

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN TỐT NGHIỆP
NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI:
XÂY DỰNG WEBSITE BẤT ĐỘNG SẢN

Người hướng dẫn: **THS. NGUYỄN THỊ VŨ THẢO**

Sinh viên thực hiện: **NGÔ LÊ NGỌC MINH**

Mã số sinh viên: **2151220169**

Lớp: **21CT4**

Đà Nẵng, 12/2025

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN TỐT NGHIỆP
NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI:
XÂY DỰNG WEBSITE BẤT ĐỘNG SẢN

Người hướng dẫn: **THS. NGUYỄN THỊ VŨ THẢO**

Sinh viên thực hiện: **NGÔ LÊ NGỌC MINH**

Mã số sinh viên: **2151220169**

Lớp: **21CT4**

Đà Nẵng, 12/2025

LỜI NÓI ĐẦU

Tên đề tài: XÂY DỰNG WEBSITE BẤT ĐỘNG SẢN

Sinh viên thực hiện: NGÔ LÊ NGỌC MINH

Mã số SV: 2151220169

Lớp: 21CT4

Trong bối cảnh chuyển đổi số diễn ra mạnh mẽ trong lĩnh vực bất động sản, các doanh nghiệp môi giới, sàn giao dịch và cá nhân kinh doanh Bất Động Sản phải đối mặt với thách thức ngày càng lớn trong việc quản lý dữ liệu tin đăng, tiếp cận khách hàng, tối ưu quy trình làm việc và nâng cao khả năng cạnh tranh. Khi số lượng tin đăng và tệp khách hàng mở rộng, việc quản lý thủ công hoặc sử dụng các kênh rời rạc (mạng xã hội, bảng tính, tin nhắn) dễ dẫn đến tình trạng dữ liệu phân tán, khó kiểm soát chất lượng tin, khó theo dõi hiệu suất môi giới, thiếu sự đồng bộ giữa các bộ phận và hạn chế khả năng giám sát hoạt động theo thời gian thực.

Xuất phát từ những khó khăn đó, việc xây dựng một hệ thống website bất động sản thống nhất – hiện đại – cập nhật dữ liệu realtime trở nên vô cùng cấp thiết. Vì vậy, em lựa chọn thực hiện đề tài “Website bất động sản” nhằm nghiên cứu và xây dựng một giải pháp hỗ trợ doanh nghiệp và cá nhân môi giới tối ưu toàn bộ quy trình như: đăng và quản lý tin bất động sản, quản lý danh mục và khu vực, quản lý khách hàng tiềm năng, tương tác và tư vấn qua hệ thống chat, theo dõi hiệu suất làm việc của môi giới, quản lý tài khoản và gói dịch vụ, hiển thị tin theo khu vực, lọc nâng cao theo nhu cầu khách hàng và hỗ trợ khách hàng lưu tin – tìm kiếm – liên hệ một cách thuận tiện.

Trong quá trình triển khai đề tài, em đã vận dụng các kiến thức về phân tích – thiết kế hệ thống thông tin, xây dựng API backend bằng Spring Boot, lập trình giao diện frontend bằng React.js, thiết kế cơ sở dữ liệu MySQL, tích hợp Socket.IO cho chức năng chat realtime và áp dụng các phương pháp tối ưu quy trình quản lý dữ liệu trong môi trường có nhiều loại người dùng như Admin – Agent – User. Đề tài không chỉ mang giá trị học thuật mà còn có khả năng ứng dụng thực tế cao, giúp hoạt động môi giới chuyên nghiệp hơn, kiểm soát chất lượng tin đăng tốt hơn, nâng cao khả năng tiếp cận khách hàng, hạn chế sai sót trong quá trình quản lý và tối ưu hiệu quả vận hành trong bối cảnh chuyển đổi số của ngành bất động sản hiện nay.

Em xin chân thành cảm ơn giảng viên hướng dẫn và những người đã hỗ trợ trong suốt quá trình nghiên cứu và hoàn thiện đề tài.

CAM ĐOAN

Em xin cam đoan đồ án tốt nghiệp với đề tài xây dựng website bất động sản là thuộc quyền sở hữu duy nhất của em. Những tài liệu tham khảo được sử dụng trong đồ án đã được trích dẫn và nêu rõ trong mục Tài liệu tham khảo. Bên cạnh đó, những kết quả nghiên cứu hoàn toàn mang tính chất trung thực, không sao chép, đạo nhái từ bất kỳ công trình nào trước đây.

Nếu những lời cam đoan trên của em không chính xác, tôi xin chịu hoàn toàn trách nhiệm và chịu mọi hình thức kỷ luật từ khoa và nhà trường.

Sinh viên thực hiện

Ngô Lê Ngọc Minh

MỤC LỤC

LỜI NÓI ĐẦU	i
CAM ĐOAN.....	ii
MỤC LỤC	iii
DANH SÁCH HÌNH ẢNH	v
DANH SÁCH BẢNG	vii
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT.....	7
1.1. Tổng quan về mô hình Client Server	7
1.1.1. Client Server là gì?.....	7
1.1.1.Nguyên tắc hoạt động của mô hình Client Server	7
1.1.2.Các ưu nhược điểm của mô hình Client Server.....	8
1.2. Tổng quan về Spring Boot	10
1.2.1.Spring boot là gì?	10
1.1.2. Cách cài đặt Spring boot	12
1.3. Tổng quan về Visual Studio Code	12
1.3.1.Visual Studio Code là gì?	12
1.3.2.Những ưu điểm nổi bật của Visual Studio Code	13
1.3.3.Các lí do nên sử dụng Visual Studio Code	13
1.4. Tổng quan về Reactjs	14
1.4.1.Reactjs là gì?.....	14
1.4.2.Những đặt điểm quan trọng của Reactjs.....	15
1.4.3.Lợi ích khi sử dụng Reactjs	15
1.4.4. Lợi ích khi sử dụng Reactjs	2
1.5. Tổng quan về hệ quản trị cơ sở dữ liệu MySQL là gì?.....	2
1.5.1. Tổng quan về hệ quản trị cơ sở dữ liệu MySQL là gì?	3
1.5.2. Ưu nhược và điểm của MySQL.....	5
1.5.3 Cách cài đặt MySQL.	6

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG	7
2.1. Phát biểu bài toán	7
2.2. Phân tích chức năng	7
2.3. Xác định tác nhân	9
2.4. Biểu đồ useCase	10
2.4.1. Biểu đồ use case cho quản trị viên (Admin)	10
2.4.2. Biểu đồ Use Case cho người môi giới (Agent)	12
2.4.3. Biểu đồ Use Case cho khách hàng	13
2.5. Sơ đồ lớp	15
2.6. Sơ đồ tuần tự chức năng.....	16
2.6.1.Mô tả tuần tự đăng kí.....	16
2.6.2. Mô tả người dùng đăng nhập.....	17
2.6.3. Mô tả tuần tự Agent đăng bài	19
2.6.4.Mô tả tuần tự chức năng xóa bài đăng.....	22
2.6.5. Mô tả tuần tự chức năng user chat với agent.....	24
2.6.6.Mô tả tuần tự chức năng nâng cấp gói.....	27
2.6.7. Mô tả tuần tự chatbox tự động AI	30
2.7. Thiết kế cơ sở dữ liệu.....	33
CHƯƠNG 3: XÂY DỰNG HỆ THỐNG VÀ PHÁT TRIỂN ỨNG DỤNG.40	
3.1. Cấu trúc thư mục dự án	40
3.1.1. Cấu trúc thư mục Reactjs	40
3.1.2. Cấu trúc thư mục Spring Boot	43
3.2. Giao diện website.....	47
KẾT LUẬN	58
TÀI LIỆU THAM KHẢO	62

DANH SÁCH HÌNH ẢNH

Hình 1. 1 Mô hình Cline Server	7
Hình 1. 2 Mô hình nguyên lí hoạt động của mô hình Client Server	8
Hình 1. 3 Logo Spring Boot	10
Hình 1. 4 Logo Visual Studio Code	12
Hình 1. 5 Logo React	14
Hình 1. 6 Cách thức hoạt động của Reactjs	2
Hình 1. 7 Logo MySql.....	3
Hình 2. 1 Biểu đồ Use Case quản trị viên	10
Hình 2. 2 Biểu đồ Use Case cho người môi giới	12
Hình 2. 3 Biểu đồ Use Case khách hàng.....	13
Hình 2. 4 Sơ đồ lớp	15
Hình 2. 5 Sơ đồ tuần tự đăng ký	16
Hình 2. 6 Sơ đồ tuần tự người dùng đăng nhập	17
Hình 2. 7 Sơ đồ tuần tự agent đăng bài	19
Hình 2. 8 Sơ đồ tuần tự xóa bài.....	22
Hình 2. 9 Sơ đồ tuần tự khách hàng chat với agent	24
Hình 2. 10 Sơ đồ tuần tự agent nâng cấp gói	27
Hình 2. 11 Sơ đồ tuần tự khách chat với chatbox AI.....	30
Hình 2. 12 Sơ đồ RD	33
Hình 3. 1 Giao diện đăng nhập	47
Hình 3. 2 Giao diện đăng kí	48
Hình 3. 3 Trang chủ.....	48
Hình 3. 4 Giao diện tìm kiếm.....	49
Hình 3. 5 Giao diện bắt động sản mới.....	50
Hình 3. 6 Giao diện dashboard.....	50
Hình 3. 7 Giao diện quản lý user.....	51

Hình 3. 8 Giao diện quản lý bất động sản.....	52
Hình 3. 9 Giao diện gửi thông báo	52
Hình 3. 10 Giao diện quản lý bất động sản.....	53
Hình 3. 11 Giao diện agent chat với user.....	53
Hình 3. 12 Giao diện nâng cấp gói.....	54
Hình 3. 13 Trang yêu thích.....	55
Hình 3. 14 Chi tiết bất động sản.....	55
Hình 3. 15 Giao diện user chat với agent.....	56
Hình 3. 16 Giao diện chi tiết bất động sản.....	56
Hình 3. 17 Giao diện chat với AI.....	57

DANH SÁCH BẢNG

Bảng 2. 1 Bảng agent đăng ký.....	16
Bảng 2. 2 Bảng Mô tả người dùng đăng nhập.....	18
Bảng 2. 3 Bảng mô tả agent đăng bài.....	19
Bảng 2. 4 Bảng mô tả agent xóa bài.....	22
Bảng 2. 5 Bảng mô tả user chat với agent.....	24
Bảng 2. 6 Bảng mô tả agent nâng cấp gói.....	27
Bảng 2. 7 Bảng chatbox tự động AI.....	30

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

Ký hiệu	Tiếng Anh
Frontend	Front-end (UI/UX, TailwindCSS, Shadcn)
Backend	Back-end (API, CRUD, REST)
Spring Boot	Spring Boot Backend
Database	SQL Database
Sever	Sever/ Deplopment
Github	Version Control (GitHub)

MỞ ĐẦU

GIỚI THIỆU TỔNG QUAN

Trong bối cảnh thị trường bất động sản phát triển nhanh và mức độ cạnh tranh ngày càng cao, nhu cầu ứng dụng công nghệ để quản lý thông tin và hỗ trợ giao dịch trở nên cấp thiết hơn bao giờ hết. Các sàn giao dịch và đội ngũ môi giới hiện nay phải xử lý một lượng lớn dữ liệu về dự án, khách hàng, tin đăng và tương tác tư vấn. Nếu vẫn quản lý thủ công hoặc sử dụng các công cụ rời rạc, doanh nghiệp rất dễ đối mặt với tình trạng dữ liệu phân tán, khó kiểm soát tính minh bạch, xử lý phản hồi chậm và hạn chế trong việc mở rộng quy mô hoạt động.

Trên thế giới, nhiều nền tảng bất động sản như Zillow, Realtor hay Rightmove đã triển khai các hệ thống số hóa toàn diện, cho phép người dùng tìm kiếm thông tin theo tiêu chí nâng cao, xem bản đồ, lưu tin yêu thích, liên hệ môi giới trực tiếp và phân tích giá thị trường theo thời gian thực. Một số nghiên cứu quốc tế còn đề xuất mô hình quản lý bất động sản dựa trên điện toán đám mây, giúp đồng bộ thông tin, tăng tốc độ truy xuất dữ liệu và hỗ trợ nhiều nhóm người dùng khác nhau từ xa.

Tại Việt Nam, quá trình số hóa ngành bất động sản đang diễn ra mạnh mẽ với sự xuất hiện của các nền tảng như Batdongsan.com.vn, Chotot Nhà Đất, OneHousing... Tuy nhiên, đa số nền tảng hiện nay chỉ tập trung vào việc đăng tin và tìm kiếm cơ bản, trong khi việc quản lý môi giới, theo dõi khách hàng tiềm năng, phân cấp người dùng, kiểm duyệt nội dung và hỗ trợ tư vấn trực tuyến vẫn còn hạn chế hoặc không tùy chỉnh sâu cho từng mô hình doanh nghiệp. Điều này đặt ra yêu cầu cần xây dựng các hệ thống linh hoạt hơn, có khả năng tích hợp đa tính năng phục vụ đồng thời Admin – Agent – User.

Đề tài “Xây dựng website bất động sản” được thực hiện với mục tiêu phát triển một mô hình nền tảng hỗ trợ đầy đủ nghiệp vụ trong lĩnh vực môi giới bất động sản. Hệ thống tập trung vào các chức năng như quản lý tin đăng, quản lý môi giới, phân quyền người dùng, theo dõi khách hàng tiềm năng, hiển thị bất động sản theo bộ lọc nâng cao, tư vấn qua hệ thống chat, cũng như cung cấp dashboard thống kê dành cho Admin. Đề tài kế thừa các mô hình nghiên cứu trong và ngoài nước, đồng thời áp dụng các kiến trúc công nghệ hiện đại nhằm đảm bảo khả năng mở rộng, bảo mật dữ liệu và mang đến trải nghiệm thao tác trực quan, thuận tiện cho cả môi giới và khách hàng.

MỤC TIÊU ĐỀ TÀI

Mục tiêu của đề tài là xây dựng và hoàn thiện một website bất động sản trực tuyến cho phép người dùng đăng ký gói dịch vụ để xem thông tin bất động sản và trao đổi trực tiếp với môi giới. Hệ thống hướng đến việc cung cấp đầy đủ các tính năng cần thiết, dễ sử dụng, giao diện thân thiện và đảm bảo tính bảo mật trong quá trình quản lý thông tin.

Đề tài tập trung làm rõ cơ sở lý luận liên quan đến việc cung cấp và quản lý thông tin bất động sản trên nền tảng số, bao gồm quy trình đăng tin của môi giới (Agent), cơ chế hiển thị thông tin, quản lý dữ liệu và phân quyền giữa các nhóm người dùng (User – Agent – Admin).

Đồng thời, đề tài cũng xem xét các quy định pháp luật Việt Nam liên quan đến cung cấp thông tin bất động sản, hoạt động môi giới, và bảo mật dữ liệu cá nhân, nhằm đảm bảo hệ thống vận hành phù hợp với quy định hiện hành. Trên cơ sở đó, đề tài đề xuất các giải pháp như tăng cường kiểm duyệt bài đăng của Admin, cải thiện cơ chế xác thực Agent, nâng cao độ chính xác của dữ liệu và hoàn thiện chính sách bảo mật thông tin trên website.

Cuối cùng, đề tài hướng đến phát triển một nền tảng xem bất động sản trực tuyến hiện đại – an toàn – minh bạch, hỗ trợ người dùng tìm kiếm thông tin, liên hệ tư vấn với môi giới và quản lý tài khoản hiệu quả; đồng thời nâng cao uy tín của môi giới và giúp Admin quản lý hệ thống tập trung, hạn chế rủi ro và bảo đảm quyền lợi của tất cả các nhóm người dùng.

Mục tiêu cụ thể của đề tài bao gồm:

- Xây dựng hệ thống quản lý tin đăng bất động sản cho phép môi giới tạo bài đăng, cập nhật thông tin, quản lý hình ảnh và trạng thái giao dịch một cách thuận tiện và minh bạch.
- Phát triển chức năng quản lý danh mục và dữ liệu bất động sản (loại hình, khu vực, tiện ích, giá, diện tích...), giúp hệ thống tổ chức thông tin một cách khoa học và dễ tra cứu.
- Hoàn thiện module phân quyền người dùng giữa ba nhóm: Quản trị viên (Admin), Môi giới (Agent) và Khách hàng (User), đảm bảo mỗi vai trò chỉ được thực hiện các chức năng phù hợp.

- Tích hợp hệ thống báo cáo – thống kê theo thời gian hỗ trợ Admin theo dõi số lượng tin đăng, lượt xem, lượt liên hệ, tỷ lệ chuyển đổi và tình trạng hoạt động của người dùng.
- Xây dựng cơ chế tìm kiếm & lọc thông minh cho khách hàng dựa trên loại hình, vị trí, giá, diện tích và các tiêu chí nâng cao, nhằm cải thiện trải nghiệm tìm kiếm bất động sản.
- Thiết kế giao diện người dùng (UI/UX) hiện đại, trực quan, dễ thao tác cho cả khách hàng lẫn môi giới; đồng thời tối ưu tốc độ tải dữ liệu và hiển thị thông tin.
- Áp dụng kiến trúc hệ thống hiện đại để đảm bảo hiệu năng, tính mở rộng và độ ổn định khi số lượng người truy cập tăng cao, đặc biệt vào các thời điểm thị trường sôi động.
- Xây dựng mô hình hệ thống mẫu hoàn chỉnh, phục vụ mục đích học thuật, thực hành và có thể triển khai thử nghiệm trong thực tế doanh nghiệp môi giới.

Mục tiêu này không chỉ giúp đề tài đáp ứng các yêu cầu về mặt học thuật mà còn hướng tới việc xây dựng một sản phẩm có khả năng ứng dụng thực tế cao trong bối cảnh chuyển đổi số của ngành bất động sản. Hệ thống được định hướng như một giải pháp hỗ trợ nền tảng cho các sàn giao dịch, môi giới và người tìm kiếm bất động sản trong việc quản lý thông tin, tiếp cận dữ liệu nhanh chóng và tương tác hiệu quả hơn.

Bên cạnh đó, đề tài đặc biệt chú trọng vào thiết kế giao diện người dùng (UI/UX) thân thiện, rõ ràng và dễ sử dụng cho nhiều nhóm đối tượng khác nhau như khách hàng, môi giới và quản trị viên. Giao diện phải hỗ trợ thao tác nhanh, hạn chế sai sót, giúp người dùng dễ dàng tìm kiếm bất động sản, gửi yêu cầu tư vấn, theo dõi tin đăng hoặc cập nhật dữ liệu. Việc đặt người dùng làm trọng tâm trong thiết kế là yếu tố then chốt để hệ thống đạt được tính ứng dụng cao và phù hợp với nhu cầu thực tế của thị trường.

Tổng thể, đề tài hướng đến việc kết hợp giữa kiến thức chuyên môn và yêu cầu thực tiễn, nhằm xây dựng một hệ thống website bất động sản hoàn chỉnh, đồng bộ, ổn định và dễ mở rộng. Hệ thống không chỉ hỗ trợ quản lý tin đăng, môi giới và khách hàng một cách khoa học mà còn tạo nền tảng vững chắc cho các giải pháp nâng cao trong tương lai như gợi ý thông minh, phân tích dữ liệu thị trường hay tự động hóa quy trình môi giới. Quá trình thực hiện đề tài cũng giúp người nghiên cứu củng cố các kỹ năng quan trọng như phân tích – thiết kế hệ thống, xây dựng cơ sở dữ liệu, lập trình web toàn stack và triển khai các mô hình công nghệ hiện đại vào thực tế.

ĐỐI TƯỢNG NGHIÊN CỨU

Đối tượng nghiên cứu:

- Website thương mại điện tử chuyên về bất động sản.
- Người dùng hệ thống gồm: Quản trị viên, Môi giới, và Người mua/Người xem tin.

Phạm vi nghiên cứu:

- Phần mềm web-based (chạy trên trình duyệt, không phải mobile app).
- Các tính năng chính bao gồm: đăng tin, tìm kiếm, lọc, yêu thích, thông báo, nhấn tin, và phân quyền người dùng.
- Không tập trung vào quy trình pháp lý giao dịch nhà đất thực tế (hợp đồng, công chứng).
- Tập trung vào quản lý dữ liệu, luồng xử lý và trải nghiệm người dùng trong môi trường web.

PHẠM VI NGHIÊN CỨU

Phạm vi nghiên cứu của đề tài tập trung vào việc xây dựng mô hình website quản lý và khai thác thông tin bất động sản ở mức thử nghiệm, nhưng vẫn đảm bảo khả năng vận hành trong thực tế. Về mặt chức năng, hệ thống được phát triển xoay quanh các nghiệp vụ trọng tâm như quản lý tin đăng, quản lý danh mục (loại hình, khu vực, tiện ích), hiển thị thông tin bất động sản cho khách hàng, quản lý tài khoản người dùng và môi giới, xử lý quy trình duyệt tin, quản lý gói dịch vụ và thống kê mức độ tương tác. Các chức năng nâng cao như tích hợp thanh toán trực tuyến, định giá tự động bằng AI, gợi ý bất động sản theo hành vi, quảng cáo thông minh hay quản lý CRM chuyên sâu không nằm trong trọng tâm của đề tài và được định hướng ở phần nghiên cứu mở rộng.

Về mặt công nghệ, đề tài giới hạn trong phạm vi các công nghệ web phổ biến gồm HTML, CSS, JavaScript/TypeScript và React.js cho phần giao diện người dùng; Java Spring Boot cho backend; cùng hệ quản trị cơ sở dữ liệu quan hệ MySQL để lưu trữ thông tin tin đăng, giao dịch và hồ sơ người dùng. Trong trường hợp cần xử lý hoạt động theo thời gian thực, hệ thống có thể sử dụng WebSocket để đồng bộ thông báo hoặc trạng thái duyệt tin. Ứng dụng được triển khai dưới dạng website, chưa phát triển phiên bản mobile native hay hạ tầng cloud phân tán quy mô lớn.

Phạm vi người dùng của hệ thống bao gồm quản trị viên, môi giới và khách hàng có nhu cầu tìm kiếm hoặc theo dõi các bất động sản quan tâm. Các đối tượng như nhà đầu tư dự án quy mô lớn, đơn vị quảng cáo, ngân hàng hoặc sàn giao dịch liên kết

không nằm trong phạm vi triển khai ở giai đoạn hiện tại. Hệ thống được xây dựng theo hướng mô phỏng (prototype), chưa tích hợp với các dịch vụ hỗ trợ như thanh toán online, hệ thống định danh điện tử (eKYC), ký hợp đồng số hay đồng bộ với phần mềm quản lý doanh nghiệp của sàn BĐS.

Dữ liệu trong quá trình thử nghiệm được mô phỏng dựa trên đặc điểm của dữ liệu thực: thông tin bất động sản, hình ảnh, loại hình giao dịch, khu vực, tiện ích và dữ liệu tương tác. Điều này giúp đảm bảo hệ thống vận hành đúng logic nhưng vẫn phù hợp với điều kiện triển khai về thời gian và nguồn lực.

Việc xác định phạm vi nghiên cứu như trên nhằm đảm bảo đề tài có tính khả thi, tập trung vào những nghiệp vụ quan trọng nhất của một nền tảng bất động sản hiện đại, tránh dàn trải và đảm bảo chất lượng sản phẩm. Các yếu tố nâng cao như tích hợp AI, mở rộng hệ thống đa chi nhánh, xử lý dữ liệu lớn hoặc kết nối trực tiếp với hệ thống pháp lý – tài chính được xem là hướng phát triển tương lai, không nằm trong phạm vi bắt buộc của đề tài. Cách định phạm vi này giúp đề tài đạt được sự cân bằng giữa tính học thuật, tính thực tiễn và khả năng hoàn thiện trong giới hạn thời gian thực hiện.

PHƯƠNG PHÁP NGHIÊN CỨU

Sử dụng phương pháp phân tích nghiệp vụ để thu thập yêu cầu từ mô hình hoạt động thực tế của nhà hàng, xác định các chức năng chính và xây dựng kiến trúc hệ thống phù hợp. Quá trình này bao gồm phân tích luồng xử lý, phân tích vai trò người dùng, xác định các module cần thiết và thiết kế cấu trúc backend – frontend theo mô hình client-server trên nền tảng Java Spring Boot và React.js.

Áp dụng các công cụ mô hình hóa hệ thống như UML để xây dựng sơ đồ use case, sơ đồ lớp, sơ đồ trình tự, sơ đồ hoạt động và mô hình ERD cho cơ sở dữ liệu MySQL. Phương pháp mô hình hóa giúp chuẩn hóa cấu trúc dữ liệu, đảm bảo tính logic giữa các thành phần và là cơ sở quan trọng cho quá trình triển khai hệ thống.

Tiến hành cài đặt và phát triển hệ thống bằng Java Spring Boot ở tầng backend, React.js ở tầng frontend và MySQL làm hệ quản trị cơ sở dữ liệu. Trong quá trình lập trình, áp dụng các nguyên tắc thiết kế phần mềm, mô hình RESTful API và phân quyền bảo mật. Sau khi xây dựng các chức năng, tiến hành kiểm thử đơn vị, kiểm thử tích hợp, kiểm thử giao diện và mô phỏng các tình huống thực tế để đảm bảo tính chính xác và ổn định của hệ thống.

Sau khi thử nghiệm, hệ thống được đánh giá dựa trên mức độ đáp ứng yêu cầu nghiệp vụ, hiệu năng xử lý, tính thân thiện của giao diện và khả năng hoạt động trong

môi trường thực tế. Từ các kết quả đánh giá, tiến hành điều chỉnh, tối ưu truy vấn MySQL, cải thiện API Spring Boot, tối ưu giao diện React.js và bổ sung những chức năng cần thiết nhằm hoàn thiện hệ thống theo hướng ổn định, hiệu quả và dễ mở rộng.

Phương pháp nghiên cứu của đề tài kết hợp giữa phân tích nghiệp vụ và mô hình hóa hệ thống để xác định yêu cầu, thiết kế kiến trúc và xây dựng giải pháp phần mềm phù hợp. Các biểu đồ UML và mô hình ERD được sử dụng nhằm chuẩn hóa luồng xử lý và cấu trúc dữ liệu. Hệ thống được triển khai theo mô hình client–server với Java Spring Boot cho backend, React.js cho frontend và MySQL làm cơ sở dữ liệu. Trong quá trình phát triển, áp dụng RESTful API, phân quyền bảo mật và kiểm thử nhiều mức (đơn vị, tích hợp, giao diện) để đánh giá mức độ chính xác và ổn định. Kết quả thử nghiệm được tổng hợp để tối ưu hiệu năng, cải thiện giao diện và hoàn thiện hệ thống theo hướng ổn định, hiệu quả và dễ mở rộng.

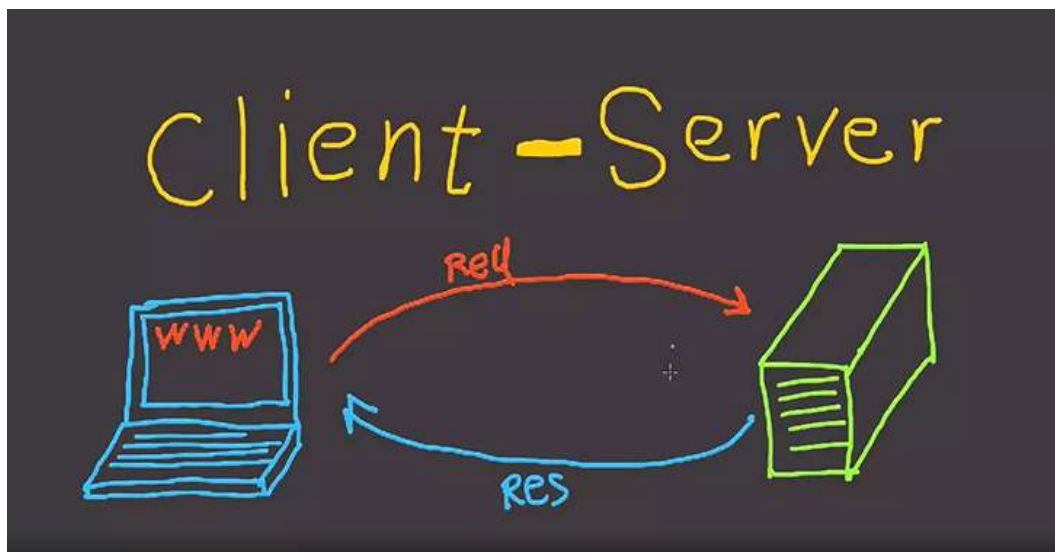
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Tổng quan về mô hình Client Server

1.1.1. Client Server là gì?

Client server là mô hình mạng máy tính gồm có 2 thành phần chính đó là máy khách (client) và máy chủ (server). Server chính là nơi giúp lưu trữ tài nguyên cũng như cài đặt các chương trình dịch vụ theo đúng như yêu cầu của client. Ngược lại, Client bao gồm máy tính cũng như các loại thiết bị điện tử nói chung sẽ tiến hành gửi yêu cầu đến server.

Mô hình mạng Client Server sẽ cho phép mạng tập trung các ứng dụng có cùng chức năng tại một hoặc nhiều dịch vụ file chuyên dụng. Chúng sẽ trở thành trung tâm của hệ thống. Hệ điều hành của mô hình Client server sẽ cho phép người dùng chia sẻ đồng thời cùng một loại tài nguyên mà không giới hạn vị trí địa lý.



Hình 1. 1 Mô hình Cline Server

1.1.1. Nguyên tắc hoạt động của mô hình Client Server

Nguyên tắc hoạt động:

- Client :

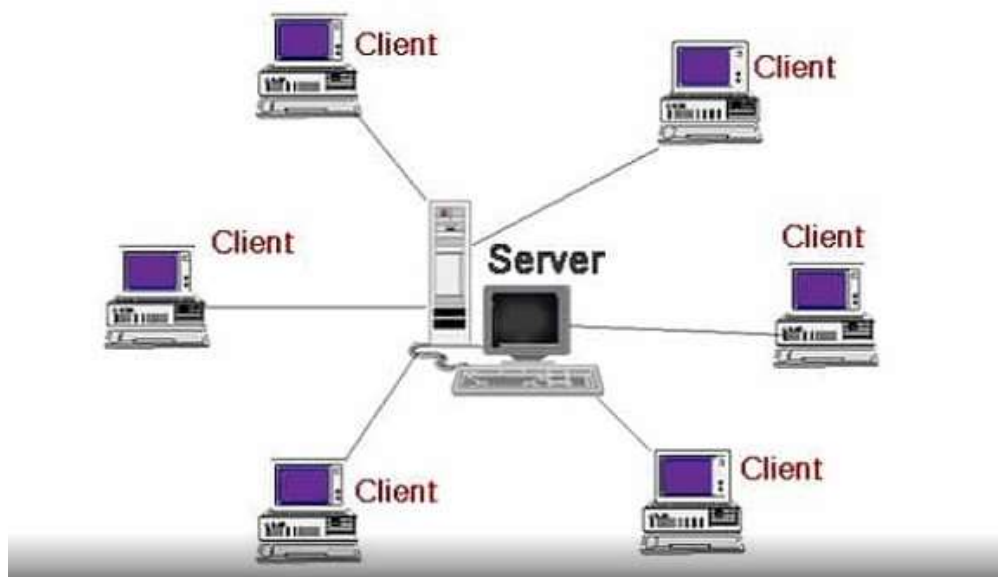
Client chính là khách hàng sử dụng dịch vụ. Nó có thể là một tổ chức hay cá nhân cụ thể nào đó. Và khi khái niệm này được sử dụng trong lĩnh vực kỹ thuật số thì cũng mang ý nghĩa tương tự như vậy. Trong Client Server thì Client chính là một máy tính

(Host). Chúng có khả năng nhận thông tin từ nhà cung cấp và sử dụng dịch vụ cụ thể (Server).

- **Server:**

Server là từ dùng để nói về một máy chủ hoặc một phương tiện được sử dụng để phục vụ các dịch vụ nào đó. Khi khái niệm này được sử dụng trong lĩnh vực công nghệ thì Server là một máy tính từ xa. Chúng có chức năng là cung cấp các thông tin (dữ liệu) cho một dịch vụ cụ thể nào đó hoặc quyền truy cập đối với dịch vụ.

Hiểu một cách đơn giản thì Client là nguồn yêu cầu và Server phải phục vụ theo nó.



Hình 1. 2 Mô hình nguyên lí hoạt động của mô hình Client Server

1.1.2. Các ưu nhược điểm của mô hình Client Server

Ưu điểm:

- **Tập trung:** khả năng kiểm soát tập trung (Centralization) đã được tích hợp sẵn. Tất cả mọi thông tin cần thiết đều sẽ được đặt ở một vị trí duy nhất. Tính năng này giúp cho mọi sự cố trong mạng đều sẽ được giải quyết ở cùng một nơi thống nhất. Đồng thời, việc cập nhật cơ sở tài nguyên, dữ liệu cũng sẽ dễ dàng hơn rất nhiều.
- **Bảo mật:** tất cả các dữ liệu đều sẽ được bảo vệ một cách tối đa nhờ vào hệ thống kiến trúc tập trung của mạng, giúp người dùng kiểm soát truy cập để chỉ có những ai được cấp quyền truy cập thì mới được thực hiện các thao tác cần thiết. Vì vậy chúng ta cần phải áp đặt thông tin đăng nhập cũng như Username

hay Password. Nếu dữ liệu của chúng ta bị mất thì các file sẽ được khôi phục một cách vô cùng dễ dàng chỉ từ một bản sao lưu duy nhất mà thôi.

- Khả năng mở rộng: Mô hình mạng kết nối Client Server có khả năng mở rộng vô cùng tốt. Chỉ cần người dùng cần sử dụng bất cứ lúc nào thì họ cũng có thể tăng được số lượng tài nguyên của mình.
- Khả năng truy cập: Tất cả mọi Client đều có khả năng đăng nhập được vào hệ thống mạng máy tính. Điều này sẽ giúp cho tất cả các nhân viên đều có thể truy cập thông tin của công ty một cách dễ dàng mà không cần phải dùng một terminal mode hoặc một bộ xử lý nào khác.

Nhược điểm :

- Tắc nghẽn lưu lượng: Trong trường hợp có quá nhiều Client tạo request từ cùng một Server thì nó có thể sẽ làm cho kết nối chậm hơn. Trong trường hợp xấu nhất còn có thể xuất hiện hiện tượng crash. Khi một server bị quá tải thì sẽ tạo ra nhiều vấn đề khi truy cập thông tin.
- Độ bền: Client Server là mạng tập trung chính vì thế, khi Server chính xảy ra sự cố hoặc bị nhiễu thì cũng đồng nghĩa với việc toàn bộ hệ thống mạng sẽ bị gián đoạn.
- Chi phí: Chi phí được sử dụng để thiết lập và bảo trì Server trong Client Server thường sẽ khá cao. Lý do là vì các hệ thống mạng có sức mạnh rất lớn cũng đồng nghĩa với việc giá để chi cho việc này là rất đắt. Chính vì vậy, không phải ai cũng có khả năng chi trả và sử dụng.
- Bảo trì: Khi các Server thực hiện triển khai để làm việc thì nó cũng sẽ hoạt động một cách không ngừng nghỉ. Vì vậy chúng ta cần phải quan tâm đến việc bảo trì hệ thống đúng mức. Khi xảy ra bất cứ vấn đề gì cũng cần phải giải quyết ngay lập tức. Đó là lí do cần phải có một nhà quản lý mạng chuyên biệt để tiến hành duy trì hoạt động của Server khi chúng được đưa vào và sử dụng.
- Tài nguyên: không phải tất cả tài nguyên hiện có trên Server đều sử dụng được.

1.2. Tổng quan về Spring Boot



Hình 1. 3 Logo Spring Boot

1.2.1. Spring boot là gì?

Spring Boot là một framework mở rộng của Spring Framework, được phát triển nhằm đơn giản hóa quá trình xây dựng các ứng dụng Java hiện đại. Thay vì phải cấu hình thủ công nhiều thành phần như ở Spring truyền thống, Spring Boot cung cấp cơ chế tự động cấu hình (auto-configuration), đi kèm với hệ sinh thái Starter giúp lập trình viên có thể khởi tạo và triển khai một ứng dụng nhanh chóng, ít cấu hình hơn nhưng vẫn đảm bảo tính linh hoạt và mạnh mẽ.

Spring Boot hỗ trợ xây dựng ứng dụng theo kiến trúc đa tầng, phù hợp cho các hệ thống web, API RESTful, microservices và các ứng dụng doanh nghiệp. Framework này tích hợp sẵn máy chủ web nhúng như Tomcat hoặc Jetty, giúp ứng dụng có thể chạy độc lập mà không cần cấu hình máy chủ bên ngoài. Với khả năng quản lý dependency tự động, kiểm thử dễ dàng, tích hợp mạnh mẽ với Spring Security, JPA/Hibernate và nhiều công nghệ khác, Spring Boot trở thành một trong những lựa chọn phổ biến nhất để phát triển backend trong các dự án hiện nay.

Spring Boot được xây dựng dựa trên nền tảng Spring Framework, nhưng được tối ưu để giúp lập trình viên giảm bớt khối lượng công việc cấu hình trong quá trình phát triển ứng dụng. Với triết lý “Convention over Configuration” (Ưu tiên quy ước hơn cấu hình), Spring Boot định nghĩa sẵn nhiều quy tắc mặc định thông minh, cho phép người dùng tập trung vào viết logic nghiệp vụ thay vì tốn thời gian thiết lập môi trường hay kết nối các thành phần. Đồng thời, Spring Boot cung cấp sẵn các Starter POM – tập hợp phụ thuộc (dependency) được cấu hình sẵn cho từng nhóm chức năng như Web, JPA, Security, Thymeleaf..., giúp quá trình tích hợp thư viện trở nên đơn

giản và đồng bộ hơn.

Một điểm nổi bật khác của Spring Boot là hỗ trợ máy chủ nhúng (embedded server) như Tomcat, Jetty hoặc Undertow. Điều này cho phép ứng dụng chạy như một file *.jar* độc lập mà không cần cài đặt hay cấu hình máy chủ web bên ngoài, từ đó đơn giản hóa việc triển khai (deployment) và mở đường cho kiến trúc microservices. Ngoài ra, Spring Boot cũng cung cấp hệ thống Actuator, giúp theo dõi trạng thái ứng dụng, quản lý cấu hình runtime và giám sát hiệu năng – rất hữu ích trong môi trường thực tế.

Các tính năng và ứng dụng của Spring boot

Spring Boot cung cấp nhiều tính năng mạnh mẽ giúp đơn giản hóa quá trình phát triển ứng dụng Java. Một trong những tính năng quan trọng nhất là tự động cấu hình (Auto Configuration), cho phép framework tự động thiết lập các thành phần cần thiết dựa trên thư viện mà ứng dụng đang sử dụng, giúp lập trình viên không phải cấu hình thủ công như ở Spring truyền thống. Bên cạnh đó, Spring Boot hỗ trợ Starter Dependencies, tức các gói phụ thuộc được cấu hình sẵn, giúp người dùng dễ dàng tích hợp những module phổ biến như Spring Web, Spring Data JPA, Spring Security hay Thymeleaf mà không cần tự quản lý từng thư viện nhỏ lẻ.

Một ưu điểm nổi bật khác là Spring Boot tích hợp máy chủ web nhúng như Tomcat, Jetty hoặc Undertow, giúp ứng dụng có thể chạy trực tiếp dưới dạng file *.jar* mà không cần triển khai lên server ngoài. Điều này đặc biệt thuận tiện trong phát triển microservices, nơi mỗi dịch vụ có thể chạy độc lập và triển khai nhanh chóng. Spring Boot còn cung cấp Spring Boot Actuator, một module hỗ trợ giám sát và kiểm tra sức khỏe ứng dụng thông qua các endpoint chuyên dụng, phù hợp cho việc theo dõi hệ thống trong môi trường production.

Về mặt ứng dụng, Spring Boot được sử dụng rộng rãi trong việc xây dựng các hệ thống web backend, API RESTful, ứng dụng doanh nghiệp, hệ thống thương mại điện tử, ứng dụng quản lý và đặc biệt là các kiến trúc microservices. Nhờ khả năng mở rộng linh hoạt, tích hợp mạnh mẽ với hệ sinh thái Spring và cộng đồng hỗ trợ lớn, Spring Boot trở thành lựa chọn hàng đầu cho những dự án yêu cầu sự ổn định, tốc độ phát triển nhanh và khả năng bảo trì lâu dài.

1.1.2. Cách cài đặt Spring boot

Spring Boot có thể chạy trên các hệ điều hành như Windows, macOS và Linux, cách cài đặt rất đơn giản:

- Windows: Cài JDK 17+ và Maven, sau đó truy cập start.spring.io để tạo dự án Spring Boot.
- macOS: Dùng Homebrew cài đặt bằng lệnh `brew install openjdk@17 maven`.
- Ubuntu/Linux: Dùng lệnh `sudo apt install openjdk-17-jdk maven`.

Kiểm tra phiên bản bằng: “`java -version`” hoặc “`mvn -v`”

1.3. Tổng quan về Visual Studio Code



Hình 1. 4 Logo Visual Studio Code

1.3.1. Visual Studio Code là gì?

Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn mở và đa nền tảng do Microsoft phát triển, được sử dụng rộng rãi trong lĩnh vực lập trình nhờ sự nhẹ, nhanh và khả năng mở rộng linh hoạt. VS Code hỗ trợ hầu hết các ngôn ngữ lập trình phổ biến như JavaScript, TypeScript, Java, Python, PHP, C#, Go... thông qua hệ thống extensions phong phú trên Marketplace. Với giao diện trực quan, dễ sử dụng và khả năng tùy chỉnh cao, VS Code mang đến môi trường lập trình hiện đại, thích hợp cho cả người mới bắt đầu lẫn lập trình viên chuyên nghiệp.

Bên cạnh các tính năng cơ bản của một trình soạn thảo mã nguồn, VS Code còn tích hợp mạnh mẽ các công cụ hỗ trợ như IntelliSense (gợi ý mã thông minh), Debugger (gỡ lỗi trực tiếp), Terminal tích hợp, Git Control, và khả năng đồng bộ hóa cấu hình (Setting Sync). Nhờ kiến trúc mở rộng linh hoạt, người dùng có thể cài đặt thêm các plugin để hỗ trợ framework hoặc ngôn ngữ chuyên biệt như Spring Boot, React, Angular, Node.js, Docker,... Điều này giúp VS Code trở thành một lựa chọn phổ biến trong phát triển ứng dụng web, API và hệ thống đa nền tảng.

1.3.2. Những ưu điểm nổi bật của Visual Studio Code

Visual Studio Code sở hữu nhiều ưu điểm nổi bật giúp nó trở thành một trong những công cụ lập trình được sử dụng rộng rãi nhất hiện nay. Trước hết, VS Code có hiệu năng nhẹ và tốc độ xử lý nhanh, cho phép khởi động và vận hành mượt mà ngay cả trên các máy cấu hình trung bình, điều mà nhiều IDE truyền thống không làm được. Bên cạnh đó, VS Code hỗ trợ hệ thống mở rộng (Extension Marketplace) vô cùng phong phú, cho phép người dùng cài đặt thêm các plugin phục vụ cho bất kỳ ngôn ngữ hoặc nền tảng lập trình nào như Java, Python, PHP, Node.js, Spring Boot, React hay Docker.

Một ưu điểm quan trọng khác là VS Code tích hợp nhiều công cụ hỗ trợ lập trình mạnh mẽ như IntelliSense (gợi ý mã thông minh), Debugging (gỡ lỗi trực tiếp), Terminal tích hợp, cùng khả năng quản lý mã nguồn thông qua Git Control. Những tính năng này giúp lập trình viên làm việc hiệu quả hơn, giảm sai sót và tăng tốc độ phát triển sản phẩm. Ngoài ra, VS Code có khả năng tùy chỉnh cao, cho phép người dùng thay đổi giao diện, phím tắt, cấu hình môi trường và đồng bộ hóa cài đặt giữa các thiết bị dễ dàng thông qua tính năng Settings Sync

Với thiết kế hiện đại, giao diện trực quan và cộng đồng hỗ trợ lớn, Visual Studio Code không chỉ là một trình soạn thảo mã nguồn đơn thuần mà còn là một môi trường phát triển linh hoạt, mạnh mẽ và phù hợp cho nhiều loại dự án từ nhỏ đến lớn. Đây cũng chính là lý do VS Code được lựa chọn rộng rãi trong các đề tài và dự án phát triển phần mềm.

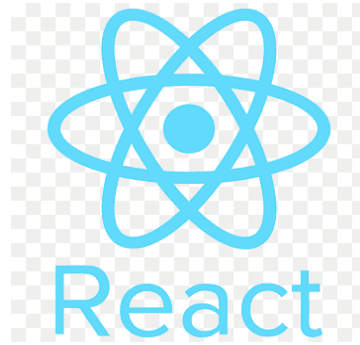
1.3.3. Các lý do nên sử dụng Visual Studio Code

- Visual Studio Code mang rất nhiều ưu điểm vượt trội so với bất kỳ IDE nào khác
- Hỗ trợ đa nền tảng: Linux, Mac, Windows,...
- Hỗ trợ đa ngôn ngữ: C/C++, C#, F#, JavaScript, JSON, Visual Basic, HTML, CSS,...
- Ít dụng lượng
- Tính năng mạnh mẽ
- Intellisense chuyên nghiệp
- Giao diện thân thiện

- Kiến trúc mạnh mẽ và người dùng có thể khai thác mở rộng
- Số lượng người sử dụng lớn tạo nên cộng đồng hỗ trợ rộng rãi.

Visual Studio Code là phần mềm miễn phí. Chính vì vậy ứng dụng chuyên biên tập, soạn thảo Code này trở nên phổ biến nhất hiện nay.

1.4. Tổng quan về Reactjs



Hình 1. 5 Logo React

1.4.1. Reactjs là gì?

ReactJS là một thư viện JavaScript mã nguồn mở dùng để phát triển giao diện người dùng (frontend), được xây dựng và phát triển bởi Facebook (Meta). ReactJS cho phép xây dựng các ứng dụng web theo mô hình component-based, trong đó giao diện được chia thành các thành phần độc lập, có khả năng tái sử dụng cao và dễ dàng mở rộng. Cơ chế hoạt động của React dựa trên Virtual DOM, giúp tối ưu quá trình cập nhật giao diện bằng cách chỉ render những phần thực sự thay đổi, từ đó nâng cao hiệu năng và trải nghiệm người dùng.

Bên cạnh đó, ReactJS áp dụng cơ chế one-way data binding, giúp kiểm soát luồng dữ liệu chặt chẽ, giảm thiểu lỗi trong quá trình phát triển ứng dụng. Với cú pháp JSX, React cho phép lập trình viên viết HTML trực tiếp trong JavaScript, giúp mã nguồn trở nên trực quan và dễ bảo trì hơn. ReactJS có khả năng tích hợp linh hoạt với nhiều công nghệ backend như Spring Boot, NodeJS, Django thông qua các API, đồng thời hỗ trợ tốt các thư viện quản lý trạng thái và định tuyến. Trong đề tài này, ReactJS được sử dụng để xây dựng giao diện người dùng, xử lý các chức năng tương tác trên nền web và kết nối dữ liệu với hệ thống backend thông qua RESTful API.

Ngoài ra, ReactJS sở hữu hệ sinh thái phong phú với nhiều thư viện và công cụ hỗ trợ phát triển như React Router để điều hướng trang trong ứng dụng Single Page Application (SPA), Redux hoặc Zustand để quản lý trạng thái toàn cục, cùng với các

thư viện gọi API như Axios, Fetch giúp kết nối dữ liệu từ phía máy chủ. ReactJS cũng hỗ trợ tốt việc xây dựng giao diện responsive khi kết hợp với các framework CSS như Tailwind CSS, Bootstrap, giúp tối ưu hiển thị trên nhiều thiết bị. Nhờ cộng đồng phát triển lớn mạnh và khả năng mở rộng linh hoạt, ReactJS không chỉ phù hợp với các dự án nhỏ mà còn đáp ứng tốt yêu cầu của các hệ thống web quy mô lớn, có độ ổn định cao và khả năng phát triển lâu dài.

1.4.2. Những đặc điểm quan trọng của Reactjs.

Các đặc điểm quan trọng cần nắm vững khi lập trình ReactJS bao gồm:

Components (thành phần): React cho phép người dùng tạo ra các thành phần độc lập, dễ dàng quản lý và có thể tái sử dụng. Mỗi component có thể chứa mã HTML, CSS và JavaScript riêng biệt. Có 2 loại component chính là Class component

Virtual DOM (DOM ảo): React tạo ra một bản sao ảo của DOM và so sánh với trạng thái trong component của DOM thực (real DOM), sau đó cập nhật những phần khác biệt, thay vì phải cập nhật toàn bộ DOM thực mỗi khi có thay đổi.

Single-Page Applications (SPA): React tích hợp tốt với các thư viện định tuyến (routing) và quản lý trạng thái của ứng dụng nên thích hợp với việc phát triển các SPA.

Hệ sinh thái đa dạng: Với nhiều thư viện và công cụ bổ sung như React Router, Redux, Material-UI, ..

1.4.3. Lợi ích khi sử dụng Reactjs

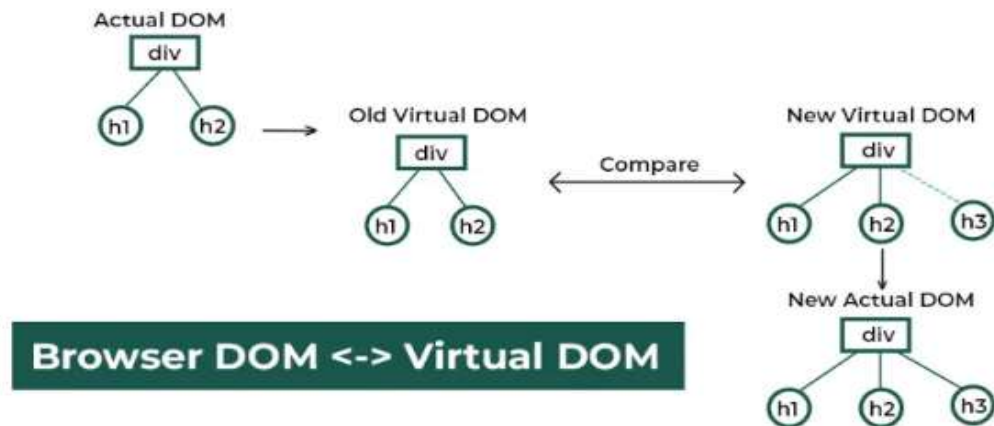
Phù hợp với nhiều loại website: Không cần code nhiều như khi tạo các website chỉ thuần dùng Javascript hay HTML, ReactJS giúp cho việc khởi tạo website trở nên dễ dàng hơn dù là website thương mại điện tử hay website sản phẩm/dịch vụ thông thường.

Tái sử dụng Component: Các component độc lập, có khả năng tái sử dụng giúp giảm số lượng code trùng lặp, dễ bảo trì code và có thể được dùng 1 phần cho các dự án khác nhau nếu các Component được xây dựng đủ tốt, đủ chặt chẽ.

Thư viện hỗ trợ phát triển web được ưa chuộng: Có thể thấy rõ mức độ quan tâm đối với lập trình ReactJS trên toàn thế giới qua số liệu từ Google Trend. Cũng chẳng nói đâu xa, trên một số trang tuyển dụng việc làm IT hàng đầu tại Việt Nam, số lượng tuyển dụng cho vị trí React Developer rất nhiều với mức lương hấp dẫn. Điều đó cho thấy độ phổ biến của thư viện Javascript này.

1.4.4. Lợi ích khi sử dụng Reactjs

Về cơ bản, React hoạt động bằng cách tạo ra một Virtual DOM (DOM ảo) và cập nhật DOM của trình duyệt khi có thay đổi.



Hình 1. 6 Cách thức hoạt động của Reactjs

Các bước chi tiết về cách thức hoạt động của Reactjs:

- Khởi tạo ứng dụng React với các lệnh khởi tạo và một component gốc để xây dựng giao diện người dùng.
- Xây dựng component độc lập, thể hiện cho một phần cụ thể của giao diện. Mỗi component là một hàm JavaScript tạo ra một phần tử HTML.
- Khi người dùng tương tác với ứng dụng React, cụ thể là trạng thái trong một component bất kỳ nào thay đổi, React sẽ gọi hàm render() và đồng thời một cây Elements mới được tạo ra. Nó được gọi là Virtual DOM.
- React chạy thuật toán diffing để xác định những thay đổi trong DOM ảo, so sánh nó với DOM hiện tại để tìm ra sự khác biệt khi trạng thái trong component thay đổi.
- Cập nhật những thay đổi ở những nơi cần thiết của DOM thực (real DOM) thay vì phải cập nhật toàn bộ.
- Hiển thị website với trạng thái mới với các mục cập nhật được thêm vào UI.

1.5. Tổng quan về hệ quản trị cơ sở dữ liệu MySQL là gì?

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System – RDBMS) phổ biến và được sử dụng rộng rãi trên toàn thế giới. MySQL hoạt động theo mô hình lưu trữ dữ liệu dạng bảng (table), trong đó các bảng được liên kết với nhau thông qua các khóa (keys) như khóa chính (Primary Key) và khóa ngoại (Foreign Key). Đây là hệ quản trị cơ sở dữ liệu mã nguồn mở, được phát triển bởi Oracle Corporation, nổi bật với tính ổn định, tốc độ xử lý cao và khả năng mở

rộng linh hoạt, phù hợp cho nhiều loại ứng dụng từ nhỏ đến lớn, bao gồm hệ thống web, ứng dụng thương mại điện tử, dịch vụ trực tuyến và các hệ thống quản lý doanh nghiệp.

MySQL hỗ trợ đầy đủ các chuẩn SQL (Structured Query Language) để thao tác dữ liệu, bao gồm truy vấn, lọc dữ liệu, cập nhật, xóa và quản lý cấu trúc bảng. Bên cạnh đó, MySQL cung cấp nhiều công cụ tối ưu hóa, cơ chế quản lý giao dịch (transaction), kiểm soát toàn vẹn dữ liệu và cơ chế chỉ mục (index) giúp tăng hiệu năng xử lý truy vấn. Với khả năng tương thích tốt với nhiều ngôn ngữ lập trình và framework, MySQL trở thành lựa chọn phổ biến trong mô hình phát triển web hiện đại, đặc biệt là khi kết hợp với backend Java Spring Boot.

Ngoài ra, MySQL hỗ trợ nhiều loại storage engine như InnoDB, MyISAM, trong đó InnoDB được sử dụng rộng rãi nhờ khả năng hỗ trợ giao dịch ACID, khóa ngoại và cơ chế khóa dòng (row-level locking), giúp đảm bảo an toàn dữ liệu khi nhiều người dùng thao tác đồng thời. MySQL cũng dễ dàng triển khai trên nhiều môi trường như localhost, cloud server, Docker hoặc các nền tảng dịch vụ cơ sở dữ liệu (Database as a Service).

Trong đề tài, MySQL được lựa chọn làm hệ quản trị cơ sở dữ liệu chính do khả năng xử lý nhanh, phù hợp với các nghiệp vụ của chuỗi nhà hàng như quản lý thực đơn, đơn hàng, tồn kho, người dùng, bàn ăn và các báo cáo doanh thu. Với cấu trúc rõ ràng, tính hiệu quả và độ tin cậy cao, MySQL đảm bảo hệ thống hoạt động ổn định và đáp ứng tốt nhu cầu mở rộng trong tương lai.



Hình 1. 7 Logo MySql

1.5.1. Tổng quan về hệ quản trị cơ sở dữ liệu MySQL là gì?

Hệ thống website bất động sản sử dụng MySQL để lưu trữ:

- Thông tin người dùng, môi giới, admin.

- Danh sách tin đăng, hình ảnh, vị trí, giá.
- Các mối quan hệ như yêu thích, thông báo, nhắn tin.

Các tính năng của MySQL

MySQL có các tính năng cơ bản như sau:

➤ **Truy vấn SQL linh hoạt:**

MySQL sử dụng ngôn ngữ SQL (Structured Query Language) để thực hiện các thao tác như SELECT, INSERT, UPDATE, DELETE, cho phép truy vấn dữ liệu mạnh mẽ và chính xác. Ngoài ra, MySQL hỗ trợ các hàm xử lý dữ liệu phức tạp, truy vấn con, hàm tổng hợp, và các biểu thức điều kiện giúp người dùng dễ dàng thao tác và phân tích dữ liệu.

➤ **Hỗ trợ giao dịch (Transactions):**

MySQL hỗ trợ giao dịch ACID (Atomicity, Consistency, Isolation, Durability), đảm bảo dữ liệu luôn an toàn và nhất quán trong quá trình xử lý. Người dùng có thể sử dụng các lệnh như START TRANSACTION, COMMIT, và ROLLBACK để kiểm soát các hoạt động giao dịch, đặc biệt hữu ích trong các hệ thống ngân hàng, thương mại điện tử hay quản lý tài chính.

➤ **Hệ thống sao lưu và phục hồi dữ liệu:**

MySQL cung cấp các công cụ sao lưu như mysqldump, mysqlpump, và khả năng replication (nhân bản) giữa các máy chủ. Nhờ đó, dữ liệu có thể được sao lưu, phục hồi hoặc nhân rộng dễ dàng để đảm bảo tính sẵn sàng và an toàn trong trường hợp xảy ra sự cố.

➤ **Chỉ mục (Indexing):**

MySQL hỗ trợ chỉ mục đơn, chỉ mục phức hợp và chỉ mục toàn văn (FULLTEXT) giúp tăng tốc độ tìm kiếm và tối ưu hóa truy vấn. Người dùng có thể tạo primary key, unique key, hoặc foreign key để đảm bảo ràng buộc và tính toàn vẹn dữ liệu.

➤ **Bảo mật dữ liệu:**

MySQL cung cấp nhiều cơ chế bảo mật như xác thực người dùng, mã hóa SSL/TLS, phân quyền truy cập theo từng tài khoản hoặc cơ sở dữ liệu. Điều này giúp bảo vệ dữ liệu trước các truy cập trái phép và tấn công mạng.

➤ **Hiệu năng và khả năng mở rộng:**

MySQL có khả năng xử lý khối lượng dữ liệu lớn với tốc độ cao nhờ vào các cơ chế như query cache, bộ nhớ đệm (buffer pool) và phân vùng bảng (partitioning). Ngoài ra, MySQL hỗ trợ replication và clustering, giúp mở rộng hệ thống khi có nhiều người truy cập đồng thời

➤ **Hỗ trợ đa nền tảng và dễ tích hợp:**

MySQL hoạt động trên hầu hết các hệ điều hành như Windows, macOS, Linux, và tương thích với nhiều ngôn ngữ lập trình như Java, PHP, Python, Node.js. Đây là lý do MySQL thường được sử dụng trong các ứng dụng web, hệ thống doanh nghiệp và dịch vụ thương mại điện tử.

1.5.2. Ưu nhược và điểm của MySQL

Ưu điểm:

- MySQL có khả năng lưu trữ và xử lý dữ liệu có cấu trúc một cách hiệu quả, phù hợp với các ứng dụng cần tính nhất quán và chính xác cao.
- Hỗ trợ truy vấn phức tạp với ngôn ngữ SQL, cho phép lọc, sắp xếp, kết nối (JOIN) và thống kê dữ liệu linh hoạt, đáp ứng tốt cho các hệ thống lớn.
- Đảm bảo tính toàn vẹn dữ liệu nhờ vào việc sử dụng các ràng buộc (constraint) như primary key, foreign key, unique, not null và check.
- Tốc độ truy vấn nhanh khi được tối ưu hóa thông qua hệ thống chỉ mục (index) và bộ nhớ đệm (cache), giúp giảm thời gian truy xuất dữ liệu.
- MySQL có khả năng sao lưu và phục hồi dữ liệu mạnh mẽ, đồng thời hỗ trợ nhân bản dữ liệu (replication) giữa các máy chủ để đảm bảo tính sẵn sàng cao.
- Tính tương thích cao với nhiều hệ điều hành và ngôn ngữ lập trình khác nhau như Java, PHP, Python, Node.js... giúp dễ dàng tích hợp trong nhiều dự án web và doanh nghiệp.

Nhược điểm:

- MySQL không linh hoạt với dữ liệu phi cấu trúc hoặc thay đổi thường xuyên, vì dữ liệu phải tuân thủ theo mô hình bảng cố định.
- Khi hệ thống có lượng dữ liệu rất lớn, hiệu suất truy vấn có thể giảm nếu không tối ưu chỉ mục và cấu trúc cơ sở dữ liệu hợp lý.
- Việc thay đổi cấu trúc bảng (schema) trong quá trình vận hành đôi khi phức tạp và có thể ảnh hưởng đến dữ liệu hiện có.
- MySQL có giới hạn trong xử lý giao dịch song song (concurrency) khi so sánh với các hệ quản trị cơ sở dữ liệu cao cấp khác như PostgreSQL hoặc Oracle.
- Phiên bản miễn phí thiếu một số tính năng nâng cao như mã hóa toàn bộ dữ liệu, giám sát hiệu suất chuyên sâu hay công cụ phục hồi tự động.

1.5.3 Cách cài đặt MySQL.

Quá trình cài đặt MySQL được thực hiện thông qua bộ cài MySQL Installer, là trình cài đặt chính thức do Oracle cung cấp, giúp người dùng dễ dàng thiết lập hệ quản trị cơ sở dữ liệu MySQL trên máy tính cá nhân. Đầu tiên, người dùng truy cập trang chủ MySQL, tải về bộ cài phù hợp với hệ điều hành và khởi chạy trình cài đặt. Trong bước lựa chọn thành phần, có thể chọn cài đặt các gói cần thiết cho quá trình phát triển như MySQL Server, MySQL Workbench, MySQL Shell và các công cụ hỗ trợ quản lý cơ sở dữ liệu.

Trong quá trình cài đặt MySQL Server, người dùng tiến hành cấu hình các thông số ban đầu, bao gồm lựa chọn mức độ bảo mật, thiết lập mật khẩu cho tài khoản quản trị (*root*), cấu hình cổng mặc định (3306), chế độ khởi động dịch vụ và kiểu xác thực người dùng. Các tùy chọn này giúp đảm bảo MySQL hoạt động ổn định, an toàn và phù hợp với môi trường phát triển phần mềm.

Sau khi hoàn tất các bước cấu hình, trình cài đặt tự động tạo dịch vụ MySQL trên hệ điều hành và khởi động máy chủ cơ sở dữ liệu. Người dùng có thể đăng nhập và quản lý hệ thống thông qua MySQL Workbench, công cụ được sử dụng phổ biến để tạo cơ sở dữ liệu, thiết kế bảng, chạy truy vấn và theo dõi hoạt động của server. Việc cài đặt thành công MySQL là tiền đề quan trọng để triển khai hệ thống quản lý chuỗi nhà hàng, đảm bảo môi trường dữ liệu ổn định cho quá trình xây dựng và kiểm thử chức năng của đề tài.

CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Phát biểu bài toán

Trong bối cảnh thị trường bất động sản ngày càng phát triển mạnh, số lượng dự án – giao dịch – và nhu cầu tra cứu thông tin của khách hàng tăng lên nhanh chóng, việc quản lý theo cách thủ công trở nên không còn phù hợp. Các đơn vị môi giới, chủ đầu tư và sàn giao dịch thường gặp nhiều khó khăn như: dữ liệu phân tán, khó kiểm soát hồ sơ tài sản, sai sót khi nhập liệu, chậm trễ trong việc cập nhật thông tin dự án, thiếu sự kết nối giữa các nhóm tư vấn – chăm sóc khách hàng – quản lý bán hàng, và hạn chế trong việc tổng hợp báo cáo phục vụ việc ra quyết định.

Bài toán đặt ra là cần xây dựng một hệ thống website quản lý bất động sản giúp số hóa toàn bộ quy trình nghiệp vụ. Hệ thống phải hỗ trợ quản lý danh mục tài sản, tình trạng giao dịch, hồ sơ pháp lý, thông tin chủ sở hữu, quản lý đội ngũ môi giới, theo dõi lịch sử chăm sóc khách hàng và tối ưu quy trình bán hàng. Ngoài ra, hệ thống cần cung cấp cổng thông tin cho khách hàng, cho phép họ xem dự án, lọc tài sản theo nhu cầu, đặt lịch xem nhà, gửi yêu cầu tư vấn hoặc theo dõi tiến độ mua bán.

Hệ thống cũng phải đảm bảo dữ liệu được cập nhật theo thời gian thực, phân quyền rõ ràng giữa Admin – Môi giới – Khách mua, đồng thời giữ tính ổn định và an toàn khi nhiều người cùng truy cập. Hơn nữa, giải pháp cần hỗ trợ khả năng mở rộng cho nhiều chi nhánh, nhiều sàn giao dịch hoặc nhiều dự án khác nhau trong tương lai.

Tóm lại, bài toán đặt ra là xây dựng một hệ thống quản lý – kinh doanh bất động sản toàn diện, giúp tự động hóa quy trình vận hành, nâng cao hiệu suất làm việc, giảm rủi ro sai sót, cải thiện trải nghiệm khách hàng và tạo nền tảng cho chuyển đổi số của doanh nghiệp trong lĩnh vực nhà đất.

2.2. Phân tích chức năng

Hệ thống được chia gồm Admin, Agent, và User. Mỗi loại người dùng có quyền hạn và chức năng khác nhau, phù hợp với vai trò hệ thống như sau:

Admin – Người quản trị hệ thống

Admin là nhóm người dùng có quyền hạn cao nhất, chịu trách nhiệm duy trì hoạt động ổn định của toàn bộ nền tảng. Các chức năng chính bao gồm:

- Quản lý người dùng: Admin có thể thêm mới, chỉnh sửa, khóa hoặc xóa tài khoản của Agent và User. Điều này đảm bảo hệ thống chỉ duy trì những tài khoản hợp lệ và hoạt động đúng quy định.
- Quản lý tin đăng: Tất cả tin đăng của môi giới đều được Admin kiểm duyệt trước khi hiển thị. Admin có quyền chỉnh sửa, ẩn hoặc xóa các tin vi phạm chính sách nhằm đảm bảo chất lượng nội dung.
- Quản lý danh mục hệ thống: Admin tạo và điều chỉnh các danh mục như loại bất động sản, khu vực, tiện ích... giúp dữ liệu được chuẩn hóa và thống nhất trên toàn hệ thống.
- Quản lý môi giới: Admin theo dõi danh sách môi giới, thống kê số lượng tin đăng, đánh giá hiệu suất và quá trình hoạt động. Qua đó, Admin dễ dàng giám sát và hỗ trợ đội ngũ môi giới khi cần.
- Gửi thông báo hệ thống: Admin có thể gửi thông báo đến toàn bộ người dùng hoặc một nhóm cụ thể, hỗ trợ truyền tải thông tin nhanh chóng và kịp thời.
- Quản lý phản hồi: Admin tiếp nhận các phản ánh, khiếu nại của khách hàng hoặc môi giới, xử lý và điều phối nhằm duy trì trải nghiệm người dùng tốt nhất.
- Xem báo cáo thống kê: Hệ thống cung cấp cho Admin các báo cáo về tin đăng, số lượng tài khoản, giao dịch, tình trạng người dùng còn/hết hạn và các số liệu phân tích khác giúp đưa ra quyết định quản lý hiệu quả.

Agent – Môi giới bất động sản

Agent là nhóm người dùng trực tiếp tạo ra nội dung trên hệ thống thông qua các tin đăng bất động sản. Họ là cầu nối giữa khách hàng và tài sản. Các chức năng chính bao gồm:

- Đăng tin bất động sản: Agent có thể tạo mới tin đăng kèm hình ảnh, mô tả, thông tin vị trí, giá bán/cho thuê. Đây là chức năng cốt lõi trong hoạt động của môi giới.
- Quản lý tin đăng cá nhân: Agent được chỉnh sửa, cập nhật, ẩn hoặc xóa tin đăng của mình, cũng như đánh dấu giao dịch đã hoàn thành.
- Quản lý khách hàng tiềm năng: Agent có thể xem danh sách khách hàng đã lưu tin, gửi yêu cầu tư vấn hoặc quan tâm đến bất động sản của mình.

- Chat & tư vấn khách hàng: Hệ thống hỗ trợ nhắn tin trực tiếp giữa Agent và khách hàng để trao đổi chi tiết, đặt lịch xem nhà hoặc tư vấn giá.
- Nhận thông báo: Agent nhận được thông báo từ Admin và từ khách hàng giúp không bỏ lỡ yêu cầu, phản hồi hoặc cảnh báo từ hệ thống.
- Quản lý hồ sơ: Agent có thể cập nhật thông tin cá nhân, ảnh đại diện, chứng chỉ nghề nghiệp giúp tăng độ uy tín và minh bạch đối với khách hàng.
- Theo dõi hiệu quả hoạt động: Hệ thống cung cấp thống kê lượt xem, lượt yêu thích, lượt liên hệ... giúp Agent đánh giá chất lượng tin đăng và tối ưu nội dung.

User – Khách hàng

User có nhu cầu mua bất động sản, trực tiếp tìm kiếm và tiếp cận thông tin bất động sản. Họ có các chức năng sau:

Khách chưa đăng nhập:

- Tìm kiếm & lọc tin: User có thể tìm bất động sản theo nhiều tiêu chí như loại hình, giá, khu vực, diện tích, tiện ích... giúp tối ưu hóa quá trình tìm kiếm.
- Xem chi tiết tin đăng: Khách hàng có thể xem toàn bộ thông tin như ảnh, mô tả, vị trí bản đồ, giá và các thông tin liên hệ của môi giới.

Khách đã đăng nhập

- Lưu tin yêu thích: User có thể lưu lại các tin quan tâm để tiện theo dõi và so sánh sau này.
- Liên hệ & chat với môi giới: Hệ thống hỗ trợ tương tác trực tiếp qua chat hoặc gửi yêu cầu tư vấn.
- Nhận thông báo: User nhận thông báo khi tin đã lưu thay đổi giá, có tin liên quan mới, hoặc có phản hồi từ Agent.
- Quản lý tài khoản: User cập nhật hồ sơ cá nhân, đổi mật khẩu, xem lịch sử tìm kiếm và danh sách tin đã lưu.

2.3. Xác định tác nhân

Trong hệ thống có 3 loại người dùng chính: Admin, Agent và User

- **Admin:**

Là người dùng có quyền hạn cao nhất trong hệ thống. Admin có vai trò quản lý

toàn bộ hoạt động của hệ thống như phê duyệt tin đăng, quản lý người dùng (môi giới và khách hàng), xử lý phản hồi, thống kê dữ liệu, và gửi thông báo đến các tài khoản khác.

- **Agent:**

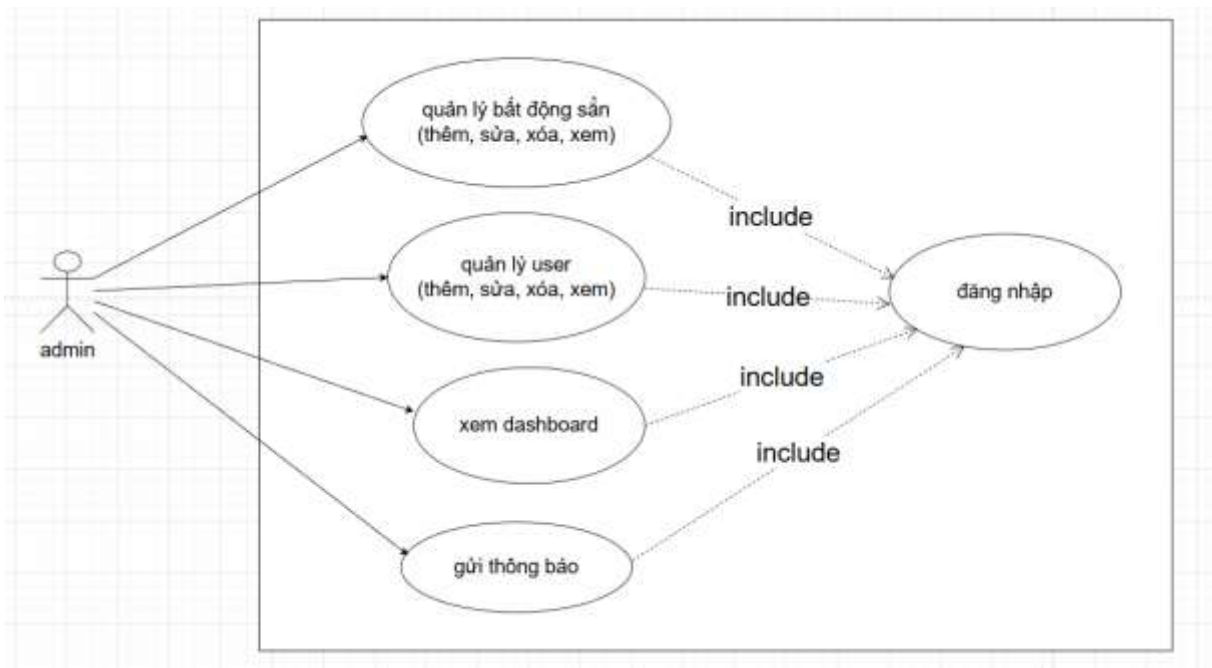
Là người dùng có quyền đăng tin bất động sản, chỉnh sửa hoặc xóa tin của mình, quản lý danh sách khách hàng tiềm năng, trò chuyện và tư vấn trực tiếp cho khách hàng. Môi giới chịu trách nhiệm cập nhật thông tin chính xác, hình ảnh và giá cả của bất động sản mà họ quản lý.

- **User:**

Là người dùng cuối của hệ thống, có thể tìm kiếm, xem chi tiết, lọc và lưu các tin bất động sản yêu thích. Khách hàng có thể gửi yêu cầu tư vấn hoặc trò chuyện với môi giới, nhận thông báo về các tin mới hoặc thay đổi giá, đồng thời quản lý hồ sơ cá nhân của mình.

2.4. Biểu đồ useCase

2.4.1. Biểu đồ use case cho quản trị viên (Admin)



Hình 2. 1 Biểu đồ Use Case quản trị viên

Mô tả Use Case quản trị viên (Admin)

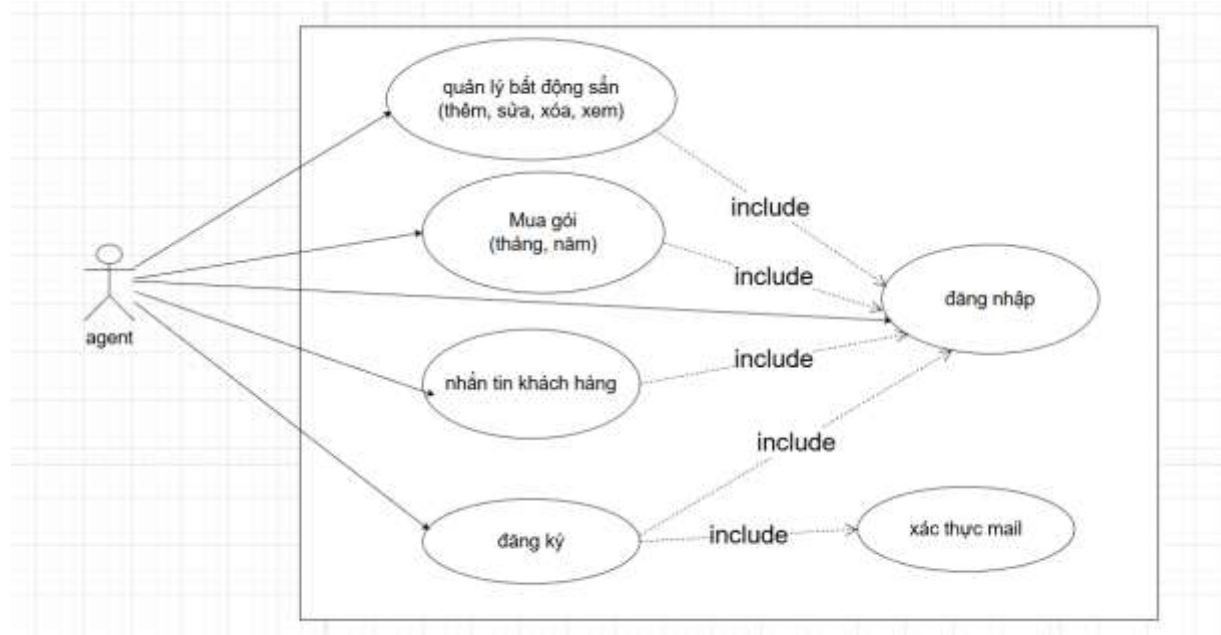
Trong hệ thống quản lý bất động sản, Admin là người chịu trách nhiệm điều hành và giám sát toàn bộ hoạt động trên nền tảng. Sau khi đăng nhập bằng tài khoản quản trị hợp lệ, Admin có thể truy cập vào tất cả các module như quản lý người dùng,

agent, tin đăng bất động sản, danh mục hệ thống, phản hồi – báo cáo và bảng thống kê. Từ giao diện quản trị, Admin có quyền thực hiện các thao tác như thêm, chỉnh sửa, vô hiệu hóa hoặc xóa dữ liệu và tài khoản, cũng như phê duyệt các tin đăng mà Agent gửi lên. Mọi hành động quản trị đều được hệ thống xử lý tự động, cập nhật dữ liệu theo thời gian thực và ghi lại log phục vụ kiểm tra hoặc truy vết sau này.

Trong quá trình thao tác, nếu Admin chọn quản lý bất động sản, hệ thống sẽ hiển thị danh sách tin đăng để Admin có thể xem chi tiết, chỉnh sửa thông tin hoặc phê duyệt những tin đang chờ duyệt. Đối với phần quản lý người dùng và agent, Admin có thể xem hồ sơ, khóa hoặc mở khóa tài khoản và điều chỉnh quyền truy cập khi cần thiết. Bên cạnh đó, Admin cũng có thể quản lý các danh mục dùng chung như loại bất động sản, hướng nhà, tiện ích... nhằm đảm bảo dữ liệu luôn đồng bộ và chính xác. Khi có báo cáo vi phạm hoặc khiếu nại từ người dùng, Admin truy cập mục phản hồi để xem chi tiết và xử lý theo quy định. Module thống kê cho phép Admin theo dõi số lượng người dùng, agent, tin đăng, lượt xem và các hoạt động nổi bật để hỗ trợ việc đưa ra quyết định quản trị.

Trong quá trình hoạt động, nếu phát sinh lỗi như thiếu quyền hạn, dữ liệu không tồn tại, lỗi hệ thống hoặc phiên đăng nhập hết hạn, hệ thống sẽ đưa ra thông báo phù hợp và hướng dẫn Admin thao tác tiếp theo. Khi hoàn thành công việc, Admin có thể đăng xuất để kết thúc phiên quản trị một cách an toàn. Tổng thể, Use Case Admin mô tả quy trình làm việc trung tâm của người quản trị trong hệ thống bất động sản, đảm bảo tính minh bạch, an toàn dữ liệu và hiệu quả vận hành.

2.4.2. Biểu đồ Use Case cho người môi giới (Agent)



Hình 2. 2 Biểu đồ Use Case cho người môi giới

Mô tả Use Case cho người môi giới (Agent)

Agent là nhóm người dùng đóng vai trò tạo ra nguồn dữ liệu chính cho hệ thống bất động sản. Họ sử dụng nền tảng để đăng tải và quản lý các tin đăng, tương tác với khách hàng và duy trì các hoạt động bán hàng thông qua hệ thống các gói dịch vụ. Để truy cập được các chức năng quan trọng như đăng tin, mua gói hay nhắn tin, Agent cần có tài khoản hợp lệ và đã hoàn tất bước xác thực thông tin nếu hệ thống yêu cầu.

Khi tham gia hệ thống, Agent có thể tự tạo và quản lý danh sách bất động sản của riêng mình: đăng tin mới, chỉnh sửa nội dung, cập nhật trạng thái hoặc xóa những tin không còn phù hợp. Ngoài ra, Agent có thể mua các gói dịch vụ theo tháng hoặc theo năm để gia tăng số lượng hoặc ưu tiên hiển thị tin đăng. Quá trình mua gói được hỗ trợ bởi hệ thống thanh toán và ngay sau khi hoàn tất, gói dịch vụ sẽ được kích hoạt để Agent sử dụng.

Bên cạnh đó, nền tảng cũng cung cấp tính năng nhắn tin trực tiếp giữa Agent và khách hàng quan tâm đến bất động sản. Các cuộc trò chuyện được ghi nhận và đồng bộ theo thời gian thực, đảm bảo Agent có thể phản hồi khách hàng nhanh chóng và chuyên nghiệp.

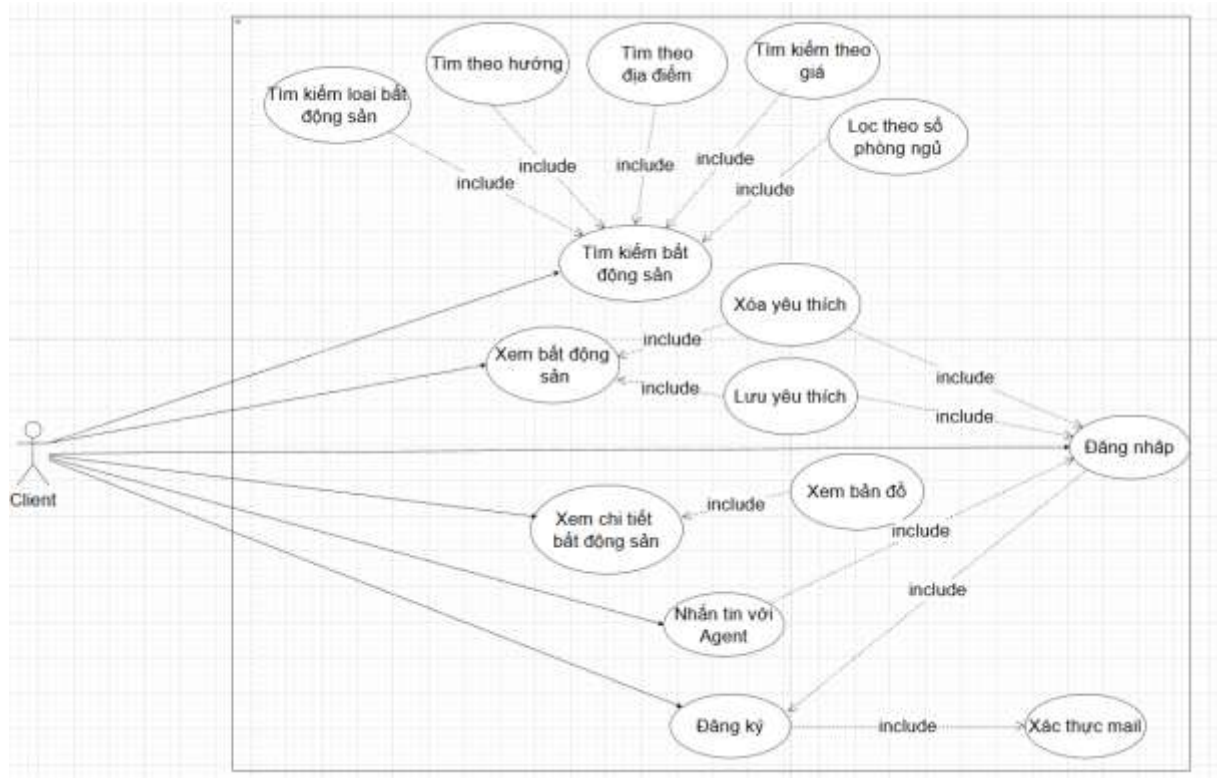
Ở góc độ truy cập hệ thống, Agent có thể đăng ký tài khoản mới, nhận mã OTP hoặc đường dẫn xác thực để kích hoạt tài khoản. Sau khi đăng nhập thành công, Agent được phép sử dụng toàn bộ các chức năng liên quan đến bất động sản, giao dịch và

trao đổi thông tin. Mọi thao tác của Agent—từ quản lý tin đăng, mua gói cho đến gửi tin nhắn—đều được hệ thống xử lý và ghi log chi tiết, đồng thời hiển thị kết quả để Agent nắm bắt.

Trong quá trình sử dụng, một số tình huống ngoại lệ có thể xảy ra như đăng nhập thất bại do sai thông tin, tài khoản chưa xác thực email, gói dịch vụ hết hạn không thể đăng tin mới, lỗi dữ liệu khi thao tác hoặc lỗi kết nối khi gửi tin nhắn. Ở mỗi trường hợp, hệ thống đều đưa ra thông báo phù hợp để Agent biết cách xử lý tiếp theo.

Tổng thể, Use Case dành cho Agent mô tả toàn bộ hành trình mà một nhân viên môi giới tương tác với hệ thống bất động sản: từ đăng ký – đăng nhập, quản lý nội dung, mua dịch vụ cho đến trao đổi trực tiếp với khách hàng. Hệ thống đảm bảo quá trình làm việc của Agent diễn ra liên tục, rõ ràng, minh bạch và đúng quy trình.

2.4.3. Biểu đồ Use Case cho khách hàng



Hình 2. 3 Biểu đồ Use Case khách hàng

Mô tả Use Case cho khách hàng

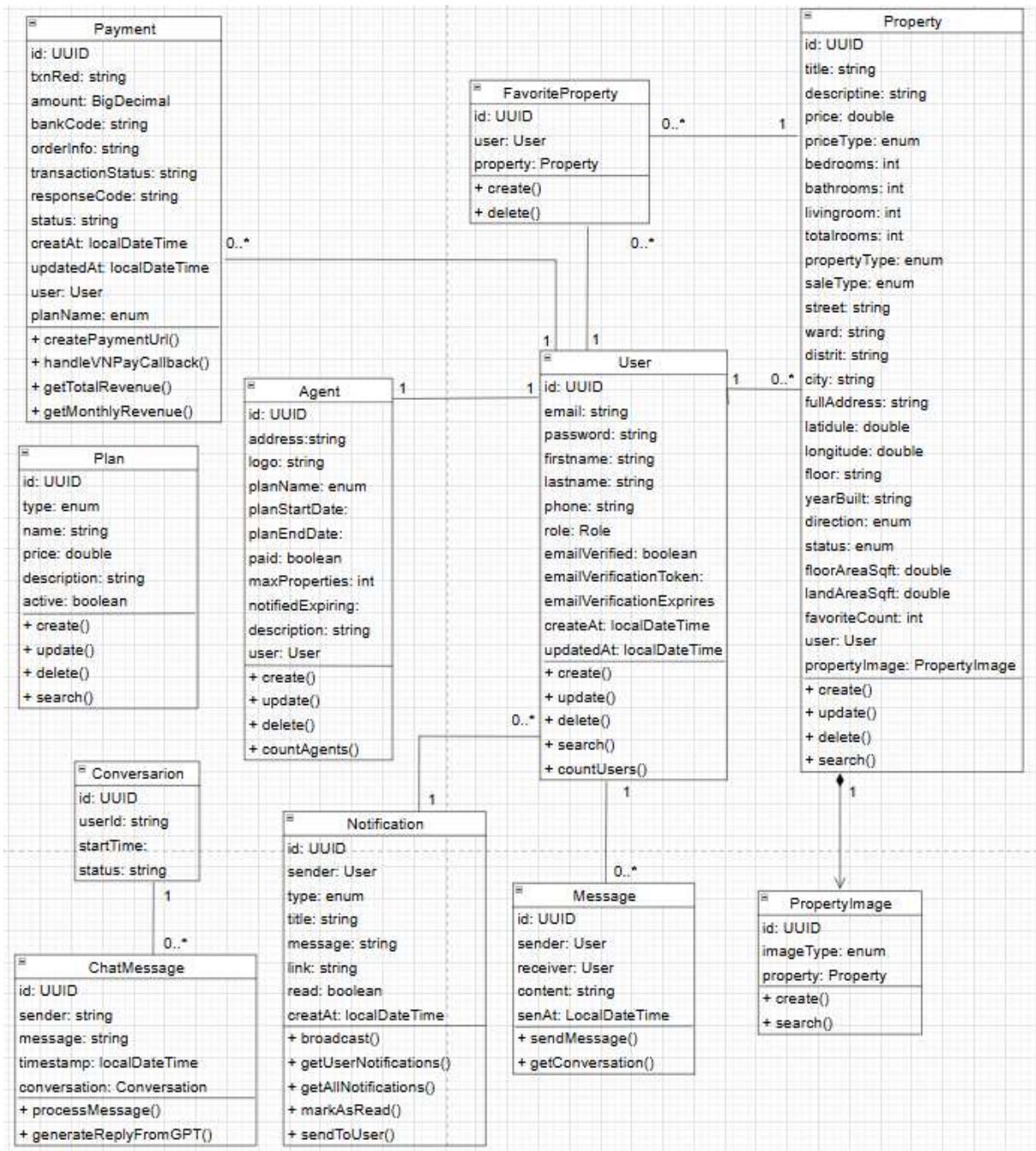
User là đối tượng truy cập hệ thống để tìm kiếm và xem thông tin bất động sản. Họ có thể áp dụng các bộ lọc theo giá, vị trí, loại bất động sản hay số phòng ngủ để tìm ra những lựa chọn phù hợp nhu cầu. Khi chọn một bất động sản, hệ thống hiển thị

đầy đủ thông tin chi tiết như mô tả, hình ảnh, giá bán – cho thuê, vị trí trên bản đồ cũng như thông tin của Agent phụ trách.

Trong quá trình sử dụng, User có thể thực hiện các thao tác tương tác như lưu tin yêu thích, xóa khỏi danh sách yêu thích hoặc nhấn tin trực tiếp với Agent. Những chức năng này yêu cầu User phải đăng nhập nên hệ thống sẽ tự động chuyển sang trang đăng nhập nếu chưa có phiên xác thực. Đồng thời, người dùng cũng có thể thực hiện đăng ký tài khoản mới và xác thực email để sử dụng đầy đủ các tính năng nâng cao.

Hệ thống đảm bảo dữ liệu luôn được cập nhật theo thao tác của User, ví dụ như khi lưu hoặc xóa bất động sản khỏi danh sách yêu thích. Các bước tìm kiếm, xem tin và xử lý tương tác đều được hệ thống phản hồi rõ ràng. Trong trường hợp phát sinh lỗi, chẳng hạn không tìm thấy kết quả phù hợp, tin đăng đã bị gỡ hoặc bản đồ không tải được, hệ thống sẽ hiển thị thông báo tương ứng để User dễ dàng nhận biết và tiếp tục thao tác.

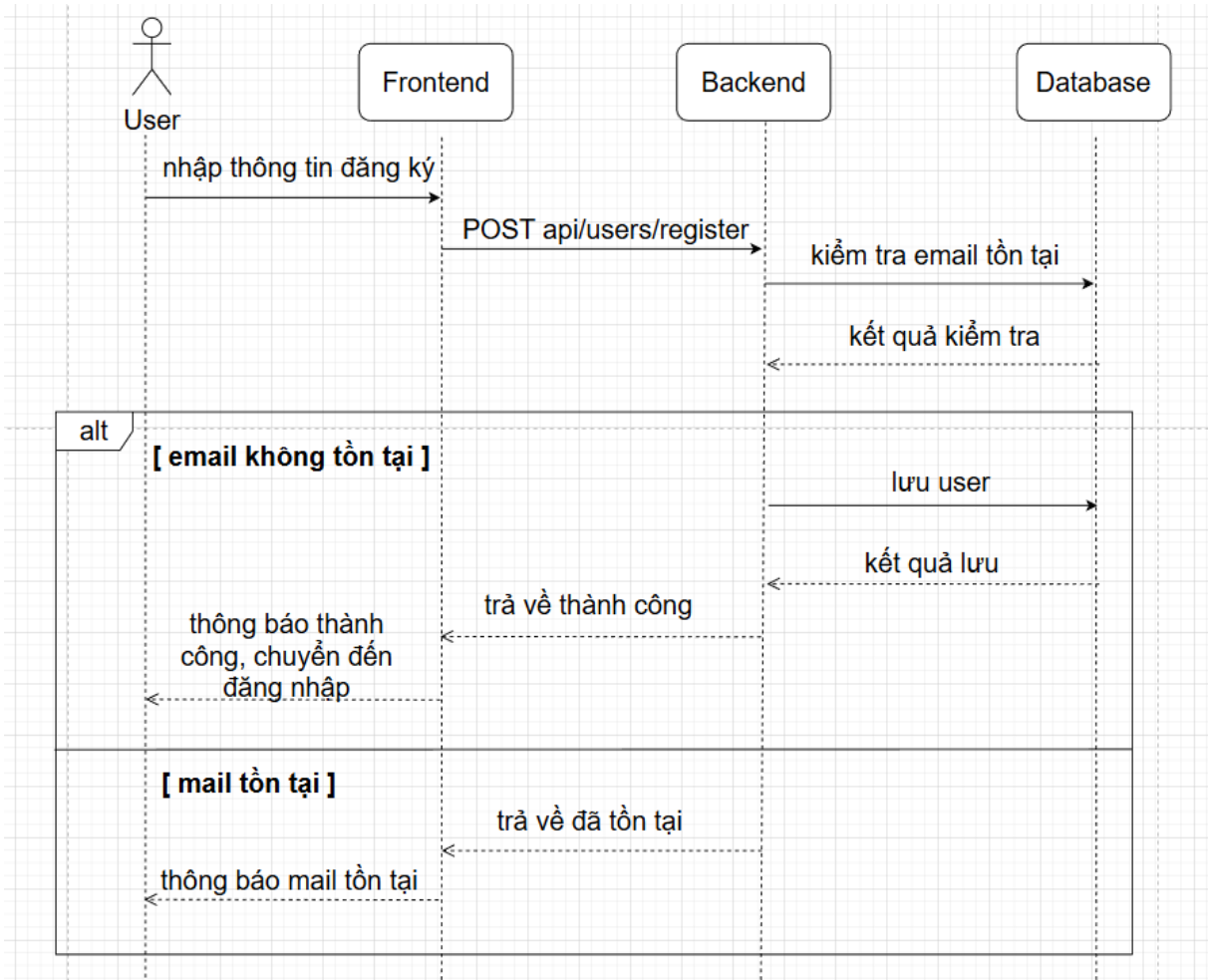
2.5. Sơ đồ lớp



Hình 2. 4 Sơ đồ lớp

2.6. Sơ đồ tuần tự chức năng

2.6.1. Mô tả tuần tự đăng kí



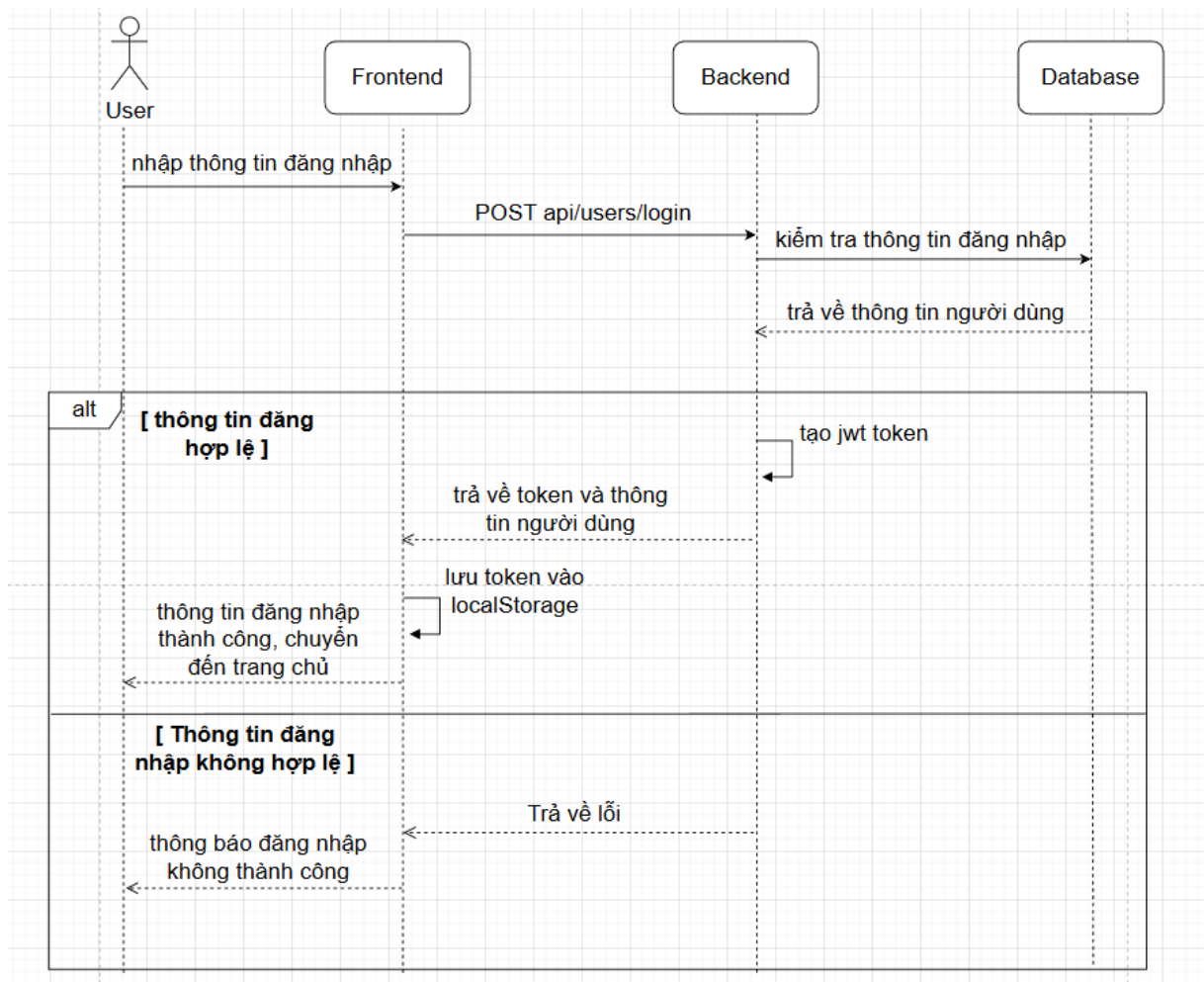
Hình 2. 5 Sơ đồ tuần tự đăng kí

Bảng 2. 1 Bảng agent đăng ký

Tên Use Case	Đăng ký
Actor	Admin, Môi giới (Agent), Khách hàng (User)
Mô tả	UC cho phép người dùng tạo tài khoản mới để có thể đăng nhập và sử dụng các chức năng của hệ thống.
Tiền điều kiện	Người dùng chưa có tài khoản trong hệ thống.
Hậu điều kiện	Người dùng đăng ký thành công và có thể đăng nhập vào hệ thống.
Luồng sự kiện chính	Người dùng chọn chức năng đăng ký 1. Form đăng ký hiển thị. 2. Người dùng nhập thông tin vào form.

	3. Nếu người dùng nhập đúng email thì hệ thống sẽ tạo tài khoản. 4. Nếu người dùng nhập sai email thì chuyển sang luồng nhánh A1. UC kết thúc.
Luồng sự kiện rẽ nhánh	Luồng nhánh A1: Đăng ký thất bại 1. Hệ thống yêu cầu nhập lại thông tin. 2. Nếu người dùng đồng ý thì quay về bước 2 của luồng sự kiện chính, nếu không thì quay về trang chủ và UC kết thúc.

2.6.2. Mô tả người dùng đăng nhập

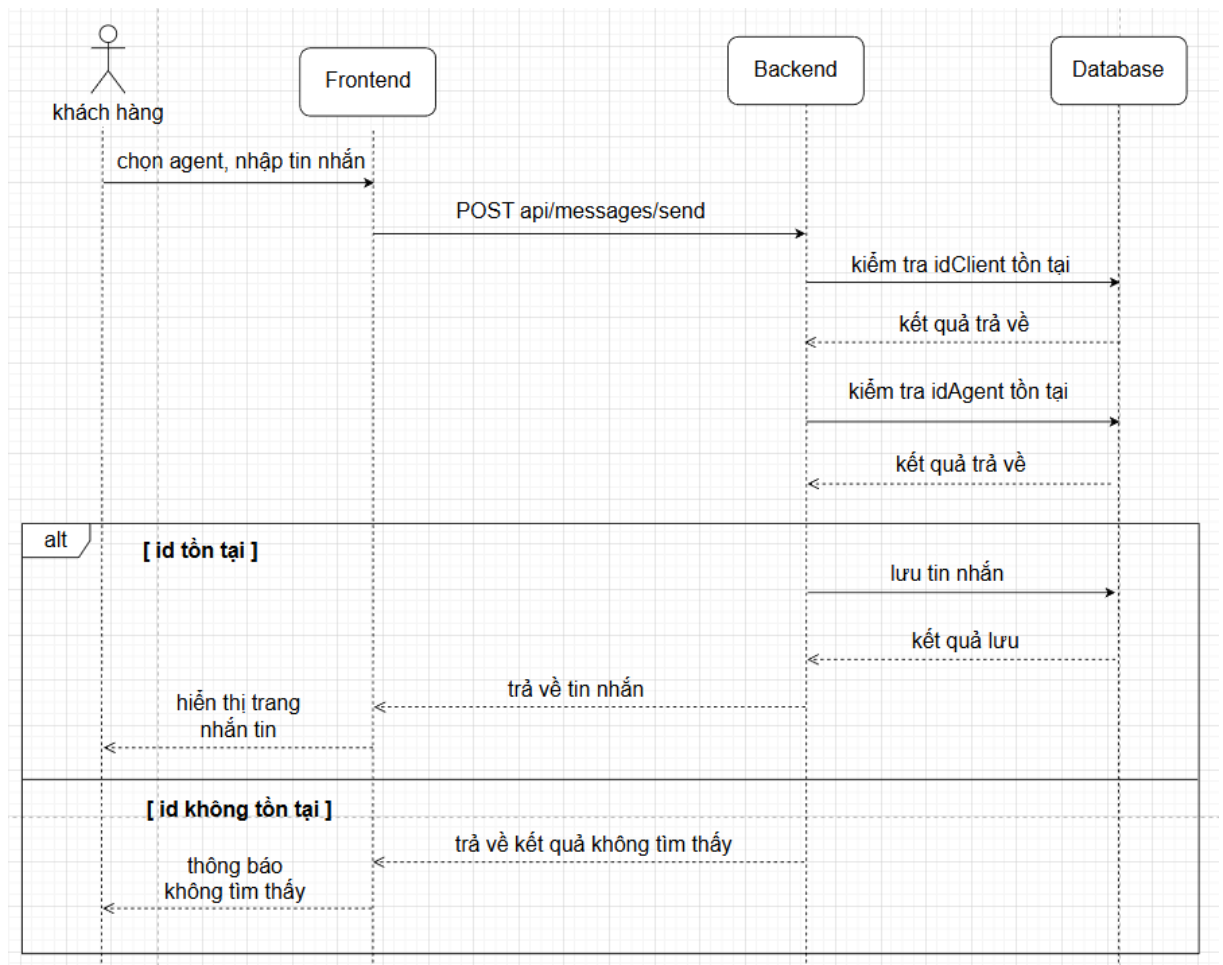


Hình 2. 6 Sơ đồ tuần tự người dùng đăng nhập

Bảng 2. 2 Bảng Mô tả người dùng đăng nhập

Tên Use Case	Đăng nhập
Actor	Admin, Môi giới (Agent), Khách hàng (User)
Mô tả	UC cho phép người dùng đăng nhập vào hệ thống để sử dụng các chức năng phù hợp với vai trò của mình.
Tiền điều kiện	Người dùng chưa đăng nhập vào hệ thống
Hậu điều kiện	Người dùng đăng nhập vào thành công và có thể sử dụng chức năng của hệ thống
Luồng sự kiện chính	Người dùng chọn chức năng đăng nhập. 1. Form đăng nhập hiển thị. 2. Người dùng chọn phương thức đăng nhập bằng email/password. 3. Người dùng nhập các thông tin cần thiết với từng phương thức đăng nhập. 4. Nếu việc đăng nhập thành công thì người dùng có thể sử dụng các chức năng của hệ thống. Nếu nhập sai thông tin thì chuyển sang luồng nhánh A1. 5. UC kết thúc.
Luồng sự kiện rẽ nhánh	Luồng nhánh A1: Người dùng đăng nhập không thành công 1. Hệ thống thông báo quá trình đăng nhập không thành công. 2. Hệ thống yêu cầu người dùng nhập lại thông tin theo từng phương thức đăng nhập. 3. Nếu người dùng đồng ý thì quay về bước 3 của luồng sự kiện chính, nếu không đồng ý thì UC kết thúc.

2.6.3. Mô tả tuần tự Agent đăng bài



Hình 2. 7 Sơ đồ tuần tự agent đăng bài

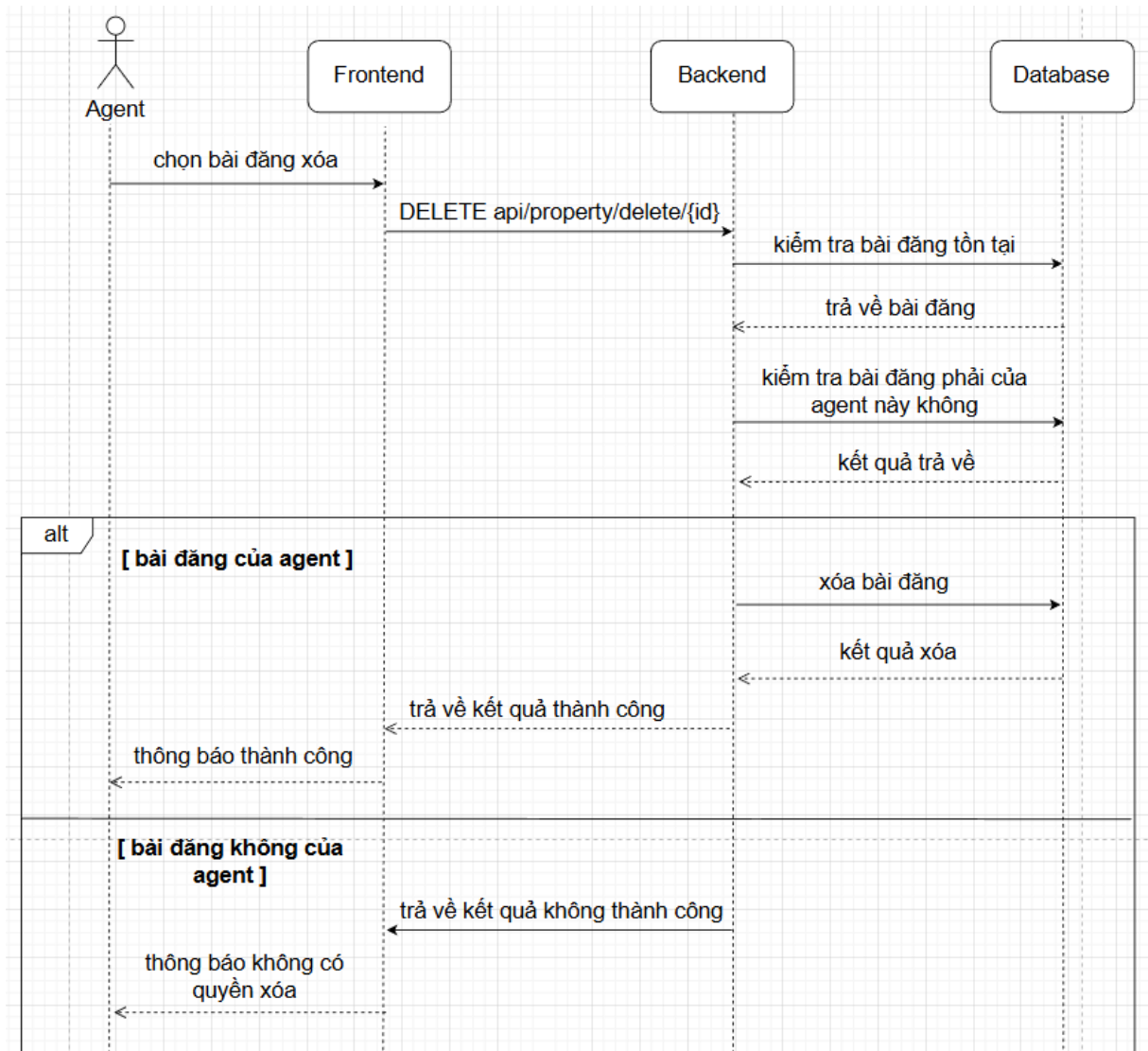
Bảng 2. 3 Bảng mô tả agent đăng bài

Tên Use Case	Đăng bài
Actor	Agent (người dùng nội bộ)
Mô tả	Use Case cho phép người dùng nhập nội dung chat và nhận phản hồi từ AI. Hệ thống lưu trữ toàn bộ nội dung hội thoại (tin nhắn người dùng và phản hồi của AI) để quản lý lịch sử trò chuyện.
Tiền điều kiện	1. Agent đã đăng nhập hệ thống. 2. Cloudinary API hoạt động bình

	thường. 3. File upload hợp lệ (đúng định dạng và dung lượng cho phép).
Hậu điều kiện	1. Ảnh được lưu thành công trên Cloudinary (nếu không có lỗi). 2. Log hệ thống ghi lại kết quả upload. 3. Giao diện hiển thị URL ảnh hoặc thông báo lỗi.
Luồng sự kiện chính	1. Agent chọn ảnh và bấm “Đăng”. 2. Frontend gửi file ảnh và tên folder đến Backend (POST /api/upload). 3. Backend đọc file thành dạng byte[[]]. 4. Backend gửi dữ liệu ảnh lên Cloudinary API. 5. Cloudinary xử lý và trả về kết quả upload. 6. Backend ghi log thành công và trả URL ảnh về giao diện.
Luồng sự kiện rẽ nhánh	1. Đăng bài thành công • Ảnh được upload lên Cloudinary (URL, publicId...). • Backend ghi log “đăng bài thành công”. • Hệ thống trả về thông báo thành công cho Agent. 1. Đăng bài thất bại (lỗi Cloudinary, lỗi hệ thống, lỗi IOException) • Upload ảnh thất bại hoặc backend ném ra ngoại lệ.

	• Backend ghi log lỗi. • Giao diện hiển thị: “Không thể đăng bài. Vui lòng thử lại.” 3. File không hợp lệ (client gửi file null/định dạng sai) • File null, định dạng sai hoặc vượt quá dung lượng. • Backend trả lỗi 400: “File không hợp lệ”. • Giao diện yêu cầu Agent chọn lại ảnh.
--	--

2.6.4. Mô tả tuần tự chức năng xóa bài đăng



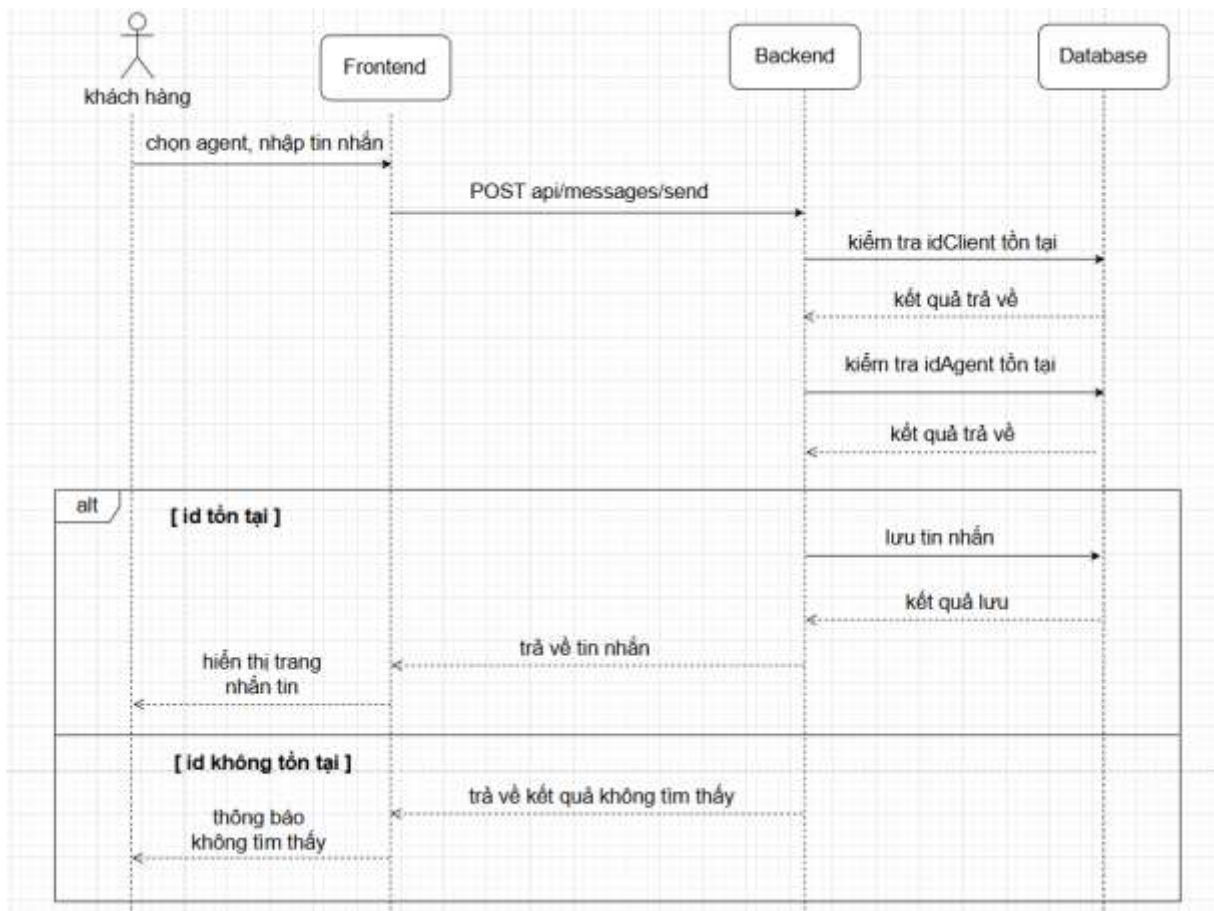
Hình 2. 8 Sơ đồ tuần tự xóa bài

Bảng 2. 4 Bảng mô tả agent xóa bài

Tên Use Case	Xóa bài
Actor	Agent (người dùng nội bộ)
Mô tả	Use Case cho phép Agent xóa một bài đăng (property) khỏi hệ thống thông qua giao diện quản lý. Hệ thống sẽ kiểm tra tính hợp lệ của ID bài đăng trước khi thực hiện xóa. Kết quả xóa sẽ được trả về cho giao diện và ghi log hệ thống.

Tiền điều kiện	1. Agent đã đăng nhập hệ thống. 2. Bài đăng (property) tồn tại trong hệ thống. 3. Agent có quyền xóa bài đăng. 4. Hệ thống và CSDL hoạt động bình thường.
Hậu điều kiện	1. Bài đăng được xóa khỏi hệ thống (nếu thành công). 2. Hệ thống ghi log kết quả xóa. 3. Giao diện hiển thị thông báo xóa thành công hoặc thất bại.
4. Luồng sự kiện chính	1. Agent chọn bài đăng và bấm “Xóa bài”. 2. Giao diện gửi ID bài đăng lên Backend. 3. Controller nhận ID và kiểm tra tính hợp lệ. 4. Backend gửi yêu cầu xóa property theo ID đến database. 5. Database thực hiện xóa và trả kết quả về cho controller. 6. Backend trả thông báo xóa thành công về giao diện. 7. Giao diện hiển thị thông báo “Xóa thành công”.
Luồng sự kiện rẽ nhánh	1. Xóa bài đăng thành công • Bài đăng được xóa khỏi database. • Backend ghi log: “Xóa bài thành công”. • Giao diện hiển thị thông báo thành công cho Agent. 2. Xóa bài đăng thất bại • ID không tồn tại hoặc lỗi hệ thống. • Backend ghi log lỗi. • Giao diện hiển thị thông báo: • “Xóa bài thất bại. Vui lòng thử lại.”

2.6.5. Mô tả tuần tự chức năng user chat với agent



Hình 2. 9 Sơ đồ tuần tự khách hàng chat với agent

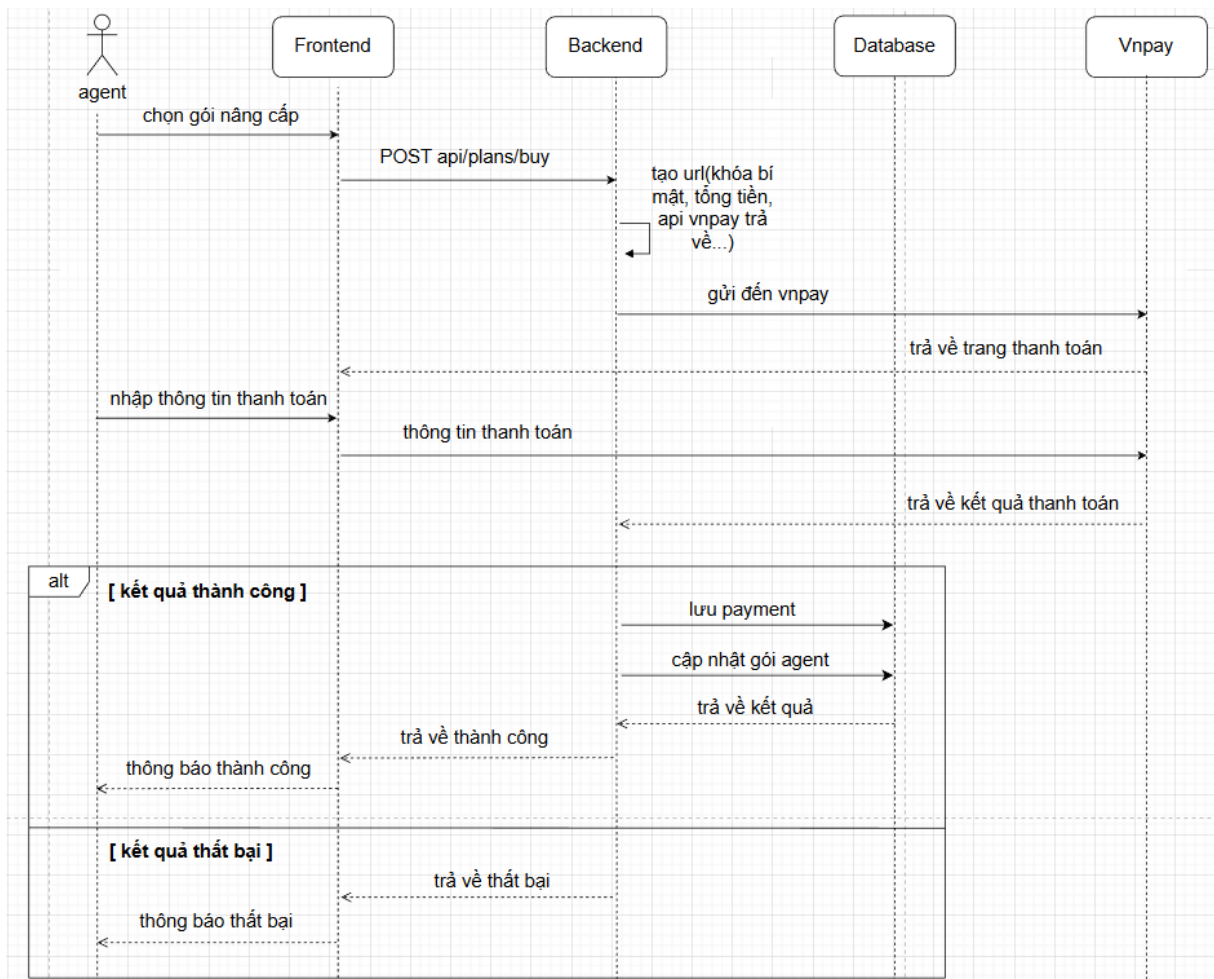
Bảng 2. 5 Bảng mô tả user chat với agent

Tên Use Case	Gửi và xem tin nhắn giữa User và Agent
Actor	User, Agent (người dùng nội bộ)
Mô tả	Use Case cho phép User và Agent gửi tin nhắn qua hệ thống. Backend nhận nội dung tin nhắn, kiểm tra ID người gửi/người nhận, lưu vào DB và trả kết quả. Ngoài ra, hệ thống cho phép User/Agent xem danh sách tin nhắn giữa hai bên.
Tiền điều kiện	1. User và Agent đã đăng nhập vào hệ thống

	2. Cả hai ID đều tồn tại trong cơ sở dữ liệu 3. Nội dung tin nhắn hợp lệ không rỗng
Hậu điều kiện	1. Tin nhắn được lưu vào DB (nếu gửi thành công). 2. Hệ thống ghi log lưu trữ. 3. Danh sách tin nhắn được trả về đúng theo cặp user-agent.
Luồng sự kiện chính	1. Gửi tin nhắn User/Agent nhập nội dung tin nhắn và bấm “Gửi”. Frontend gửi request chứa senderId, receiverId, content đến Backend. Backend kiểm tra tồn tại của senderId và receiverId trong DB. DB xác nhận và trả kết quả kiểm tra. Backend lưu tin nhắn vào DB. DB phản hồi “lưu thành công”. Backend trả về kết quả gửi tin thành công cho Frontend. 2. Xem danh sách tin nhắn User/Agent mở giao diện xem hội thoại. Frontend gửi request userId, agentId đến Backend. Backend truy vấn DB để lấy danh sách tin nhắn giữa 2 ID này. DB trả về danh sách tin nhắn. Backend trả dữ liệu về Frontend. Giao diện hiển thị danh sách tin nhắn.

Luồng sự kiện rẽ nhánh	1. Gửi tin nhắn thành công • DB lưu bản ghi message thành công. • Backend trả về trạng thái “success” cùng timestamp. 2. Gửi tin nhắn thất bại • ID người gửi hoặc người nhận không tồn tại → Backend trả lỗi 404 “User not found”. • Lỗi DB (không lưu được) → Backend trả lỗi 500 “Lỗi hệ thống”. 3. Nội dung tin nhắn không hợp lệ • Backend trả lỗi 400 “Nội dung tin nhắn không được để trống”. • Giao diện hiển thị thông báo yêu cầu nhập lại. 4. Không có tin nhắn nào • DB trả danh sách rỗng. • Giao diện hiển thị “Chưa có tin nhắn”.
-------------------------------	--

2.6.6. Mô tả tuần tự chức năng nâng cấp gói



Hình 2. 10 Sơ đồ tuần tự agent nâng cấp gói

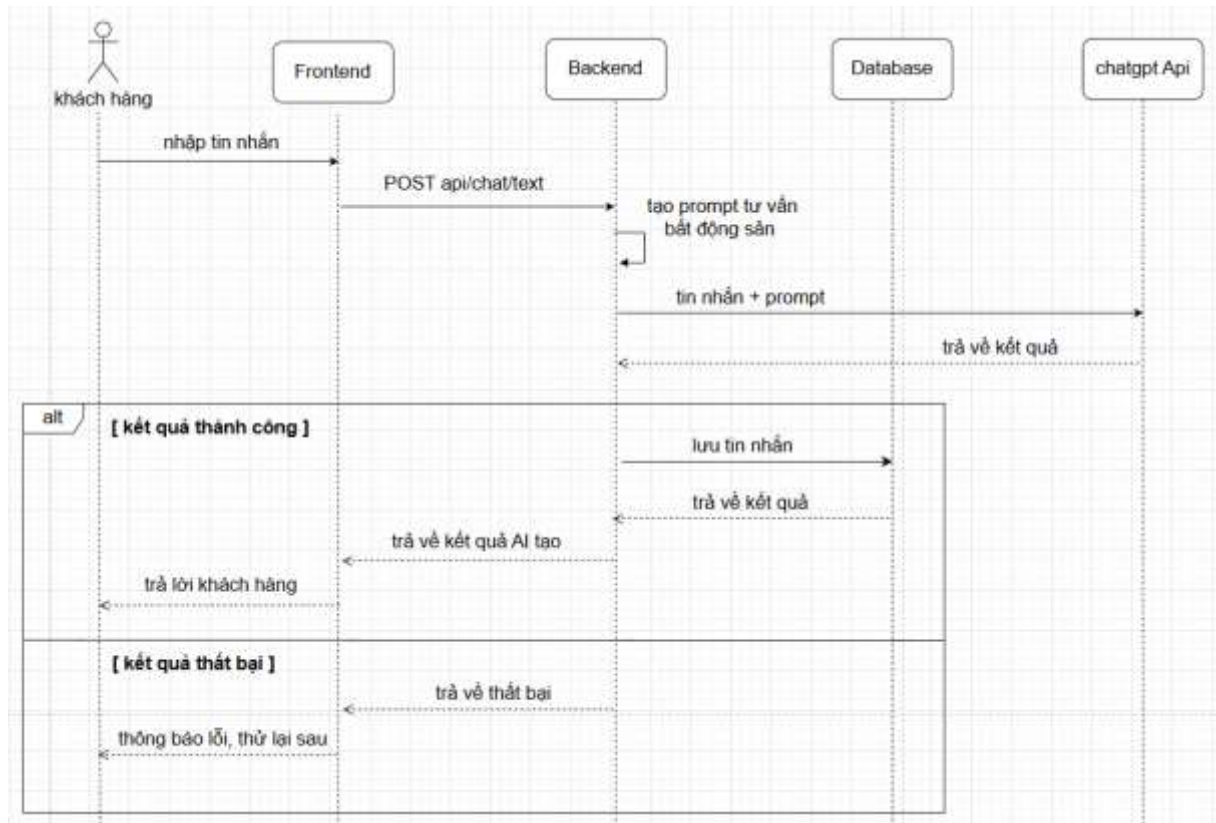
Bảng 2. 6 Bảng mô tả agent nâng cấp gói

Tên Use Case	Thanh toán nâng cấp gói dịch vụ qua VnPay
Actor	Agent (người dùng nội bộ muốn nâng cấp gói)
Mô tả	Use Case cho phép Agent chọn gói nâng cấp (planType), thực hiện thanh toán thông qua cổng VnPay. Backend tạo Payment, tạo URL thanh toán, nhận callback từ VnPay để xác nhận trạng thái giao dịch và cập nhật gói cho Agent nếu thanh toán thành công.
Tiền điều kiện	1. Agent đã đăng nhập hệ thống. 2. Các gói nâng cấp (planType) được cấu hình sẵn.

	3. Hệ thống thanh toán VnPay hoạt động bình thường. 4. DB hoạt động và có thể lưu thông tin Payment.
Hậu điều kiện	1. Nếu thanh toán thành công: • Gói của Agent được nâng cấp theo planType. • Payment được cập nhật trạng thái “Success”. 2. Nếu thanh toán thất bại: • Payment lưu trạng thái “Failed”. 3. Hệ thống ghi log quá trình thanh toán. 4. Giao diện thông báo kết quả cho Agent.
Luồng sự kiện chính	1. Agent chọn gói nâng cấp (planType) và bấm “Thanh toán”. 2. Frontend gửi request userId, planType, price đến Backend. 3. Backend lưu thông tin Payment ở trạng thái “Pending” vào DB. 4. Backend gọi API của VnPay để tạo URL thanh toán. 5. VnPay trả về PaymentUrl. 6. Backend trả PaymentUrl về cho Frontend. 7. Frontend redirect Agent sang trang thanh toán của VnPay. 8. Agent nhập thông tin thanh toán và xác nhận. 9. VnPay gửi callback về Backend để thông báo kết quả giao dịch. 10. Backend xác nhận thông tin thanh toán và lưu trạng thái mới vào DB. 11. Backend kiểm tra: thanh toán thành công hay thất bại.

Luồng sự kiện rẽ nhánh	1. Thanh toán thành công - Backend cập nhật trạng thái Payment = “Success”. - Backend cập nhật gói của Agent theo planType. - DB lưu lại thông tin cập nhật. - Backend trả kết quả “Nâng cấp thành công” về Frontend. - Frontend hiển thị thông báo thành công. 2. Thanh toán thất bại - VnPay trả lỗi hoặc Backend xác nhận giao dịch không hợp lệ. - Backend cập nhật Payment = “Failed”. - Backend trả về thông báo lỗi. - Frontend hiển thị “Thanh toán thất bại”. 3. Callback lỗi / Không nhận được callback - Backend không nhận được thông báo từ VnPay trong thời gian cho phép. - Payment giữ trạng thái “Pending”. - Frontend hiển thị “Không xác nhận được thanh toán” 4. Không thể tạo URL thanh toán - API VnPay trả lỗi khi tạo PaymentUrl. - Backend trả về lỗi “Không thể tạo giao dịch”. - Frontend yêu cầu Agent thử lại.
-------------------------------	--

2.6.7. Mô tả tuần tự chatbox tự động AI



Hình 2. 11 Sơ đồ tuần tự khách chat với chatbox AI

Bảng 2. 7 Bảng chatbox tự động AI

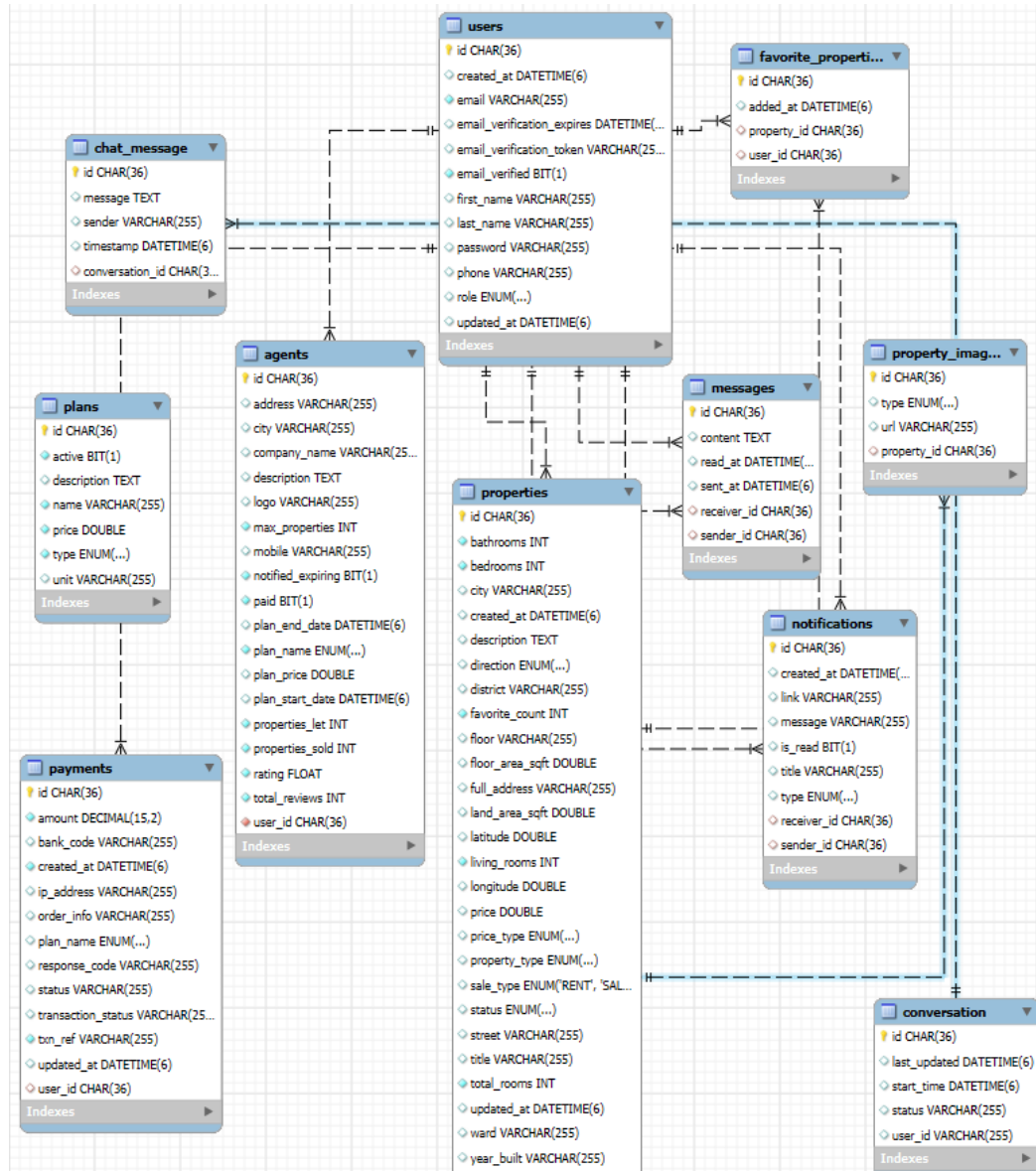
Tên Use Case	Chat với AI
Actor	Người dùng (User / Khách hàng)
Mô tả	Use Case cho phép người dùng nhập nội dung chat và nhận phản hồi từ AI. Hệ thống lưu trữ toàn bộ nội dung hội thoại (tin nhắn người dùng và phản hồi của AI) để quản lý lịch sử trò chuyện.
Tiền điều kiện	Người dùng đã đăng nhập vào hệ thống Dịch vụ AI (OpenAI API) hoạt động bình thường.
Hậu điều kiện	Tin nhắn người dùng và phản hồi AI được lưu vào cơ sở dữ liệu. Phản hồi AI hiển thị trên giao diện người

	dùng.
Luồng sự kiện chính	1. Người dùng nhập nội dung chat trên giao diện. 2. Frontend gửi yêu cầu POST /api/chat (userId, message) đến ChatController. 3. ChatService kiểm tra hoặc tạo hội thoại mới. 4. Hệ thống lưu tin nhắn người dùng. 5. ChatService gọi OpenAI API để tạo phản hồi AI. 6. Hệ thống lưu phản hồi của AI, cập nhật hội thoại và trả về kết quả cho giao diện.
Luồng sự kiện rẽ nhánh	1 Email đã tồn tại • Hệ thống phát hiện email đã có trong hệ thống. • Hiện thị: “Email đã được sử dụng, vui lòng chọn email khác.” • Người dùng có thể quay lại bước 3 để nhập lại. 2 Email không hợp lệ / sai định dạng • Hệ thống kiểm tra định dạng email sai. • Hiện thị: “Email không hợp lệ.” • Quay lại bước 3. 3 Thiếu hoặc sai thông tin bắt buộc • Ví dụ: mật khẩu quá ngắn, thiếu họ tên, thiếu CMND/CCCD (đối với Agent). • Hiện thị: “Vui lòng nhập đầy đủ và đúng thông tin.” • Quay lại bước 3. 4 Không gửi được email xác thực • Lỗi kết nối mail server.

	• Hệ thống tạo tài khoản nhưng đang ở trạng thái “Chưa xác thực”. • Hiện thị: “Không gửi được email xác thực, vui lòng thử lại sau.” 5 Lỗi hệ thống • Hệ thống gặp sự cố khi tạo tài khoản. • Hiện thị: “Đăng ký thất bại. Vui lòng thử lại sau.” • Người dùng có thể quay lại bước 1
--	--

2.7. Thiết kế cơ sở dữ liệu

Sơ đồ RD



Hình 2. 12 Sơ đồ RD

Bảng dữ liệu users

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã định danh người dùng
created_at	DATETIME	NOT NULL	Thời điểm tạo tài khoản
email	VARCHAR(255)	UNIQUE	Email đăng

			nhập
email_verified	BIT(1)	NOT NULL	Trạng thái xác thực email
email_verification_token	VARCHAR(255)	NULL	Mã xác thực email
password	VARCHAR(255)	NOT NULL	Mật khẩu đã hash
first_name	VARCHAR(255)	NULL	Tên
last_name	VARCHAR(255)	NULL	Họ
phone	VARCHAR(255)	NULL	Số điện thoại
role	ENUM('USER','AGENT','ADMIN')	NOT NULL	Phân quyền hệ thống
updated_at	DATETIME	NULL	Thời điểm cập nhật

Bảng dữ liệu agents

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã agent
address	VARCHAR(255)	NULL	Địa chỉ văn phòng
name	VARCHAR(255)	NOT NULL	Tên agent
description	TEXT	NULL	Giới thiệu agent
mobile	VARCHAR(255)	NULL	Số điện thoại
email	VARCHAR(255)	NULL	Email
logo	VARCHAR(255)	NULL	Logo
plan	VARCHAR(255)	NULL	Plan hiện tại
agent_plan_expire	DATETIME	NULL	Thời điểm hết hạn gói
agent_plan_data	DATETIME	NULL	Thời điểm kích hoạt gói
total_properties	INT	DEFAULT 0	Tổng số bất động sản
properties_left	INT	DEFAULT 0	Số bất động sản còn lại được đăng theo gói
plan_id	CHAR(36)	FK	Khóa ngoại tới bảng plans
user_id	CHAR(36)	FK	Liên kết tới bảng users

created_at	DATETIME	NOT NULL	Thời điểm tạo
updated_at	DATETIME	NULL	Thời điểm cập nhật

Bảng dữ liệu properties

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã bất động sản
created_at	DATETIME	NOT NULL	Thời gian đăng
bedrooms	INT	NULL	Phòng ngủ
bathrooms	INT	NULL	Phòng tắm
area	DOUBLE	NULL	Diện tích
address	VARCHAR(255)	NOT NULL	Địa chỉ
title	VARCHAR(255)	NOT NULL	Tiêu đề
description	TEXT	NULL	Mô tả
latitude	DOUBLE	NULL	Vĩ độ
longitude	DOUBLE	NULL	Kinh độ
price	DOUBLE	NULL	Giá
price_label	VARCHAR(255)	NULL	Label (USD/m2, theo tháng...)
property_type	ENUM(...)	NOT NULL	Loại bất động sản
status	ENUM('AVAILABLE','SOLD','RENTED')	NOT NULL	Trạng thái
floor	INT	NULL	Tầng
total_floors	INT	NULL	Tổng số tầng
rooms	INT	NULL	Tổng phòng
garage	INT	NULL	Số garage
furnished	ENUM('YES','NO','PARTIAL')	NULL	Nội thất

year_built	VARCHAR(255)	NULL	Năm xây dựng
agent_id	CHAR(36)	FK	Thuộc agent nào
user_id	CHAR(36)	FK	Người đăng
updated_at	DATETIME	NULL	Cập nhật gần nhất

Bảng dữ liệu property_images

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã ảnh
property_id	CHAR(36)	FK NOT NULL	Thuộc bất động sản nào
name	VARCHAR(255)	NOT NULL	Tên file ảnh
url	VARCHAR(255)	NOT NULL	URL ảnh
is_featured	BOOLEAN	DEFAULT FALSE	Ảnh đại diện
created_at	DATETIME	NOT NULL	Thời điểm thêm ảnh
updated_at	DATETIME	NULL	Cập nhật ảnh

Bảng dữ liệu favorite_properties

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã yêu thích
property_id	CHAR(36)	FK	Bất động sản được yêu thích
user_id	CHAR(36)	FK	Người yêu thích
added_at	DATETIME	NOT NULL	Thời điểm lưu

Bảng dữ liệu Favorites (Lưu yêu thích)

Trường	Kiểu	Required	Mô tả
id	UUID	Not null	Khóa chính
used_id	UUID (FK)	Not null	Người dùng
property_id	UUID (FK)	Not null	Thuộc chi nhánh
created_at	DATETIME	Not null	Bất động sản

Bảng dữ liệu plans

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã gói
title	VARCHAR(255)	NOT NULL	Tên gói
description	TEXT	NULL	Mô tả
price	DOUBLE	NOT NULL	Giá
plan_type	ENUM('MONTHLY','YEARLY')	NOT NULL	Chu kỳ
max_properties	INT	NOT NULL	Số lượng BĐS tối đa được đăng
created_at	DATETIME	NOT NULL	Tạo
updated_at	DATETIME	NULL	Cập nhật

Bảng dữ liệu payments

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã thanh toán
amount	DECIMAL(15,2)	NOT NULL	Số tiền
status	ENUM('PENDING','SUCCESS','FAILED')	NOT NULL	Trạng thái
pay_method	VARCHAR(255)	NOT NULL	Phương thức thanh toán

description	VARCHAR(255)	NULL	Ghi chú
user_id	CHAR(36)	FK	Người thanh toán
plan_id	CHAR(36)	FK	Gói liên quan
payment_data	VARCHAR(255)	NULL	Dữ liệu cổng thanh toán
created_at	DATETIME	NOT NULL	Thời điểm tạo
updated_at	DATETIME	NULL	Thời điểm cập nhật

Bảng dữ liệu messages

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã tin nhắn
content	TEXT	NOT NULL	Nội dung
created_at	DATETIME	NOT NULL	Thời điểm tạo
sender_id	CHAR(36)	FK	Người gửi
receiver_id	CHAR(36)	FK	Người nhận

Bảng dữ liệu notifications

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã thông báo
created_at	DATETIME	NOT NULL	Thời điểm tạo
title	VARCHAR(255)	NOT NULL	Tiêu đề
content	TEXT	NULL	Nội dung
is_read	BIT(1)	DEFAULT 0	Đã đọc hay chưa
sender_id	CHAR(36)	FK	Người gửi

receiver_id	CHAR(36)	FK	Người nhận
-------------	----------	----	------------

Bảng dữ liệu conversation

Trường	Kiểu	Ràng buộc	Mô tả
id	CHAR(36)	PK	Mã cuộc trò chuyện
created_at	DATETIME	NOT NULL	Thời gian tạo
user_id	CHAR(36)	FK	Người dùng
agent_id	CHAR(36)	FK	Agent
updated_at	DATETIME	NULL	Cập nhật

CHƯƠNG 3: XÂY DỰNG HỆ THỐNG VÀ PHÁT TRIỂN ỨNG DỤNG

3.1. Cấu trúc thư mục dự án

3.1.1. Cấu trúc thư mục Reactjs

```
-- src/
| |-- App.css           // CSS giao diện app
| |-- App.tsx           // Component gốc App
| |-- index.css         // CSS global
| |-- main.tsx          // Điểm mount React
| |-- vite-env.d.ts     // Khai báo type cho Vite
| |
| |-- assets/           // Tài nguyên tĩnh
| | |-- react.svg       // Logo React
| |
| |-- components/       // Component dùng lại
| | |-- ui/             // Bộ UI (shadcn)
| | | |-- avatar.tsx    // UI avatar
| | | |-- badge.tsx     // UI badge
| | | |-- button.tsx    // UI button
| | | |-- card.tsx      // UI card
| | | |-- command.tsx   // UI command palette
| | | |-- dialog.tsx    // UI dialog
| | | |-- dropdown-menu.tsx // UI dropdown menu
| | | |-- input.tsx     // UI input
| | | |-- label.tsx     // UI label
| | | |-- navigation-menu.tsx // UI menu điều hướng
| | | |-- popover.tsx   // UI popover
| | | |-- progress.tsx  // UI thanh tiến trình
| | | |-- scroll-area.tsx // UI vùng cuộn
| | | |-- select.tsx    // UI select
| | | |-- separator.tsx // UI separator
| | | |-- skeleton.tsx  // UI skeleton loading
```

```
| | |-- slider.tsx      // UI slider
| | |-- table.tsx      // UI table
| | |-- tabs.tsx       // UI tabs
| | |-- textarea.tsx   // UI textarea
| |
| |-- contexts/        // React context
| | |-- AuthContext.tsx // Context xác thực
| |
| |-- layouts/         // Layout khung trang
| | |-- AdminLayout.tsx // Layout admin
| | |-- AgentLayout.tsx // Layout agent
| | |-- UserLayout.tsx  // Layout user
| |
| |-- lib/             // Thư viện nội bộ
| | |-- utils.ts       // Hàm tiện ích chung
| |
| |-- pages/           // Các trang chính
| | |-- admin/         // Trang admin
| | | |-- Dashboard.tsx // Trang dashboard admin
| | | |-- LocationPage.tsx // Quản lý địa điểm
| | | |-- NotificationPage.tsx // Quản lý thông báo
| | | |-- PropertyPage.tsx // Quản lý bất động sản
| | | |-- UserPage.tsx   // Quản lý người dùng
| | |
| | |-- agent/         // Trang cho agent
| | | |-- AgentPlanPage.tsx // Trang gói agent
| | | |-- PropertyAgentPage.tsx // BĐS của agent
| | | |
| | | |-- chat/
| | | | |-- ChatAgentPage.tsx // Chat của agent
| | | |
| | |-- auth/         // Trang auth
| | | |-- LoginPage.tsx // Trang đăng nhập
```

```
| | | |-- RegisterAgentPage.tsx // Đăng ký agent
| | | |-- RegisterPage.tsx    // Đăng ký user
| | |
| | |-- chat/                // Trang chat chung
| | | |-- ChatPage.tsx       // Trang chat
| | |
| | |-- user/                // Trang phía user
| |   |-- ChatAiPage.tsx     // Trang chat AI
| |   |-- FavoritesPage.tsx  // Trang yêu thích
| |   |-- Home.tsx           // Trang chủ user
| |   |-- ProfilePage.tsx    // Trang hồ sơ
| |   |-- PropertyDetailPage.tsx // Chi tiết BĐS
| |   |-- SaveSearchDialogProps.tsx // Popup lưu tìm kiếm
| |   |-- SearchPage.tsx     // Trang tìm kiếm
| |
| | |-- component/
| |   |-- BottomNavigation.tsx // Thanh điều hướng dưới
| |
| |-- routes/                // Định nghĩa route
| | |-- AdminRoutes.tsx      // Route cho admin
| | |-- AgentRouter.tsx      // Route cho agent
| | |-- GuestRoute.tsx       // Route cho khách
| | |-- ProtectedRoute.tsx   // Route bảo vệ
| | |-- UserRoutes.tsx       // Route cho user
| |
| |-- services/              // Service gọi API
| | |-- auth.service.ts      // Service xác thực
| |
| |-- type/                  // Định nghĩa type
| |   |-- property.ts        // Kiểu dữ liệu Property
| |
| |-- utils/                  // Tiện ích frontend
| |   |-- jwt.ts              // Xử lý token JWT
```

| | -- useNotifications.ts // Hook thông báo

3.1.2. Cấu trúc thư mục Spring Boot

```
src/
├─ main/
│ └─ java/
│   └─ com/ngocminh/batdongsan_be/
│     └─ BatdongsanBeApplication.java // Điểm khởi động Spring Boot
│     |
│     └─ config/ // Cấu hình hệ thống
│       └─ CloudinaryConfig.java // Cấu hình Cloudinary (upload ảnh)
│       └─ CorsConfig.java // Cấu hình CORS cho API
│       └─ OpenAIConfig.java // Cấu hình OpenAI
│       └─ VNPayConfig.java // Cấu hình thanh toán VNPay
│       └─ WebSocketConfig.java // Cấu hình WebSocket realtime
│       |
│       └─ security/ // Bảo mật + JWT
│         └─ CustomUserDetailsService.java // Load user cho Spring Security
│         └─ Endpoints.java // Khai báo whitelist/secure endpoints
│         └─ JwtAuthenticationFilter.java // Filter kiểm tra JWT
│         └─ JwtService.java // Tạo/validate token JWT
│         └─ SecurityConfig.java // Cấu hình Security tổng
│         |
│         └─ controller/ // REST API controller
│           └─ AgentController.java // API môi giới (agent)
│           └─ AuthController.java // API đăng nhập/đăng ký
│           └─ ChatController.java // API chat realtime
│           └─ DashboardController.java // API dashboard/thống kê
│           └─ FavoritePropertyController.java // API BĐS yêu thích
│           └─ LocationController.java // API vị trí/khu vực
│           └─ MessageController.java // API tin nhắn
│           └─ NotificationController.java // API thông báo
│           └─ PaymentController.java // API thanh toán
│           └─ PlanController.java // API gói dịch vụ
```

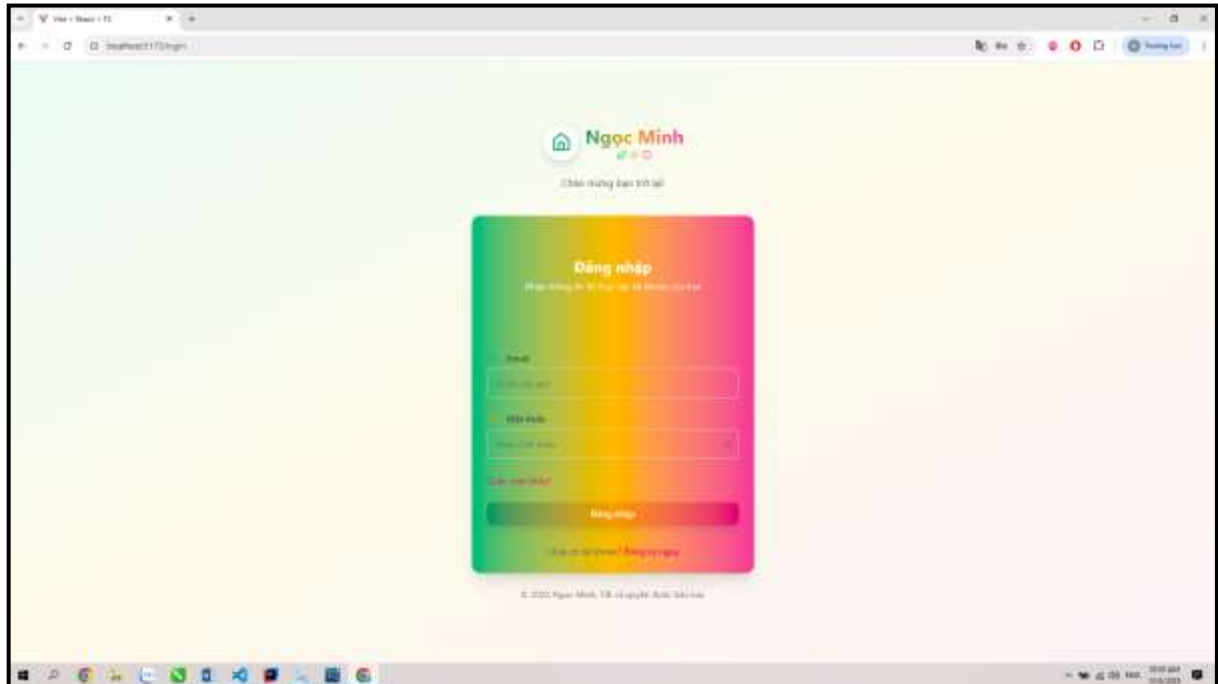
```
| | | └─ PropertyController.java      // API bất động sản
| | | └─ PropertyImageController.java // API ảnh bất động sản
| | | └─ UploadController.java        // API upload file
| | | └─ UserController.java          // API người dùng
| | |
| | └─ dto/                          // DTO trao đổi dữ liệu
| |   └─ FavoritePropertyDTO.java     // DTO BĐS yêu thích
| |   |
| |   └─ request/                    // Dữ liệu FE gửi lên
| |     └─ AgentRequest.java          // Tạo/cập nhật agent
| |     └─ AuthRequest.java           // Request đăng nhập
| |     └─ BroadcastRequest.java      // Gửi broadcast thông báo
| |     └─ ChatRequest.java           // Gửi tin nhắn chat
| |     └─ PropertyCreateRequest.java // Tạo bất động sản mới
| |     └─ SendMessageRequest.java    // Gửi message thường
| |     └─ UploadRequest.java         // Thông tin upload file
| |     └─ UserFilterRequest.java     // Lọc người dùng
| |     └─ UserRequest.java           // Tạo/cập nhật người dùng
| |     |
| |     └─ response/                 // Dữ liệu trả về FE
| |       └─ AgentResponse.java       // Thông tin agent
| |       └─ AuthResponse.java        // Thông tin đăng nhập/token
| |       └─ ChatPartnerResponse.java // Thông tin đối tác chat
| |       └─ ChatResponse.java        // Nội dung cuộc chat
| |       └─ MessageResponse.java     // Tin nhắn
| |       └─ NotificationResponse.java // Thông báo
| |       └─ PropertyDetailResponse.java // Chi tiết BĐS
| |       └─ PropertyListImgResponse.java // Danh sách ảnh BĐS
| |       └─ PropertyResponse.java    // BĐS rút gọn/list
| |       └─ UserDetailResponse.java  // Thông tin chi tiết user
| |       └─ UserResponse.java        // User rút gọn/list
| |       |
| |       └─ mapper/                 // MapStruct mapper DTO ↔ Entity
```

```
| | | └─ FavoritePropertyMapper.java    // Map BĐS yêu thích
| | | └─ NotificationMapper.java        // Map thông báo
| | | └─ UserDetailMapper.java          // Map chi tiết user
| | | └─ UserMapper.java                // Map user
| | |
| | └─ model/                          // Entity ánh xạ bảng DB
| | | └─ Agent.java                    // Entity môi giới
| | | └─ Appointment.java              // Lịch hẹn xem nhà
| | | └─ ChatMessage.java              // Tin nhắn chat
| | | └─ Conversation.java             // Cuộc hội thoại
| | | └─ FavoriteProperty.java         // BĐS yêu thích
| | | └─ Location.java                 // Vị trí/khu vực
| | | └─ Message.java                  // Tin nhắn (thường)
| | | └─ Notification.java             // Thông báo
| | | └─ Payment.java                  // Thanh toán
| | | └─ Plan.java                     // Gói dịch vụ
| | | └─ Property.java                 // Bất động sản
| | | └─ PropertyImage.java            // Ảnh bất động sản
| | | └─ Review.java                   // Đánh giá BĐS/agent
| | | └─ SavedSearch.java              // Tìm kiếm đã lưu
| | | └─ TransportLink.java            // Liên kết giao thông
| | | └─ User.java                     // Người dùng hệ thống
| | |
| | └─ repository/                     // Tầng truy vấn DB (JPA)
| | | └─ AgentRepository.java          // Repo agent
| | | └─ ChatMessageRepository.java    // Repo tin nhắn chat
| | | └─ ConversationRepository.java   // Repo cuộc hội thoại
| | | └─ FavoritePropertyRepository.java // Repo BĐS yêu thích
| | | └─ LocationRepository.java       // Repo vị trí
| | | └─ MessageRepository.java        // Repo message
| | | └─ NotificationRepository.java   // Repo thông báo
| | | └─ PaymentRepository.java        // Repo thanh toán
| | | └─ PlanRepository.java           // Repo gói dịch vụ
```

```
| | | └─ PropertyImageRepository.java // Repo ảnh BĐS
| | | └─ PropertyRepository.java // Repo bất động sản
| | | └─ UserRepository.java // Repo người dùng
| | |
| | └─ security/ // (dự phòng/nơi thêm class security khác)
| |
| | └─ service/ // Tầng service nghiệp vụ
| | └─ AgentPlanScheduler.java // Scheduler kiểm tra hạn gói agent
| | └─ AgentService.java // Xử lý logic môi giới
| | └─ AuthService.java // Xử lý đăng nhập/đăng ký
| | └─ ChatService.java // Xử lý chat
| | └─ CloudinaryService.java // Xử lý upload ảnh Cloudinary
| | └─ FavoritePropertyService.java // Xử lý BĐS yêu thích
| | └─ LocationService.java // Xử lý vị trí/khu vực
| | └─ MailService.java // Gửi email
| | └─ MessageService.java // Xử lý message
| | └─ NotificationService.java // Gửi/xử lý thông báo
| | └─ PaymentService.java // Xử lý thanh toán
| | └─ PlanService.java // Xử lý gói dịch vụ
| | └─ PropertyImageService.java // Xử lý ảnh BĐS
| | └─ PropertyService.java // Xử lý nghiệp vụ BĐS
| | └─ UserService.java // Xử lý người dùng
| |
| └─ resources/
| └─ application.yml // Cấu hình ứng dụng (DB, port,...)
|
└─ test/
    └─ java/com/ngocminh/batdongsan_be/
        └─ BatdongsanBeApplicationTests.java // Test khởi động context Spring
```

3.2. Giao diện website

- Giao diện đăng nhập, đăng kí



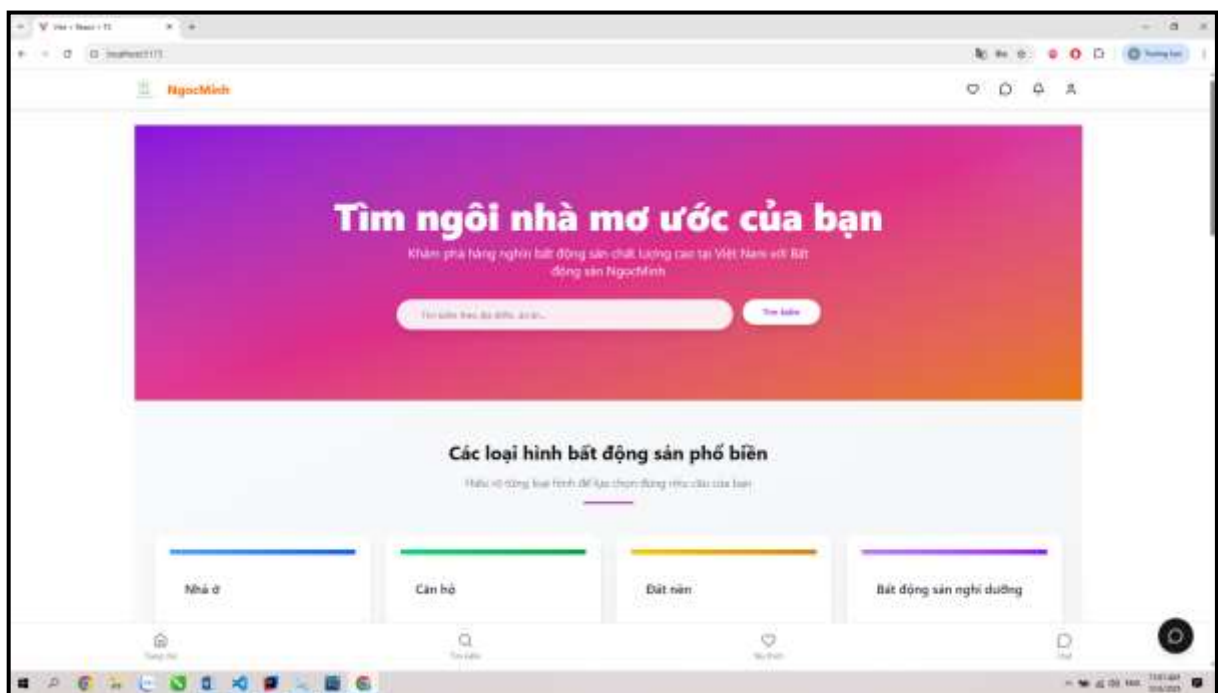
Hình 3. 1 Giao diện đăng nhập

Màn hình đăng nhập cho phép Admin, Môi giới và Khách hàng nhập email và mật khẩu để truy cập hệ thống. Tất cả người dùng đều dùng chung giao diện đăng nhập, hệ thống tự phân quyền sau khi xác thực.



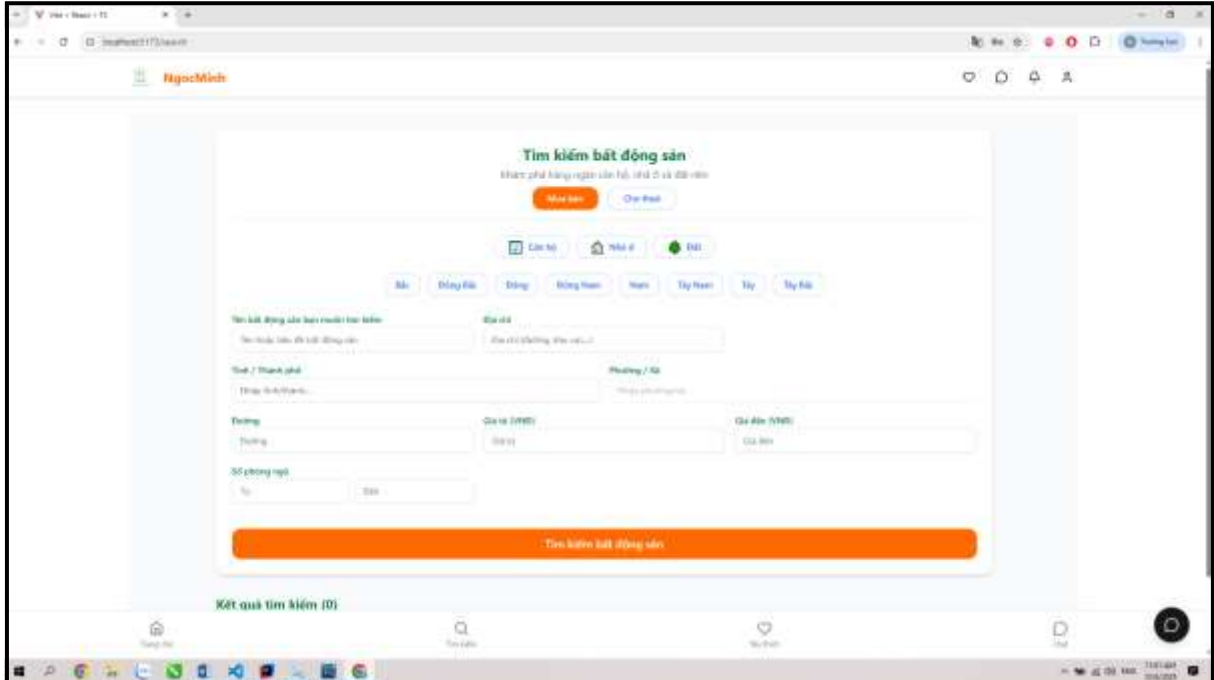
Hình 3. 2 Giao diện đăng kí

Màn hình đăng ký cho phép Admin, Môi giới và Khách hàng tạo tài khoản mới bằng cách nhập thông tin cơ bản như email, mật khẩu và xác nhận mật khẩu. Hệ thống tạo tài khoản và gán đúng vai trò sau khi đăng ký thành công.



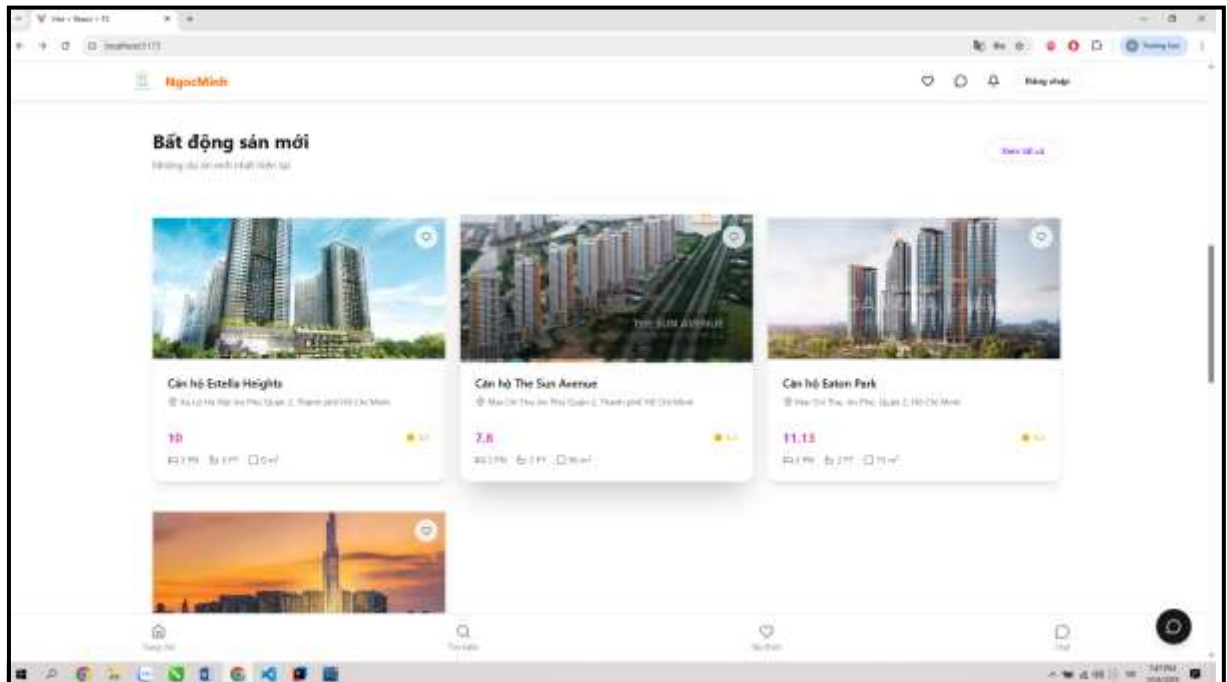
Hình 3. 3 Trang chủ

Trang chủ hiển thị tổng quan hệ thống và các chức năng chính dành cho Admin, Môi giới và Khách hàng. Giao diện tự tùy biến theo vai trò: Admin quản trị, Môi giới quản lý bài đăng, Khách hàng xem và tìm kiếm bất động sản.



Hình 3. 4 Giao diện tìm kiếm

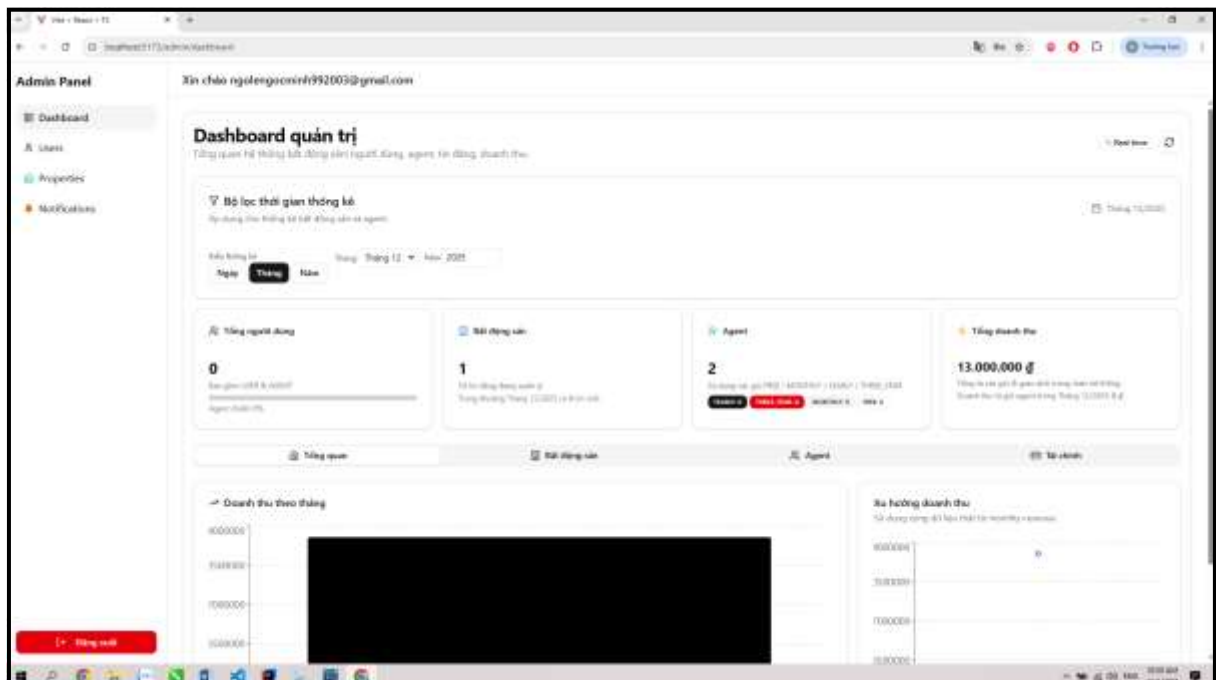
Chức năng tìm kiếm cho phép Khách hàng, Môi giới và Admin nhập từ khóa hoặc chọn bộ lọc để tìm bất động sản theo nhu cầu. Hệ thống trả về danh sách kết quả phù hợp, hiển thị thông tin cơ bản và cho phép xem chi tiết.



Hình 3. 5 Giao diện bất động sản mới

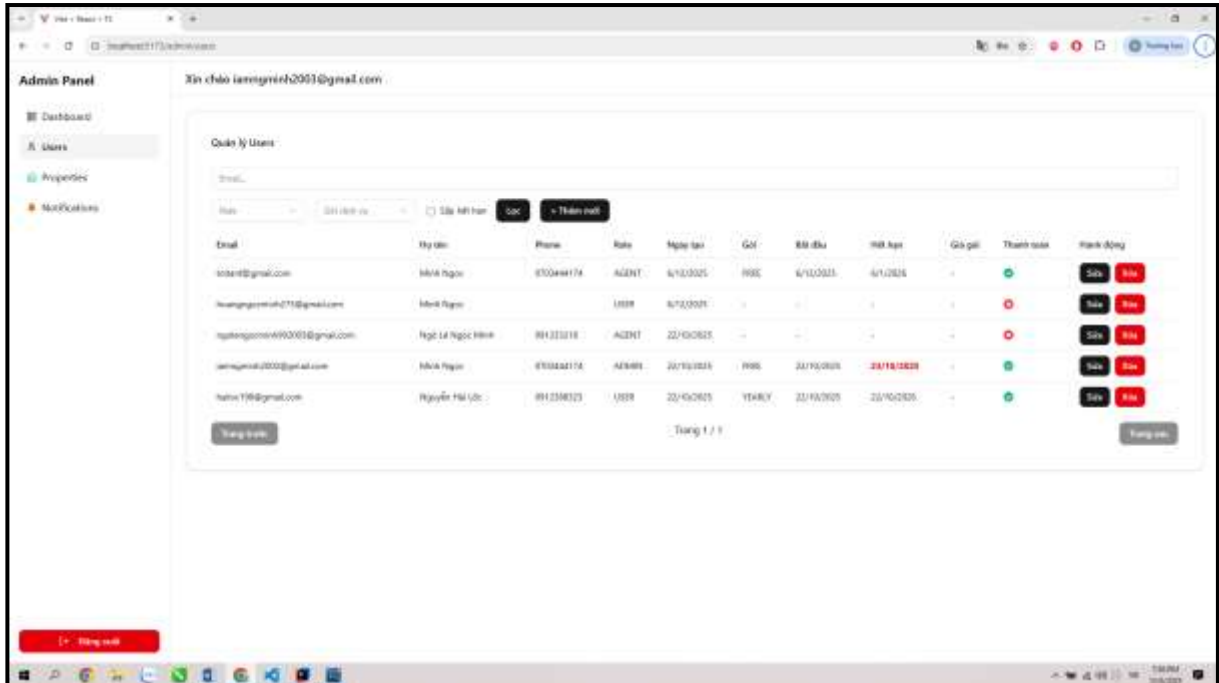
Giao diện bất động sản mới cho phép Môi giới và Admin tạo bài đăng mới với đầy đủ thông tin như tiêu đề, mô tả, giá, hình ảnh và vị trí. Hệ thống hiển thị form nhập liệu trực quan, hỗ trợ upload hình và lưu bài đăng sau khi hoàn tất.

➤ Giao diện phía admin



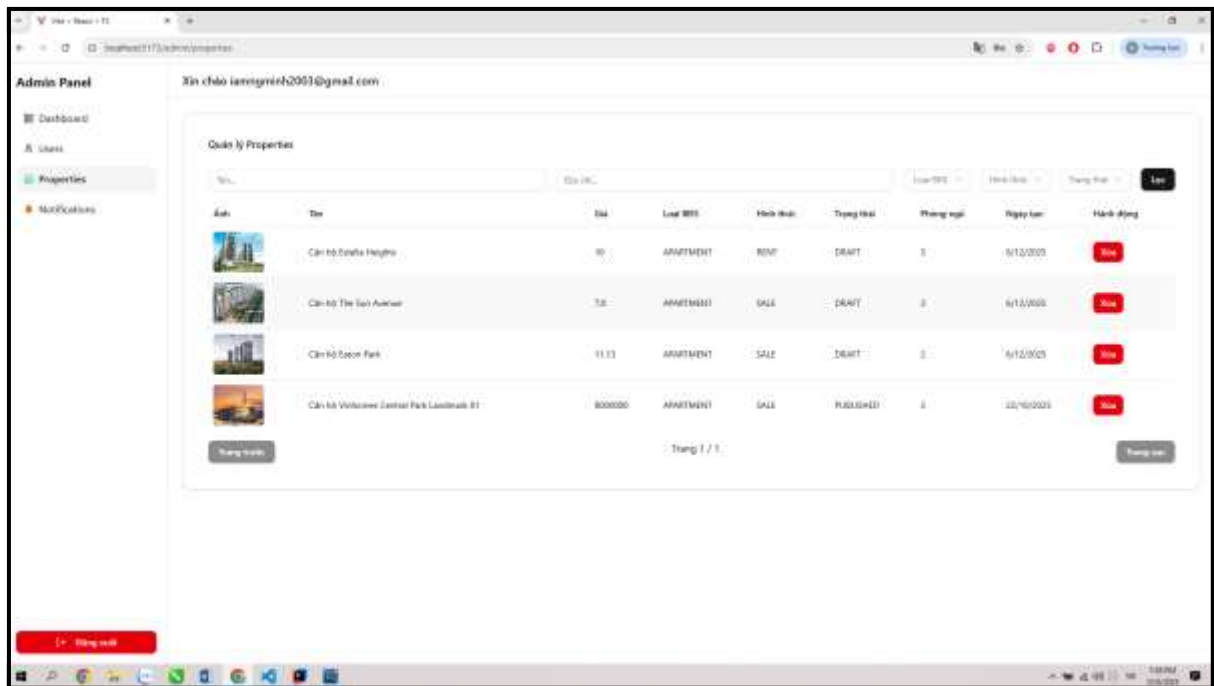
Hình 3. 6 Giao diện dashboard

Giao diện Dashboard hiển thị tổng quan hệ thống với các số liệu chính, biểu đồ và trạng thái hoạt động cho Admin, Môi giới và Khách hàng. Hệ thống tự tùy biến nội dung theo vai trò: Admin xem quản trị, Môi giới xem bài đăng – giao dịch, Khách hàng xem lịch sử và thông tin cá nhân.



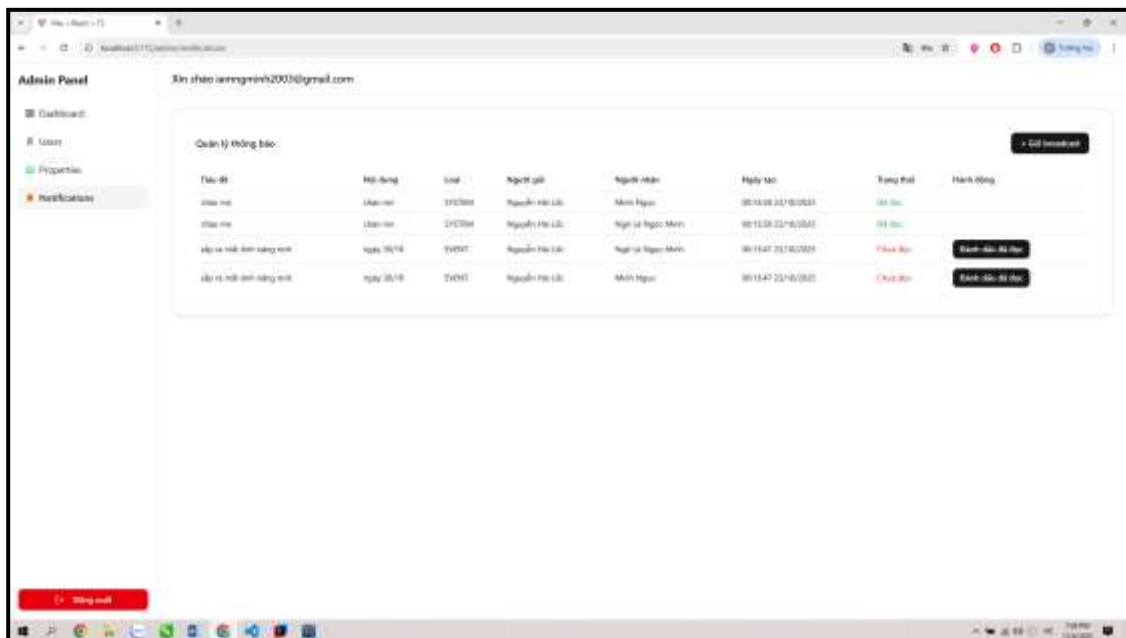
Hình 3. 7 Giao diện quản lý user

Giao diện quản lý tài khoản User cho phép Admin xem danh sách người dùng, bao gồm thông tin cơ bản và trạng thái hoạt động. Admin có thể chỉnh sửa quyền, khóa/mở tài khoản và tìm kiếm nhanh theo tên, email hoặc vai trò.



Hình 3. 8 Giao diện quản lý bất động sản

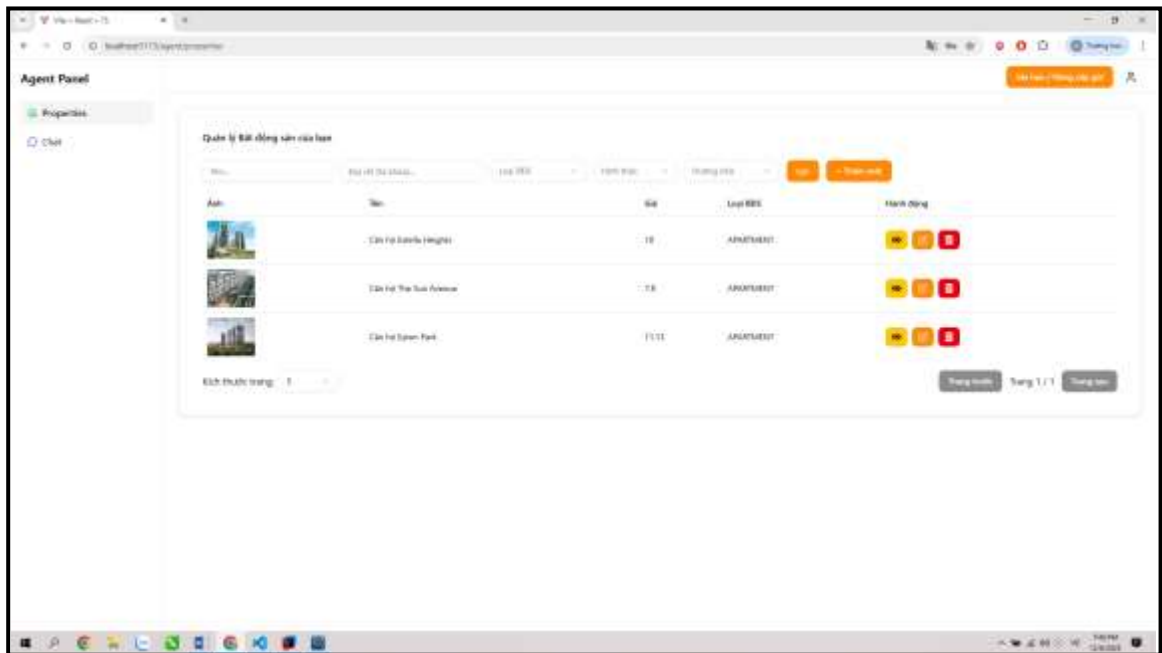
Giao diện quản lý bất động sản hiển thị danh sách toàn bộ bài đăng với thông tin cơ bản và trạng thái. Người dùng có quyền phù hợp (Admin/Môi giới) có thể xem, chỉnh sửa, cập nhật, lọc và xóa bài đăng trực tiếp trên giao diện.



Hình 3. 9 Giao diện gửi thông báo

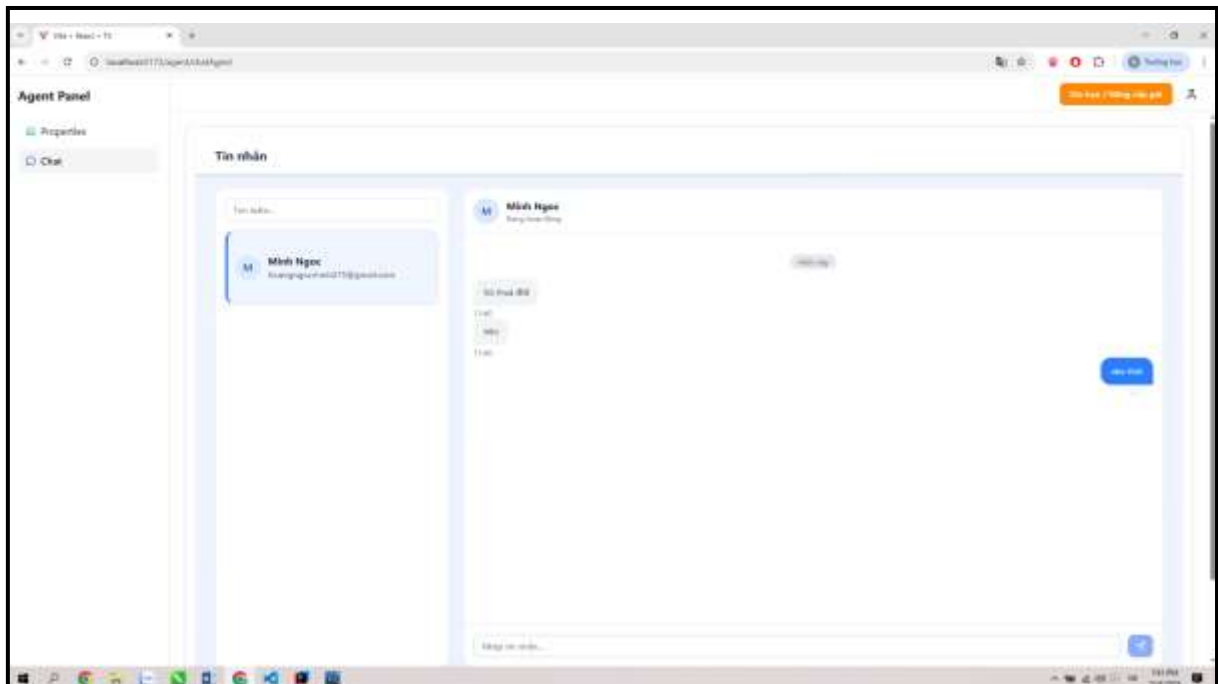
Giao diện gửi thông báo cho phép Admin tạo và gửi thông báo đến Môi giới và Khách hàng theo từng nhóm hoặc từng người. Hệ thống cung cấp form nhập nội dung, chọn đối tượng nhận và quản lý lịch sử các thông báo đã gửi.

➤ Giao diện phía agent



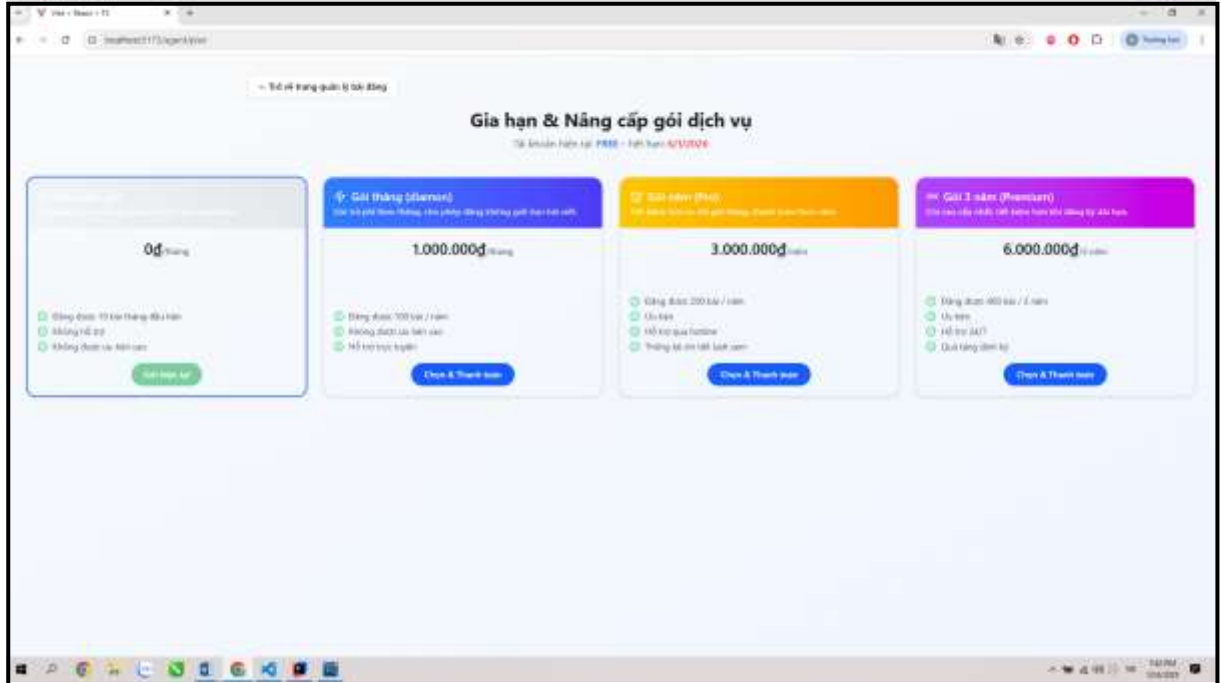
Hình 3. 10 Giao diện quản lý bất động sản

Giao diện quản lý bất động sản cho phép Agent xem, tạo, chỉnh sửa và cập nhật các bài đăng do chính mình quản lý. Hệ thống hiển thị danh sách bài đăng kèm trạng thái, công cụ lọc và thao tác nhanh để quản lý hiệu quả.



Hình 3. 11 Giao diện agent chat với user

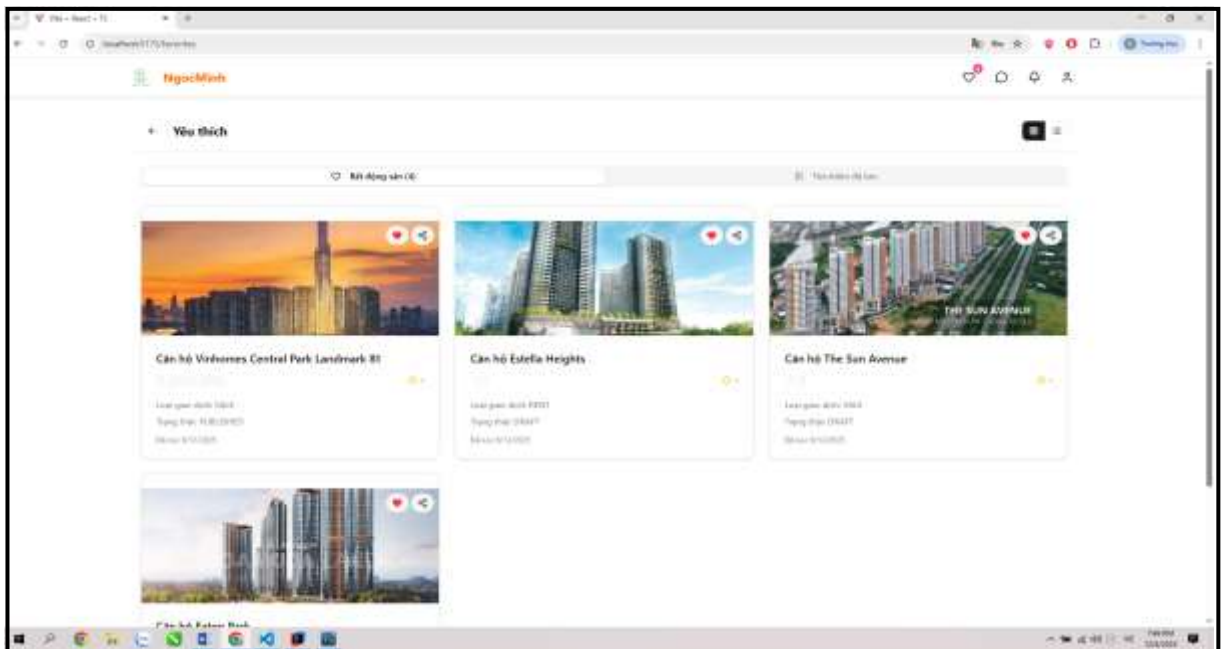
Giao diện chat cho phép Agent trao đổi trực tiếp với User về nhu cầu và thông tin bất động sản. Hệ thống hiển thị danh sách hội thoại, nội dung tin nhắn theo thời gian thực và hỗ trợ gửi văn bản, hình ảnh.



Hình 3. 12 Giao diện nâng cấp gói

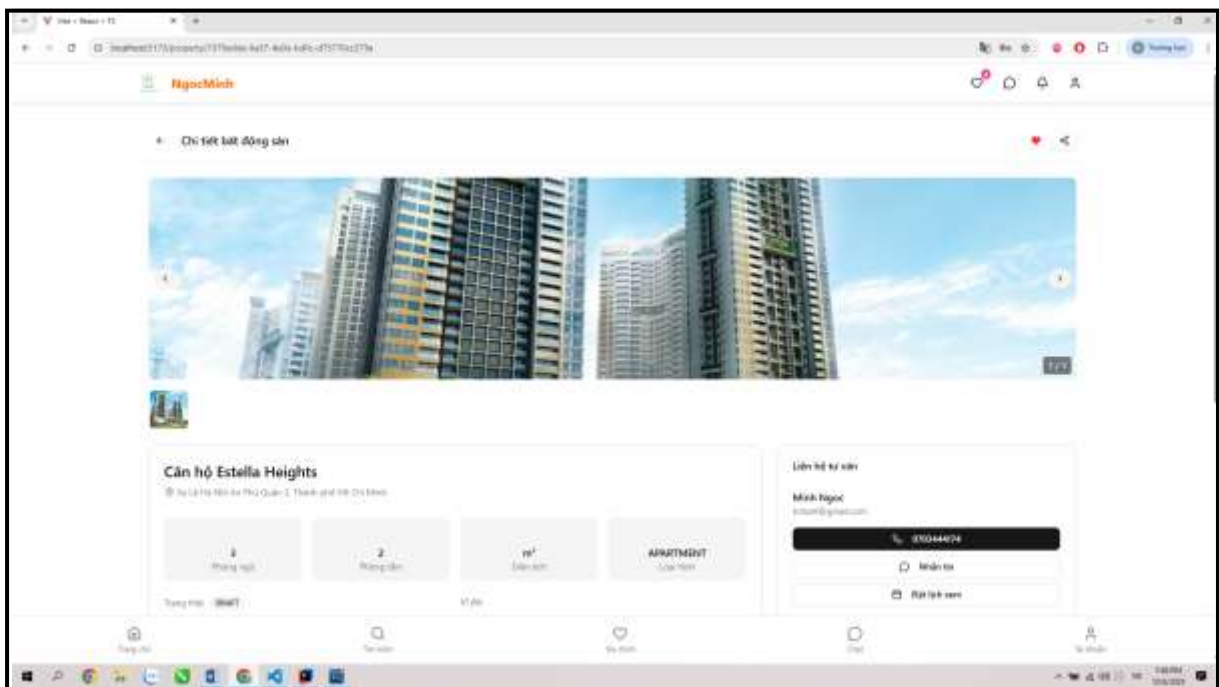
Giao diện nâng cấp gói hiển thị các gói dịch vụ dành cho Agent, bao gồm thông tin quyền lợi, thời hạn và chi phí. Agent có thể chọn gói phù hợp và tiến hành thanh toán để nâng cấp hoặc gia hạn tài khoản.

➤ Giao diện phía user



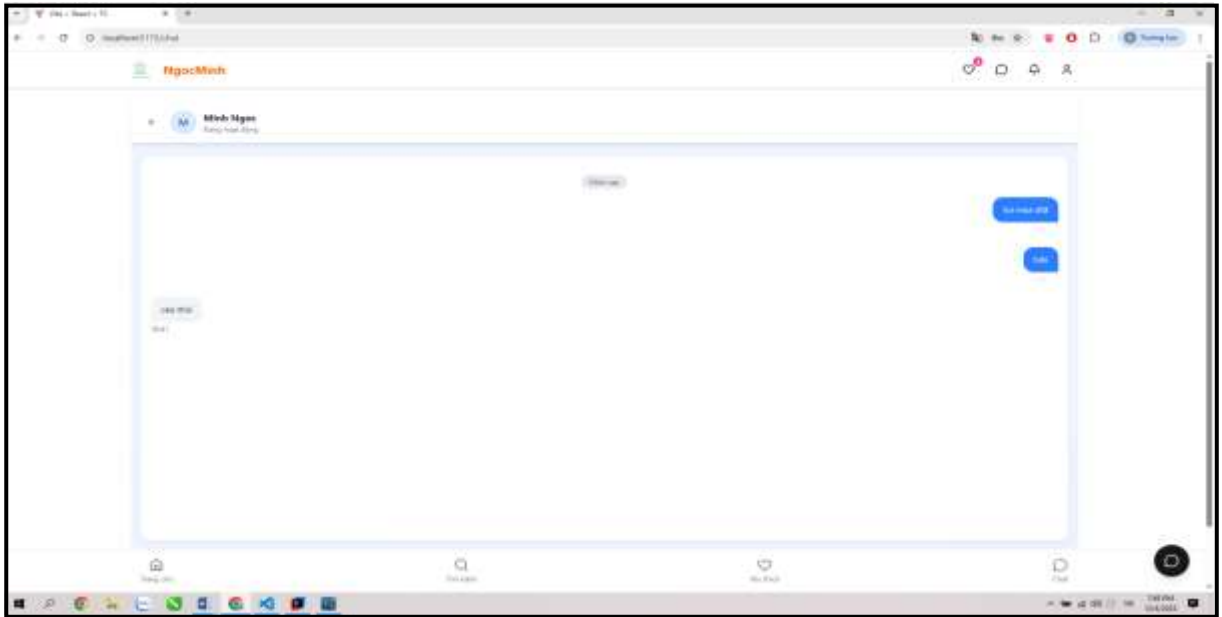
Hình 3. 13 Trang yêu thích

Trang yêu thích cho phép User lưu và xem lại các bất động sản đã đánh dấu ♥ trước đó. Hệ thống hiển thị danh sách bất động sản yêu thích kèm thông tin cơ bản và cho phép mở xem chi tiết bất cứ lúc nào.



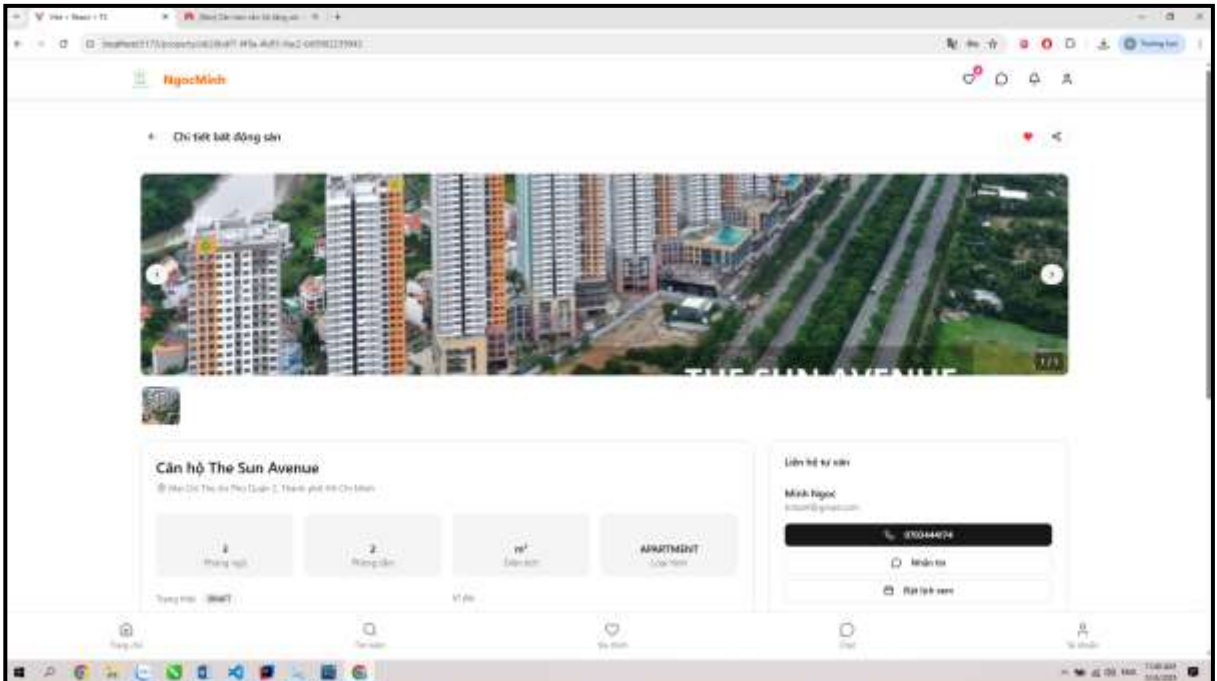
Hình 3. 14 Chi tiết bất động sản

Giao diện chi tiết bất động sản hiển thị hình ảnh, mô tả, giá, diện tích, số phòng và các thông tin quan trọng của căn hộ/nhà. User có thể xem slideshow, thả tim, chia sẻ, và liên hệ trực tiếp với Agent qua gọi điện hoặc nhắn tin.

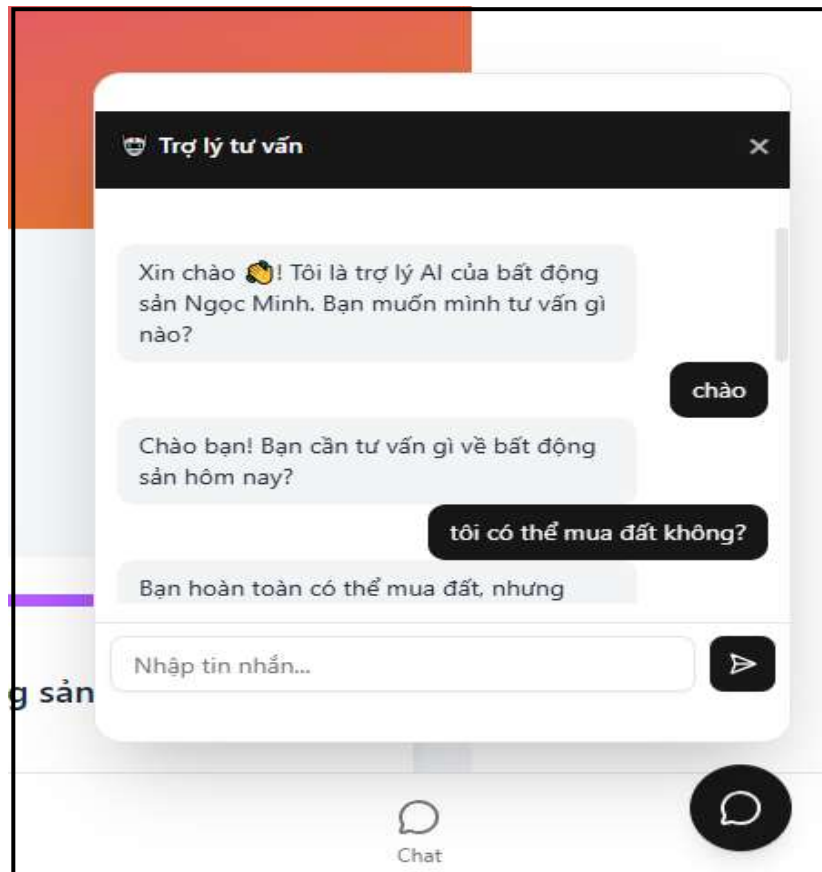


Hình 3. 15 Giao diện user chat với agent

Giao diện chat cho phép User trao đổi trực tiếp với Agent về thông tin bất động sản hoặc nhu cầu tư vấn. Hệ thống hiển thị danh sách tin nhắn theo thời gian thực và hỗ trợ gửi văn bản, hình ảnh và xem lịch sử cuộc trò chuyện.



Hình 3. 16 Giao diện chi tiết bất động sản



Hình 3. 17 Giao diện chat với AI

Giao diện chat với AI cho phép User đặt câu hỏi và nhận phản hồi tự động về bất động sản, tìm kiếm hoặc hỗ trợ hệ thống. AI trả lời theo thời gian thực và lưu lại toàn bộ lịch sử trò chuyện để User xem lại khi cần.

KẾT LUẬN

Đề tài xây dựng website bất động sản là một dự án quan trọng nhằm cung cấp nền tảng cho người dùng tham khảo, xem và trao đổi thông tin về các chung cư, nhà đất cho thuê và mua bán. Hệ thống giúp các doanh nghiệp và cá nhân quản lý tin đăng, giao dịch và thông tin khách hàng một cách hiệu quả, đồng thời cho phép người dùng công khai truy cập, tìm kiếm và lưu lại các bất động sản quan tâm.

Trong hệ thống, Admin chịu trách nhiệm quản lý toàn bộ website, kiểm duyệt tin đăng, phân quyền cho môi giới (Agent) và quản lý danh mục bất động sản. Agent (Môi giới) có thể đăng tin, quản lý các bất động sản của mình, chăm sóc khách hàng và trao đổi thông tin trực tiếp với người dùng. User (Khách hàng) có thể tìm kiếm, lọc tin theo tiêu chí, lưu các tin yêu thích, xem chi tiết bất động sản và liên hệ trực tiếp với môi giới để trao đổi hoặc thuê/mua.

Hệ thống giúp tối ưu hóa trải nghiệm người dùng, tăng tính minh bạch và thuận tiện trong việc tìm kiếm, tham khảo thông tin bất động sản, đồng thời cải thiện hiệu suất quản lý cho Admin và Agent. Đây là bước quan trọng giúp website trở thành kênh thông tin đáng tin cậy, phục vụ cộng đồng và tăng khả năng cạnh tranh trên thị trường bất động sản.

NHỮNG KẾT QUẢ ĐẠT ĐƯỢC

Việc xây dựng hệ thống quản lý website bất động sản mang lại nhiều kết quả đáng kể, bao gồm:

- **Tăng hiệu quả và năng suất:** Hệ thống giúp quản lý và tổ chức thông tin bất động sản, giao dịch, người dùng một cách hiệu quả, từ đó nâng cao năng suất làm việc.
- **Tiết kiệm thời gian:** Hệ thống giúp giảm thiểu thời gian tìm kiếm thông tin, quản lý tin đăng và theo dõi giao dịch.
- **Tăng tính đồng bộ và linh hoạt:** Hệ thống cho phép đồng bộ dữ liệu giữa các vai trò người dùng (Admin, Môi giới, Khách hàng) và linh hoạt trong việc quản lý tin đăng, yêu thích và thông báo.
- **Nâng cao khả năng quản lý:** Hệ thống giúp tổ chức các tin đăng, danh sách khách hàng và nhiệm vụ một cách rõ ràng, hỗ trợ việc phát triển kinh doanh bất động sản bền vững.

- **Giảm thiểu trùng lặp công việc:** Hệ thống đảm bảo mỗi tin đăng, mỗi yêu cầu hay nhiệm vụ chỉ được xử lý một lần, hạn chế nhầm lẫn và trùng lặp.

Sau một thời gian nghiên cứu và thực hành với framework ReactJS và ngôn ngữ JavaScript, website bất động sản đã đạt được kết quả khá tốt. Mặc dù vẫn còn một số điểm chưa hoàn chỉnh, nhưng các chức năng cơ bản đã được triển khai và đang dần được hoàn thiện:

HẠN CHẾ

Bên cạnh những kết quả đã đạt được, dự án vẫn còn một số hạn chế cần cải thiện:

- Website tuy đã hoàn thiện nhưng các tính năng chưa phong phú như mong muốn; phần gợi ý bất động sản cho người dùng chưa được triển khai.
- Việc thiết kế cơ sở dữ liệu với MySQL còn gặp một số vấn đề, dẫn đến hiệu suất chưa tối ưu do chưa khai thác hết khả năng của hệ quản trị.
- Một số tính năng vẫn đang trong quá trình phát triển và chưa hoàn thiện.

HƯỚNG PHÁT TRIỂN

- Nghiên cứu và tối ưu cơ sở dữ liệu MySQL, kết hợp với Spring Boot để nâng cao hiệu suất và đáp ứng tốt hơn nhu cầu người dùng.
- Tìm hiểu thêm các thư viện UI và công nghệ frontend để nâng cao giao diện trực quan, đẹp mắt và sinh động hơn.
- Mở rộng quy mô hệ thống với nhiều tính năng bổ sung, phục vụ đa dạng nhu cầu người dùng.
- Thêm các phương thức thanh toán trực tuyến nhằm nâng cao trải nghiệm giao dịch cho khách hàng.

HẠN CHẾ

Mặc dù hệ thống đã hoàn thiện các chức năng cốt lõi như quản lý người dùng (Admin, Agent), đăng bài, quản lý bài đăng, tin nhắn, thông báo, lịch hẹn,... theo cấu trúc dữ liệu trong BDDS, nhưng vẫn còn tồn tại một số hạn chế nhất định do giới hạn về thời gian và phạm vi đề tài.

Trước hết, một số tính năng nâng cao liên quan đến quản lý dữ liệu bất động sản chưa được triển khai đầy đủ. Hệ thống mới chỉ hỗ trợ đăng bài và quản lý danh sách bài viết theo quy trình cơ bản. Những chức năng như kiểm duyệt bài tự động, phân tích nội dung bài đăng, gợi ý tối ưu SEO hoặc tự động cảnh báo nội dung không

hợp lệ vẫn chưa được phát triển, mặc dù đã có các bảng dữ liệu hỗ trợ như `real_estate_posts`, `real_estate_likes`, `media_files`.

Bên cạnh đó, mô-đun thống kê và báo cáo vẫn còn đơn giản. Hiện tại hệ thống mới chỉ hỗ trợ xem số lượng bài đăng, số lượt tương tác hoặc doanh thu/hoa hồng (nếu có) ở mức tổng quan. Các báo cáo chuyên sâu hơn như thống kê theo khu vực, theo loại hình bất động sản, theo hiệu suất Agent, biểu đồ phân tích xu hướng hoặc dự báo dữ liệu vẫn chưa được tích hợp. Điều này làm hạn chế khả năng hỗ trợ ra quyết định của nhà quản lý.

Về bảo mật, dù hệ thống đã có cơ chế quản lý tài khoản, phân quyền Admin – Agent và lưu nhật ký hoạt động, nhưng vẫn chưa tích hợp các biện pháp bảo mật nâng cao như kiểm soát truy cập theo vai trò chi tiết (RBAC), xác thực đa yếu tố, hạn chế số lần đăng nhập sai, mã hóa dữ liệu nhạy cảm hoặc cảnh báo hoạt động bất thường. Đây là những yếu tố quan trọng khi hệ thống vận hành trong môi trường có nhiều tác nhân tham gia.

Ngoài ra, hệ thống hiện chỉ vận hành trên nền tảng web. Trong bối cảnh môi giới, tư vấn và khách hàng cần sự linh hoạt, việc chưa có ứng dụng di động khiến trải nghiệm sử dụng giảm tính cơ động, đặc biệt là với Agent thường xuyên làm việc ngoài hiện trường.

Cuối cùng, hệ thống chưa hỗ trợ tốt mô hình đa chi nhánh hoặc đa công ty. BĐS có nhiều bảng cho phép mở rộng như `users`, `roles`, `real_estate_posts`, tuy nhiên cơ chế tách dữ liệu theo chi nhánh, phân quyền theo tổ chức, hay đồng bộ dữ liệu thời gian thực giữa nhiều đơn vị chưa được phát triển. Điều này hạn chế hệ thống khi triển khai cho các doanh nghiệp quy mô lớn hoặc các sàn giao dịch có nhiều chi nhánh.

HƯỚNG PHÁT TRIỂN

Từ những hạn chế trên, hệ thống có nhiều tiềm năng mở rộng để trở thành giải pháp hoàn chỉnh hơn.

Một trong những hướng quan trọng là mở rộng mô-đun quản lý bài đăng theo hướng thông minh hơn, tích hợp AI để nhận diện nội dung không hợp lệ, hỗ trợ viết mô tả tối ưu SEO, phân loại bài đăng tự động theo dữ liệu vị trí hoặc loại hình. Các bảng như `media_files`, `real_estate_posts`, `locations` hoàn toàn phù hợp để phát triển thêm các tính năng này.

Bên cạnh đó, hệ thống có thể bổ sung bộ công cụ báo cáo nâng cao, hỗ trợ biểu đồ trực quan, dashboard tổng quan theo thời gian thực, phân tích dữ liệu theo khu vực,

theo loại hình, theo hiệu suất Agent hoặc theo chiến dịch marketing. Việc kết hợp dữ liệu từ post_views, real_estate_likes, appointments sẽ giúp xây dựng các báo cáo sâu hơn và hỗ trợ quyết định chính xác hơn.

Về bảo mật, hệ thống cần được trang bị thêm MFA, hạn chế đăng nhập sai, mã hóa dữ liệu quan trọng, và cơ chế giám sát phiên hoạt động nhằm đảm bảo an toàn cho thông tin người dùng, đặc biệt khi có sự tham gia của nhiều Agent và Admin.

Hệ thống cũng có thể được mở rộng lên nền tảng di động, đáp ứng nhu cầu Agent thao tác nhanh, cập nhật thông tin bài đăng, xem lịch hẹn hoặc nhận thông báo mọi lúc mọi nơi.

Cuối cùng, hướng phát triển dài hạn là xây dựng mô hình đa chi nhánh, hỗ trợ phân tách dữ liệu theo tổ chức, đồng bộ hóa dữ liệu giữa nhiều sàn giao dịch và quản lý tập trung. Đây là bước quan trọng để hệ thống có thể triển khai trong thực tế ở các doanh nghiệp môi giới quy mô lớn.

TÀI LIỆU THAM KHẢO

Nội dung lý thuyết được trình bày trong báo cáo được tham khảo từ những nguồn sau:

- [1]. <https://viblo.asia/>
- [2]. <https://itnavi.com.vn/job>
- [3]. <https://wiki.matbao.net>
- [4]. <https://vietnix.vn>
- [5]. <https://fptcloud.com>