Thiết kế chi tiết lớp ChatBoxViewController được miêu tả như hình vẽ. Lớp ChatBoxViewController phụ thuộc vào 2 lớp là MessagingViewController và ChatBoxModel. ChatBoxViewController hiển thị danh sách các nhóm chat, MessagingViewController hiển thị danh sách các tin nhắn trong 1 nhóm chat cụ thể, ChatBoxModel có nhiệm vụ lấy ra dữ liệu cho ChatBoxViewController hiển thị, sau đó MessagingViewController được ChatBoxViewController gọi đến và lớp MessageModel lấy dữ liệu cho MessagingViewController hiển thị. Lớp MessagingViewController cần tham số đầu vào là chatBoxId để có thể lấy ra những tin nhắn thuộc nhóm chat. Lớp VideoCallViewController thực hiện nhiệm vụ xử lý logic khi thực hiện gọi video, lớp MessagingViewController gọi đến lớp VideoCallViewController và hiển thị.

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG



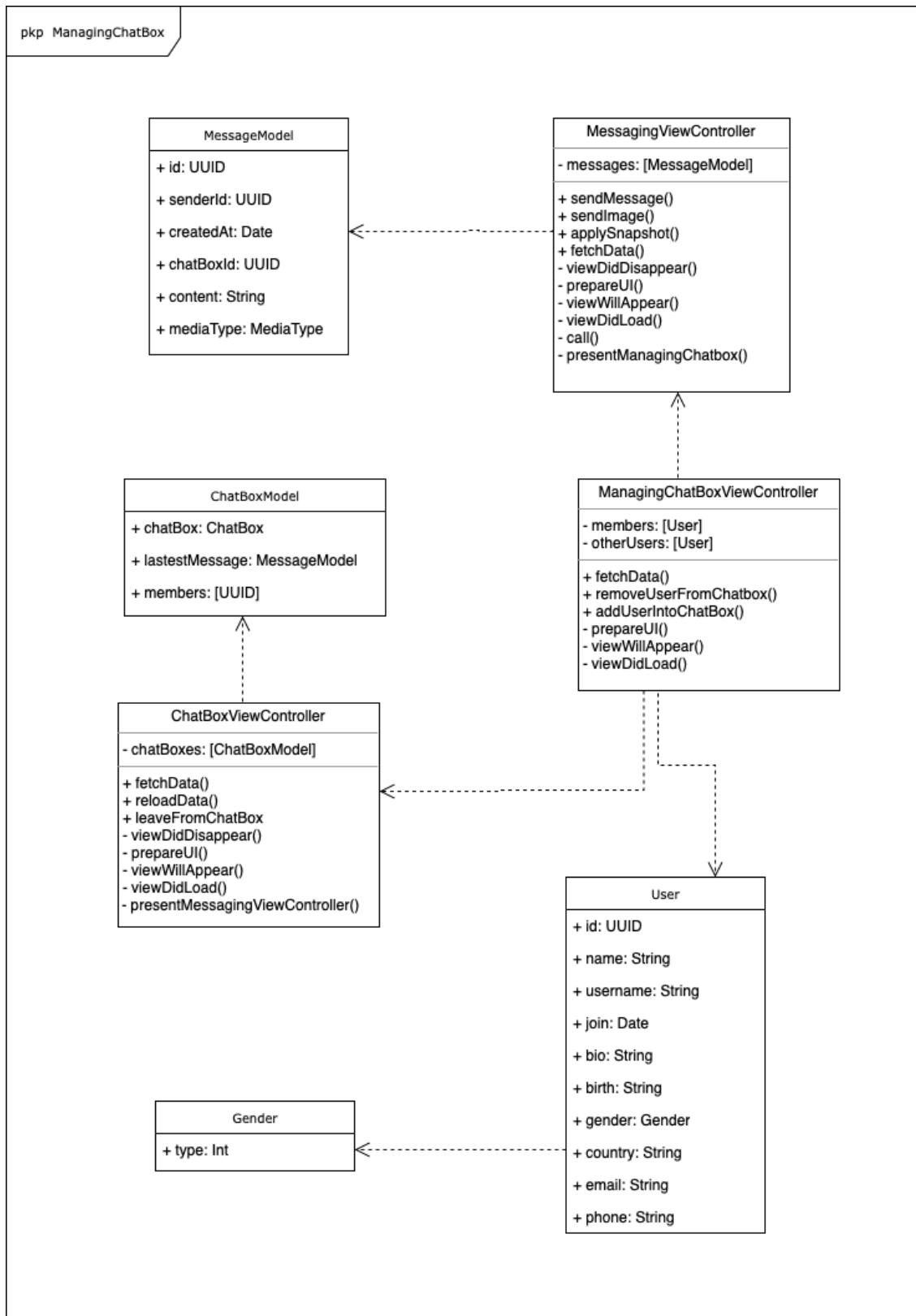
Hình 4.9: Thiết kế chi tiết lớp ChatBoxViewController

b, Thiết kế chi tiết lớp ManagingChatBoxViewController

Thiết kế chi tiết lớp **ManagingChatBoxViewController** được miêu tả như hình vẽ. Lớp **ManagingChatBoxViewController** phụ thuộc vào 2 lớp là **MessagingViewController** và **ChatBoxViewController**. **ManagingChatBoxViewController** có nhiệm vụ hiển thị danh sách thành viên thuộc nhóm và những người dùng không thuộc

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG

nhóm, ManagingChatBoxViewController cần tham số đầu vào là danh sách những thành viên thuộc nhóm. Khi sự kiện thêm hoặc xoá người dùng, ManagingChatBoxViewController thông báo đến MessagingViewController và ChatBoxViewController để chúng cập nhật dữ liệu.



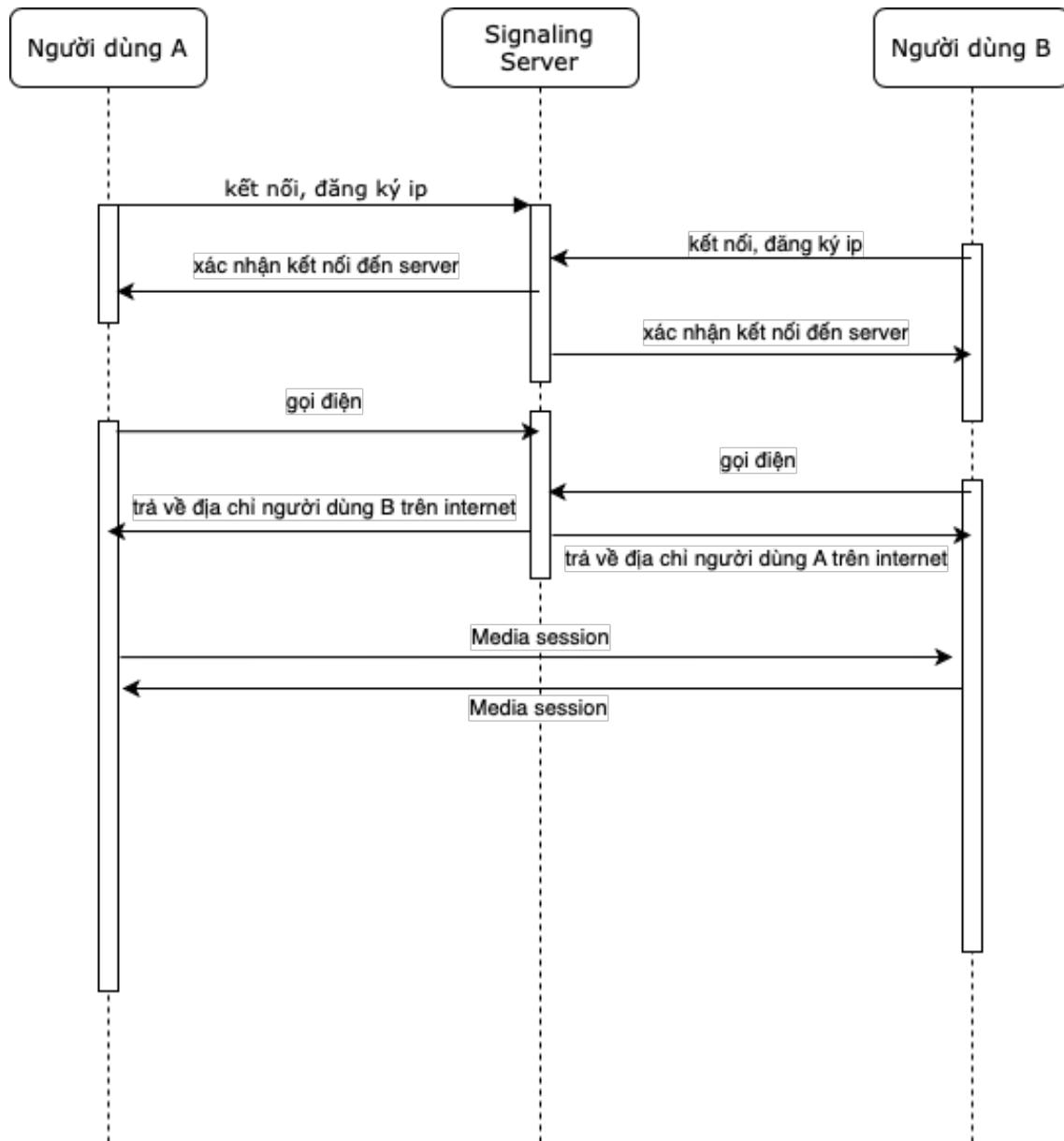
Hình 4.10: Thiết kế chi tiết lớp ManagingChatBoxViewController

4.2.3 Biểu đồ tuần tự tính năng gọi điện video

Theo biểu đồ mô tả, khi ứng dụng trên client iOS chạy sẽ tự động gửi ip của mình cho signaling server, nếu 2 client muốn gọi đàm thoại video cho nhau, chúng sẽ gửi yêu cầu lên signaling server và lấy về ip của nhau. Khi 2 client có được ip của nhau, kết nối peer-to-peer được tạo ra. Khi đường truyền thiết lập và mở, các luồng media stream được khởi tạo và nhờ đó 2 client iOS gửi dữ liệu media trực tiếp cho nhau.

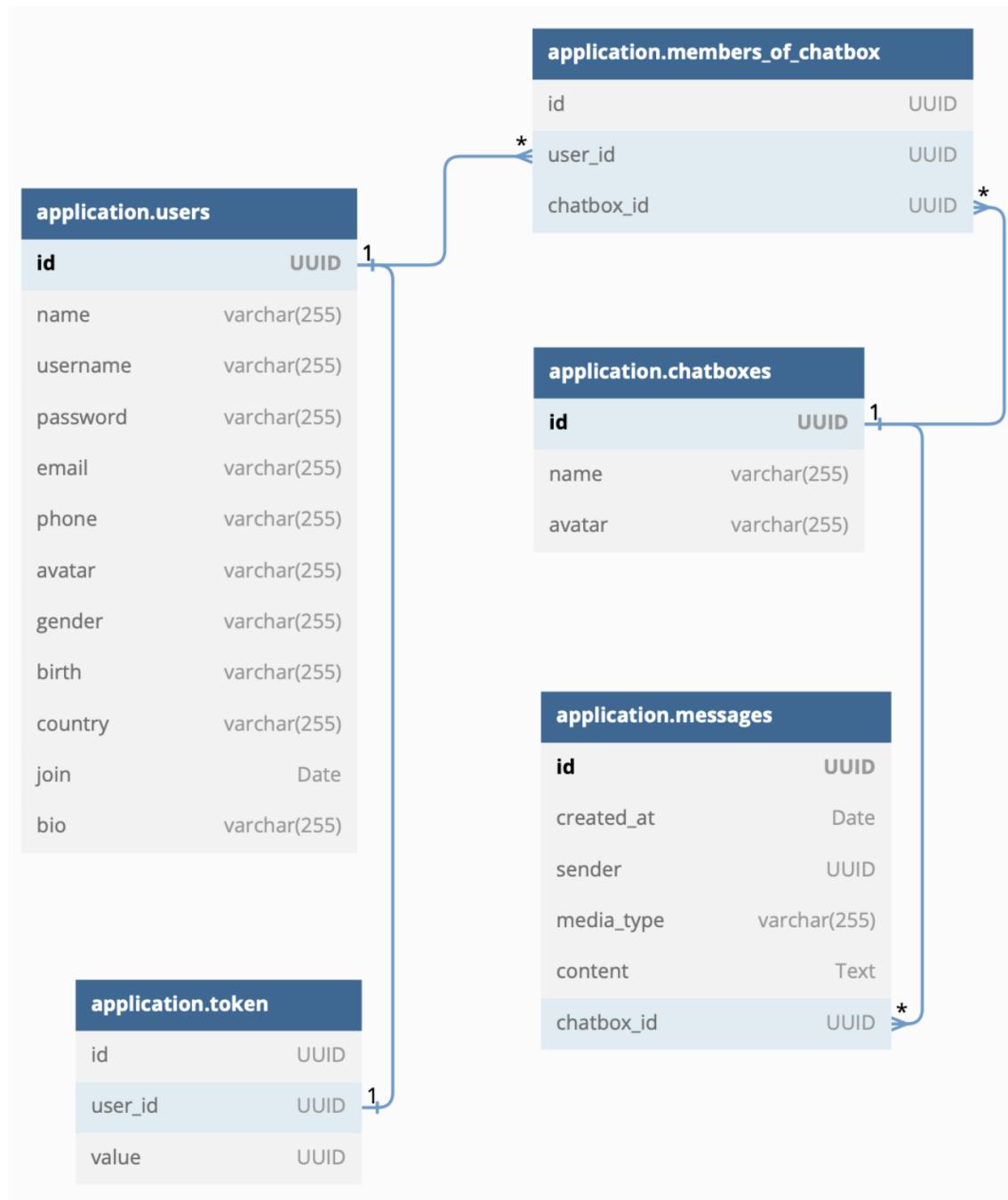
Media stream là một hình thức stream dữ liệu âm thanh và hình ảnh. Yêu tố này hình thành bằng cách gọi hàm getUserMedia của WebRTC để khởi tạo khi làm việc cục bộ.

Media stream phát huy vai trò cho phép truy cập vào stream của một máy tính. Điều này diễn ra khi một kết nối WebRTC được thiết lập với thiết bị khác.



Hình 4.11: Biểu đồ tuần tự tính năng gọi điện video

4.2.4 Thiết kế cơ sở dữ liệu



Hình 4.12: Sơ đồ thiết kế cơ sở dữ liệu

Bảng dữ liệu User

Mục đích: Lưu thông tin của người dùng.

Danh sách thuộc tính:

Thứ tự	Tên	Kiểu dữ liệu	Mô tả	Có bắt buộc	Có ràng buộc
1	id	UUID	Id của người dùng	Có	Khoá chính
2	name	Varchar(255)	Tên của người dùng	Có	
3	username	Varchar(255)	Tên đăng nhập của người dùng	Có	Duy nhất
4	password	Varchar(255)	Mật khẩu	Có	
5	gender	Varchar(255)	Giới tính của người dùng	Không	
6	bio	Varchar(255)	Thông tin giới thiệu bản thân của người dùng	Không	
7	birth	Varchar(255)	Ngày tháng năm sinh của người dùng	Không	
8	phone	Varchar(255)	Số điện thoại của người dùng	Không	
9	email	Varchar(255)	Email của người dùng	Không	
10	country	Varchar(255)	Quốc gia của người dùng	Không	
11	join	Date	Quốc gia của người dùng	Không	
12	avatar	Varchar(255)	Quốc gia của người dùng	Không	

Bảng 4.1: Bảng dữ liệu User

Bảng dữ liệu Chatbox

Mục đích: Lưu thông tin phòng chat.

Danh sách thuộc tính:

Thứ tự	Tên	Kiểu dữ liệu	Mô tả	Có bắt buộc	Có ràng buộc
1	id	UUID	Id chatbox (id của phòng chat)	Có	Khoá chính
2	name	Varchar(255)	Tên chatbox (tên của phòng chat)	Không	
3	avartar	Varchar(255)	Id lưu ảnh đại diện chatbox	Không	

Bảng 4.2: Bảng dữ liệu Chatbox

Bảng dữ liệu Message

Mục đích: Lưu tin nhắn của các phòng chat.

Danh sách thuộc tính:

Thứ tự	Tên	Kiểu dữ liệu	Mô tả	Có bắt buộc	Có ràng buộc
1	id	UUID	Id của tin nhắn	Có	Khoá chính
2	created_at	Date	Ngày tin nhắn được gửi	Có	
3	sender	UUID	Id người gửi tin nhắn	Có	
4	media_type	Varchar(255)	Kiểu dữ liệu của tin nhắn	Có	
5	content	Varchar(255)	Nội dung tin nhắn	Có	
6	chatbox_id	Varchar(255)	Id chatbox (id của phòng chat)	Có	

Bảng 4.3: Bảng dữ liệu User

Bảng dữ liệu Member Of Chatbox

Mục đích: Lưu id thành viên trong phòng chat.

Danh sách thuộc tính:

Thứ tự	Tên	Kiểu dữ liệu	Mô tả	Có bắt buộc	Có ràng buộc
1	id	UUID	Id trường dữ liệu	Có	Khoá chính
2	user_id	UUID	Id của người dùng	Có	
3	chatbox_id	UUID	Id chatbox (id của phòng chat)	Có	

Bảng 4.4: Bảng dữ liệu members_of_chatbox

4.3 Xây dựng ứng dụng

4.3.1 Thư viện và công cụ sử dụng

Các thư viện và công cụ sử dụng để xây dựng đề tài "Phần mềm nhắn tin, gọi điện video trực tuyến":

Công cụ	Mục đích	Địa chỉ URL
Swift	Ngôn ngữ lập trình back-end và iOS	https://developer.apple.com/swift/
Visual Code	IDE lập trình	https://code.visualstudio.com
Xcode	IDE lập trình	https://developer.apple.com/xcode/
PostgreSQL	Hệ quản trị cơ sở dữ liệu quan hệ	https://www.postgresql.org
MongoDB	Hệ quản trị cơ sở dữ liệu phi cấu trúc	https://www.mongodb.com
Vapor	Framework lập trình back-end	https://vapor.codes
UIKit	Framework xây dựng giao diện	https://developer.apple.com/documentation/uikit
Alamofire	Framework hỗ trợ request API	https://github.com/Alamofire/Alamofire

Nuke	Framework hỗ trợ xử lý ảnh bất đồng bộ, cache ảnh vào bộ nhớ trên thiết bị,..	https://github.com/kean/Nuke
Combine	Framework của Apple hỗ trợ xử lý bất đồng bộ, đa luồng,..	https://developer.apple.com/documentation/combine
StarScream	Framework hỗ trợ WebSocket	https://github.com/daltoniam/Starscream
WebRTC	Framework hỗ trợ xây dựng tính năng gọi điện video trực tuyến	https://webrtc.org
Docker	Tạo môi trường để chạy Vapor, PostgreSQL, MongoDB	https://www.docker.com

Bảng 4.5: Bảng danh sách thư viện và công cụ sử dụng

4.3.2 Kết quả đạt được

Mô tả	Thông tin
Số dòng code	13950
Số lượng lớp	116
Số giao diện	12
Dung lượng mã nguồn và tài nguyên	13.8MB
Môi trường lập trình	MacOS

Bảng 4.6: Thông tin kết quả đạt được

4.3.3 Minh họa các chức năng chính

Kết quả đạt được là xây dựng thành công một ứng dụng nhắn tin và gọi điện video trực tuyến trên nền tảng iOS và http server trên nền tảng MacOS với các chức năng cơ bản:

Người dùng có thể tạo tài khoản và đăng nhập vào hệ thống.

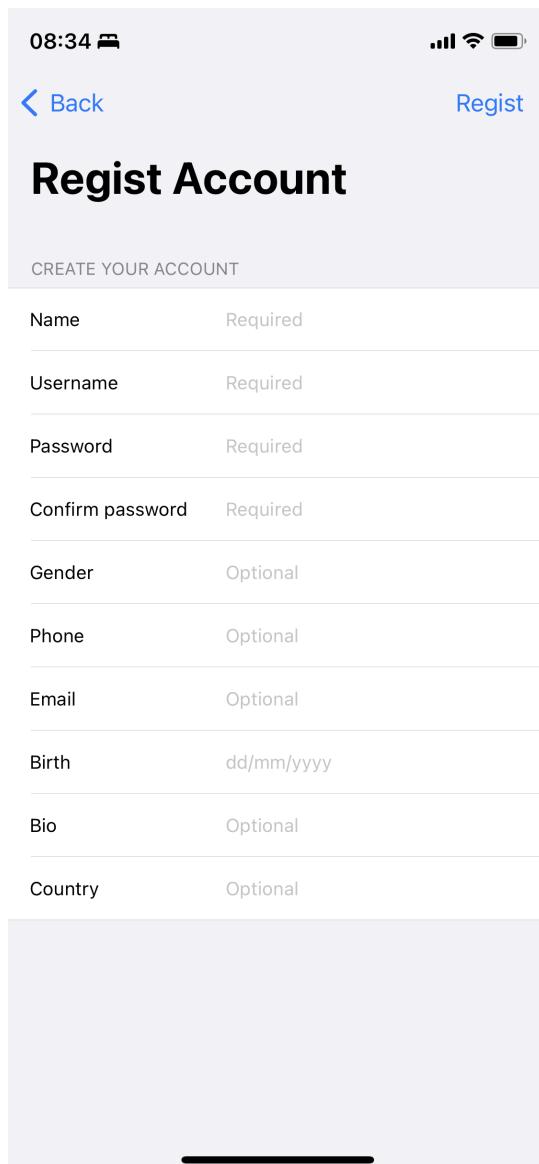
Người dùng đã đăng nhập có thể xem được thông tin công khai của những người dùng khác, chỉnh sửa thông tin cá nhân của bản thân, tạo phòng chat và nhắn tin

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG

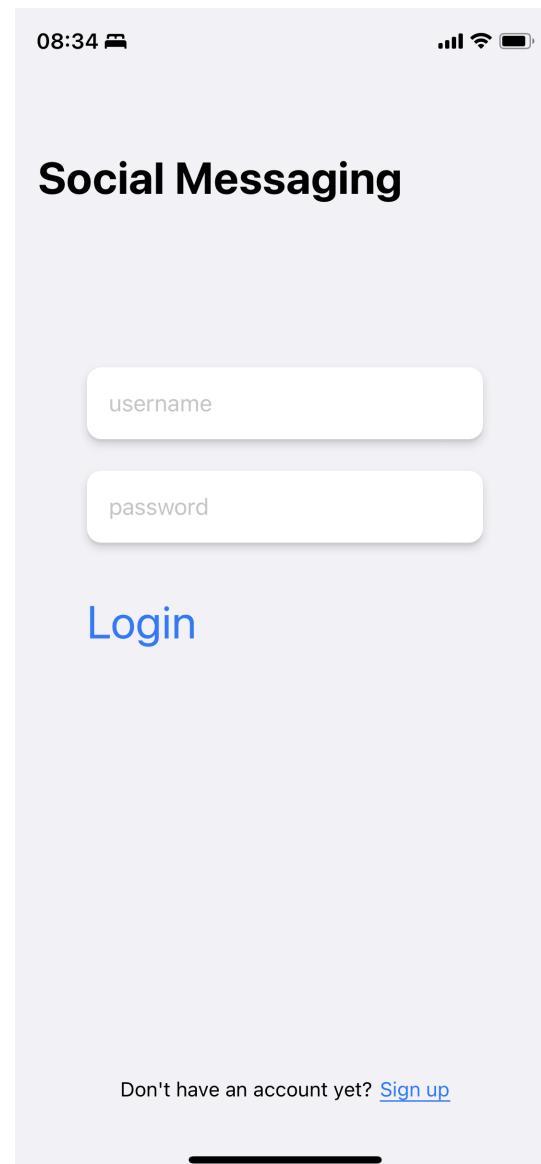
hoặc gọi điện video với người dùng khác. Ngoài ra người dùng đã đăng nhập có thể tạo nhóm chat và có thể quản lý thành viên trong nhóm.

Người dùng có thể gửi đi tin nhắn là văn bản hoặc ảnh.

Hình ảnh minh họa sử dụng ứng dụng:

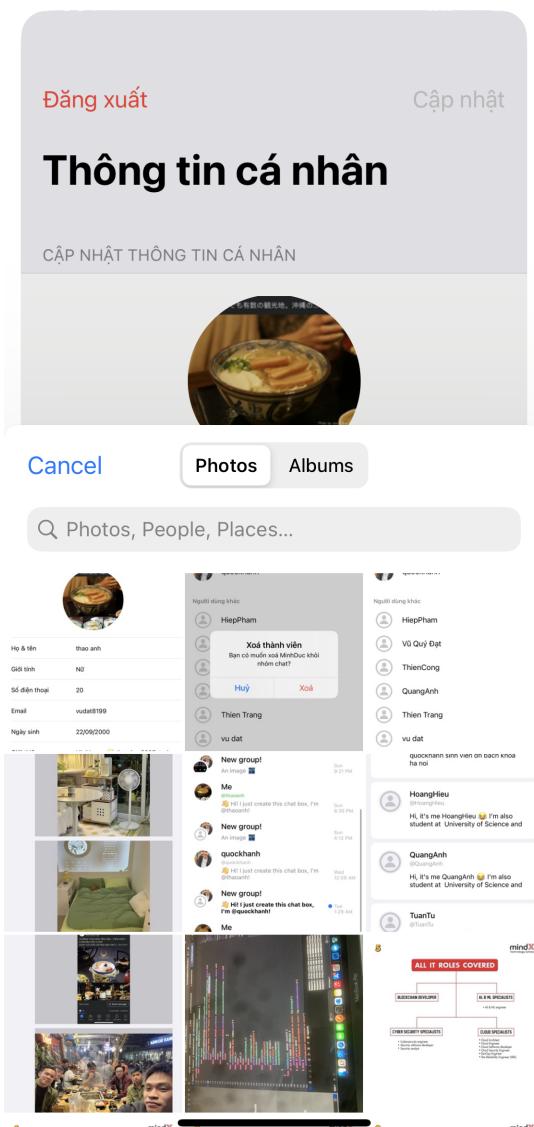


Hình 4.13: Màn hình đăng ký

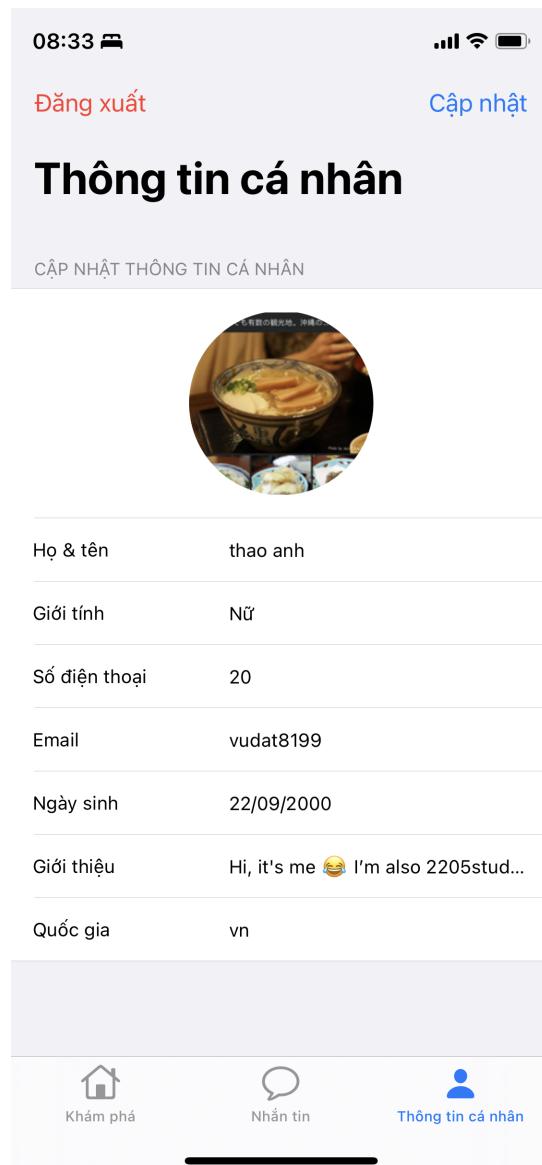


Hình 4.14: Màn hình đăng nhập

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG

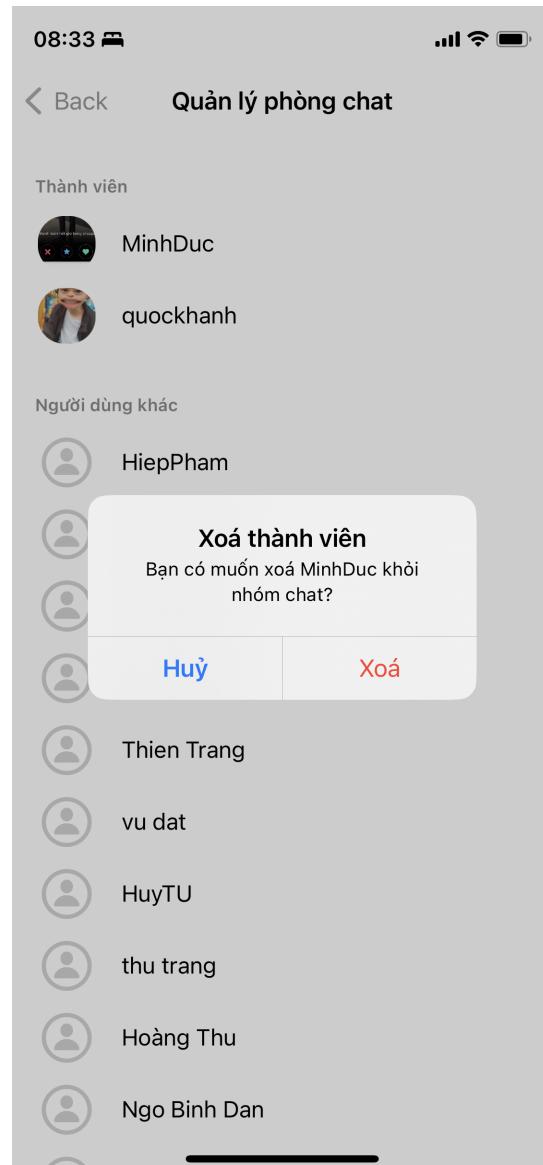


Hình 4.15: Màn hình thay đổi avatar



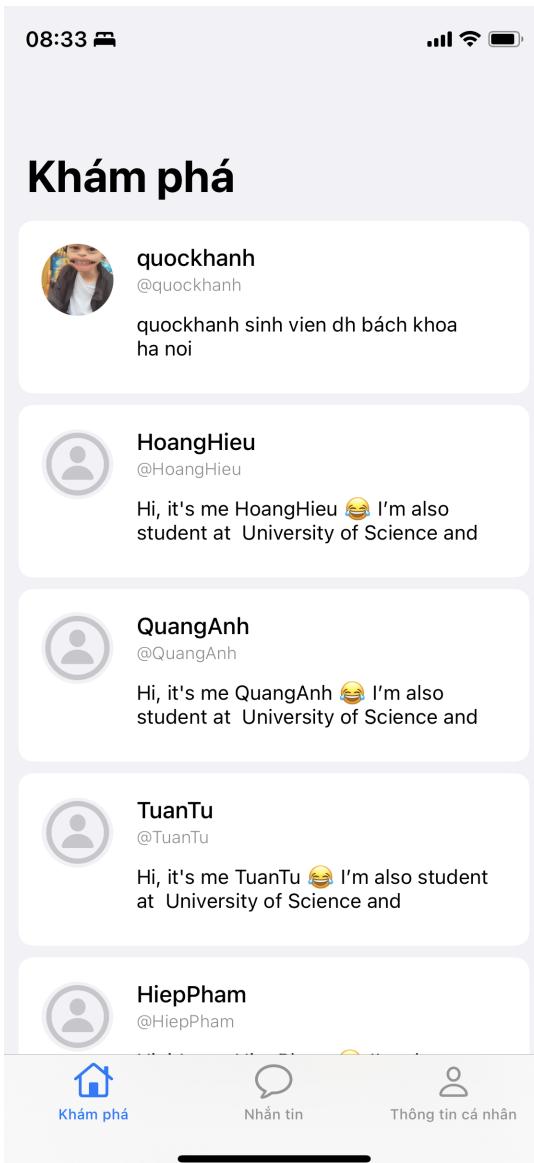
Hình 4.16: Màn hình xem thông tin bản thân

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG

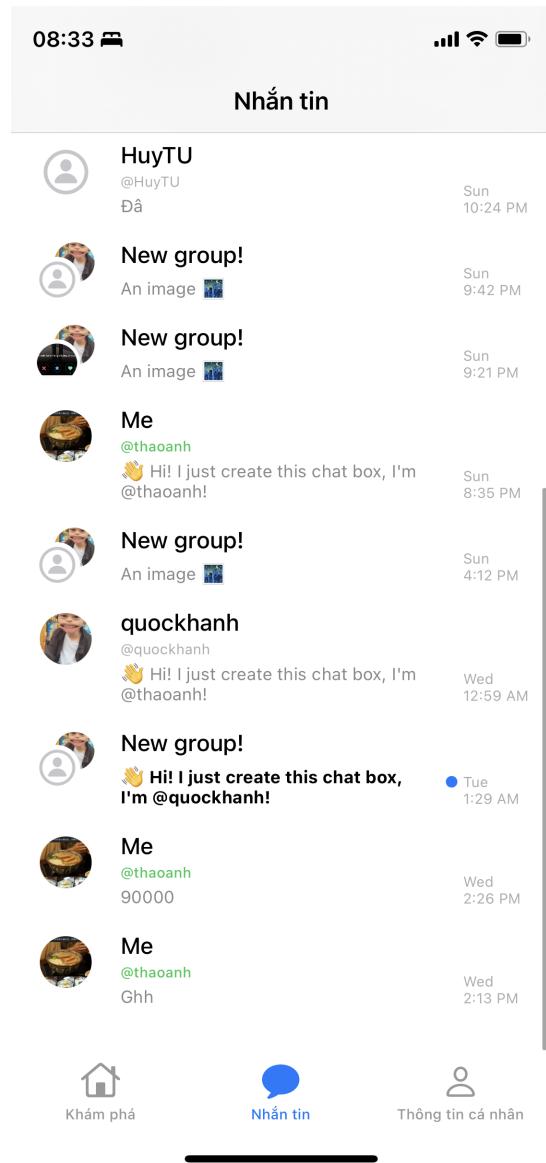


Hình 4.17: Màn hình quản lý thành viên chat **Hình 4.18:** Màn hình xoá thành viên khỏi chat box

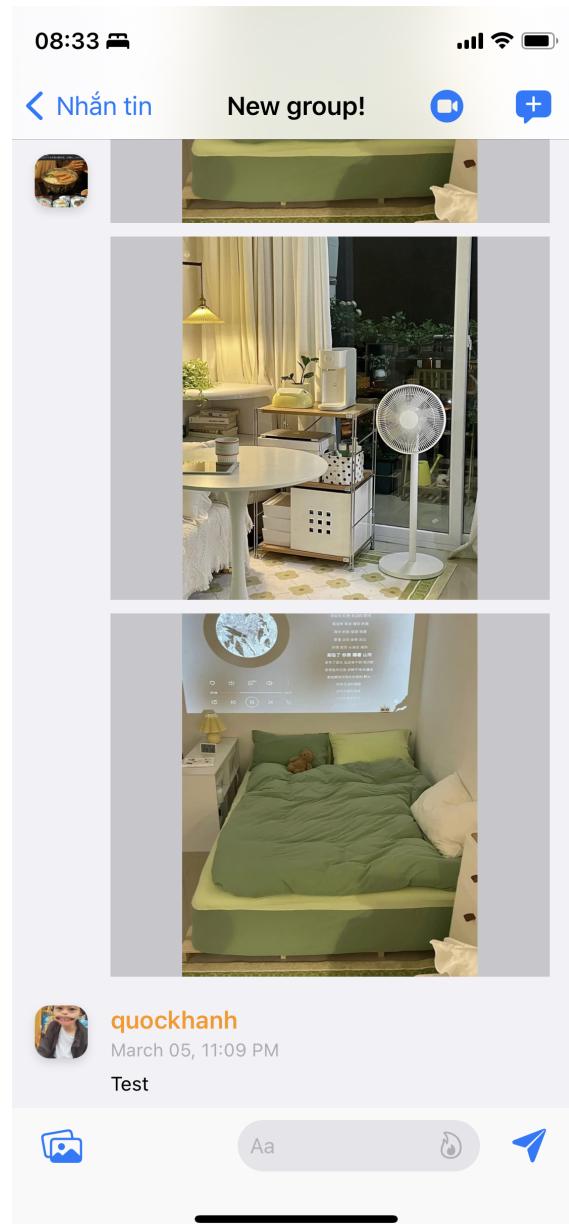
CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG



Hình 4.19: Màn hình danh sách người dùng



Hình 4.20: Màn hình danh sách chat box



Hình 4.21: Màn hình nhắn tin

4.4 Kiểm thử

STT	Chức năng	Đầu vào	Đầu ra	Kết quả
1	Đăng ký	Nhập đầy đủ và đúng yêu cầu của các trường bắt buộc	Thông báo đăng ký tài khoản thành công và hiển thị màn hình đăng nhập	Đạt

CHƯƠNG 4. PHÁT TRIỂN VÀ TRIỂN KHAI ỨNG DỤNG

2	Đăng ký	Nhập thiêu các trường bắt buộc	Hiển thị thông báo yêu cầu người dùng kiểm tra lại thông tin tài khoản	Đạt
3	Đăng nhập	Nhập đúng cả 2 trường với tài khoản và mật khẩu đã tồn tại trong cơ sở dữ liệu	Đăng nhập thành công và hiển thị màn hình chính	Đạt
4	Đăng nhập	Nhập sai thông tin 1 hoặc cả 2 trường	Hiển thị thông báo yêu cầu người dùng kiểm tra lại tài khoản và mật khẩu	Đạt
5	Đăng nhập	Nhập thiêu 1 hoặc cả 2 trường	Yêu cầu người dùng kiểm tra lại tài khoản và mật khẩu	Đạt
6	Đăng xuất	Ấn nút đăng xuất	Xoá dữ liệu người dùng hiện tại và hiển thị màn hình đăng nhập	Đạt
7	Xem thông tin người dùng	Ấn vào 1 người dùng ở màn hình chính	Hiển thị màn hình thông tin người dùng	Đạt
8	Tạo phòng chat	Kéo thanh người dùng từ phải sang trái	Phản hồi cho người dùng bằng âm thanh và cập nhật danh sách nhóm chat	Đạt
9	Gửi tin nhắn văn bản	Nhập tin nhắn văn bản và ấn gửi	Cập nhật tin nhắn mới trên màn hình xem tin nhắn và màn hình danh sách nhóm chat	Đạt
10	Gửi tin nhắn ảnh	Chọn ảnh muốn gửi và ấn gửi	Cập nhật tin nhắn mới trên màn hình xem tin nhắn và màn hình danh sách nhóm chat	Đạt
11	Gọi điện video	Ấn vào nút gọi trên màn hình chat	Nhìn thấy video và nghe, nói chuyện được với người đang gọi tới	Đạt

12	Thêm người dùng vào nhóm chat	Chọn nút thêm trên màn hình chat sau đó chọn người dùng muốn thêm vào nhóm và ấn xác nhận	Cập nhật danh sách người dùng	Đạt
13	Xoá người dùng khỏi nhóm chat	Chọn nút thêm trên màn hình chat sau đó chọn người dùng muốn xoá khỏi nhóm và ấn xác nhận	Cập nhật danh sách người dùng	Đạt
14	Rời khỏi nhóm chat	Tại màn hình danh sách nhóm chat, kéo thanh nhóm chat từ phải sang trái và ấn xác nhận	Xoá nhóm chat khỏi danh sách nhóm chat	Đạt
15	Chỉnh sửa thông tin cá nhân công khai	Chỉnh sửa thông tin bất kỳ sau đó chọn cập nhật trên màn hình thông tin cá nhân	Cập nhật thông tin trên màn hình thông tin cá nhân và phản hồi bằng âm thanh	Đạt
16	Chỉnh sửa ảnh đại diện	Chọn ảnh bất kỳ sau đó chọn cập nhật trên màn hình thông tin cá nhân	Cập nhật thông tin trên màn hình thông tin cá nhân và phản hồi bằng âm thanh	Đạt

Bảng 4.7: Kết quả kiểm thử

4.5 Triển khai

Back-end được triển khai trên Docker, phần back-end có thể cài đặt được ở trên nhiều nền tảng, miễn là cài đặt Docker. Dưới đây là các bước cài đặt và triển khai trên MacOS và Ubuntu:

Cài đặt:

Bước 1: Cài đặt Docker.

Cài đặt Docker trên máy chủ ở trang chủ của Docker.

Sau khi cài đặt Docker, mở Docker lên và chạy thử 1 container để kiểm tra bằng cách mở cửa sổ dòng lệnh và chạy:

`docker run hello-world`

Sau đó chạy:

```
docker run -it ubuntu bash
```

Đây là kết quả sau khi chạy:

```
docker run hello-world
```

```
Unable to find image 'hello-world:latest' locally
```

```
latest: Pulling from library/hello-world
```

```
0e03bdcc26d7: Pull complete
```

Digest:

```
sha256:49a1c8800c94df04e9658809b006fd8a686cab8028d33cfba2cc049724254202
```

```
Status: Downloaded newer image for hello-world:latest
```

```
docker run -it ubuntu bash
```

Bước 2: Cấu hình Docker.

Cấu hình PostgreSQL trên Docker:

- Tạo file docker-compose.yml tại thư mục Thesis/Back-end.
- Thêm đoạn cấu hình sau vào file docker-compose.yml:

```
17 version: '3.7'
18
19 volumes:
20   db_data:
21
22 x-shared_environment: &shared_environment
23   LOG_LEVEL: ${LOG_LEVEL:-debug}
24   DATABASE_HOST: db
25   DATABASE_NAME: vapor_database
26   DATABASE_USERNAME: vapor_username
27   DATABASE_PASSWORD: vapor_password
28
29 services:
30   app:
31     image: thesis:latest
32     build:
33       context: .
34       environment:
35         <<: *shared_environment
36     depends_on:
37       - db
38     ports:
39       - '8080:8080'
40     # user: '0' # uncomment to run as root for testing purposes even though Dockerfile defines 'vapor' user.
41     command: ["serve", "--env", "production", "--hostname", "0.0.0.0", "--port", "8080"]
42   db:
43     image: postgres:14-alpine
44     volumes:
45       - db_data:/var/lib/postgresql/data/pgdata
46     environment:
47       PGDATA: /var/lib/postgresql/data/pgdata
48       POSTGRES_USER: vapor_username
49       POSTGRES_PASSWORD: vapor_password
50       POSTGRES_DB: vapor_database
51     ports:
52       - '5432:5432'
```

version: '3.7' - là phiên bản của Docker compose

Phiên bản PostgreSQL sử dụng:

postgres:14-alpine

DATABASE USERNAME: vaporusername - là username của database

POSTGRES PASSWORD: vaporpassword - là mật khẩu của database

POSTGRES DB: vapordatabase - là tên của database

Cách cấu hình port forwarding cho PostgreSQL database trên Docker

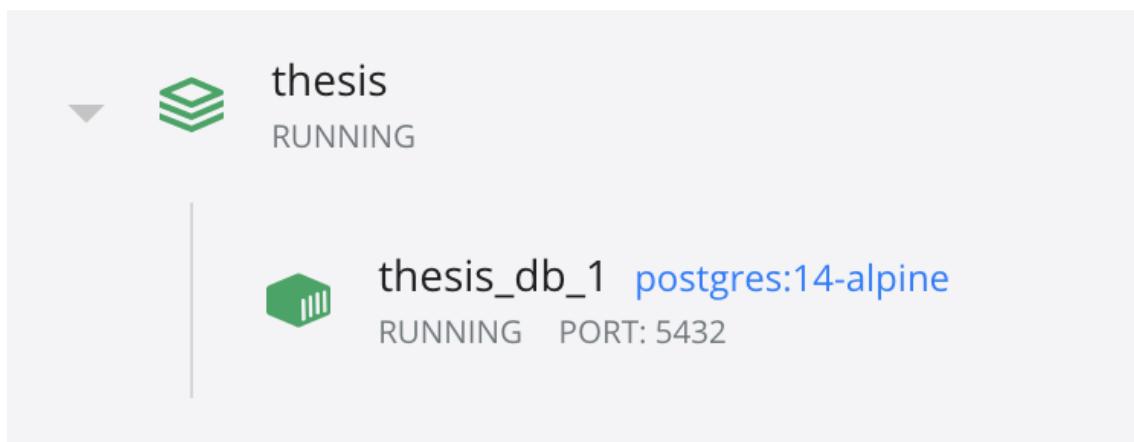
ports:

- '5432:5432'

- Mở cửa sổ dòng lệnh và chạy:

docker-compose up db

- Mở Docker References lên và kiểm tra:



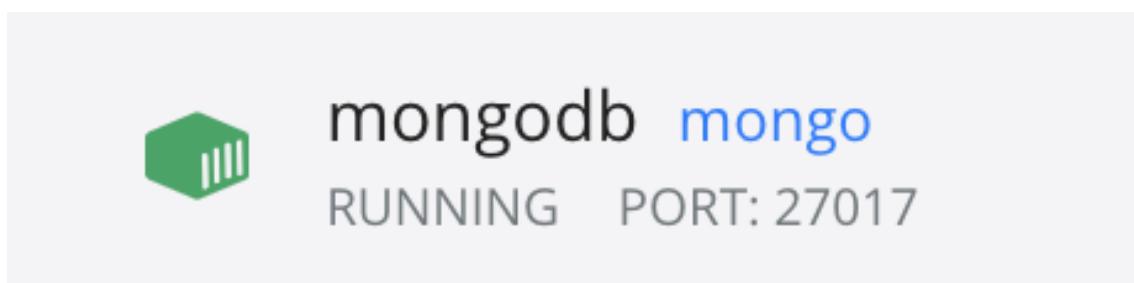
Hình 4.22: Container thông báo đang chạy PostgreSQL

Cấu hình MongoDB trên Docker:

- Mở cửa sổ dòng lệnh và chạy:

docker run --name mongodb -d -p 27017:27017 mongo

- Mở Docker References lên và kiểm tra:



Hình 4.23: Container thông báo đang chạy MongoDB

Bước 3: Cấu hình HTTP server.

Từ thư mục mã nguồn của đồ án, vào đường dẫn Thesis/Back-end, Tại đây thư mục Signaling là phần service cho tính năng gọi video trực tuyến, các thư mục còn lại dùng để triển khai http server và websocket.

Triển khai http server trên MacOS:

- Từ thư mục Back-end, mở mã nguồn trên XCode IDE bằng cách mở file Package.swift.

- Cấu hình http server trong file configure.swift:

Cấu hình ip

```
let host = "192.168.1.24"
```

Cấu hình port

```
app.http.server.configuration.port = 8080
```

Cấu hình dung lượng lớn nhất của http body

```
app.routes.defaultMaxBodySize = "20mb"
```

Cấu hình tên database

```
databaseName = "vapordatabase"
```

Cấu hình port database

```
databasePort = 5432
```

Cấu hình hostname database

```
hostname: Environment.get("DATABASEHOST") ?? "localhost"
```

Cấu hình username database

```
username: Environment.get("DATABASEUSERNAME") ?? "vaporusername"
```

Cấu hình password database

```
password: Environment.get("DATABASEPASSWORD") ?? "vaporpassword"
```

Cấu hình MongoDB

```
try app.databases.use(.mongo(connectionString: "mongodb://localhost:27017/mongo"),  
as: .mongo)
```

```
try app.initializeMongoDB(connectionString: "mongodb://localhost:27017/mongo")
```

- Bắt đầu chạy phần mềm trên Xcode bằng tổ hợp phím command + R

Triển khai http server trên Ubuntu:

Cấu hình server trên Ubuntu giống với hướng dẫn trên MacOS.

Chạy phần mềm trên Ubuntu:

- Mở cửa sổ dòng lệnh và bắt đầu cài đặt Swift bằng lệnh sau: sudo apt install clang libpython2.7 libpython2.7-dev
wget https://swift.org/builds/swift-5.3-release/ubuntu2004/swift-5.3-RELEASE/swift-5.3-RELEASE-ubuntu20.04.tar.gz
tar xzf swift-5.3-RELEASE-ubuntu20.04.tar.gz
sudo mv swift-5.3-RELEASE-ubuntu20.04 /usr/share/swift
echo "export PATH=/usr/share/swift/usr/bin:\$PATH" » ./zshrc
source ./zshrc
- Sau khi đã cài đặt xong Swift, chạy lệnh sau để bắt đầu biên dịch mã nguồn:
swift build
- Sau khi biên dịch thành công, chạy lệnh sau để chạy phần mềm với ip và port được chỉ định:
swift run Run –hostname 192.168.1.24 –port 8080

Bước 4: Chạy signaling server phục vụ cho tính năng gọi điện video trực tuyến.

- Từ thư mục Back-end, trỏ vào thư mục signaling từ cửa sổ dòng lệnh, chạy lệnh:
swiftc main.swift WebSocketServer.swift WebSocketClient.swift -o server
.server
- server sẽ bắt đầu chạy.

Bước 5: Cài đặt ứng dụng trên iOS.

- Từ thư mục mã nguồn của đồ án Thesis, mở thư mục ChatApp sau đó mở file ChatApp.xcodeproj.
- Sửa mục nhà phát triển thành nhóm phát triển đính kèm với tài khoản Apple iCloud.
- Chạy phần mềm bằng tổ hợp phím command + R.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

Xuất phát từ nhu cầu thực tế của công ty và khả năng của bản thân, cùng với tin thần muôn thử, muôn tìm hiểu về cách hoạt động của ứng dụng nhắn tin và gọi video, em đã thực hiện đồ án xây dựng Phần mềm nhắn tin, gọi điện video trực tuyến. Kết quả của đồ án đã đáp ứng được các yêu cầu đề ra ban đầu. Em đã xây dựng thành công một ứng dụng chạy trên nền tảng iOS để nhắn tin và gọi điện video. Thông qua các khảo sát các ứng dụng nhắn tin và gọi video đã có trước đó, em đã phân tích ra được các tính năng mà ứng dụng cần có: tạo tài khoản, chỉnh sửa thông tin cá nhân, nhắn tin văn bản, gửi tin nhắn ảnh và video, xem thông tin của người dùng khác, gọi điện video, tạo một phòng trò chuyện chỉ bao gồm những thành viên trong 1 dự án cụ thể, xoá thành viên khỏi nhóm chat, xoá nhóm chat.

Đồ án tuy đã đạt được những mục tiêu đề ra tuy nhiên do giới hạn kiến thức của bản thân và thời gian làm đồ án là có hạn nên không thể tránh khỏi những sai sót không đáng có: tổ chức các luồng dữ liệu trong ứng dụng chưa tối ưu, thời gian gửi ảnh lên server chưa nhanh như mong muốn. Với những hạn chế nêu trên của đồ án, em sẽ tìm hiểu và khắc phục trong tương lai để mang đến trải nghiệm tốt nhất cho người dùng.

5.2 Hướng phát triển của đồ án

Phần mềm nhắn tin, gọi điện video trực tuyến sẽ tiếp tục xây dựng nhiều tính năng, cải tiến nhiều hơn để mang lại trải nghiệm tốt hơn cho người dùng với các tính năng trong tương lai đó là: cho phép gọi điện video nhiều hơn 2 người, tìm kiếm lịch sử tin nhắn, chia sẻ được nhiều kiểu file trong phần tin nhắn, quét mã qrcode để kết bạn cũng như nhắn tin với nhau, cho phép đăng bài viết.

TÀI LIỆU THAM KHẢO

- [1] Apple, *The Swift Programming Language*. 2022-09-12. [Online]. Available: <https://books.apple.com/vn/book/the-swift-programming-language-swift-5-7/id881256329>.
- [2] Logan Wright, Tim Condon Tanner Nelson, *Server-Side Swift with Vapor*. [Online]. Available: <https://www.kodeco.com/books/server-side-swift-with-vapor/v3.0> (visited on 03/06/2023).
- [3] Dao Tuan Anh, *Giới thiệu về webrtc và hướng tiếp cận media server*. [Online]. Available: <https://viblo.asia/p/gioi-thieu-ve-webrtc-va-huong-tiep-can-media-server-maGK7k3MKj2> (visited on 03/06/2023).
- [4] Vu Ngoc Anh, *Sử dụng socket để kết nối giữa ios app và web server*. [Online]. Available: <https://viblo.asia/p/su-dung-socket-de-ket-noi-giuua-ios-app-va-web-server-Klov164LR5b9> (visited on 03/06/2023).
- [5] Vapor, *Fluent*. [Online]. Available: <https://docs.vapor.codes/fluent/overview/> (visited on 03/06/2023).
- [6] Vapor, *Fluent*. [Online]. Available: <https://docs.vapor.codes/fluent/overview/> (visited on 03/06/2023).
- [7] Vapor, *Docker deploys*. [Online]. Available: <https://docs.vapor.codes/deploy/docker/> (visited on 03/06/2023).

PHỤ LỤC