

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN I



ĐỒ ÁN TỐT NGHIỆP

XÂY DỰNG ỨNG DỤNG HELPO TỐI ƯU HOÁ
KẾT NỐI KHÁCH HÀNG VÀ NGƯỜI CUNG CẤP
DỊCH VỤ CHĂM SÓC TẠI GIA

GIẢNG VIÊN HƯỚNG DẪN: TS. Đỗ Thị Liên

SINH VIÊN THỰC HIỆN: B21DCCN626 – Lê Minh Quang

B21DCCN110 – Phạm Thanh Sơn

B21DCCN686 – Đỗ Đức Thiện

HỌC VIỆN CÔNG NGHỆ Bưu Chính Viễn Thông
KHOA CÔNG NGHỆ THÔNG TIN I



ĐỒ ÁN TỐT NGHIỆP

**XÂY DỰNG ỨNG DỤNG HELPO TỐI ƯU HOÁ
KẾT NỐI KHÁCH HÀNG VÀ NGƯỜI CUNG CẤP
DỊCH VỤ CHĂM SÓC TẠI GIA**

GIẢNG VIÊN HƯỚNG DẪN: TS. Đỗ Thị Liên

SINH VIÊN THỰC HIỆN: B21DCCN626 – Lê Minh Quang

B21DCCN110 – Phạm Thanh Sơn

B21DCCN686 – Đỗ Đức Thiện

Hà Nội, năm 2025

LỜI CẢM ƠN

Trước tiên, chúng em muốn gửi lời tri ân sâu sắc đến quý thầy cô khoa Công nghệ thông tin 1 - những người đã tận tâm truyền đạt kiến thức cho chúng em, từ những kiến thức cơ bản nhất cho đến những kiến thức chuyên sâu và phức tạp hơn. Nhờ sự dày công dạy dỗ, hướng dẫn tận tình của các thầy cô, chúng em đã có được nền tảng kiến thức vững chắc, đủ để tiến hành thực hiện và hoàn thành đề tài quan trọng này một cách thuận lợi. Đặc biệt, chúng em muốn bày tỏ lòng biết ơn chân thành nhất tới cô Đỗ Thị Liên - người đã trực tiếp giản hướng dẫn, chỉ bảo tận tình và luôn sẵn lòng giải đáp mọi thắc mắc của chúng em. Cô đã tạo mọi điều kiện thuận lợi để em có thể hoàn thành đề tài một cách tốt nhất.

Tuy nhiên, do kiến thức và kinh nghiệm của chúng em vẫn còn hạn chế, nên không tránh khỏi đề tài này vẫn còn nhiều thiếu sót, hạn chế về cả nội dung lẫn hình thức trình bày. Chúng em rất mong nhận được sự thông cảm từ quý thầy cô. Đồng thời, chúng em cũng kính mong quý thầy cô sẽ tận tình đóng góp ý kiến quý báu, chỉ ra những hạn chế và gợi ý những cải tiến cần thiết để nhóm em có thể hoàn thiện, nâng cao chất lượng các mô hình nghiên cứu trong tương lai, đưa ra những sản phẩm hoàn chỉnh và toàn diện nhất.

Một lần nữa nhóm em xin chân thành cảm ơn!

NHẬN XÉT VÀ ĐÁNH GIÁ

Điểm:

(Bằng chữ:)

Hà Nội, ngày.....tháng.....năm 2025

CÁN BỘ-GIẢNG VIÊN HƯỚNG DẪN

TS. Đỗ Thị Liên

NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN

Điểm: (Bằng chữ:)

Hà Nội, ngày.....tháng.....năm 2025

CÁN BỘ-GIẢNG VIÊN PHẢN BIỆN

MỤC LỤC

ĐẶT VÂN ĐỀ	1
CHƯƠNG 1. TỔNG QUAN VỀ ỨNG DỤNG HEPO TỐI ƯU HÓA KẾT NỐI KHÁCH HÀNG VÀ NGƯỜI CUNG CẤP DỊCH VỤ TẠI GIA	3
1.1. Khảo sát các hệ thống kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia	3
1.2. Xác định yêu cầu ứng dụng kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia	4
1.2.1. Các đối tượng sử dụng.....	4
1.2.2. Yêu cầu chức năng	5
1.2.3. Yêu cầu phi chức năng	5
1.3. Thiết kế tương tác cho ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia.....	6
CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG.....	9
2.1. Mô hình tổng quan hệ thống	9
2.1.1. Tổng quan về hệ thống	9
2.1.2. Mô hình phân rã chức năng	10
2.2. Phương pháp tiếp cận và giải quyết vấn đề	12
2.2.1. Phương pháp xây dựng phần mềm.....	12
2.2.2. Kiến trúc phần mềm	13
2.2.3. Công nghệ xây dựng.....	16
2.3. Phân tích hệ thống	27
2.3.1. Tổng quan nghiệp vụ	27
2.3.2. Đặc tả Usecase	28
2.3.3. Biểu đồ lớp.....	38
2.4. Thiết kế hệ thống	43
2.4.1. Biểu đồ lớp thiết kế	43
2.4.2. Biểu đồ tuần tự cho các tương tác quan trọng	43
2.4.3. Thiết kế luồng xử lý Chatbot	49
2.5. Cơ sở dữ liệu	51
CHƯƠNG 3. THỬ NGHIỆM VÀ ĐÁNH GIÁ HỆ THỐNG	53
3.1. Dữ liệu thực nghiệm.....	53
3.1.1. Tập dữ liệu công việc.	53
3.1.2. Tập dữ liệu thông tin và chính sách	53
3.1.3. Dữ liệu người dùng thử nghiệm.....	54
3.2. Cài đặt thực nghiệm.	54

3.2.1. Độ đo	54
3.2.2. Phương pháp thực nghiệm.	54
3.2.3. Các phương pháp được sử dụng để so sánh	55
3.3. Kết quả thực nghiệm.	56
3.3.1. RAG với Hybrid Recommend	56
3.3.2. ARAG	57
3.4. Thủ nghiệm ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia	57
3.4.1. Giao diện cho khách hàng	57
3.4.2. Giao diện cho nhân viên	63
3.4.3. Giao diện cho quản trị viên.....	69

DANH MỤC HÌNH ẢNH

Ảnh 1. Hệ thống giao diện của khách hàng	6
Ảnh 2. Hệ thống giao diện của nhân viên.....	7
Ảnh 3. System Overview Diagram.....	9
Ảnh 4. Kiến trúc tổng thể.....	13
Ảnh 5. MVVM architecture	14
Ảnh 6. Repository Pattern.....	15
Ảnh 7. Ngôn ngữ lập trình Kotlin	17
Ảnh 8. Jetpack Compose.....	17
Ảnh 9. XML	18
Ảnh 10. Ngôn ngữ lập trình JavaScript	18
Ảnh 11. NodeJs	18
Ảnh 12. ExpressJS framework	19
Ảnh 13. Firebase.....	19
Ảnh 14. Ngôn ngữ lập trình Python	20
Ảnh 15. FastAPI	20
Ảnh 16. LangChain.....	20
Ảnh 17. LangGraph	21
Ảnh 18. Hình minh họa RAG	21
Ảnh 19. Hình minh họa ARAG.....	22
Ảnh 20. Sơ đồ Hybrid Recommandation System	23
Ảnh 21. Vercel.....	24
Ảnh 22. Pinecone.....	24
Ảnh 23. Chroma	25
Ảnh 24. Cloudinary	25
Ảnh 25. SePay	25
Ảnh 26. Gemini	26
Ảnh 27. Groq.....	26
Ảnh 28. Hugging Face Spaces	27
Ảnh 29. Usecase tổng quan	29
Ảnh 30. Biểu đồ lớp phân tích	38
Ảnh 31. Biểu đồ lớp thiết kế	43
Ảnh 32. Biểu đồ tuần tự chức năng đăng ký cho nhân viên	44
Ảnh 33. Biểu đồ tuần tự chức năng đăng nhập cho khách hàng.....	44
Ảnh 34. Biểu đồ tuần tự đăng bài tuyển tìm người thực hiện dịch vụ	45
Ảnh 35. Biểu đồ tuần tự xem chi tiết công việc đã đăng.....	45
Ảnh 36. Biểu đồ tuần tự quản lý ứng viên	46

Ảnh 37. Biểu đồ tuần tự chức năng đăng nhập của nhân viên.....	47
Ảnh 38. Biểu đồ tuần tự nhân viên ứng tuyển công việc	47
Ảnh 39. Biểu đồ tuần tự chức năng huỷ ứng tuyển.....	48
Ảnh 40. Biểu đồ tuần tự chức năng xem lịch làm việc	48
Ảnh 41. Biểu đồ tuần tự tư vấn công việc kết hợp chatbot	49
Ảnh 42. Luồng hoạt động xử lý gợi ý công việc.....	49
Ảnh 43. Luồng hoạt động xử lý tư vấn thông tin	50
Ảnh 44. Luồng hoạt động lịch sử trò chuyện.....	50
Ảnh 45. Bảng cơ sở dữ liệu.....	51
Ảnh 46. Biểu đồ kết quả thực nghiệm	56
Ảnh 47. GD trang chủ của khách hàng chưa đăng nhập	58
Ảnh 48. GD lựa chọn đăng nhập/ đăng ký.....	58
Ảnh 49. GD đăng ký của khách hàng	58
Ảnh 50. GD trang chủ của khách hàng đã đăng nhập	58
Ảnh 51. GD đăng nhập của khách hàng	59
Ảnh 52. GD chọn ví trí làm việc	60
Ảnh 53. GD đăng ký dịch vụ dọn dẹp	60
Ảnh 54. GD chọn thời gian làm việc	60
Ảnh 55. GD xác nhận và thanh toán.....	60
Ảnh 56. GD mã thanh toán QR	61
Ảnh 57. GD công việc đã đăng	61
Ảnh 58. GD chi tiết công việc	61
Ảnh 59. GD danh sách ứng viên	62
Ảnh 60. GD chấp nhận ứng viên	62
Ảnh 61. GD đánh giá nhân viên	63
Ảnh 62. GD sau khi đánh giá nhân viên	63
Ảnh 63. GD bài đánh giá	63
Ảnh 64. GD nhân viên đăng nhập	64
Ảnh 65. GD trang chủ cho nhân viên	64
Ảnh 66. Danh sách công việc dọn dẹp	65
Ảnh 67. Chi tiết công việc dọn dẹp	65
Ảnh 68. Lịch cần thực hiện của việc	65
Ảnh 69. Phạm vi công việc dọn dẹp.....	65
Ảnh 70. Chức năng ứng tuyển.....	66
Ảnh 71. Chức năng huỷ ứng tuyển.....	66
Ảnh 72. Thông báo ứng tuyển thành công.....	66
Ảnh 73. Hiển thị thông báo cảnh báo	66

Ảnh 74. Hiển thị thông báo huỷ thành công	67
Ảnh 75. Lịch làm việc.....	67
Ảnh 76. Chi tiết lịch một công việc.....	67
Ảnh 77. GD cài đặt.....	68
Ảnh 78. Danh sách công việc đã ứng tuyển.....	68
Ảnh 79. Cập nhật hồ sơ nhân viên	68
Ảnh 80. Chọn địa chỉ	68
Ảnh 81. GD tư vấn với chatbot	69
Ảnh 82. GD đăng nhập Admin.....	70
Ảnh 83. GD màn hình chính Admin	70
Ảnh 84. GD đăng ký nhân viên.....	70
Ảnh 85. GD quản lý nhân viên.....	71
Ảnh 86. GD quản lý doanh thu	71
Ảnh 87. GD quản lý yêu cầu.....	71
Ảnh 88. GD thông tin cá nhân (Admin)	72

DANH MỤC BẢNG BIỂU

Bảng 1. Khảo sát hệ thống trên thị trường	3
Bảng 2. Tổng quan nghiệp vụ	28
Bảng 3. Đặc tả usecase đăng ký của khách hàng	30
Bảng 4. Đặc tả usecase đăng nhập của khách hàng	31
Bảng 5. Đặc tả usecase đăng bài tìm người thực hiện dịch vụ	32
Bảng 6. Đặc tả usecase đăng bài tìm người thực hiện dịch vụ	33
Bảng 7. Đặc tả usecase đánh giá nhân viên	34
Bảng 8. Đặc tả usecase đăng nhập của nhân viên	35
Bảng 9. Đặc tả usecase nhân viên ứng tuyển công việc	35
Bảng 10. Đặc tả usecase nhân viên huỷ ứng tuyển công việc	36
Bảng 11. Đặc tả usecase nhân viên xem lịch làm việc theo ngày chọn	37
Bảng 12. Đặc tả usecase nhân viên được tư vấn bởi chatbot.....	38
Bảng 13. Mô tả thuộc tính của lớp thực thể.....	41
Bảng 14. Mối quan hệ của các lớp thực thể.....	42
Bảng 15. Kết quả đánh giá thực nghiệm độ đo theo topk=300.....	56
Bảng 16. Yêu cầu phần mềm khi triển khai	74
Bảng 17. Phân chia công việc cho các thành viên trong nhóm	82

CÔNG THỨC

Công thức 1. Công thức Precision@k	54
Công thức 2. Công thức Recall@k.....	54
Công thức 3. Công thức F1-Score	54
Công thức 4. Công thức Haversine	55

DANH MỤC CÁC CHỮ VIẾT TẮT

TỪ	TIẾNG ANH	TIẾNG VIỆT
AI	Artificial Intelligence	Trí tuệ nhân tạo
API	Application Programming Interface	Giao diện lập trình ứng dụng
ARAG	Agentic Retrieval-Augmented Generation	Sinh nội dung tăng cường truy xuất có tác nhân
CPU	Central Processing Unit	Bộ xử lý trung tâm
HTML	Hyper Text Markup Language	Ngôn ngữ đánh dấu siêu văn bản
HTTP	HyperText Transfer Protocol	Giao thức truyền tải siêu văn bản
JSON	JavaScript Object Notation	Ký hiệu đối tượng JavaScript
LLM	Large Languge Model	Mô hình ngôn ngữ lớn
REST	Representative State Transfer	Chuyển trạng thái đại diện
SQL	Structured Query Language	Ngôn ngữ truy vấn cơ sở dữ liệu
UI	User Interface	Giao diện người dùng
XML	Extensible Markup Language	Ngôn ngữ đánh dấu mở rộng

ĐẶT VĂN ĐỀ

Trong bối cảnh nền kinh tế phát triển mạnh mẽ và nhu cầu sử dụng dịch vụ chăm sóc tại gia ngày càng tăng, việc kết nối hiệu quả giữa khách hàng và người cung cấp dịch vụ trở thành một yêu cầu thiết yếu. Các nền tảng hiện có tuy đã phần nào đáp ứng nhu cầu này nhưng vẫn tồn tại nhiều hạn chế về tính minh bạch, khả năng quản lý và trải nghiệm người dùng. Điều này đặt ra yêu cầu xây dựng một giải pháp tối ưu hơn, vừa đảm bảo sự tiện lợi, vừa nâng cao độ tin cậy trong quá trình kết nối.

Đề tài “**Xây dựng ứng dụng HELPO tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia**” được thực hiện nhằm giải quyết những vấn đề trên. Ứng dụng không chỉ hỗ trợ khách hàng dễ dàng tìm kiếm dịch vụ phù hợp mà còn giúp người cung cấp dịch vụ chủ động tiếp cận công việc, đồng thời đảm bảo quy trình quản lý minh bạch và an toàn. Bên cạnh đó, hệ thống tích hợp các công nghệ hiện đại như xác thực bảo mật, đồng bộ dữ liệu thời gian thực và chatbot hỗ trợ, nhằm mang đến trải nghiệm tối ưu cho người dùng.

Báo cáo này sẽ trình bày toàn bộ quá trình nghiên cứu, phân tích, thiết kế và triển khai hệ thống, từ việc khảo sát nhu cầu thực tế, xây dựng kiến trúc phần mềm đến thử nghiệm và đánh giá hiệu quả. Thông qua đề tài, nhóm mong muốn đóng góp một giải pháp công nghệ thiết thực, góp phần nâng cao chất lượng dịch vụ chăm sóc tại gia và thúc đẩy xu hướng ứng dụng công nghệ vào đời sống.

– Những nhiệm vụ cần thực hiện:

- ❖ **Khảo sát và phân tích nhu cầu thực tế:** Tìm hiểu thị trường dịch vụ chăm sóc tại gia, đánh giá các nền tảng hiện có, phân tích ưu điểm và hạn chế.
- ❖ **Nghiên cứu cơ sở lý thuyết và công nghệ liên quan:** Tìm hiểu các mô hình kiến trúc phần mềm phù hợp cho ứng dụng di động và hệ thống quản trị.
- ❖ **Thiết kế hệ thống:** Xây dựng kiến trúc tổng thể cho ứng dụng, bao gồm phân tích nghiệp vụ, thiết kế cơ sở dữ liệu và các sơ đồ UML (Use Case, Sequence Diagram, Class Diagram).
- ❖ **Phát triển và triển khai ứng dụng:** Xây dựng ứng dụng di động cho khách hàng và nhân viên trên nền tảng Android.
- ❖ **Tích hợp công nghệ AI và Chatbot hỗ trợ:** Xây dựng chatbot dựa trên mô hình ARAG để hỗ trợ người dùng tra cứu thông tin, gợi ý công việc và tư vấn dịch vụ.
- ❖ **Thử nghiệm và đánh giá hệ thống:** Đánh giá chất lượng gợi ý và trải nghiệm người dùng thông qua các độ đo như Precision@k.

- Cấu trúc của đồ án gồm 3 chương chính:

Chương 1. Tổng quan về ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia

1. Khảo sát các hệ thống kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia
2. Xác định yêu cầu ứng dụng kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia
3. Thiết kế tương tác cho ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia
4. Tiêu kết chương 1

Chương 2. Phân tích và thiết kế hệ thống

1. Mô hình tổng quan hệ thống
2. Phương pháp tiếp cận và giải quyết vấn đề
 - 2.2.1. Phương pháp xây dựng phần mềm
 - 2.2.2. Kiến trúc phần mềm
 - 2.2.3. Công nghệ xây dựng
3. Phân tích hệ thống
4. Thiết kế hệ thống
5. Cơ sở dữ liệu
6. Tiêu kết chương 2

Chương 3. Thủ nghiệm và đánh giá hệ thống

1. Dữ liệu thực nghiệm
2. Cài đặt thực nghiệm
 - 3.2.1. Độ đo
 - 3.2.2. Phương pháp thực nghiệm
 - 3.2.3. Các phương pháp được sử dụng để so sánh
3. Kết quả thực nghiệm
4. Thủ nghiệm ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia
5. Tiêu kết chương 3

Sau khoảng thời gian thực hiện đồ án, do vẫn còn hạn chế về kiến thức và kinh nghiệm, thời gian để thực hiện và nghiên cứu tương đối ngắn nên chắc chắn không thể tránh khỏi thiếu sót. Nhóm em rất mong nhận được sự góp ý thầy cô để nội dung của đồ án được hoàn thiện hơn. Nhóm em xin chân thành cảm ơn.

CHƯƠNG 1. TỔNG QUAN VỀ ỦNG DỤNG HEPO TỐI ƯU HOÁ KẾT NỐI KHÁCH HÀNG VÀ NGƯỜI CUNG CẤP DỊCH VỤ TẠI GIA

Chương 1 tập trung trình bày bối cảnh hình thành đề tài và nhu cầu thực tiễn của thị trường dịch vụ chăm sóc tại gia. Nội dung chương bao gồm việc khảo sát các nền tảng kết nối hiện có, phân tích những hạn chế còn tồn tại, từ đó xác định bài toán cần giải quyết. Trên cơ sở đó, chương làm rõ mục tiêu xây dựng ứng dụng Helpo, đồng thời đề xuất các yêu cầu chức năng, phi chức năng và phác họa các luồng tương tác cơ bản của hệ thống.

1.1. Khảo sát các hệ thống kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia

Các nền tảng kết nối dịch vụ hiện nay có thể chia thành hai nhóm chính: (1) các ứng dụng chuyên biệt về giúp việc và chăm sóc tại gia; (2) các nền tảng tìm việc tự do hoặc chợ rao vặt.

Bảng sau tổng hợp đặc điểm và hạn chế của một số hệ thống tiêu biểu:

TÊN SẢN PHẨM	ĐẶC ĐIỂM CHÍNH	HẠN CHẾ NỔI BẬT
Nền tảng tuyển dụng chung (Chợ Tốt, Facebook Groups)	Dễ đăng bài, tiếp cận số lượng lớn người dùng; đa dạng loại dịch vụ	Không có cơ chế xác minh người cung cấp dịch vụ; thiếu hệ thống đánh giá chuẩn hóa; không có quản lý lịch làm việc; không hỗ trợ xử lý khiếu nại
JupViec.vn	Cung cấp dịch vụ giúp việc theo giờ; có đánh giá khách hàng; có quản lý đơn hàng	Hệ sinh thái dịch vụ hạn chế (tập trung vào dọn dẹp); nhân viên không chủ động ứng tuyển theo yêu cầu; chưa hỗ trợ đa dịch vụ trong cùng một nền tảng
bTaskee	Giao diện hiện đại; có cơ chế ghép việc tự động	Phạm vi chủ yếu xoay quanh giúp việc và vệ sinh; chưa hỗ trợ các dịch vụ chăm sóc, bảo trì, sửa chữa chuyên sâu

Bảng 1. Khảo sát hệ thống trên thị trường

Qua bảng so sánh các nền tảng kết nối dịch vụ chăm sóc tại gia hiện có, có thể nhận thấy mỗi hệ thống đều sở hữu những ưu điểm nhất định nhưng vẫn tồn tại nhiều hạn chế mang tính hệ thống. Các nền tảng tuyển dụng chung như Chợ Tốt hay Facebook Groups có lợi thế về khả năng tiếp cận người dùng và sự đa dạng dịch vụ, tuy nhiên lại thiếu cơ chế xác minh người cung cấp, không có hệ thống đánh giá chuẩn hóa, cũng như chưa hỗ trợ quản lý lịch làm việc và xử lý khiếu nại, dẫn đến mức độ tin cậy chưa cao. Trong khi đó, các ứng dụng chuyên biệt như JupViec.vn hay bTaskee đã bước đầu giải quyết được bài toán quản lý đơn hàng và trải nghiệm người dùng thông qua giao diện thân thiện và cơ chế ghép việc, nhưng lại bị giới hạn về phạm vi dịch vụ, chủ yếu tập trung vào lĩnh vực giúp việc và vệ sinh, chưa đáp ứng được nhu cầu tổng hợp của hộ gia đình trong các lĩnh vực chăm sóc, bảo trì và sửa chữa.

Bên cạnh đó, phần lớn các hệ thống hiện nay chưa tối ưu hóa quá trình ghép nối giữa khách hàng và người cung cấp dịch vụ, thể hiện ở việc nhân viên chưa được chủ động ứng tuyển linh hoạt hoặc chưa có cơ chế gợi ý công việc phù hợp dựa trên năng lực và khu vực làm việc. Đồng thời, công tác quản lý và xử lý khiếu nại còn thiếu quy trình thống nhất, khiến trải nghiệm người dùng chưa đồng đều và khó đảm bảo tính minh bạch. Từ những phân tích trên, có thể thấy nhu cầu về một nền tảng như Helpo là rõ ràng, với mục tiêu xây dựng một hệ sinh thái dịch vụ đa dạng, minh bạch, có cơ chế đánh giá chuẩn hóa, ghép nối hiệu quả và quy trình xử lý khiếu nại rõ ràng, qua đó nâng cao chất lượng kết nối giữa khách hàng và người cung cấp dịch vụ.

1.2. Xác định yêu cầu ứng dụng kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia

1.2.1. Các đối tượng sử dụng

Hệ thống Helpo phục vụ ba nhóm đối tượng chính, mỗi nhóm đảm nhiệm một vai trò nghiệp vụ riêng biệt và tương tác với hệ thống theo các chức năng khác nhau. Việc xác định rõ các đối tượng sử dụng giúp làm rõ phạm vi hoạt động của hệ thống và là cơ sở cho quá trình phân tích, thiết kế các chức năng tương ứng.

– *Khách hàng* là nhóm người dùng có nhu cầu sử dụng các dịch vụ chăm sóc tại gia. Họ thực hiện các thao tác như đăng ký nhu cầu dịch vụ, lựa chọn người cung cấp phù hợp, theo dõi trạng thái công việc và đưa ra đánh giá sau khi dịch vụ hoàn tất. Khách hàng đóng vai trò trung tâm trong việc khởi tạo các đơn dịch vụ và quyết định chất lượng kết nối trong hệ thống.

– *Người cung cấp dịch vụ (nhân viên)* là nhóm người dùng đã được xét duyệt và cho phép tham gia cung cấp dịch vụ trên hệ thống. Nhân viên có thể xem danh sách công việc, chủ động ứng tuyển, nhận và thực hiện công việc theo yêu cầu của khách hàng.

Ngoài ra, nhân viên còn nhận được đánh giá và phản hồi sau mỗi công việc, từ đó ảnh hưởng trực tiếp đến mức độ uy tín và cơ hội được gợi ý công việc trong hệ thống.

– *Quản trị viên (Admin)* là đối tượng chịu trách nhiệm giám sát toàn bộ hoạt động của hệ thống. Admin có quyền quản lý tài khoản người dùng, theo dõi các đơn dịch vụ và đơn ứng tuyển, cũng như tiếp nhận và xử lý các khiếu nại phát sinh. Vai trò của quản trị viên giúp đảm bảo hệ thống vận hành ổn định, minh bạch và tuân thủ các chính sách đã đề ra.

1.2.2. Yêu cầu chức năng

Hệ thống Helpo được thiết kế nhằm đáp ứng đầy đủ các yêu cầu chức năng phục vụ cho ba nhóm đối tượng chính là khách hàng, người cung cấp dịch vụ và quản trị viên. Trước hết, hệ thống hỗ trợ chức năng đăng ký và đăng nhập, bao gồm xác thực tài khoản thông thường và đăng nhập nhanh thông qua Google Sign-In, giúp người dùng dễ dàng tiếp cận ứng dụng.

Đối với *khách hàng*, hệ thống cho phép tạo và đăng bài tuyển dụng dịch vụ tại gia, theo dõi danh sách công việc đã đăng, cũng như quản lý trạng thái của từng công việc trong suốt vòng đời, bao gồm các trạng thái: thanh toán, đang tuyển, đang thực hiện, hoàn thành và hủy. Khách hàng có thể xem danh sách các nhân viên đã ứng tuyển, thực hiện thao tác duyệt hoặc từ chối ứng viên phù hợp, đồng thời đưa ra đánh giá sau khi công việc kết thúc nhằm phản ánh chất lượng dịch vụ.

Đối với *người cung cấp dịch vụ*, hệ thống hỗ trợ xem danh sách các công việc đang tuyển theo loại hình dịch vụ và khu vực, cho phép nhân viên chủ động ứng tuyển vào các công việc phù hợp với năng lực của mình. Trong quá trình thực hiện công việc, nhân viên có thể theo dõi trạng thái công việc và trao đổi thông tin với khách hàng thông qua chức năng trò chuyện thời gian thực.

Bên cạnh đó, *quản trị viên* được cung cấp các chức năng quản lý toàn diện, bao gồm quản lý tài khoản người dùng, bài đăng dịch vụ, doanh thu và hóa đơn, cũng như giám sát hoạt động của hệ thống.

Ngoài ra, hệ thống tích hợp chatbot ARAG nhằm hỗ trợ người dùng tra cứu thông tin ứng dụng, chính sách sử dụng, cũng như hỗ trợ tìm kiếm và ứng tuyển công việc một cách thuận tiện.

1.2.3. Yêu cầu phi chức năng

Bên cạnh các yêu cầu chức năng, hệ thống Helpo còn phải đáp ứng các yêu cầu phi chức năng nhằm đảm bảo chất lượng và tính bền vững trong quá trình vận hành. Về tính bảo mật, hệ thống sử dụng *Firebase Authentication* để xác thực người dùng và áp dụng cơ chế phân quyền rõ ràng giữa các vai trò nhằm bảo vệ dữ liệu và hạn chế truy cập trái phép.

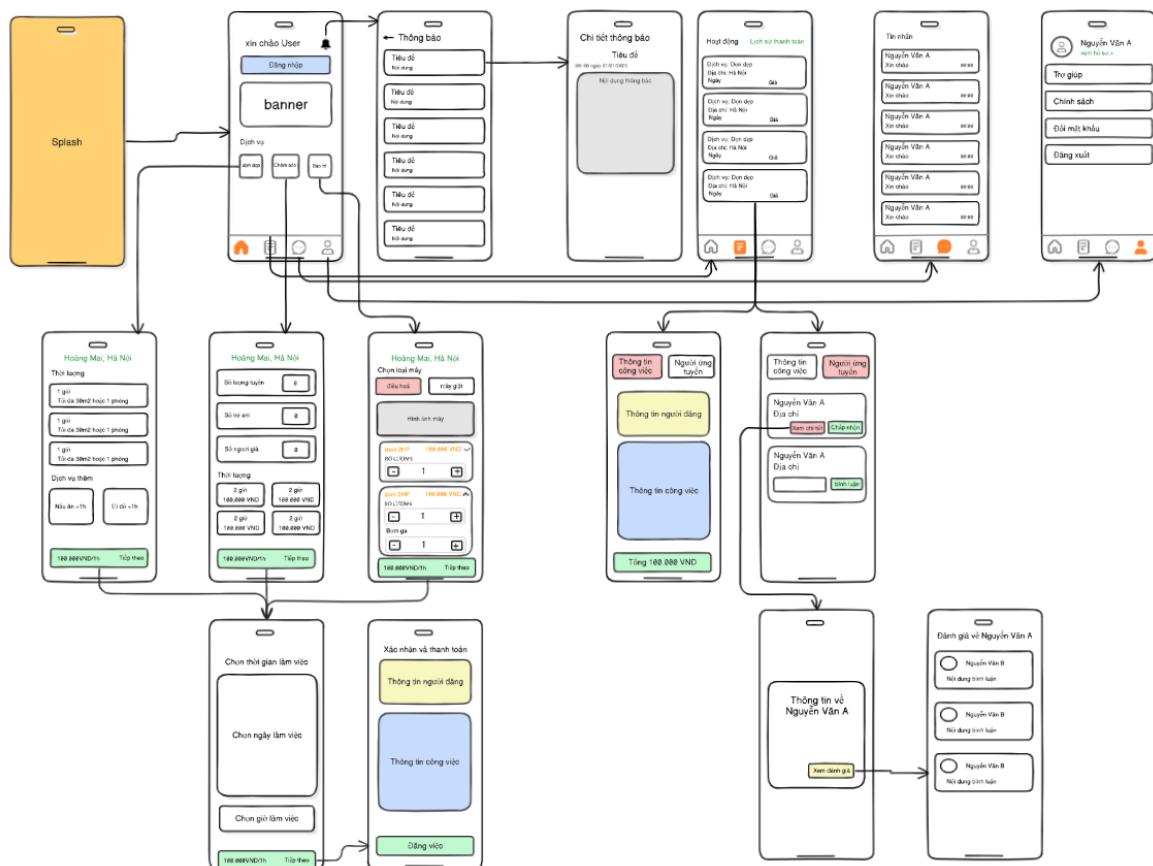
Về tính ổn định, hệ thống được thiết kế để hạn chế tối đa thời gian gián đoạn, đồng thời hỗ trợ đồng bộ dữ liệu theo thời gian thực nhằm đảm bảo thông tin giữa các bên luôn được cập nhật kịp thời. Ngoài ra, khả năng mở rộng cũng là một yêu cầu quan trọng, cho phép hệ thống dễ dàng bổ sung các loại dịch vụ mới hoặc tích hợp thêm chức năng thanh toán trực tuyến trong tương lai mà không ảnh hưởng đến kiến trúc hiện tại.

Cuối cùng, hệ thống chú trọng đến tính dễ sử dụng, với giao diện rõ ràng, trực quan, phù hợp với cả những người dùng không chuyên về công nghệ, qua đó nâng cao trải nghiệm người dùng và khả năng tiếp cận của ứng dụng.

1.3. Thiết kế tương tác cho ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia

Dưới đây là wireframe dành cho 2 vai trò người dùng: khách hàng và nhân viên

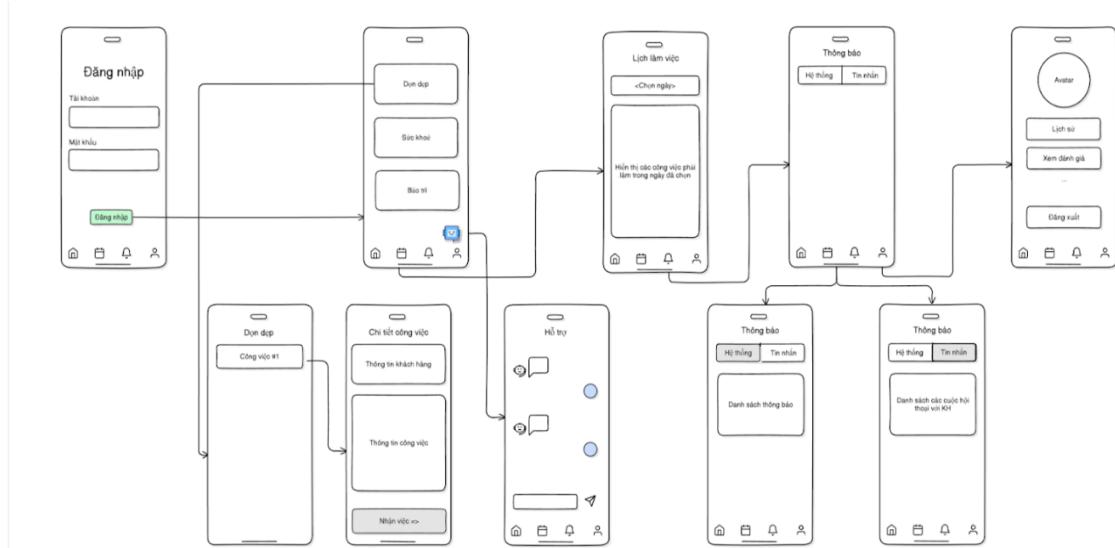
- Khách hàng



Ảnh 1. Hệ thống giao diện của khách hàng

Đồ án tốt nghiệp Đại học

– Nhân viên



Ảnh 2. Hệ thống giao diện của nhân viên

Tiểu kết chương 1

Chương 1 đã làm rõ bối cảnh hình thành và nhu cầu thực tiễn của thị trường dịch vụ chăm sóc tại gia, đồng thời chỉ ra những hạn chế còn tồn tại của các nền tảng kết nối hiện nay. Trên cơ sở phân tích đó, chương đã xác định được các vấn đề cốt lõi cần giải quyết và khẳng định sự cần thiết của việc xây dựng ứng dụng Helpo như một giải pháp thống nhất, minh bạch và đáng tin cậy.

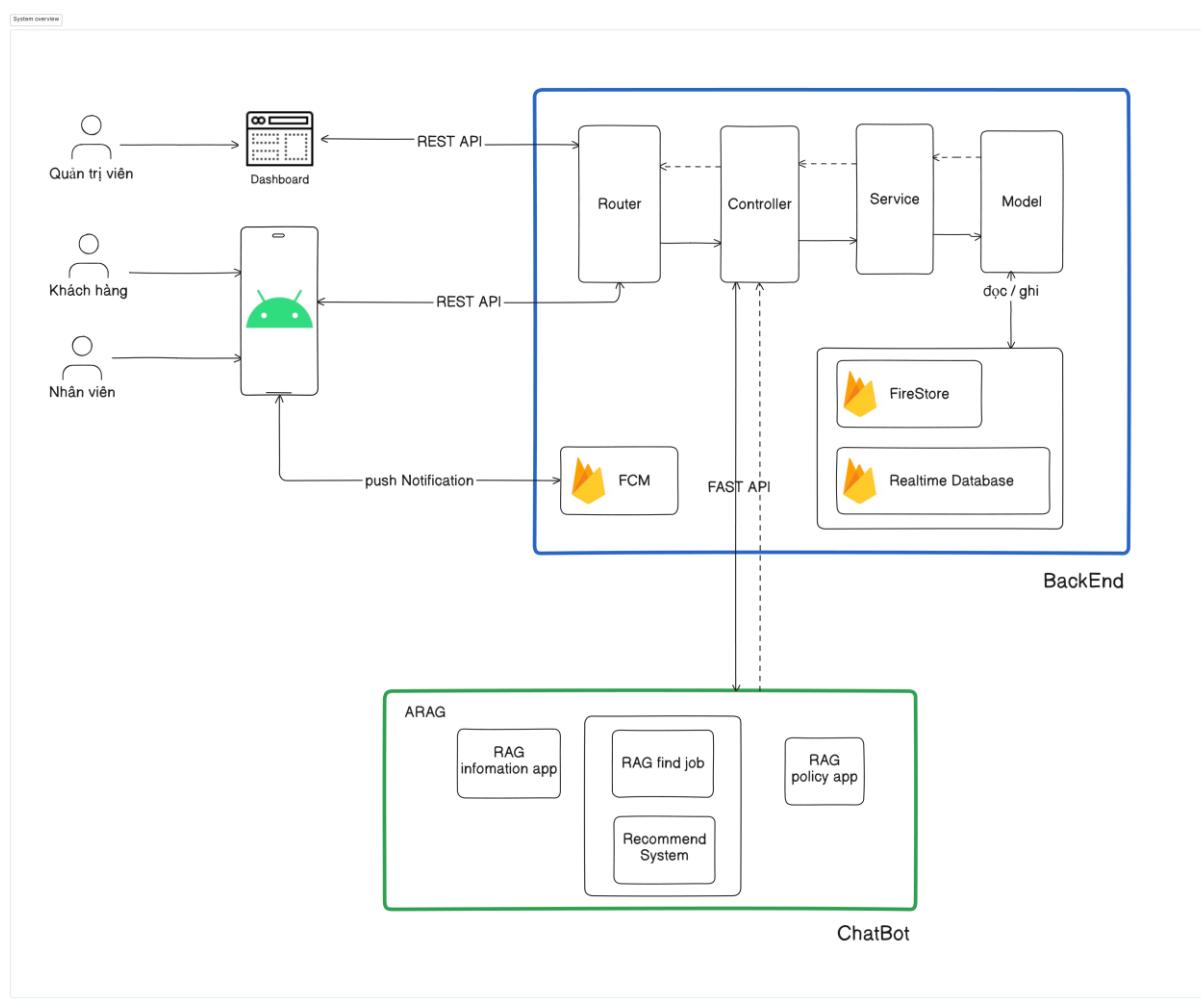
Bên cạnh việc đề xuất các yêu cầu chức năng và phi chức năng, chương cũng đã mô tả các tác nhân chính, các luồng tương tác cơ bản và định hướng thiết kế trải nghiệm người dùng. Những nội dung này đóng vai trò làm nền tảng cho quá trình thiết kế và triển khai hệ thống. Từ đó, chương 2 sẽ tập trung vào phân tích và thiết kế chi tiết kiến trúc hệ thống, các thành phần kỹ thuật và mô hình dữ liệu nhằm hiện thực hóa các yêu cầu đã được xác định.

CHƯƠNG 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

Chương 2 tập trung vào quá trình phân tích và thiết kế hệ thống ứng dụng Helpo nhằm hiện thực hóa các yêu cầu đã được xác định ở Chương 1. Nội dung chương trình bao gồm mô hình tổng quan của hệ thống, phương pháp tiếp cận và giải quyết vấn đề, bao gồm phương pháp xây dựng phần mềm, kiến trúc và các công nghệ được lựa chọn. Trên cơ sở đó, chương tiến hành phân tích chi tiết các chức năng, thiết kế hệ thống và xây dựng mô hình cơ sở dữ liệu, tạo nền tảng kỹ thuật vững chắc cho giai đoạn triển khai và phát triển ứng dụng ở các chương tiếp theo.

2.1. Mô hình tổng quan hệ thống

2.1.1. Tổng quan về hệ thống



Ảnh 3. System Overview Diagram

Hình trên trình bày mô hình tổng quan hệ thống của ứng dụng Helpo, mô tả các thành phần chính của hệ thống và mối quan hệ tương tác giữa chúng. Hệ thống được

xây dựng theo kiến trúc Client–Server, trong đó các ứng dụng phía người dùng và hệ thống quản trị giao tiếp với Backend thông qua các API chuẩn REST.

Ở phía Client, hệ thống bao gồm hai ứng dụng Android độc lập: một app dành cho khách hàng và một app khác dành cho người cung cấp dịch vụ (nhân viên). Ngoài ra, Dashboard quản trị được sử dụng bởi quản trị viên để theo dõi và quản lý hoạt động của hệ thống. Tất cả các thành phần phía Client đều gửi yêu cầu và nhận phản hồi từ Server thông qua REST API.

Phía Server đóng vai trò xử lý trung tâm, chịu trách nhiệm tiếp nhận yêu cầu từ Client, xử lý nghiệp vụ và thực hiện các thao tác đọc/ghi dữ liệu với Firebase. Trong đó, Cloud Firestore và Realtime Database được sử dụng để lưu trữ dữ liệu nghiệp vụ, lịch sử hoạt động và đồng bộ các chức năng thời gian thực như chat.

Bên cạnh các chức năng cốt lõi, hệ thống tích hợp Chatbot ARAG, bao gồm các mô-đun RAG hỗ trợ tra cứu thông tin ứng dụng, tìm kiếm công việc, tra cứu chính sách và mô-đun gợi ý (Recommendation System). Chatbot ARAG tiếp nhận truy vấn từ người dùng, xử lý thông tin và kết nối với LLM để tạo phản hồi phù hợp, sau đó trả kết quả về ứng dụng thông qua cơ chế query/response.

Mô hình tổng quan hệ thống giúp làm rõ cấu trúc tổng thể của Helpo, xác định vai trò của từng thành phần và mối liên kết giữa các hệ thống con.

2.1.2. Mô hình phân rã chức năng

a. Phần dành cho Khách hàng

- *Đăng nhập*: Khách hàng truy cập ứng dụng → hệ thống hiển thị màn hình đăng nhập (tùy chọn đăng nhập bằng Google) → Khách hàng nhập thông tin tài khoản và nhấn “Đăng nhập” → hệ thống kiểm tra thông tin hợp lệ → chuyển đến trang chủ hiển thị danh sách công việc đã đăng và các đề xuất dịch vụ.

- *Đăng ký*: Người dùng truy cập ứng dụng → chọn “Đăng ký” → hệ thống hiển thị màn hình đăng ký → Khách hàng nhập các thông tin cá nhân (họ tên, email, mật khẩu, số điện thoại, khu vực) → nhấn “Đăng ký” → hệ thống tạo tài khoản mới và chuyển về màn hình đăng nhập.

- *Đăng xuất*: Khách hàng sau khi đăng nhập thành công → vào mục “Hồ sơ cá nhân” → nhấn “Đăng xuất” → hệ thống hiển thị thông báo xác nhận → Khách hàng xác nhận → hệ thống đăng xuất và quay về màn hình đăng nhập.

- *Đăng bài tuyển dụng*: Tại trang chủ, khách hàng chọn các dịch vụ được hệ thống cung cấp (Dọn dẹp, Chăm sóc, Bảo trì) → hệ thống hiển thị form tạo công việc → Khách hàng nhập thông tin cần thiết: địa chỉ, thời gian làm, mô tả chi tiết và mức giá → nhấn “Đăng việc” → hệ thống lưu lại thông tin đơn dịch vụ và hiển thị danh sách này cho người cung cấp.

Đồ án tốt nghiệp Đại học

– *Xem danh sách đơn dịch vụ đã đăng*: Tại trang hoạt động → hệ thống hiển thị danh sách bài đăng cùng trạng thái (Đang tuyển, Đang thực hiện, Hoàn thành) → Khách hàng chọn để xem chi tiết.

– *Duyệt đơn ứng tuyển*: Tại màn hình chi tiết công việc, chọn xem “Ứng viên” → hệ thống hiển thị danh sách các nhân viên đã ứng tuyển → Khách hàng xem hồ sơ và chọn “Chấp nhận” hoặc “Từ chối” → hệ thống cập nhật trạng thái đơn ứng tuyển và gửi thông báo đến người cung cấp dịch vụ.

– *Đánh giá Nhân viên*: Sau khi công việc hoàn tất → hệ thống hiển thị form đánh giá → Khách hàng nhập điểm rating (1–5 sao) và nhận xét → nhấn “Bình luận” → hệ thống lưu thông tin review và cập nhật hồ sơ nhân viên.

b. Phần dành cho Người cung cấp dịch vụ (nhân viên)

– *Đăng ký*: Bắt buộc người cung cấp dịch vụ phải đăng ký tài khoản với quản trị viên → Thông qua một số thủ tục, xác nhận → trở thành nhân viên chính thức, sau đó mới có thể sử dụng ứng dụng này.

– *Đăng nhập*: Nhân viên điền thông tin vào form đăng nhập (email, password) → đăng nhập để vào trang chủ.

– *Xem danh sách công việc*: Sau khi đăng nhập, Nhân viên chọn tab “Trang chủ” → hệ thống hiển thị danh sách việc đang tuyển theo loại hình dịch vụ → Nhân viên chọn một công việc để xem chi tiết (mô tả, địa điểm, mức giá, thời gian).

– *Ứng tuyển công việc*: Nhân viên chọn “Ứng tuyển” → hệ thống gửi yêu cầu ứng tuyển đến Khách hàng → hiển thị thông báo “Đã ứng tuyển thành công” → công việc được lưu vào danh sách “Đang chờ duyệt”.

– *Huỷ ứng tuyển*: Trước khi Khách hàng chấp nhận, Nhân viên có thể vào mục “Ứng tuyển của tôi” → chọn công việc → nhấn “Huỷ ứng tuyển” → hệ thống cập nhật trạng thái và thông báo đến Khách hàng.

– *Nhận việc*: Khi Khách hàng chấp thuận đơn ứng tuyển → hệ thống gửi thông báo “Bạn đã được nhận” → Nhân viên xác nhận nhận việc.

– *Xem đánh giá*: Sau khi Khách hàng đánh giá → Nhân viên nhận thông báo → vào mục “Đánh giá” để xem điểm và nhận xét → hệ thống hiển thị trung bình các đánh giá trên hồ sơ.

– *Xem lịch làm việc*: Nhân viên chọn tab “Lịch làm việc” → hệ thống hiển thị các công việc đã được duyệt theo ngày/giờ.

c. Phần dành cho Admin

- *Đăng nhập hệ thống*: Admin truy cập giao diện quản trị → nhập thông tin đăng nhập → hệ thống xác thực và chuyển đến trang Dashboard tổng quan.
- *Đăng ký nhân viên*: Admin login chọn Đăng ký thành viên → Chọn người làm → Nhập: email, password, confirmPassword → Nhấn Đăng ký
- *Quản lý số lượng thành viên*: Ở trang dashboard → Chọn Quản lý thành viên → Biểu đồ số lượng thành viên theo role hiện ra
- *Xem doanh thu*: Ở trang dashboard → Chọn Quản lý công việc → Biểu đồ doanh thu theo tháng hiện ra
- *Quản lý hóa đơn*: Ở trang dashboard → Chọn Quản lý hóa đơn → Hiển thị số lượng hóa đơn theo trạng thái
- *Tìm kiếm hóa đơn*: Ở trang dashboard → Chọn Quản lý hóa đơn → Bảng trạng thái hóa đơn → Search theo tên hoặc filter
- *Xem thông tin cá nhân*: Ở trang dashboard → Chọn Thông tin cá nhân → Thông tin cá nhân hiện ra
- *Đăng xuất*: Ở trang dashboard → Chọn Đăng xuất

2.2. Phương pháp tiếp cận và giải quyết vấn đề

2.2.1. Phương pháp xây dựng phần mềm

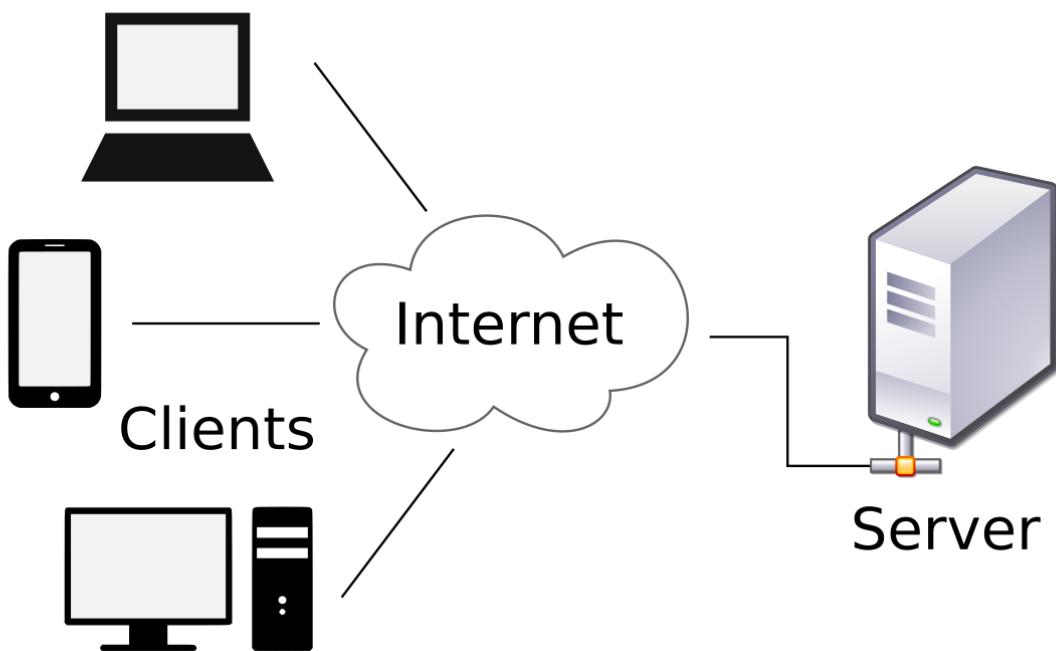
Hệ thống Helpo được chia thành bốn module chính, mỗi module độc lập về mặt chức năng nhưng có thể tương tác thông qua các API chuẩn:

- Module người dùng (User Module): Quản lý đăng ký, đăng nhập, phân quyền và hồ sơ cá nhân. Module này tích hợp Firebase Authentication để xác thực người dùng và hỗ trợ đăng nhập qua Google Sign-In.
- Module quản lý công việc (Job Management Module): Chịu trách nhiệm cho phép khách hàng đăng bài tìm người làm, nhân viên ứng tuyển công việc, quản lý trạng thái công việc (thanh toán, đang tuyển, đang thực hiện, hoàn thành, hủy) và theo dõi lịch làm việc.
- Module đánh giá (Rating & Review Module): Thu thập và quản lý đánh giá từ khách hàng về chất lượng dịch vụ, lưu trữ lịch sử đánh giá và tính toán điểm trung bình để hỗ trợ quy trình gợi ý công việc.
- Module hỗ trợ thông minh (AI Support Module): Tích hợp chatbot ARAG nhằm hỗ trợ tra cứu thông tin, tư vấn dịch vụ và gợi ý công việc phù hợp cho nhân viên dựa trên hồ sơ, vị trí và độ uy tín.

Việc phân tách module giúp nhóm phát triển tập trung triển khai từng phần một cách độc lập, rút ngắn thời gian phát triển, đồng thời dễ dàng mở rộng hoặc thay thế module khi có nhu cầu.

2.2.2. Kiến trúc phần mềm

2.2.2.1. Kiến trúc Client-Server



Ảnh 4. Kiến trúc tổng thể

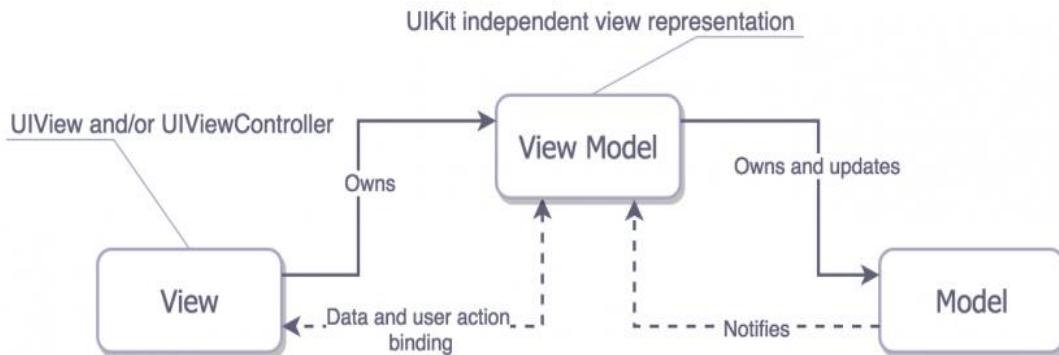
Hệ thống Helpo được thiết kế dựa trên *Client–Server Architecture*, trong đó chức năng giữa phía Client và phía Server được phân tách rõ ràng nhằm đảm bảo tính mở rộng, khả năng bảo trì và hiệu năng xử lý khi phục vụ đồng thời nhiều người dùng. Phía Client bao gồm các ứng dụng di động Android cùng với trang web dashboard chịu trách nhiệm thu thập dữ liệu đầu vào từ người dùng, gửi yêu cầu đến Server thông qua các API chuẩn REST và hiển thị kết quả phản hồi tương ứng. Phía Server đóng vai trò là trung tâm xử lý nghiệp vụ, thực hiện xác thực, phân quyền, quản lý dữ liệu và điều phối các chức năng chính của hệ thống. Đồng thời, Server còn tích hợp các mô-đun gợi ý dựa trên RS-ARAG, hỗ trợ tìm kiếm thông tin, đề xuất công việc và tư vấn cho người dùng.

Việc áp dụng kiến trúc Client–Server giúp hệ thống Helpo vận hành linh hoạt, dễ mở rộng trong tương lai và phù hợp với mô hình ứng dụng mobile-first, đồng thời tạo nền tảng vững chắc cho các bước phân tích và thiết kế chi tiết ở các mục tiếp theo.

2.2.2.2. Phần ứng dụng di động

- MVVM architecture

Ứng dụng Android của Helpo áp dụng mô hình MVVM (Model – View – ViewModel) nhằm tách biệt giữa giao diện người dùng và logic xử lý, giúp mã nguồn dễ bảo trì, dễ mở rộng và thuận lợi cho việc kiểm thử.



Ảnh 5. MVVM architecture

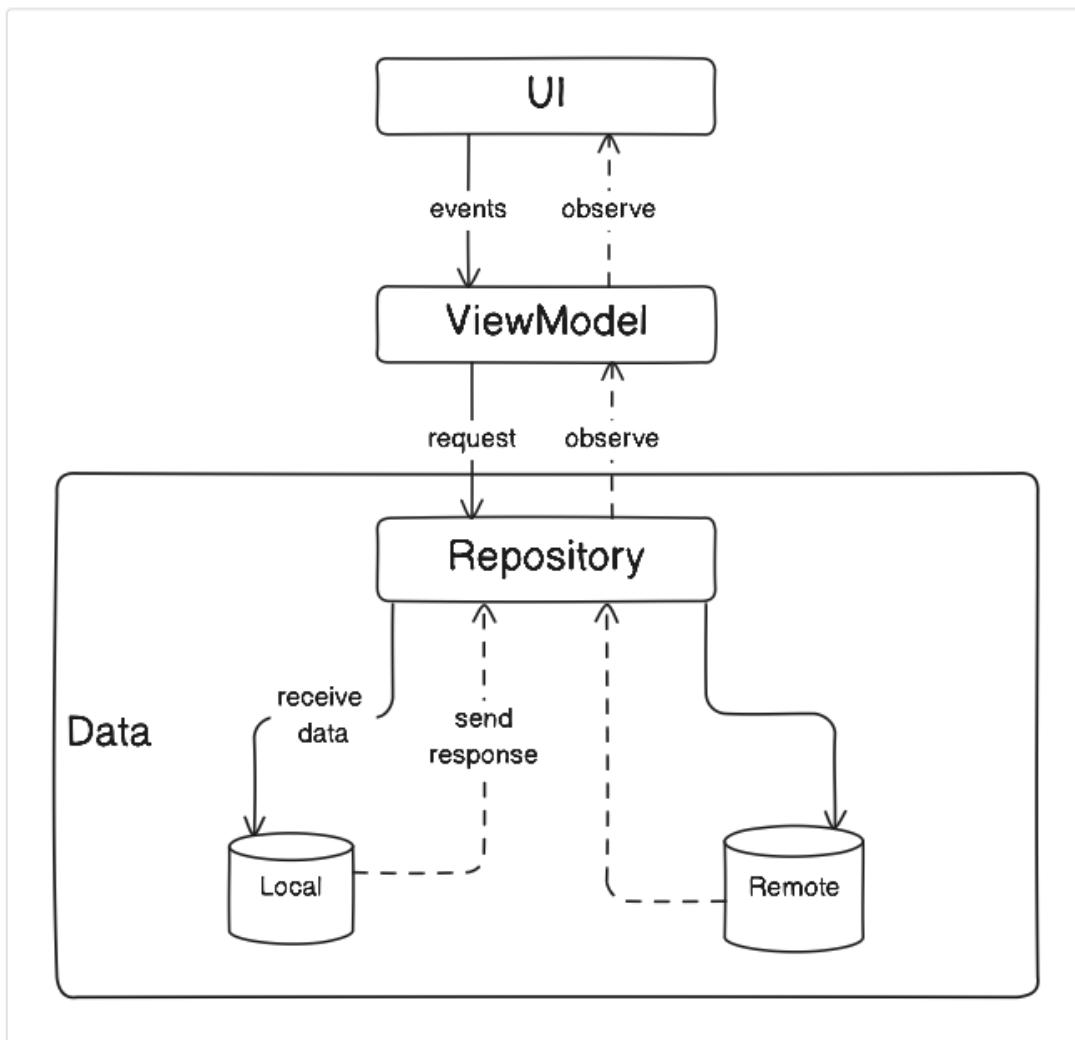
View đại diện cho phần giao diện người dùng, bao gồm các Activity, Fragment hoặc giao diện được xây dựng bằng Jetpack Compose. View chỉ chịu trách nhiệm hiển thị dữ liệu và chuyển các thao tác của người dùng đến **ViewModel**, không xử lý logic nghiệp vụ.

ViewModel đóng vai trò trung gian giữa View và Model. Thành phần này tiếp nhận yêu cầu từ View, xử lý logic liên quan đến giao diện và quản lý trạng thái hiển thị, sau đó cung cấp dữ liệu cho View thông qua các cơ chế quan sát như LiveData hoặc StateFlow. **ViewModel** không phụ thuộc trực tiếp vào View, giúp tăng khả năng tái sử dụng và thuận lợi cho việc kiểm thử.

Model chịu trách nhiệm quản lý dữ liệu của ứng dụng, bao gồm dữ liệu lấy từ Server hoặc các dịch vụ Firebase. Model không quan tâm đến cách dữ liệu được hiển thị mà chỉ tập trung vào việc cung cấp dữ liệu đúng và đầy đủ cho **ViewModel**.

Việc áp dụng *MVVM architecture* giúp tách biệt rõ ràng giữa giao diện và logic xử lý, giảm độ phức tạp của mã nguồn, đồng thời hỗ trợ tốt cho việc mở rộng và bảo trì ứng dụng trong quá trình phát triển lâu dài.

- Repository Pattern



Ảnh 6. Repository Pattern

Để hiện thực hóa mô hình MVVM một cách hiệu quả, ứng dụng Helpo áp dụng Repository Pattern nhằm tách biệt hoàn toàn tầng truy xuất dữ liệu khỏi tầng xử lý logic giao diện. Trong kiến trúc này, Repository đóng vai trò là lớp trung gian giữa ViewModel và các nguồn dữ liệu, bao gồm dữ liệu cache từ local, REST API từ Backend và các dịch vụ Firebase như Realtime Database. Nhờ đó, ViewModel không làm việc trực tiếp với dữ liệu mà chỉ tương tác thông qua các phương thức được định nghĩa trong Repository, giúp giảm sự phụ thuộc giữa các lớp và nâng cao tính tổ chức của mã nguồn.

Việc áp dụng Repository Pattern mang lại nhiều lợi ích quan trọng đối với ứng dụng di động. Trước hết, Repository cung cấp một *giao diện truy xuất dữ liệu thống nhất*, cho phép ViewModel tiếp cận dữ liệu mà không cần quan tâm đến chi tiết nguồn dữ liệu đến từ API hay cơ sở dữ liệu thời gian thực. Cách tiếp cận này giúp hạn chế lặp lại logic truy xuất dữ liệu, giảm độ phức tạp của tầng giao diện và tăng khả năng tái sử dụng mã nguồn.

Bên cạnh đó, Repository Pattern hỗ trợ tốt cho mô hình *offline-first* thông qua việc tận dụng cơ chế cache cục bộ (Room Database). Nhờ vậy, ứng dụng vẫn có thể hiển

thị dữ liệu đã lưu ngay cả khi thiết bị không có kết nối mạng, góp phần cải thiện đáng kể trải nghiệm người dùng trong các tình huống mạng không ổn định.

Ngoài ra, Repository Pattern còn giúp *nâng cao khả năng kiểm thử* của hệ thống. Do ViewModel chỉ phụ thuộc vào các interface của Repository, dữ liệu có thể được dễ dàng mock trong quá trình viết unit test mà không cần kết nối tới API hoặc Firebase thực tế. Đồng thời, kiến trúc này cũng tạo điều kiện thuận lợi cho việc mở rộng và thay đổi nguồn dữ liệu trong tương lai, ví dụ như chuyển đổi từ Firestore sang một hệ quản trị cơ sở dữ liệu khác, mà không ảnh hưởng đến ViewModel hay giao diện người dùng.

Từ những ưu điểm trên, có thể thấy Repository Pattern là lựa chọn phù hợp cho ứng dụng Helpo, đáp ứng tốt các yêu cầu về tính linh hoạt, khả năng mở rộng, độ ổn định và đảm bảo trải nghiệm người dùng nhất quán trong bối cảnh một hệ thống dịch vụ tại gia hoạt động trên nền tảng di động.

2.2.2.3. Kiến trúc phía Server

- Mô hình phân lớp

Hệ thống chia thành các lớp (layers) mỗi lớp đảm bảo nhận một vai trò riêng biệt:

- Lớp Route Layer (định tuyến yêu cầu): Tiếp nhận yêu cầu HTTP, định tuyến đến Controller tương ứng
- Lớp Controller Layer (điều khiển luồng xử lý): Nhận request từ Route gọi đến Service để xử lý logic sau đó trả về response
- Lớp Service Layer (xử lý nghiệp vụ, logic chính): Chứa logic nghiệp vụ gọi đến nhiều Model
- Lớp Model Layer (truy cập dữ liệu, tương tác Database): Định nghĩa cấu trúc dữ liệu và thao tác cơ sở dữ liệu

Hệ thống backend được xây dựng theo kiến trúc phân lớp (Layered Architecture) trong đó mã nguồn được chia thành bốn tầng chính: Model, Service, Controller và Route. Mỗi tầng đảm nhận một vai trò riêng biệt nhằm đảm bảo phân tách, mở rộng và dễ bảo trì.

2.2.3. Công nghệ xây dựng

2.2.3.1. Frontend

- Ngôn ngữ lập trình



Ảnh 7. Ngôn ngữ lập trình Kotlin

Đây là ngôn ngữ lập trình hiện đại được Google khuyến nghị sử dụng cho Android Development. Kotlin sở hữu cú pháp ngắn gọn, rõ ràng và dễ đọc hơn Java, giúp giảm thiểu lỗi và rút ngắn thời gian phát triển. Một trong những ưu điểm nổi bật của Kotlin là cơ chế null safety, giúp hạn chế các lỗi thường gặp khi truy cập biến rỗng trong quá trình chạy chương trình. Bên cạnh đó, Kotlin hỗ trợ mạnh mẽ lập trình bất đồng bộ thông qua Coroutines, giúp xử lý tác vụ nền hiệu quả hơn và mang lại trải nghiệm mượt mà cho người dùng. Nhờ những đặc điểm này, việc sử dụng Kotlin giúp quá trình phát triển ứng dụng trở nên nhanh hơn, mã nguồn an toàn hơn và dễ bảo trì hơn.

– User Interface



Ảnh 8. Jetpack Compose

Jetpack Compose được sử dụng để xây dựng giao diện cho ứng dụng Android thứ nhất. Đây là công nghệ UI hiện đại của Google dựa trên mô hình declarative, cho phép lập trình viên mô tả giao diện bằng code một cách ngắn gọn và trực quan. Compose giúp tối giản cấu trúc dự án khi không còn phụ thuộc vào nhiều file XML riêng lẻ, đồng thời loại bỏ phần lớn boilerplate code so với cách tiếp cận truyền thống. Công nghệ này tích hợp tự nhiên với ViewModel, LiveData và StateFlow, giúp việc quản lý trạng thái dễ dàng và nhất quán. Với khả năng tùy biến linh hoạt, hiệu năng tốt và công cụ Preview hỗ trợ quan sát UI theo thời gian thực, Jetpack Compose phù hợp cho các màn hình cần mức độ tương tác cao và đòi hỏi tốc độ phát triển nhanh.

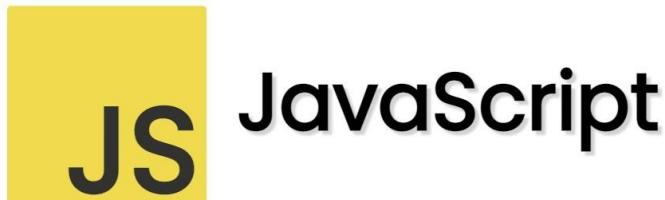


Ảnh 9. XML

XML Layout là hệ thống được sử dụng truyền thống trong ứng dụng Android để xây dựng giao diện. Cách tiếp cận này cho phép lập trình viên kiểm soát cấu trúc UI theo dạng phân cấp rõ ràng, phù hợp với các màn hình có bố cục ổn định hoặc ít thay đổi theo thời gian. XML vẫn là lựa chọn đáng tin cậy nhờ độ ổn định cao, tương thích tốt với các thư viện UI lâu đời của Android và dễ duy trì đối với các dự án đã có sẵn cơ sở nền tảng XML. Việc duy trì một ứng dụng sử dụng XML giúp đảm bảo sự ổn định của giao diện trong các tình huống phức tạp và hỗ trợ tốt cho những thành viên trong nhóm đã quen với mô hình xây dựng UI truyền thống.

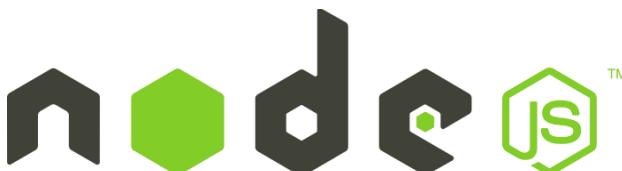
2.2.3.2. Backend

- Ngôn ngữ lập trình



Ảnh 10. Ngôn ngữ lập trình JavaScript

JavaScript là ngôn ngữ lập trình phổ biến và linh hoạt, đặc biệt phù hợp cho các ứng dụng web và hệ thống dịch vụ hướng sự kiện. JavaScript cho phép xây dựng các ứng dụng Backend có khả năng xử lý bất đồng bộ hiệu quả, đáp ứng tốt các yêu cầu về hiệu năng trong môi trường nhiều người dùng truy cập đồng thời.



Ảnh 11. Node.js

- Framework

NodeJS là môi trường chạy JavaScript phía server, được xây dựng trên V8 Engine của Google, cho phép sử dụng JavaScript để phát triển backend. Node hoạt động theo

mô hình event-driven và non-blocking I/O, tức là không chặn luồng xử lý và có khả năng xử lý nhiều request đồng thời một cách hiệu quả. Kiến trúc của Node dựa trên Event Loop, sử dụng một thread để quản lý các tác vụ bất đồng bộ, giúp tốc độ phản hồi nhanh và phù hợp cho các ứng dụng realtime. NodeJS có hệ sinh thái thư viện phong phú thông qua NPM, dễ mở rộng, dễ học vì dùng chung JavaScript cho cả frontend và backend.



Ảnh 12. ExpressJS framework

ExpressJS là một framework tối giản nhưng mạnh mẽ chạy trên NodeJS, được sử dụng phổ biến để xây dựng API và các ứng dụng web. Express cung cấp hệ thống routing rõ ràng cho các phương thức như GET, POST, PUT, DELETE; hỗ trợ middleware – nơi xử lý các chức năng như logging, xác thực, parse dữ liệu hoặc bắt lỗi; hỗ trợ phục vụ static files và tích hợp các template engines như EJS, Pug để render giao diện động. Nhờ cú pháp ngắn gọn, linh hoạt và dễ mở rộng, Express trở thành lựa chọn hàng đầu khi phát triển backend bằng Node.

2.2.3.3. CSDL



Ảnh 13. Firebase

Firebase là nền tảng phát triển ứng dụng do Google cung cấp, hỗ trợ toàn diện cho việc xây dựng backend mà không cần quản lý server (Backend-as-a-Service). Firebase cung cấp nhiều dịch vụ tích hợp sẵn như cơ sở dữ liệu thời gian thực, xác thực người dùng, lưu trữ tệp và hosting, giúp rút ngắn thời gian phát triển và đơn giản hóa hạ tầng. Với mô hình hoạt động trên nền tảng cloud, Firebase có khả năng mở rộng tự động, đảm bảo ứng dụng luôn hoạt động ổn định ngay cả khi lượng người dùng tăng cao.

2.2.3.4. AI



Ảnh 14. Ngôn ngữ lập trình Python

Python được sử dụng làm ngôn ngữ chính để xây dựng hệ thống AI nhờ cú pháp đơn giản và hệ sinh thái thư viện mạnh mẽ. Các mô hình trí tuệ nhân tạo được phát triển và huấn luyện bằng Python, sau đó triển khai thông qua FastAPI để cung cấp các API phục vụ suy luận (inference). FastAPI cho phép xử lý request nhanh, hỗ trợ bất đồng bộ và dễ dàng tích hợp mô hình AI vào hệ thống. Nhờ đó, hệ thống AI có thể hoạt động ổn định, mở rộng tốt và đáp ứng yêu cầu xử lý theo thời gian thực.



Ảnh 15. FastAPI

FastAPI là một framework web hiện đại của Python, dùng để xây dựng RESTful API với hiệu năng cao và dễ phát triển. Framework này tận dụng type hints của Python để tự động kiểm tra dữ liệu đầu vào/đầu ra, giúp giảm lỗi và tăng tính rõ ràng trong mã nguồn. FastAPI được xây dựng dựa trên Starlette (xử lý web, bất đồng bộ) và Pydantic (kiểm tra và xác thực dữ liệu). Nhờ đó, FastAPI hỗ trợ lập trình bất đồng bộ (async/await), cho phép xử lý nhiều yêu cầu đồng thời.



Ảnh 16. LangChain

LangChain là một framework hỗ trợ xây dựng ứng dụng sử dụng mô hình ngôn ngữ lớn (LLM), cho phép kết hợp LLM với dữ liệu, công cụ và các quy trình xử lý logic một cách linh hoạt. LangChain giúp tổ chức và điều phối các bước xử lý phức tạp thay vì chỉ gọi mô hình ngôn ngữ đơn lẻ.

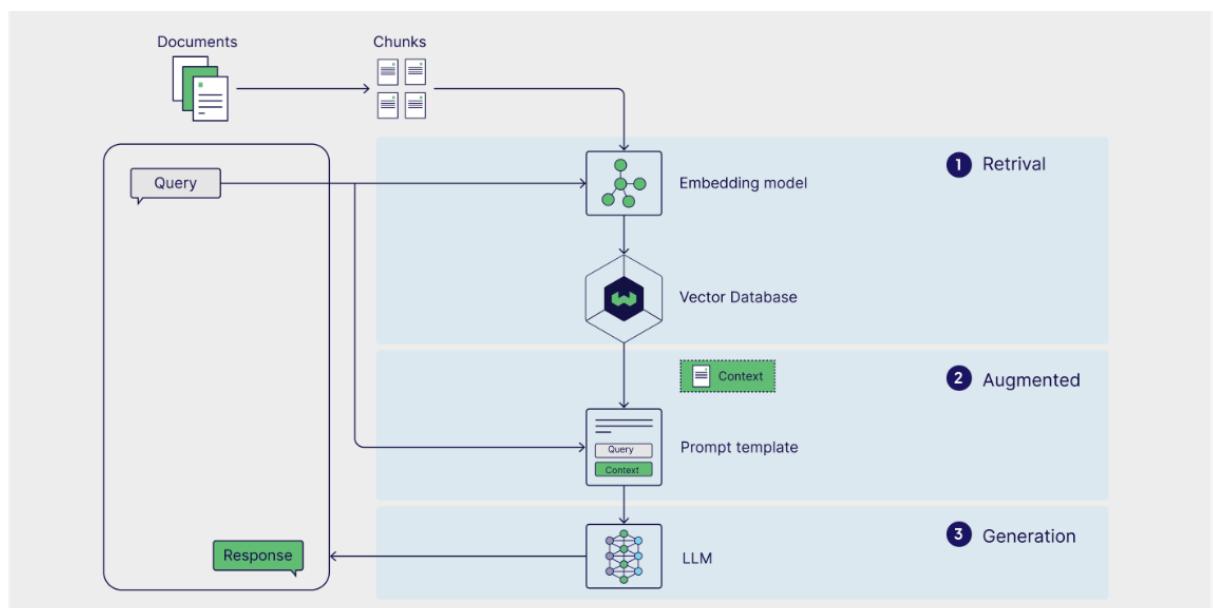


Ảnh 17. LangGraph

LangGraph là một framework mở rộng của LangChain, được thiết kế để xây dựng các luồng xử lý AI có trạng thái (stateful workflows) dưới dạng đồ thị. Thay vì xử lý tuyến tính, LangGraph cho phép mô hình ngôn ngữ hoạt động theo nhiều nhánh, vòng lặp và điều kiện rẽ nhánh, giúp kiểm soát logic xử lý phức tạp tốt hơn. LangGraph tổ chức ứng dụng AI thành các node (tác vụ) và edge (luồng chuyển tiếp), trong đó trạng thái được duy trì xuyên suốt quá trình xử lý.

2.2.3.5. ARAG (Agentic Retrieval Augmented Generation) kết hợp Hybrid Recommendation System

a. RAG

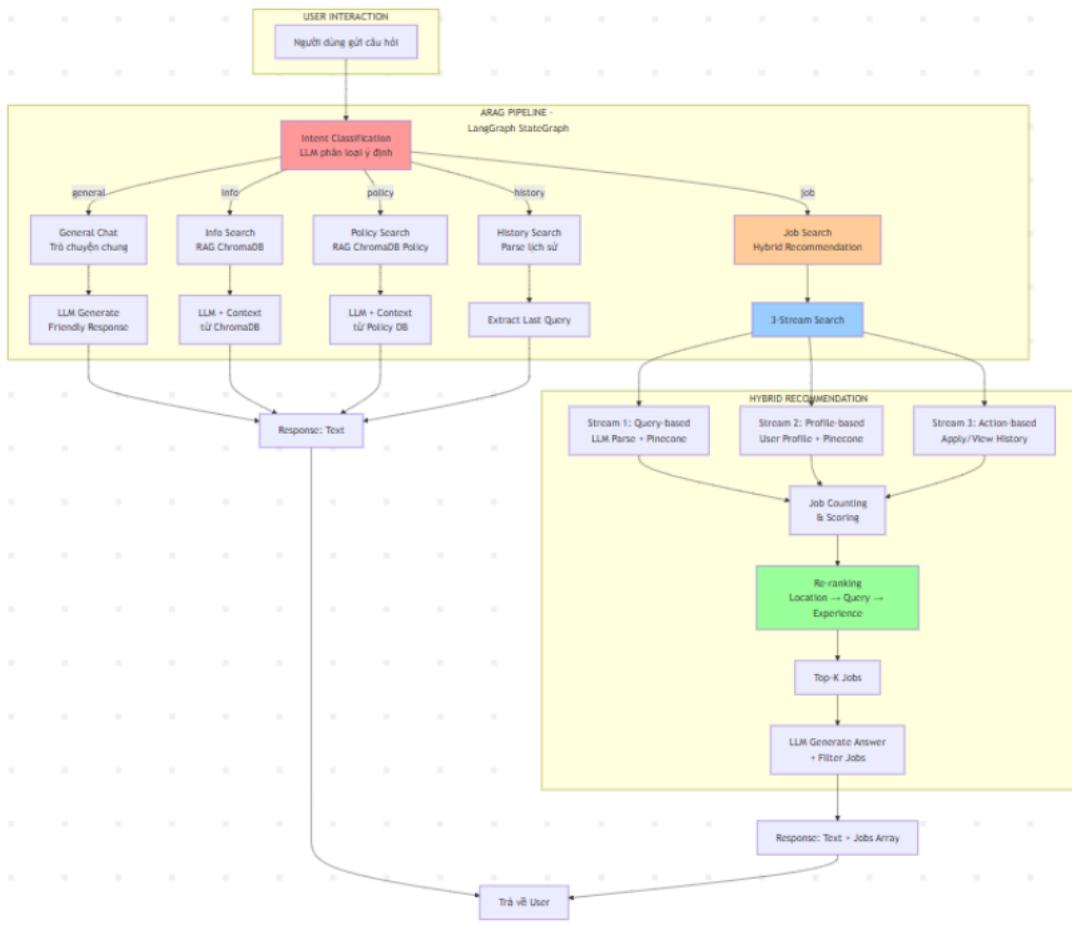


Ảnh 18. Hình minh họa RAG

Một khung công tác giúp tăng cường kiến thức tổng quát của mô hình LLM tạo sinh bằng cách cung cấp cho nó dữ liệu bổ sung có liên quan đến nhiệm vụ đang thực hiện, được truy xuất từ một nguồn dữ liệu bên ngoài. Các thành phần của RAG bao gồm: nguồn kiến thức bên ngoài, mẫu câu hỏi gợi ý và mô hình tạo sinh. Các thành phần này cho phép các ứng dụng được hỗ trợ bởi LLM tạo ra các phản hồi chính xác hơn bằng cách tận dụng dữ liệu cụ thể có giá trị cho từng nhiệm vụ.

Để tối ưu hóa quá trình kết nối giữa khách hàng và người cung cấp dịch vụ, hệ thống Helpo áp dụng phương pháp kết hợp giữa ARAG và hệ thống khuyến nghị kết hợp. Phương pháp này không chỉ cải thiện khả năng tư vấn thông qua chatbot mà còn nâng cao độ chính xác trong việc gợi ý công việc phù hợp cho nhân viên.

b. ARAG



Ảnh 19. Hình minh họa ARAG

Phương pháp mở rộng của RAG truyền thống, trong đó hệ thống không chỉ truy xuất thông tin tĩnh từ knowledge base mà còn chủ động lập kế hoạch, ra quyết định và thực hiện các hành động dựa trên ngữ cảnh hội thoại. Điểm khác biệt giữa RAG và ARAG nằm ở khả năng "agent" - khả năng tự quyết định luồng xử lý thay vì tuân theo một pipeline cố định.

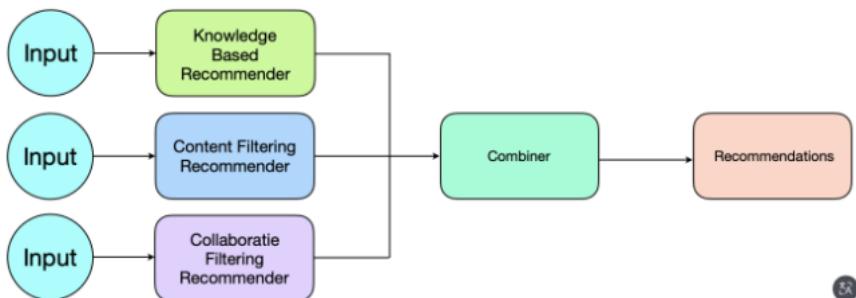
Trong hệ thống Helpo, ARAG được triển khai thông qua LangGraph, cho phép chatbot thực hiện các nhiệm vụ phức tạp theo các nhánh điều kiện:

- Intent Classification (Phân loại ý định): Khi người dùng gửi câu hỏi, hệ thống sử dụng LLM để phân loại ý định thành các nhóm chính:
 - + Truy vấn thông tin chung (general query): Hỏi về chính sách, hướng dẫn sử dụng
 - + Tìm kiếm công việc (job search): Nhân viên muốn tìm công việc phù hợp
 - + Tư vấn dịch vụ (service consultation): Khách hàng cần tư vấn về loại dịch vụ

- Retrieval Agent (Tác nhân truy xuất): Dựa trên ý định được xác định, hệ thống quyết định nguồn dữ liệu cần truy vấn:
 - + ChromaDB: Chứa embedding của chính sách, thông tin ứng dụng
 - + Pinecone: Chứa embedding của công việc, hồ sơ ứng viên
 - + Firestore: Dữ liệu nghiệp vụ thời gian thực (trạng thái công việc, đơn ứng tuyển)
- Decision Agent (Tác nhân ra quyết định): Sau khi thu thập thông tin, LLM tổng hợp và sinh phản hồi. Nếu thông tin không đủ, agent có thể:
 - + Yêu cầu người dùng cung cấp thêm thông tin
 - + Truy vấn bổ sung từ nguồn dữ liệu khác
 - + Gọi hệ thống khuyến nghị để đề xuất công việc cụ thể

Việc sử dụng ARAG giúp chatbot không chỉ trả lời câu hỏi mà còn chủ động hỗ trợ người dùng trong quá trình tìm kiếm và ứng tuyển công việc, tạo ra trải nghiệm tương tác thông minh và linh hoạt.

c. Hybrid Recommendation System (Hệ thống khuyến nghị kết hợp)



Ảnh 20. Sơ đồ Hybrid Recommandation System

Khác với các hệ thống khuyến nghị truyền thống (content-based, collaborative filtering, rule-based), hệ thống Helpo áp dụng *kiến trúc 3-Stream Search* kết hợp với *re-ranking* thông minh để tối ưu hóa độ chính xác và khả năng cá nhân hóa.

a. Quy trình tìm kiếm song song (The 3 Streams)

Hệ thống bắt đầu bằng việc kích hoạt đồng thời ba luồng dữ liệu độc lập để bao quát mọi khía cạnh nhu cầu của người dùng:

- Luồng 1 - Query-based (Dựa trên truy vấn): Groq LLM tiếp nhận và phân tích câu hỏi trực tiếp để trích xuất các điều kiện lọc, sau đó chuẩn hóa thành vector để thực hiện tìm kiếm trong Pinecone kết hợp với Metadata Filtering.
- Luồng 2 - Profile-based (Dựa trên hồ sơ): Hệ thống phân tích sâu profile và sở thích của nhân viên để xác định loại công việc và dịch vụ quan tâm nhất. Từ đó, một câu

hỏi tự nhiên được sinh ra tự động, chuyển đổi thành vector và truy vấn trong cơ sở dữ liệu để tìm ra các vị trí tương thích với năng lực cá nhân.

– Luồng 3 - Action-based (Dựa trên hành động): Dựa vào lịch sử tương tác (apply/view), hệ thống tạo mô tả chi tiết cho các công việc cũ, vector hóa chúng để tìm các job tương tự. Kết quả được xử lý qua bước đếm và sắp xếp theo tần suất xuất hiện để xác định xu hướng quan tâm thực tế của người dùng.

b. *Tổng hợp và Tối ưu hóa (Re-ranking)*

Sau khi thu thập kết quả từ cả ba luồng trên, hệ thống thực hiện bước Re-ranking để tinh lọc danh sách cuối cùng. Tại đây, một logic xếp hạng thông minh sẽ tính toán và chọn ra Top-K công việc tốt nhất dựa trên thứ tự ưu tiên khắt khe: đầu tiên là khoảng cách địa lý, kế đến là độ tương đồng của câu hỏi, và cuối cùng là sự phù hợp về kinh nghiệm làm việc. Kết quả cuối cùng trả về cho người dùng không chỉ đúng ý định tìm kiếm mà còn tối ưu hóa khả năng thực thi và cá nhân hóa sâu sắc.

2.2.3.6. Công cụ khác



Ảnh 21. Vercel

Vercel là một nền tảng triển khai ứng dụng hiện đại, hỗ trợ mạnh mẽ cho các ứng dụng JavaScript, đặc biệt là NodeJS và các API dạng serverless. Trong hệ thống, Vercel được sử dụng để triển khai backend viết bằng NodeJS. Việc sử dụng Vercel mang lại nhiều lợi ích như quá trình deploy đơn giản, tự động và không yêu cầu quản lý máy chủ truyền thống. Nhờ tích hợp CI/CD, mỗi khi cập nhật mã nguồn trên GitHub, Vercel sẽ tự động build và triển khai phiên bản mới, đảm bảo hệ thống luôn ở trạng thái mới nhất.



Ảnh 22. Pinecone

Pinecone là dịch vụ cơ sở dữ liệu vector chuyên dụng được thiết kế để lưu trữ và truy vấn các vector embedding. Trong đề tài, Pinecone được sử dụng để lưu các embedding của thông tin job (công việc). Embedding là dạng biểu diễn số hóa của dữ liệu văn bản, cho phép hệ thống hiểu ý nghĩa ngữ nghĩa của từng mô tả công việc.



Ảnh 23. Chroma

ChromaDB là một vector database mã nguồn mở, nhẹ và linh hoạt, thường được sử dụng trong các hệ thống RAG (Retrieval Augmented Generation) hoặc bài toán AI cần xử lý embedding cục bộ. Trong hệ thống, ChromaDB được sử dụng để lưu trữ các embedding của thông tin ứng viên hoặc dữ liệu ứng dụng, từ đó phục vụ việc truy vấn, so sánh và gợi ý dựa trên ngữ nghĩa.

Điểm mạnh của ChromaDB là dễ triển khai, chi phí thấp, và có thể hoạt động trực tiếp trong môi trường backend mà không cần hạ tầng cloud phức tạp. Khi kết hợp cùng mô hình AI để tạo embedding, ChromaDB giúp hệ thống tìm kiếm nhanh hơn, đặc biệt trong các tác vụ như matching ứng viên với công việc, lọc hồ sơ hoặc phân tích tương đồng nội dung. Với khả năng tối ưu truy vấn vector và độ trễ thấp, ChromaDB là lựa chọn phù hợp cho các thành phần AI chạy nội bộ.



Ảnh 24. Cloudinary

Cloudinary - dịch vụ lưu trữ đám mây đóng vai trò quan trọng trong việc quản lý và lưu trữ các tệp tài nguyên như hình ảnh, video hoặc file đính kèm của hệ thống. Thay vì lưu trữ trực tiếp trên máy chủ backend, việc đưa dữ liệu lên cloud giúp giảm tải cho hệ thống và tối ưu tốc độ truy cập. Các dịch vụ cloud thường cung cấp hạ tầng CDN (Content Delivery Network), giúp người dùng truy cập ảnh nhanh hơn, ổn định hơn từ bất kỳ vị trí địa lý nào.



Ảnh 25. SePay

SePay là dịch vụ hỗ trợ kiểm tra tiền vào/ra từ tài khoản ngân hàng một cách tự động. Trong hệ thống, SePay được tích hợp thông qua API để kiểm tra trạng thái giao

dịch chuyển khoản, giúp tự động xác nhận thanh toán của người dùng. Khả năng theo dõi giao dịch theo thời gian thực giúp giảm thiểu rủi ro sai sót khi xử lý thanh toán thủ công và tăng tốc độ phản hồi của hệ thống. Việc sử dụng SePay mang lại lợi ích lớn trong các ứng dụng có yêu cầu xác thực giao dịch, ví dụ như nạp tiền, thanh toán dịch vụ hoặc mua gói thành viên. SePay đóng vai trò như một cổng thanh toán đơn giản nhưng hiệu quả, không cần tích hợp các hệ thống phức tạp mà vẫn đảm bảo tính chính xác và an toàn cho người dùng



Ảnh 26. Gemini

Gemini là một mô hình AI/LLM tiên tiến dùng để xử lý ngôn ngữ tự nhiên, tạo văn bản và sinh embedding. Trong hệ thống, Gemini được sử dụng để phân tích nội dung, gợi ý, tóm tắt và đặc biệt là tạo embedding phục vụ các tác vụ liên quan đến Pinecone hoặc ChromaDB. Embedding tạo bởi Gemini có chất lượng cao, giúp hệ thống hiểu và so sánh nội dung dựa trên ý nghĩa chứ không chỉ dựa trên từ khóa. Ngoài việc hỗ trợ truy vấn ngữ nghĩa, Gemini còn giúp nâng cao các tính năng AI khác như tự động phân tích dữ liệu, sinh phản hồi thông minh hoặc hỗ trợ người dùng trong quá trình tìm kiếm. Việc tích hợp Gemini giúp hệ thống trở nên thông minh hơn, mang lại trải nghiệm hiện đại và đáp ứng xu hướng ứng dụng AI vào bài toán thực tế.



Ảnh 27. Groq

Groq là nền tảng cung cấp Large Language Model (LLM) với tốc độ xử lý nhanh nhất hiện nay nhờ chip LPU (Language Processing Unit) chuyên dụng. Trong project này, Groq được sử dụng để chạy mô hình **Llama 3.3 70B Versatile** - một mô hình ngôn ngữ lớn được phát triển bởi Meta, có khả năng trích xuất câu hỏi người dùng, phân loại nhóm chủ đề câu hỏi, sinh tạo câu trả lời tự nhiên



Hugging Face

Ảnh 28. Hugging Face Spaces

Hugging Face Spaces là một nền tảng cho phép người dùng triển khai ứng dụng AI trực tiếp trên web một cách đơn giản và miễn phí. Spaces hỗ trợ các framework phổ biến như Gradio, Streamlit, Static HTML, hoặc Docker, giúp việc xây dựng giao diện tương tác cho mô hình AI trở nên nhanh chóng mà không cần quản lý máy chủ.

2.3. Phân tích hệ thống

2.3.1. Tổng quan nghiệp vụ

Các tác nhân xuất hiện trong hệ thống bao gồm: *khách hàng*, *nhan viên* và *quản trị viên*. Hệ thống được xây dựng nhằm mục đích hỗ trợ kết nối nhu cầu về dịch vụ tại gia giữa khách hàng với các người cung cấp được xét duyệt từ quản trị hệ thống. Do đó để hiểu rõ hơn về hệ thống, hãy quan sát bảng từ khóa dưới đây:

TÙ LOẠI	TÙ KHÓA	GIẢI THÍCH
DANH TỪ	Dịch vụ tại gia	Các công việc được trong phạm vi (dọn dẹp nhà cửa, chăm sóc sức khỏe, bảo trì máy lạnh hoặc máy giặt) được hệ thống giới hạn và quy định rõ trong chính sách.
	Đơn dịch vụ	Kết quả của sự kiện khách đặt lịch sử dụng dịch vụ mong muốn, bao gồm: thông tin người đặt, loại dịch vụ, chi tiết công việc phải làm, địa điểm làm việc, lương, thời gian thực hiện.
	Đơn ứng tuyển	Kết quả của hành động ứng tuyển của nhân viên, chứa các thông tin về cơ bản gồm: tên, giới tính, điểm đánh giá.
ĐỘNG TỪ	Đăng tuyển	Hành động khách hàng tạo và đăng một đơn dịch vụ mới lên hệ thống, bao gồm thông tin công việc, thời gian, mức lương và địa điểm.
	Ứng tuyển	Hành động nhân viên gửi yêu cầu tham gia thực hiện một công việc đã được đăng, kèm theo hồ sơ cá nhân và thông tin liên quan.

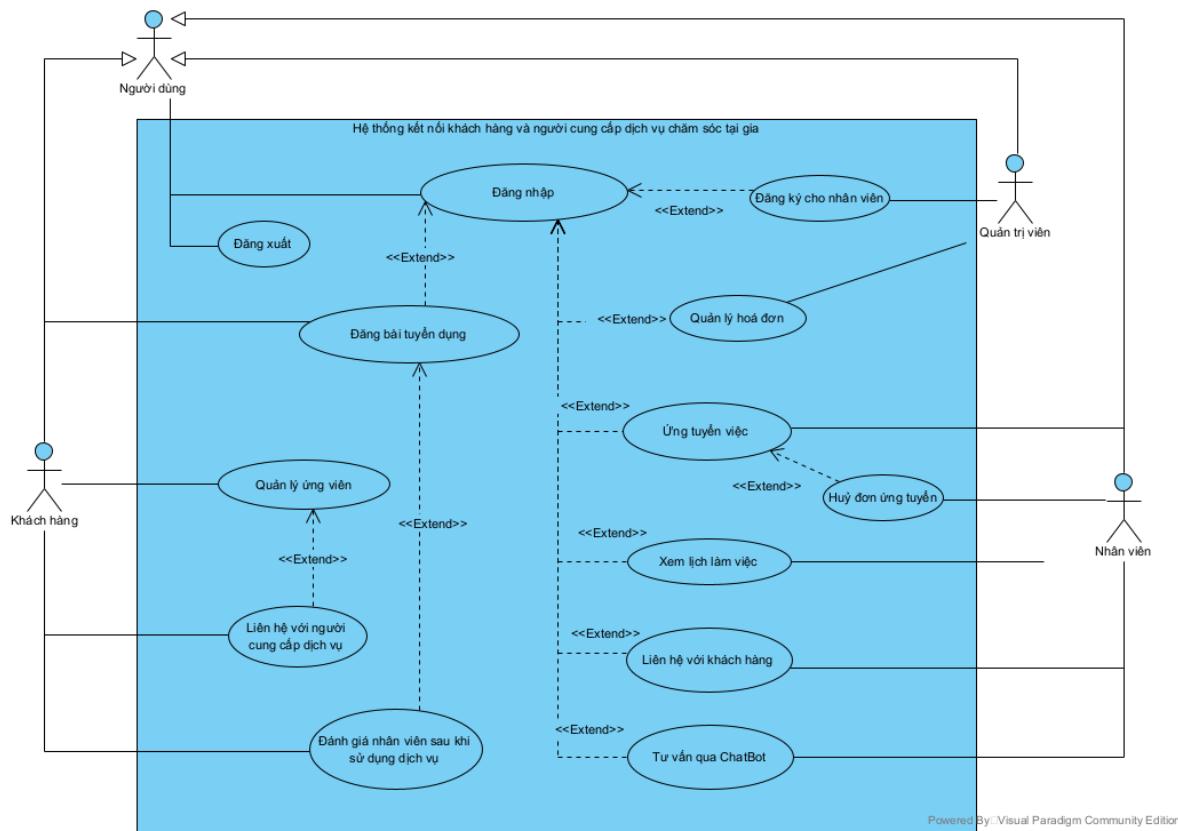
	Phê duyệt ứng viên	Hành động khách hàng xem xét và chấp nhận một nhân viên phù hợp trong danh sách các đơn ứng tuyển vào công việc của mình.
	Từ chối ứng viên	Hành động khách hàng loại bỏ những ứng viên không phù hợp, cập nhật trạng thái ứng tuyển tương ứng trong hệ thống.
	Huỷ ứng tuyển	Hành động nhân viên rút lại yêu cầu ứng tuyển của mình trước khi khách hàng đưa ra quyết định phê duyệt.
	Hoàn thành công việc	Hành động đánh dấu công việc đã được thực hiện xong bởi nhân viên và xác nhận bởi khách hàng, chuyển đơn dịch vụ sang trạng thái “Hoàn thành”.
	Đánh giá nhân viên	Hành động khách hàng cho điểm và nhận xét về hiệu quả làm việc của nhân viên sau khi công việc kết thúc; kết quả được cập nhật vào hồ sơ nhân viên.

Bảng 2. Tổng quan nghiệp vụ

2.3.2. Đặc tả Usecase

1. Usecase tổng quát

Đồ án tốt nghiệp Đại học



Ảnh 29. Usecase tổng quan

2. Các chức năng chính

– Chức năng của khách hàng

Use case name	Đăng ký
Actor	Khách hàng
Pre-condition	
Post condition	Khách hàng có tài khoản để sử dụng dịch vụ
Flow	<ol style="list-style-type: none"> Tại trang chủ, khách hàng A chọn “Đăng nhập/ Tạo tài khoản” Hệ thống hiển thị cửa sổ lựa chọn gồm 2 nút “Đăng nhập” và “Đăng ký”

Đồ án tốt nghiệp Đại học

	<p>3. Khách hàng A chọn “Đăng ký”</p> <p>4. Hệ thống hiện thị giao diện đăng ký gồm các ô nhập thông tin gồm: username, password, confirmpassword và 1 nút “Đăng ký”</p> <p>5. Khách hàng nhập username là abc@gmail.com, password là 123456, confirmpassword là 123456 và chọn “Đăng ký”</p> <p>6. Hệ thống hiển thị giao diện trang chủ với tên người dùng là abc</p>
Exception	<p>6a. Đã có tài khoản trong hệ thống, hệ thống báo tài khoản đã tồn tại</p>

Bảng 3. Đặc tả usecase đăng ký của khách hàng

Use case name	Đăng nhập
Actor	Khách hàng
Pre-condition	Khách hàng đã có tài khoản
Post condition	Khách hàng đã đăng nhập thành công
Flow	<p>1. Tại trang chủ, khách hàng A chọn “Đăng nhập/ Tạo tài khoản”</p> <p>2. Hệ thống hiển thị cửa sổ lựa chọn gồm 2 nút “Đăng nhập” và “Đăng ký”</p> <p>3. Khách hàng A chọn “Đăng nhập”</p> <p>4. Hệ thống hiện thị giao diện đăng ký gồm các ô nhập thông tin gồm: username, password và 1 nút “Đăng nhập”</p> <p>5. Khách hàng nhập username là abc@gmail.com, password là 123456 và chọn “Đăng nhập”</p> <p>6. Hệ thống hiển thị giao diện trang chủ với tên người dùng là abc</p>

Đồ án tốt nghiệp Đại học

Exception	6a. Sai thông tin đăng nhập, hệ thống thông báo sai tài khoản hoặc mật khẩu
-----------	-----------------------------------------------------------------------------

Bảng 4. Đặc tả usecase đăng nhập của khách hàng

Use case name	Đăng bài tìm người thực hiện dịch vụ															
Actor	Khách hàng															
Pre-condition	Khách hàng đã đăng nhập thành công															
Post condition	Đăng bài thành công lên hệ thống															
Flow	<p>1. Sau khi đăng nhập thành công, khách hàng A chọn dịch vụ “Dọn dẹp”.</p> <p>2. Hệ thống hiện địa chỉ cần tuyển, danh sách thời gian kèm diện tích phòng, dịch vụ thêm, 1 nút “Tiếp tục”.</p> <table border="1" style="margin-top: 20px;"> <thead> <tr> <th>STT</th> <th>Thời lượng</th> <th>Giá</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>2h hoặc 20m2</td> <td>200.000đ</td> </tr> <tr> <td>2</td> <td>3h hoặc 30m2</td> <td>300.000đ</td> </tr> <tr> <td>3</td> <td>4h hoặc 40m2</td> <td>400.000đ</td> </tr> </tbody> </table> <table border="1" style="margin-top: 20px;"> <thead> <tr> <th>STT</th> <th>Dịch vụ thêm</th> <th>Giá</th> </tr> </thead> </table>	STT	Thời lượng	Giá	1	2h hoặc 20m2	200.000đ	2	3h hoặc 30m2	300.000đ	3	4h hoặc 40m2	400.000đ	STT	Dịch vụ thêm	Giá
STT	Thời lượng	Giá														
1	2h hoặc 20m2	200.000đ														
2	3h hoặc 30m2	300.000đ														
3	4h hoặc 40m2	400.000đ														
STT	Dịch vụ thêm	Giá														

Đồ án tốt nghiệp Đại học

	<table border="1"> <tr> <td>1</td><td>Ủi đồ</td><td>50.000đ</td></tr> <tr> <td>2</td><td>Nấu Ăn</td><td>50.000đ</td></tr> </table>	1	Ủi đồ	50.000đ	2	Nấu Ăn	50.000đ
1	Ủi đồ	50.000đ					
2	Nấu Ăn	50.000đ					
Exception	<p>3. Khách hàng A chọn thời lượng 2h, dịch vụ ủi đồ và chọn nút “Tiếp theo”.</p> <p>4. Hệ thống hiển thị giao diện chọn ngày, thời gian và 1 nút “Tiếp theo”.</p> <p>5. Khách hàng A chọn danh sách ngày 20/11/2025, 22/11/2025, 25/11/2025, chọn giờ bắt đầu là 15:00 và chọn “Tiếp theo”.</p> <p>6. Hệ thống hiển thị giao diện hóa đơn và 1 nút “Đăng bài”.</p> <p>7. Khách hàng A chọn “Đăng bài”.</p> <p>8. Hệ thống hiển thị giao diện mã qr gồm 1 nút đóng giao diện và 1 nút “Lưu mã QR về máy” đồng thời hiển thị thông báo “Đăng bài thành công”</p> <p>9. Khách hàng A chọn đóng giao diện</p> <p>10. Hệ thống quay về giao diện chính</p> <p>10. Không lưu mã QR về máy để thanh toán</p> <p>10.1. Khách hàng A chọn “Hoạt động”</p> <p>10.2. Hệ thống hiện thị giao diện danh sách bài đăng công việc</p> <p>10.3 Khách hàng A chọn bài đăng mới nhất với trạng thái “Chưa thanh toán”</p> <p>10.4. Hệ thống hiện thị giao diện mã QR gồm mã QR, 1 nút đóng giao diện, 1 nút “Lưu mã QR về máy”</p> <p>10.5. Khách hàng chọn “Lưu mã QR về máy” và thực hiện thanh toán</p>						

Bảng 5. Đặc tả usecase đăng bài tìm người thực hiện dịch vụ

Use case name	Quản lý ứng viên
---------------	------------------

Đồ án tốt nghiệp Đại học

Actor	Khách hàng																		
Pre-condition	Khách hàng đã đăng nhập thành công																		
Post condition	Xem được ứng viên và lựa chọn ứng viên phù hợp																		
Flow	<p>1. Sau khi đăng nhập thành công, khách hàng A chọn “Hoạt động”.</p> <p>2. Hệ thống hiển thị danh sách bài đăng của khách hàng A:</p> <table border="1"> <thead> <tr> <th>STT</th> <th>Tên dịch vụ</th> <th>Giá</th> <th>Địa chỉ</th> <th>Trạng thái</th> <th>Mô tả</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Dọn dẹp</td> <td>120.000VNĐ</td> <td>Hà Nội</td> <td>Đang tuyển</td> <td>Dọn dẹp nhà 40m2</td> </tr> <tr> <td>2</td> <td>Chăm sóc</td> <td>250.000VND</td> <td>HCM</td> <td>Đang tuyển</td> <td>Người già: 3</td> </tr> </tbody> </table> <p>3. Khách hàng A chọn dịch vụ dọn dẹp.</p> <p>4. Hệ thống hiển thị thông tin bài đăng.</p> <p>5. Khách hàng A chọn “Ứng viên”.</p> <p>6. Hệ thống hiển thị danh sách ứng viên.</p> <p>7. Khách hàng A chọn “Chấp nhận” hoặc “Từ chối”.</p> <p>8. Hệ thống hiển thị thông báo thành công.</p>	STT	Tên dịch vụ	Giá	Địa chỉ	Trạng thái	Mô tả	1	Dọn dẹp	120.000VNĐ	Hà Nội	Đang tuyển	Dọn dẹp nhà 40m2	2	Chăm sóc	250.000VND	HCM	Đang tuyển	Người già: 3
STT	Tên dịch vụ	Giá	Địa chỉ	Trạng thái	Mô tả														
1	Dọn dẹp	120.000VNĐ	Hà Nội	Đang tuyển	Dọn dẹp nhà 40m2														
2	Chăm sóc	250.000VND	HCM	Đang tuyển	Người già: 3														
Exception	6a. Không có ai ứng tuyển vào, hệ thống hiển thị “Không có ứng viên”																		

Bảng 6. Đặc tả usecase đăng bài tìm người thực hiện dịch vụ

Đồ án tốt nghiệp Đại học

Use case name	Đánh giá nhân viên
Actor	Khách hàng
Pre-condition	Khách hàng đã đăng nhập thành công
Post condition	Đánh giá người cung cấp dịch vụ thành công
Flow	<ol style="list-style-type: none"> 1. Sau khi đăng nhập thành công, Khách hàng A vào trang “Hoạt động”. 2. Hệ thống hiển thị danh sách những bài đăng mà khách hàng A đã đăng. 3. Khách hàng A chọn bài đăng đã hoàn thành công việc. 4. Hệ thống hiển thị chi tiết thông tin công việc. 5. Khách hàng A chọn “Người ứng tuyển”. 6. Hệ thống hiển thị danh sách người ứng tuyển. 7. Khách hàng A bình luận, đánh giá số sao, và chọn “Bình luận”. 8. Hệ thống hiển thị thông báo thành công.
Exception	<p>2a. Không có bài đăng nào, hệ thống hiển thị giao diện “Không có bài đăng nào”</p>

Bảng 7. Đặc tả usecase đánh giá nhân viên

3. Chức năng của nhân viên

Use case name	Đăng nhập
Actor	Nhân viên
Pre-condition	Nhân viên được đăng ký tài khoản bởi Quản trị viên

Đồ án tốt nghiệp Đại học

Post condition	Nhân viên đăng nhập thành công và giao diện chuyển về “Trang chủ”
Flow	<ol style="list-style-type: none"> 1. Tại giao diện đăng nhập, nhân viên nhập thông tin tài khoản mật khẩu, sau đó click button “Đăng nhập” 2. Hệ thống thông báo thành công, hiển thị giao diện trang chủ
Exception	<ol style="list-style-type: none"> 2a. Sai thông tin đăng nhập, hệ thống thông báo “Sai thông tin đăng nhập”

Bảng 8. Đặc tả usecase đăng nhập của nhân viên

Use case name	Nhân viên ứng tuyển công việc
Actor	Nhân viên
Pre-condition	Nhân viên đã đăng nhập thành công hệ thống
Post condition	Nhân viên ứng tuyển thành công một công việc cụ thể đã chọn
Flow	<ol style="list-style-type: none"> 1. Sau khi đăng nhập thành công, nhân viên chọn một danh mục cụ thể (dọn dẹp/ chăm sóc sức khoẻ/ bảo trì) 2. Hệ thống hiển thị danh sách công việc đang tuyển theo danh mục. 3. Nhân viên chọn một việc cụ thể 4. Hệ thống hiển thị thông tin chi tiết công việc, bao gồm: <ul style="list-style-type: none"> - Thông tin về khách hàng - Thời gian, địa điểm làm việc - Mức lương - Phạm vi công việc 5. Nhân viên chọn button “Ứng tuyển” 6. Hệ thống hiển thị thông báo thành công
Exception	<ol style="list-style-type: none"> 2a. Nếu chưa có công việc nào trong danh mục, hệ thống hiện thị “Không có công việc nào tồn tại” 6a. Hệ thống hiển thị thông báo lỗi nếu như có lỗi xảy ra

Bảng 9. Đặc tả usecase nhân viên ứng tuyển công việc

Đồ án tốt nghiệp Đại học

Use case name	Nhân viên huỷ ứng tuyển công việc
Actor	Nhân viên
Pre-condition	Nhân viên đã đăng nhập thành công hệ thống, đã từng ứng tuyển một công việc cụ thể nhưng công việc đó chưa đến thời gian thực thi
Post condition	Nhân viên huỷ ứng tuyển thành công
Flow	<ol style="list-style-type: none"> 1. Sau khi đăng nhập thành công, nhân viên xem chi tiết một công việc đã ứng tuyển 2. Hệ thống hiển thị chi tiết công việc đã chọn 3. Nhân viên chọn button “Huỷ ứng tuyển” 4. Hệ thống hiển thị thông báo cảnh báo “Có xác nhận huỷ ứng tuyển” 5. Nhân viên chọn “có” 6. Hệ thống hiển thị thông báo huỷ thành công
Exception	<ol style="list-style-type: none"> 1a. Nếu nhân viên chọn công việc chưa từng ứng tuyển trước đó, hệ thống không có button “Huỷ ứng tuyển” 6a. Nếu có lỗi xảy ra, hệ thống hiển thị thông báo lỗi

Bảng 10. Đặc tả usecase nhân viên huỷ ứng tuyển công việc

Use case name	Nhân viên xem lịch làm việc theo ngày
Actor	Nhân viên
Pre-condition	Nhân viên đã đăng nhập hệ thống thành công
Post condition	Hệ thống hiển thị danh sách công việc theo ngày đã chọn
Flow	<ol style="list-style-type: none"> 1. Nhân viên chọn “Xem lịch”. 2. Hệ thống hiển thị danh sách công việc trong ngày hiện tại. 3. Nhân viên chọn ngày cụ thể khác. 4. Hệ thống hiển thị danh sách công việc trong ngày đã chọn.

Exception	<p>2a. Nếu trong ngày hiện tại không có công việc, hệ thống hiển thị thông báo “Không có công việc”.</p> <p>4a. Nếu trong ngày hiện tại không có công việc, hệ thống hiển thị thông báo “Không có công việc”.</p>
-----------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

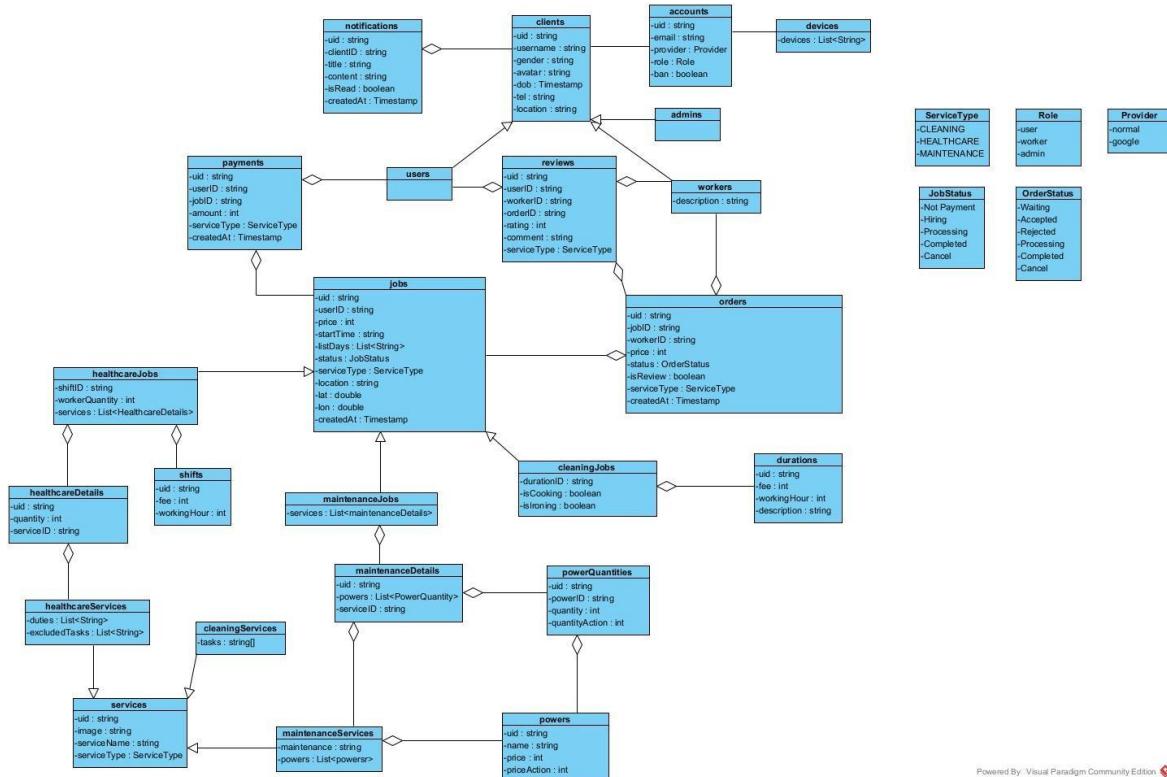
Bảng 11. Đặc tả usecase nhân viên xem lịch làm việc theo ngày chọn

Use case name	Nhân viên được tư vấn cùng chatbot
Actor	Nhân viên
Pre-condition	Nhân viên đã đăng nhập hệ thống thành công
Post condition	Hệ thống hiển thị danh sách công việc gợi ý hoặc câu trả lời phù hợp dựa trên nội dung mà khách hàng trao đổi với chatbot
Flow	<ol style="list-style-type: none"> 1. Khách hàng chọn “Tư vấn công việc” trong ứng dụng 2. Hệ thống mở giao diện chatbot và hiển thị lời chào 3. Nhân viên nhập yêu cầu, ví dụ: <ul style="list-style-type: none"> - “Tôi cần người dọn nhà ngày mai”. - “Tôi muốn tìm người chăm trẻ 3 tiếng”. - “Gợi ý giúp tôi dịch vụ phù hợp với công việc nhà”. 4. Chatbot phân tích nội dung nhập vào, xác định các thông tin quan trọng (loại dịch vụ, thời gian, địa điểm, hồ sơ của nhân viên đó). 5. Hệ thống xử lý và trả về gợi ý, ví dụ: <ul style="list-style-type: none"> - Gợi ý các gói dịch vụ phù hợp. - Gợi ý danh sách nhân viên khả dụng. - Hỏi thêm thông tin nếu khách hàng cung cấp chưa đủ. 6. Lặp lại từ bước 3.

Exception	5a. Chatbot không hiểu yêu cầu của nhân viên đang được tư vấn. 5b. Chatbot không tìm thấy dịch vụ phù hợp với yêu cầu của nhân viên.
-----------	-----------------------------------------------------------------------------------------------------------------------------------------

Bảng 12. Đặc tả usecase nhân viên được tư vấn bởi chatbot

2.3.3. Biểu đồ lớp



Powered By: Visual Paradigm Community Edition

Ảnh 30. Biểu đồ lớp phân tích

a. Lớp thực thể

STT	TÊN LỚP	THUỘC TÍNH	MÔ TẢ
1	accounts	uid, email, provider, role, ban	Thông tin đăng nhập và quyền hạn của người dùng trong hệ thống
2	clients	uid, username, gender, avatar, dob, tel, location	Thông tin người dùng chung cho cả client, worker và admin

Đồ án tốt nghiệp Đại học

3	users	uid, username, gender, avatar, dob, tel, location	Thông tin người dùng có nhu cầu đăng bài tuyển dụng dịch vụ (dọn dẹp, chăm sóc, bảo trì).
4	workers	uid, username, gender, avatar, dob, tel, location, description	Thông tin người cung cấp dịch vụ; lưu thông tin thanh toán và mô tả kỹ năng
5	admins	uid, username, gender, avatar, dob, tel, location	Thông tin quản trị viên
6	jobs	uid, userID, price, startTime, listDays, status, serviceType, location, lat, lon, createdAt	Thông tin chi tiết về loại dịch vụ, giá, và thời gian làm việc của công việc
7	cleaningJobs	uid, userID, price, startTime, listDays, status, serviceType, location, lat, lon, createdAt, durationID, isCooking, isIroning	Chi tiết công việc thuộc nhóm dọn dẹp nhà cửa
8	healthcareJobs	uid, userID, price, startTime, listDays, status, serviceType, location, lat, lon, createdAt, shiftID, workerQuantity, services	Thông tin công việc thuộc nhóm chăm sóc sức khỏe
9	healthcareDetails	uid, quantity, serviceID	Chi tiết người được chăm sóc hoặc dịch vụ y tế liên quan
10	maintenanceJobs	uid, userID, price, startTime, listDays, status, serviceType, location, lat, lon, createdAt, services	Thông tin công việc nhóm bảo trì bảo trì

Đồ án tốt nghiệp Đại học

11	maintenanceDetails	uid, powers, serviceID	Chi tiết công việc bảo trì, bao gồm danh sách thiết bị và thông số
12	powerQuantities	uid, powerID, quantity, quantityAction	Số lượng thiết bị và hành động bảo trì tương ứng
13	services	uid, image, serviceName, serviceType	Loại dịch vụ chính mà hệ thống hỗ trợ (Cleaning, Healthcare, Maintenance).
14	cleaningServices	uid, image, serviceName, serviceType, tasks	Thông tin loại dịch vụ “Dọn dẹp vệ sinh”
15	healthcareServices	uid, image, serviceName, serviceType, duties, excludedTasks	Thông tin loại dịch vụ “Chăm sóc sức khỏe”
16	maintenanceServices	uid, userID, price, startTime, listDays, status, serviceType, location, lat, lon, createdAt, services	Thông tin loại dịch vụ “Bảo trì thiết bị”
17	powers	uid, name, price, priceAction	Thông tin loại thiết bị cần bảo trì cùng giá dịch vụ
18	orders	uid, jobID, workerID, price, status, isReview, serviceType, createdAt	Thông tin đơn ứng tuyển hoặc đơn nhận việc giữa worker và client
19	reviews	uid, userID, workerID, orderID, rating, comment, serviceType	Thông tin đánh giá, nhận xét và điểm số của client cho worker sau khi hoàn thành công việc

Đồ án tốt nghiệp Đại học

20	notifications	uid, clientID, title, content, idRead, createdAt	Thông tin thông báo về cho người dùng
21	payments	uid, jobID, userID, amount, serviceType, createdAt	Thông tin thanh toán hóa đơn khách hàng
22	shifts	uid, fee, workingHour	Thông tin ca làm việc dành cho Healthcare
23	durations	uid, fee, workingHour, description	Thông tin ca làm việc dành cho Cleaning
24	devices	devices	Lưu trữ fcmToken thiết bị của người dùng

Bảng 13. Mô tả thuộc tính của lớp thực thể

b. Mối quan hệ giữa các lớp thực thể

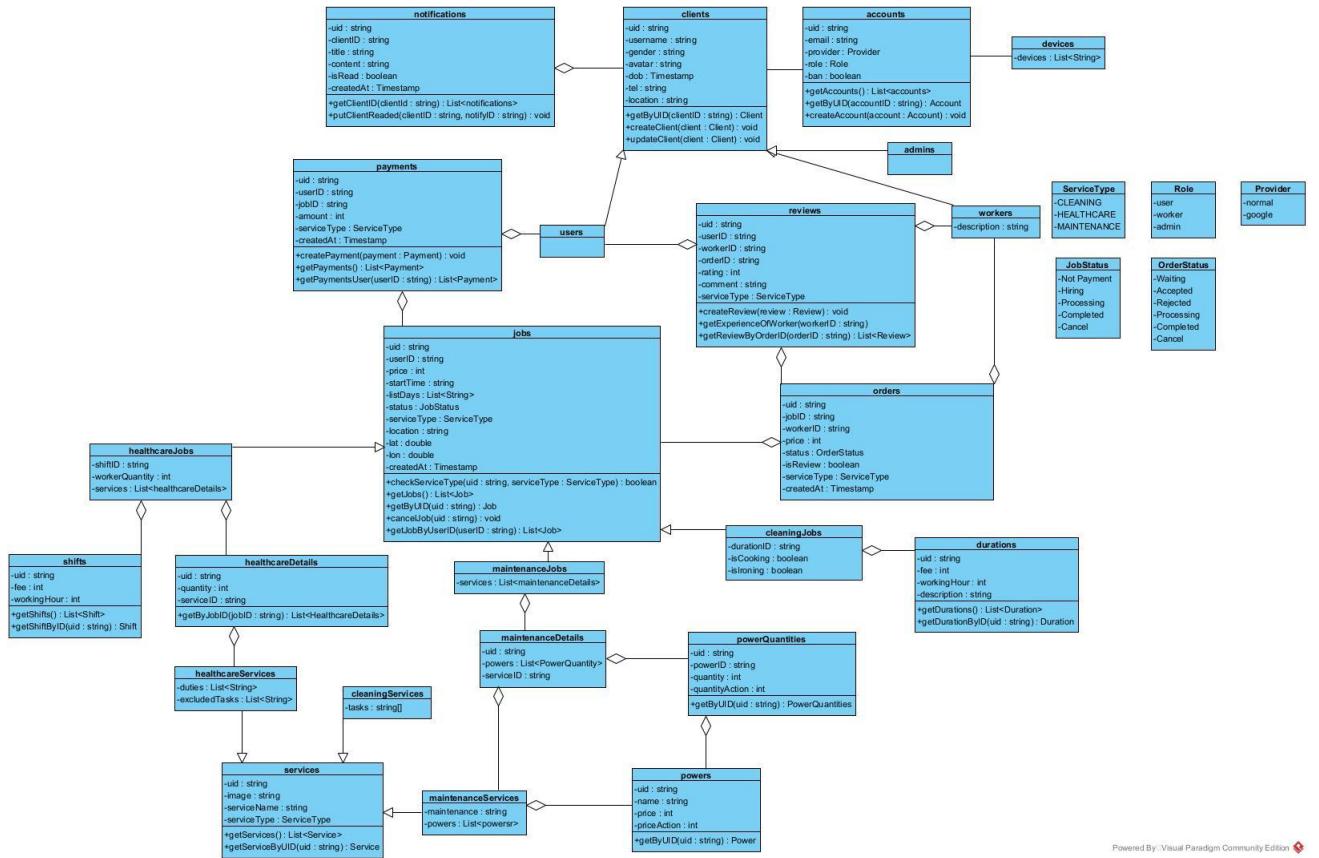
MỐI QUAN HỆ	KIỂU	GHI CHÚ
Client - Account	1 - 1	Mỗi người dùng có một tài khoản đăng nhập
Client - User/Worker/Admin	kết thừa	Ba vai trò người dùng
User - Job	1 - n	1 người dùng có thể đăng ký nhiều job
Job - Order	1 - n	Mỗi công việc có nhiều đơn ứng tuyển
Worker - Order	1 - n	1 Nhân viên có nhiều order
Job – CleaningJob/ HealthcareJob/ MaintenanceJob	kết thừa	3 loại danh mục công việc
MaintenanceDetails – Power	1 - n	Một lần bảo trì sẽ bảo trì nhiều thiết bị

Worker – Review	1 - n	Nhân viên nhận được nhiều đánh giá
Client – Review	1- n	Khách hàng có thể tạo nhiều đánh giá cho Nhân viên
Client – Notification	1 -n	Mỗi người dùng nhận nhiều thông báo
Job – Payment	1–1	Mỗi công việc có một giao dịch
User – Payment	1–n	Mỗi người dùng tạo nhiều giao dịch
Service - CleaningService/ HealthcareService/ MaintenanceService	kết thừa	3 loại dịch vụ của ứng dụng

Bảng 14. Mối quan hệ của các lớp thực thể

2.4. Thiết kế hệ thống

2.4.1. Biểu đồ lớp thiết kế



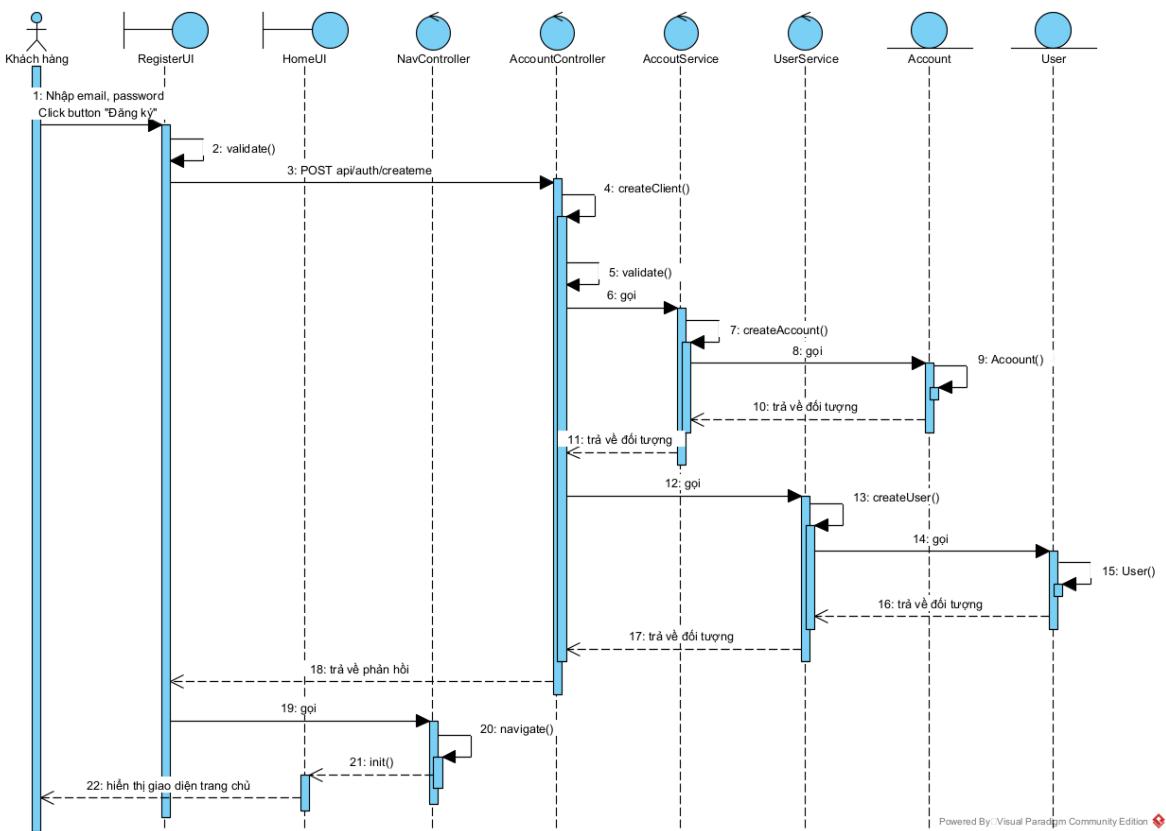
Ảnh 31. Biểu đồ lớp thiết kế

2.4.2. Biểu đồ tuần tự cho các tương tác quan trọng

1. Khách hàng

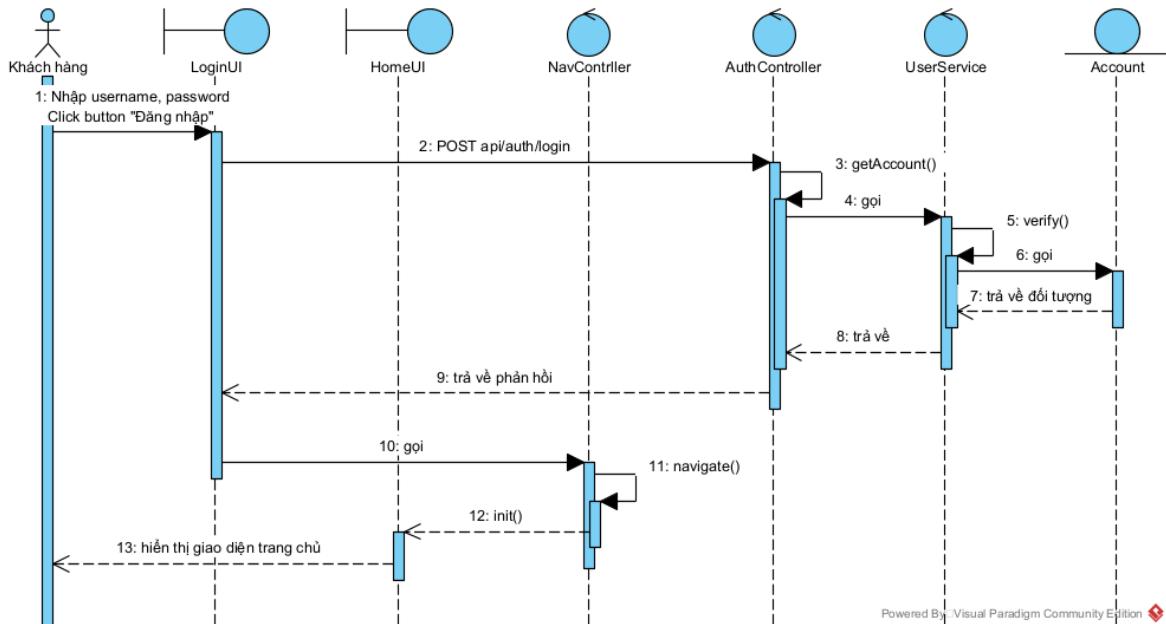
- Đăng ký

Đồ án tốt nghiệp Đại học



Ảnh 32. Biểu đồ tuần tự chức năng đăng ký cho nhân viên

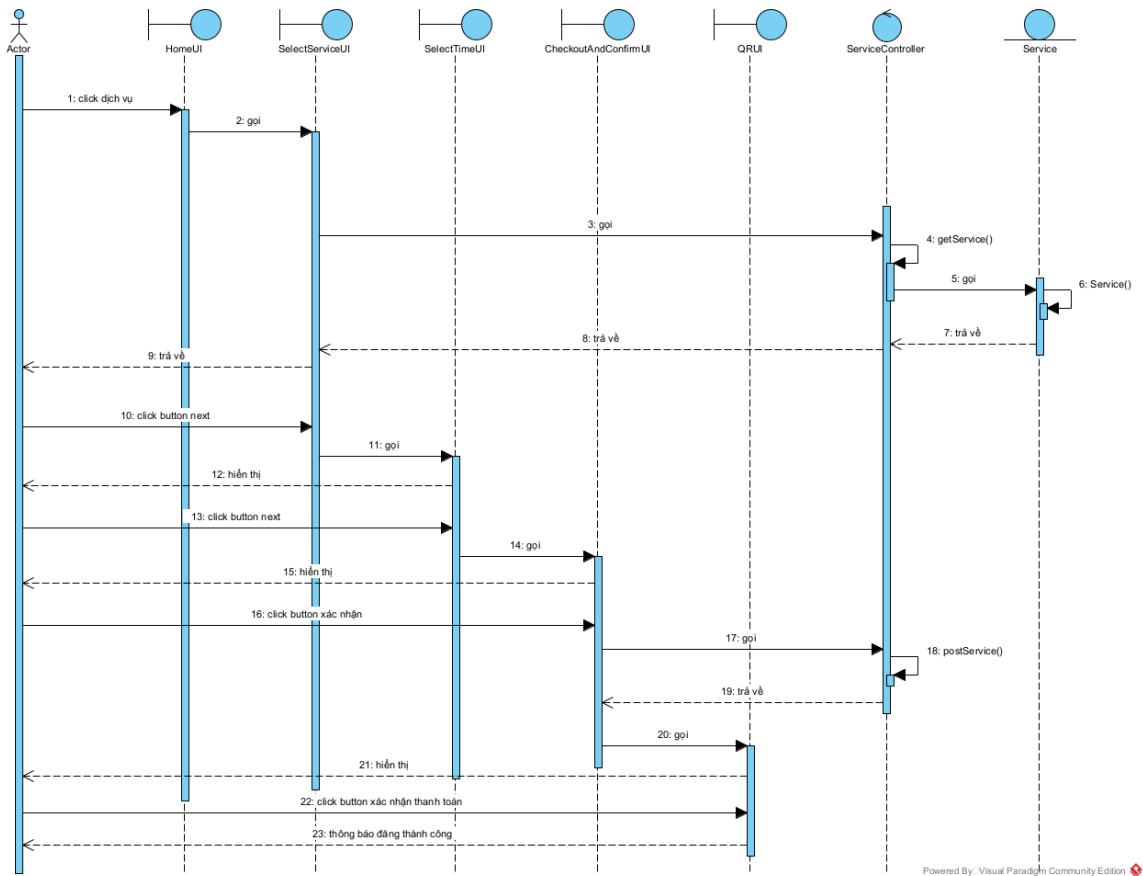
- Đăng nhập bằng tài khoản, mật khẩu



Ảnh 33. Biểu đồ tuần tự chức năng đăng nhập cho khách hàng

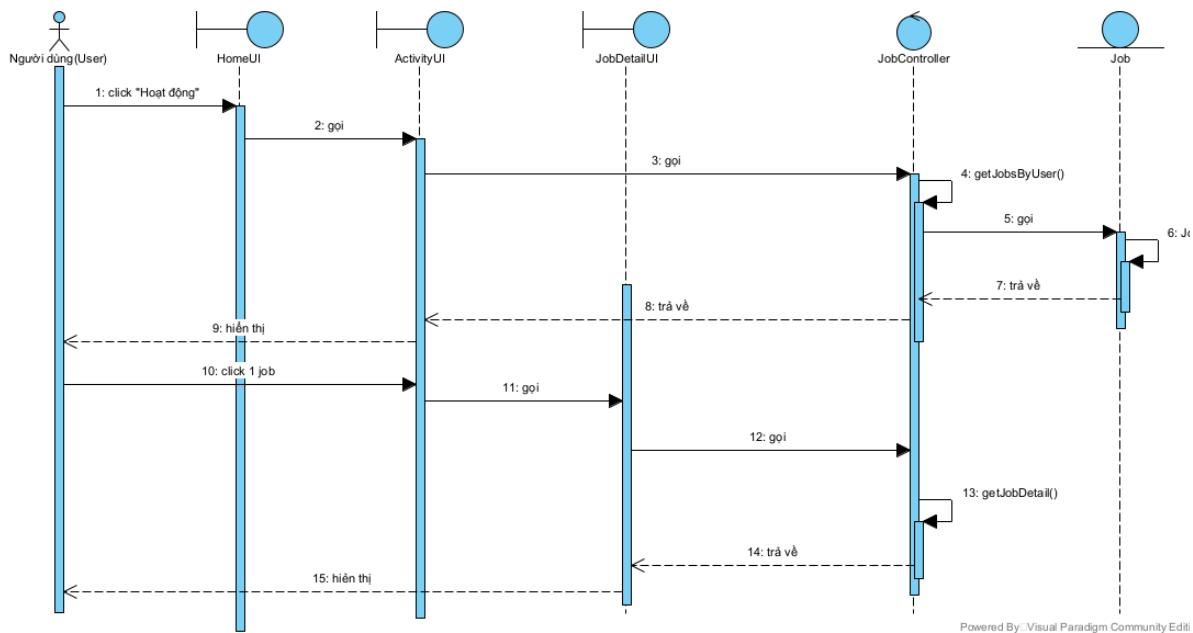
Đồ án tốt nghiệp Đại học

- Chức năng đăng bài tuyển tìm người thực hiện dịch vụ



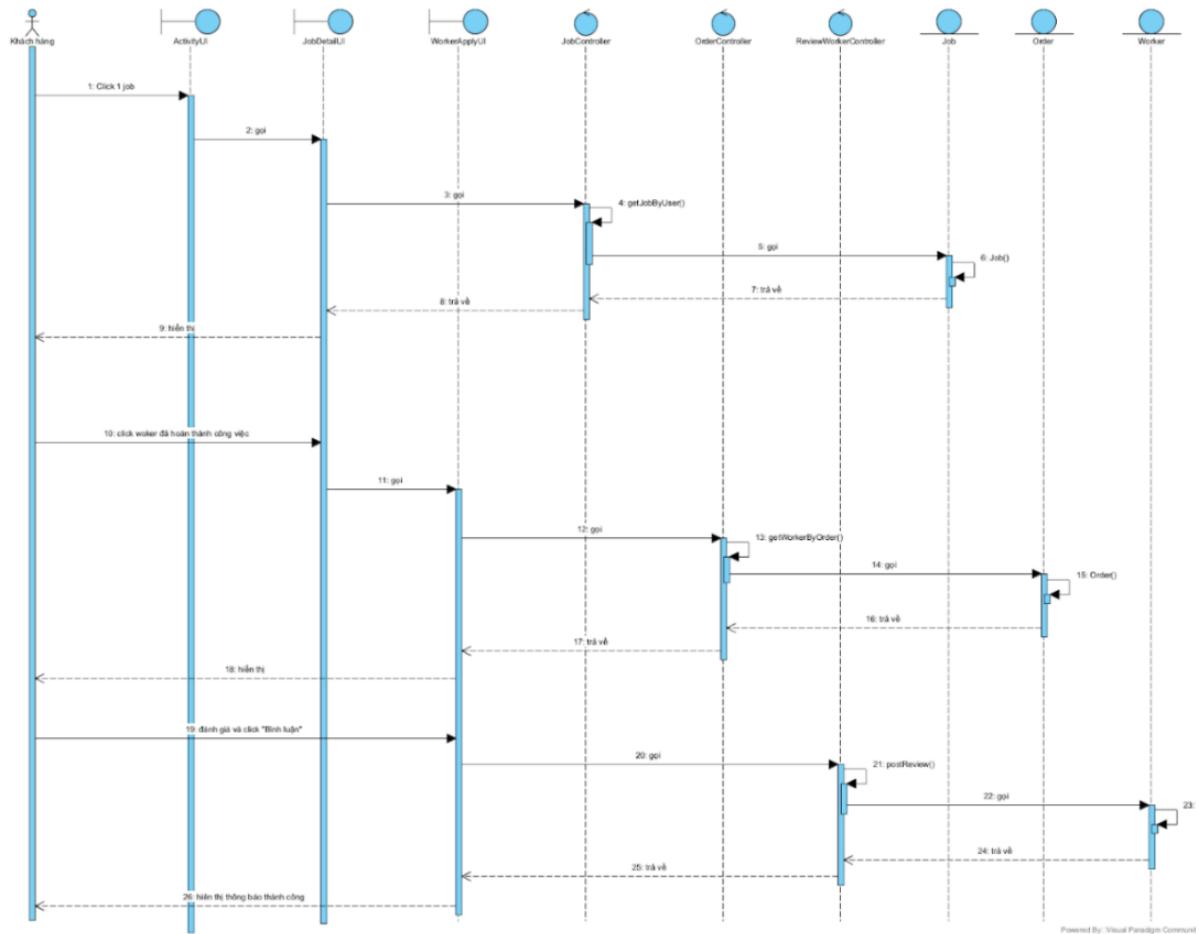
Ảnh 34. Biểu đồ tuần tự đăng bài tuyển tìm người thực hiện dịch vụ

- Chức năng xem chi tiết công việc đã đăng



Ảnh 35. Biểu đồ tuần tự xem chi tiết công việc đã đăng

- Quản lý ứng viên (phê duyệt/ từ chối)

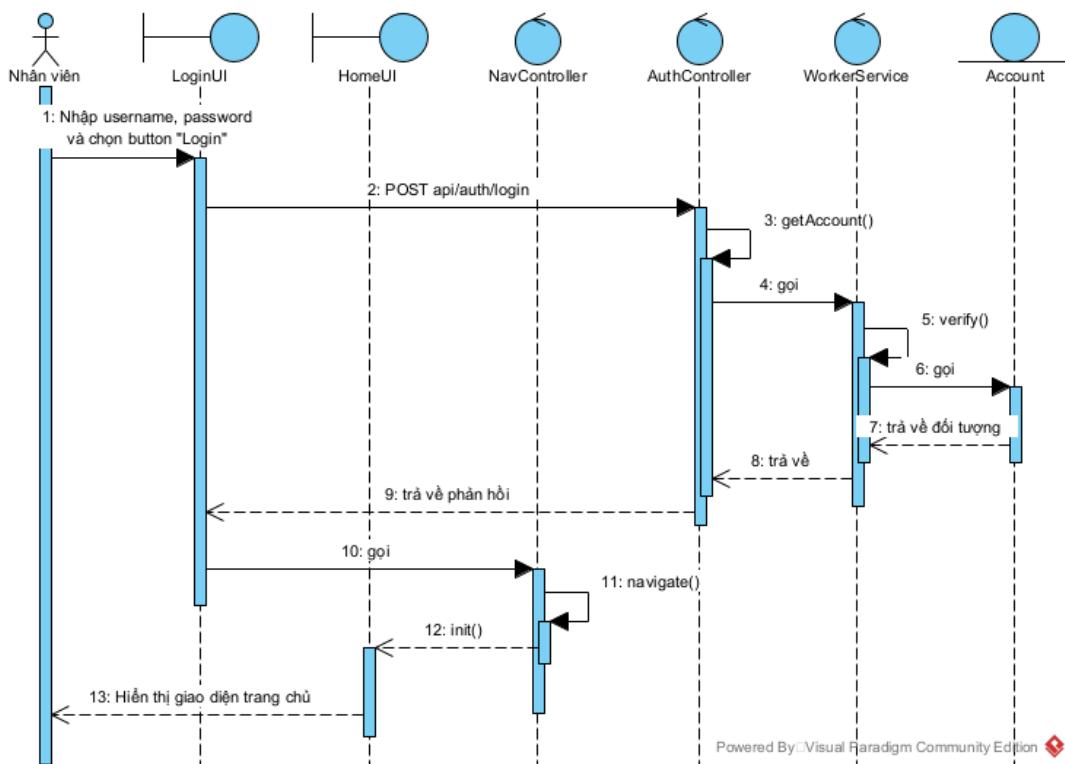


Ảnh 36. Biểu đồ tuần tự quản lý ứng viên

2. Nhân viên

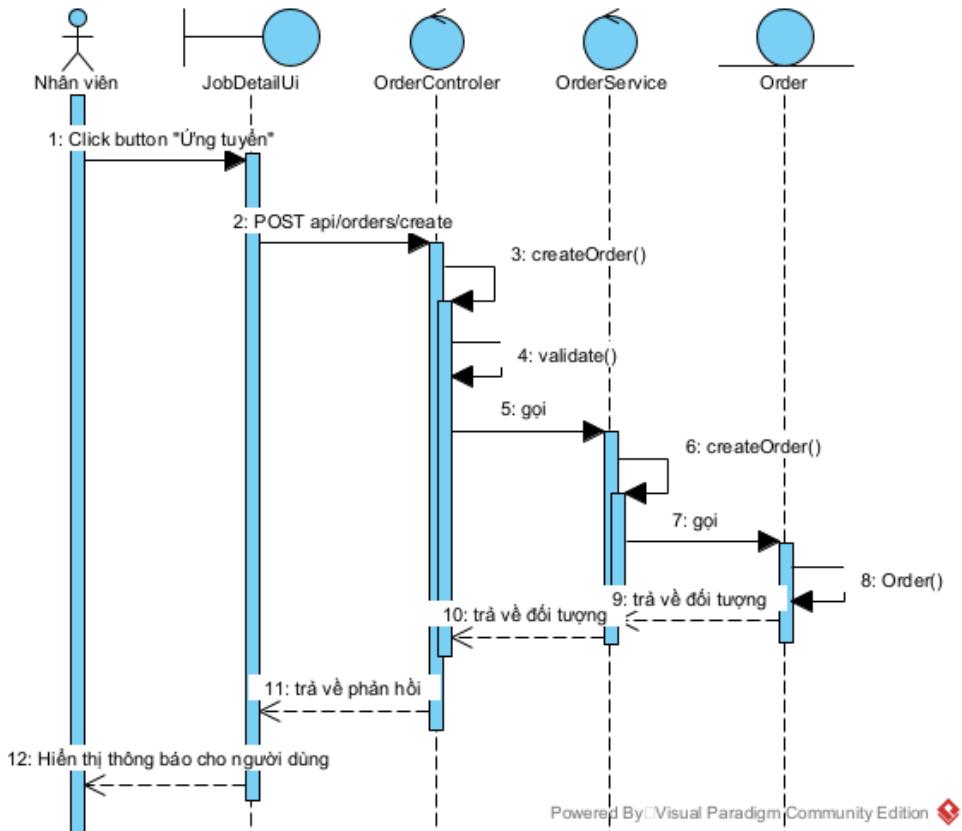
- Chức năng đăng nhập

Đồ án tốt nghiệp Đại học



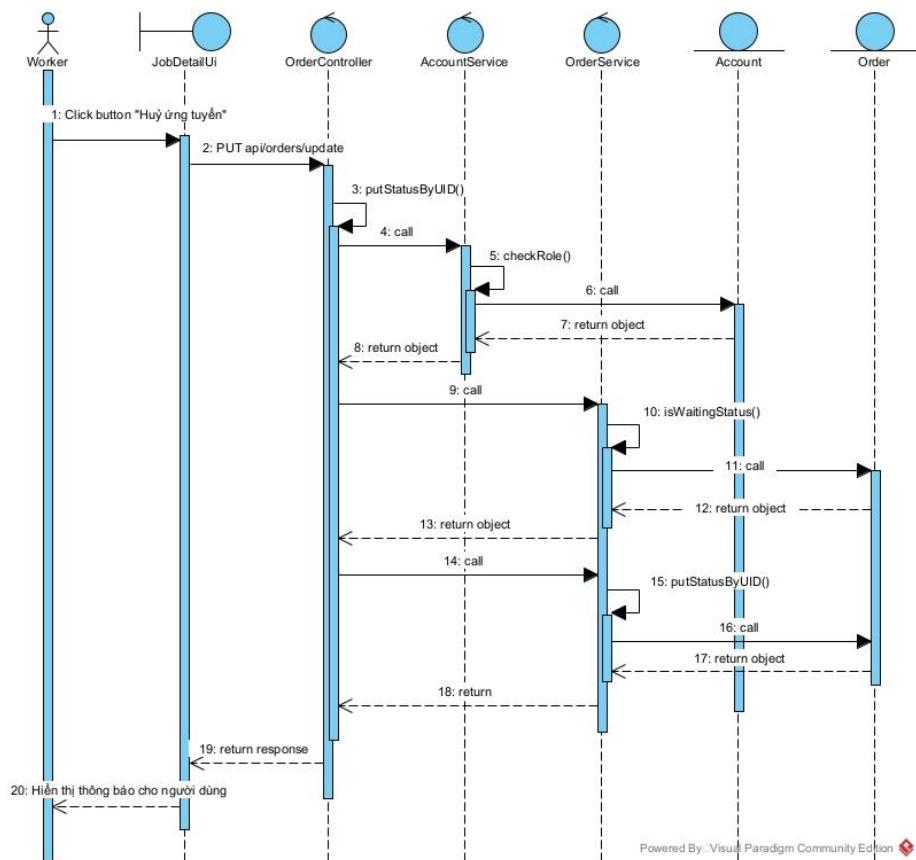
Ảnh 37. Biểu đồ tuần tự chức năng đăng nhập của nhân viên

– Chức năng ứng tuyển



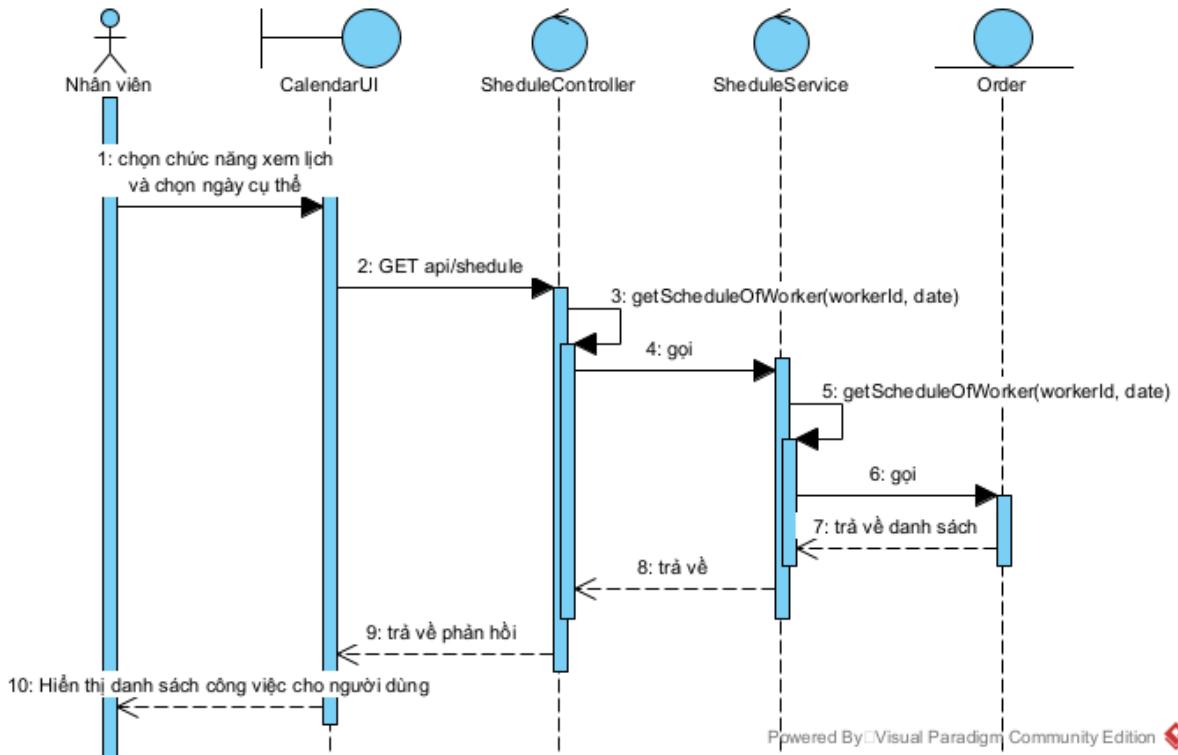
Ảnh 38. Biểu đồ tuần tự nhân viên ứng tuyển công việc

– Chức năng huỷ ứng tuyển



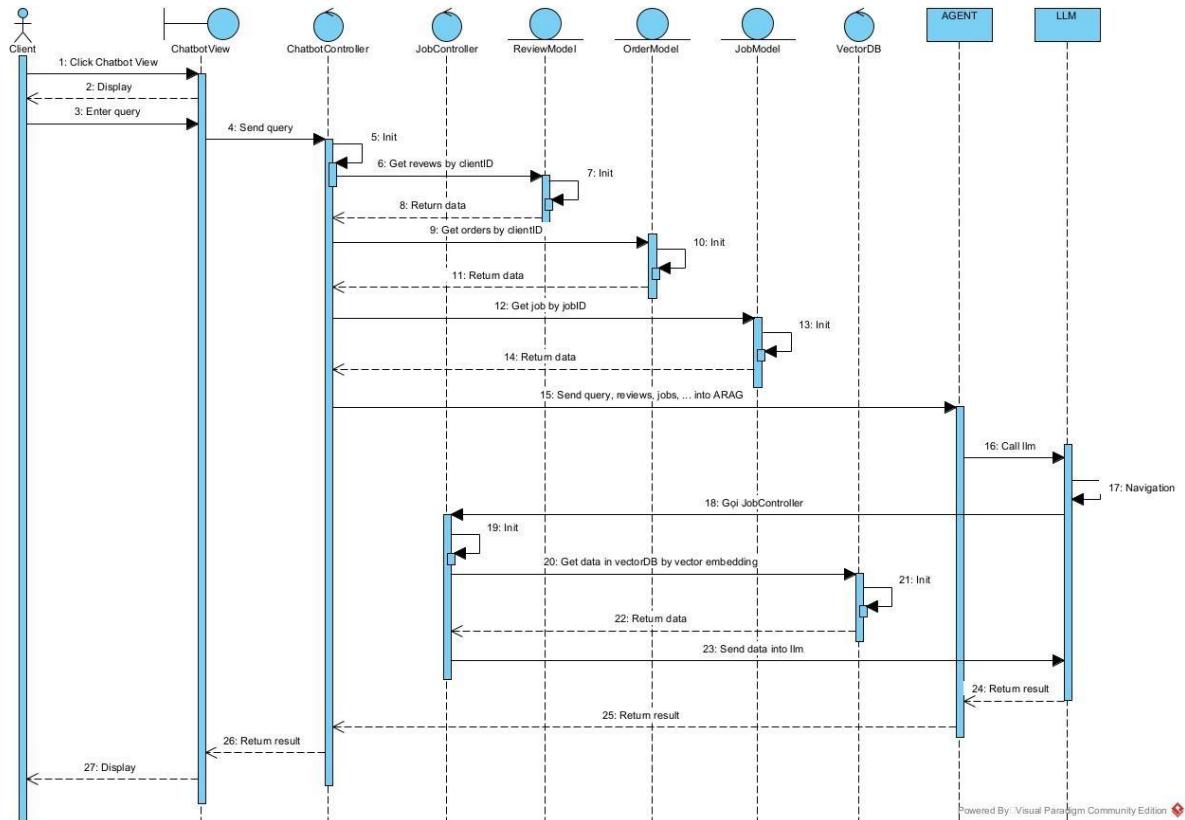
Ảnh 39. Biểu đồ tuần tự chức năng huỷ ứng tuyển

– Chức năng xem lịch làm việc



Ảnh 40. Biểu đồ tuần tự chức năng xem lịch làm việc

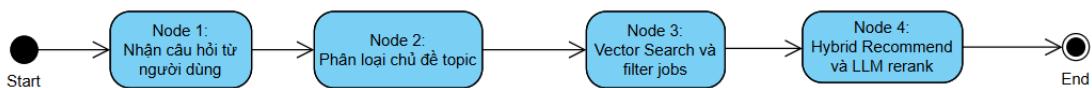
- Chức năng tư vấn công việc kết hợp chatbot



Ảnh 41. Biểu đồ tuần tự tư vấn công việc kết hợp chatbot

2.4.3. Thiết kế luồng xử lý Chatbot

2.4.3.1. Luồng hoạt động chức năng gợi ý công việc



Ảnh 42. Luồng hoạt động xử lý gợi ý công việc

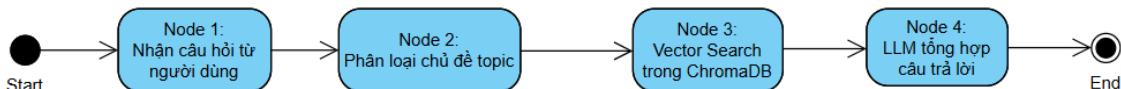
Node 1 (Backend Service): Hệ thống tiếp nhận yêu cầu từ người dùng, truy xuất dữ liệu User Profile (vị trí, kinh nghiệm, sở thích) và lịch sử hoạt động từ Database dựa trên User ID.

Node 2 (AI Controller): Sử dụng mô hình ngôn ngữ lớn (LLM) để phân tích câu hỏi người dùng, xác định ý định tìm kiếm việc làm (Topic = "job") và trích xuất các thực thể quan trọng (địa điểm, mức lương, loại hình dịch vụ).

Node 3 (Vector Search): Chuyển đổi yêu cầu thành vector và thực hiện tìm kiếm tương đồng (Similarity Search) trên Pinecone để lọc ra danh sách ứng viên công việc. Đồng thời áp dụng bộ lọc metadata để chỉ lấy các công việc đang ở trạng thái sẵn sàng (Available).

Node 4 (Hybrid Recommendation Engine): Hệ thống tính toán điểm số phù hợp (Scoring) cho từng công việc dựa trên đa chiều thông tin. Cuối cùng, LLM thực hiện bước Re-ranking để sắp xếp lại danh sách ưu tiên nhất và trả về kết quả kèm lời dẫn.

2.4.3.2. Luồng hoạt động chức năng tư vấn thông tin cơ bản về hệ thống



Ảnh 43. Luồng hoạt động xử lý tư vấn thông tin

Node 1 (Backend Service): Thu thập câu hỏi hiện tại và toàn bộ lịch sử hội thoại (conversation history) của phiên làm việc để đảm bảo ngữ cảnh liền mạch.

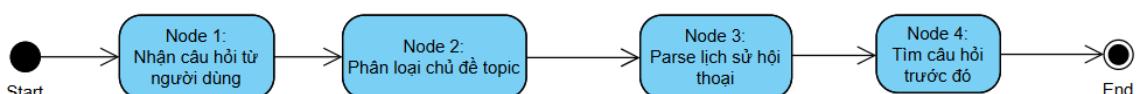
Node 2 (AI Controller): LLM thực hiện phân loại chủ đề câu hỏi vào các nhóm:

- + Info: Thông tin dịch vụ, hướng dẫn sử dụng.
- + Policy: Chính sách, quy định, điều khoản.
- + General: Trò chuyện xã giao thông thường.

Node 3 (RAG Retrieval): Dựa trên chủ đề, hệ thống sử dụng Embedding Model để chuyển câu hỏi thành vector và truy vấn vào kho tri thức tương ứng (ChromaDB Info hoặc Policy). Kỹ thuật Semantic Search giúp tìm ra các văn bản tài liệu liên quan nhất (Retrieved Context).

Node 4 (LLM Generation): LLM tiếp nhận câu hỏi, lịch sử hội thoại và các tài liệu vừa tìm được (Context) để tổng hợp và sinh ra câu trả lời tự nhiên, chính xác, bám sát tài liệu tham khảo.

2.4.3.3. Luồng hoạt động chức năng lịch sử trò chuyện



Ảnh 44. Luồng hoạt động lịch sử trò chuyện

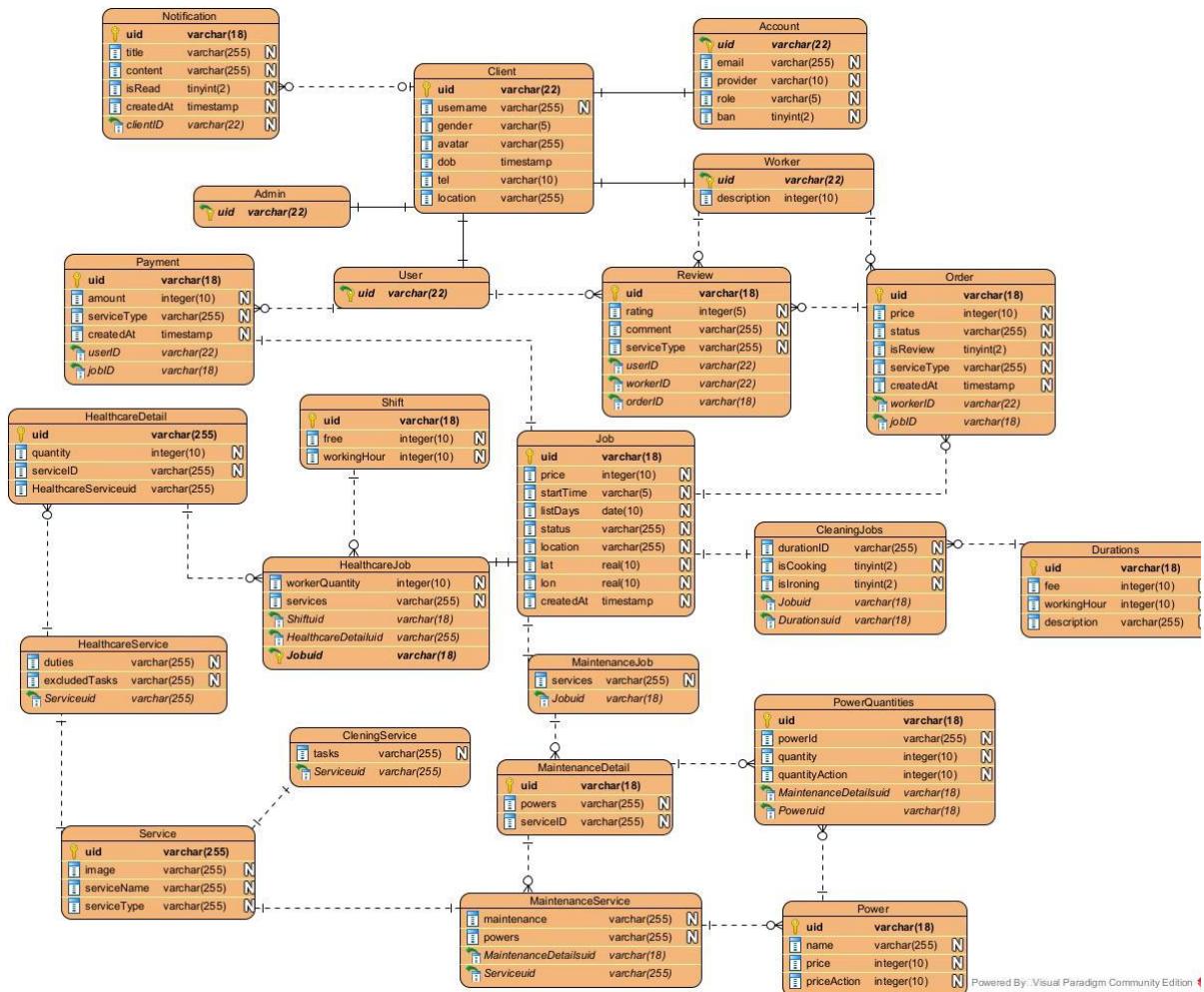
Node 1 (Backend Service): Truy xuất chuỗi dữ liệu lịch sử hội thoại (conversation_history string) được lưu trữ trong database

Node 2 (AI Controller): LLM xác định người dùng đang có ý định hỏi lại hoặc tham chiếu đến nội dung đã trao đổi trước đó (Topic = "history").

Node 3 (Data Parsing): Hệ thống thực hiện phân tích cú pháp (Parse) chuỗi lịch sử, tách biệt các lượt chat (User Message/Bot Response) thành một danh sách có cấu trúc.

Node 4 (Context Extraction): Duyệt ngược danh sách từ thời điểm hiện tại về quá khứ để tìm và trích xuất chính xác câu hỏi hoặc nội dung gần nhất mà người dùng quan tâm, sau đó phản hồi lại thông tin đó.

2.5. Cơ sở dữ liệu



Ảnh 45. Bảng cơ sở dữ liệu

Tiểu kết chương 2

Chương 2 đã trình bày chi tiết quá trình phân tích và thiết kế hệ thống Helpo, bao gồm việc xác định kiến trúc tổng thể, mô hình nghiệp vụ và các phương pháp tiếp cận để giải quyết bài toán kết nối khách hàng với người cung cấp dịch vụ chăm sóc tại gia. Hệ thống được xây dựng theo kiến trúc Client–Server, đảm bảo tính mở rộng, khả năng bảo trì và hiệu năng xử lý. Ở phía ứng dụng di động, nhóm áp dụng mô hình MVVM kết hợp Repository Pattern nhằm tách biệt rõ ràng giữa giao diện và logic xử lý, giúp mã nguồn dễ bảo trì và hỗ trợ tốt cho việc kiểm thử. Phía backend được thiết kế theo Layered Architecture với các tầng Route, Controller, Service và Model, đảm bảo tính phân tách và dễ mở rộng.

Nhìn chung, nội dung chương 2 đã khái quát việc triển khai hệ thống, đảm bảo tính logic, khả năng mở rộng và đáp ứng tốt mục tiêu tối ưu hóa kết nối giữa khách hàng và người cung cấp dịch vụ

CHƯƠNG 3. THỬ NGHIỆM VÀ ĐÁNH GIÁ HỆ THỐNG

Chương 3 tập trung vào quá trình thử nghiệm và đánh giá hệ thống ứng dụng Helpo sau khi đã hoàn thành giai đoạn phân tích và thiết kế. Nội dung chương trình bao gồm dữ liệu thực nghiệm, môi trường và phương pháp cài đặt thử nghiệm, các độ đo đánh giá cũng như phương pháp so sánh được sử dụng nhằm kiểm chứng tính hiệu quả và khả năng đáp ứng yêu cầu của hệ thống. Trên cơ sở đó, chương phân tích các kết quả thực nghiệm đạt được và tiến hành thử nghiệm thực tế ứng dụng Helpo trên các nhóm người dùng khác nhau, qua đó đánh giá mức độ hoàn thiện, tính ổn định và trải nghiệm sử dụng của hệ thống.

3.1. Dữ liệu thực nghiệm.

3.1.1. Tập dữ liệu công việc.

Hệ thống GoodJob AI sử dụng tập dữ liệu công việc thực tế bao gồm:

1. Số lượng công việc: 1,000 công việc đa dạng
2. Loại dịch vụ: 3 danh mục chính
 - + CLEANING (Dọn dẹp vệ sinh): 45%
 - + HEALTHCARE (Chăm sóc sức khỏe): 35%
 - + MAINTENANCE (Bảo trì thiết bị): 20%
3. Phạm vi địa lý: Các quận/huyện ở Việt Nam chủ yếu ở Hà Nội và Hồ Chí Minh
4. Khoảng giá: 200,000 - 5,000,000 VNĐ
5. Thông tin bổ sung: Rating, số lượng đánh giá, thời gian làm việc, địa chỉ cụ thể

3.1.2. Tập dữ liệu thông tin và chính sách

1. Tài liệu thông tin: 4 file markdown chứa thông tin chi tiết về các dịch vụ
 - + application.md: Hướng dẫn sử dụng ứng dụng
 - + cleaning.md: Thông tin dịch vụ dọn dẹp
 - + healthcare.md: Thông tin dịch vụ chăm sóc sức khỏe
 - + maintenance.md: Thông tin dịch vụ bảo trì
2. Tài liệu chính sách:
 - + policy.md chứa các quy định, chính sách khiếu nại

3.1.3. Dữ liệu người dùng thử nghiệm

1. Số lượng người dùng (user) test: 100 user
2. Số lượng nhân viên (worker) test: 100 worker
3. Số lượng công việc (job) test: 1000 job
4. Số lượng đánh giá nhân viên (review) test: 100 review
5. Số lượng yêu cầu (order) test: 100 order
6. Số lượng câu hỏi thực nghiệm: 180 queries đa dạng về tìm kiếm công việc, thông tin dịch vụ và chính sách

3.2. Cài đặt thực nghiệm.

3.2.1. Độ đo

Hệ thống được đánh giá thông qua độ đo: Precision@k, Recall@k, F1-Score

- + Precision@k: Tỷ lệ công việc phù hợp trong top-k kết quả

$$\text{Precision}@k = (\text{Số công việc phù hợp trong top-k}) / k$$

Công thức 1. Công thức Precision@k

- + Recall@k: Tỷ lệ công việc phù hợp được tìm thấy trong top-k

$$\text{Recall}@k = (\text{Số công việc phù hợp trong top-k}) / (\text{Tổng số công việc phù hợp})$$

Công thức 2. Công thức Recall@k

- + F1-Score: Trung bình điều hòa của Precision và Recall

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

Công thức 3. Công thức F1-Score

Các độ đo này cho phép đánh giá toàn diện cả độ chính xác của kết quả gợi ý công việc lẫn chất lượng hội thoại của hệ thống AI.

3.2.2. Phương pháp thực nghiệm.

Trong quá trình thực nghiệm, hệ thống GoodJob AI được triển khai theo mô hình Retrieval-Augmented Generation kết hợp cơ chế gợi ý lai (Hybrid Recommendation). Quy trình thực nghiệm bao gồm các bước chính như sau:

1. Hệ thống tiếp nhận câu truy vấn từ người dùng, thực hiện phân loại ý định (intent classification) nhằm xác định loại yêu cầu như tìm kiếm công việc, hỏi thông tin dịch vụ hoặc truy vấn chính sách.
2. Hệ thống tiến hành truy xuất dữ liệu liên quan từ kho vector bằng phương pháp tìm kiếm ngữ nghĩa dựa trên embedding. Song song với đó, các điều kiện lọc theo hồ sơ người dùng như vị trí địa lý, mức giá mong muốn, loại dịch vụ và kinh nghiệm làm việc cũng được áp dụng.
3. Hệ thống thực hiện xếp hạng lại kết quả dựa trên mô hình hybrid, kết hợp độ tương đồng ngữ nghĩa, mức độ phù hợp hồ sơ và lịch sử tương tác của người dùng.

Phương pháp tính khoảng cách địa lý (Haversine)

Để đánh giá mức độ phù hợp về mặt vị trí giữa người dùng và các công việc được gợi ý, hệ thống sử dụng công thức Haversine nhằm tính toán khoảng cách địa lý giữa hai điểm dựa trên vĩ độ và kinh độ. Khoảng cách giữa hai điểm trên bề mặt Trái Đất được xác định theo công thức:

$$\begin{aligned}\Delta\varphi &= \varphi_2 - \varphi_1 \\ \Delta\lambda &= \lambda_2 - \lambda_1 \\ a &= \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos(\varphi_1)\cos(\varphi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right) \\ c &= 2 \cdot \arctan 2(\sqrt{a}, \sqrt{1-a}) \\ d &= R \cdot c\end{aligned}$$

Công thức 4. Công thức Haversine

Trong đó:

- φ_1, φ_2 là vĩ độ của hai điểm (tính theo radian),
- λ_1, λ_2 là kinh độ của hai điểm (tính theo radian)
- R là bán kính trung bình của Trái Đất, với $R=6371\text{km}$
- d là khoảng cách địa lý giữa hai điểm (km)

Công thức Haversine được sử dụng trong bước lọc và xếp hạng kết quả nhằm ưu tiên các công việc có vị trí gần với người dùng hơn, từ đó nâng cao tính thực tiễn và khả năng ứng dụng của hệ thống trong môi trường thực tế.

3.2.3. Các phương pháp được sử dụng để so sánh

Để đánh giá hiệu quả của hệ thống GoodJob AI, nghiên cứu tiến hành so sánh với các phương pháp baseline sau:

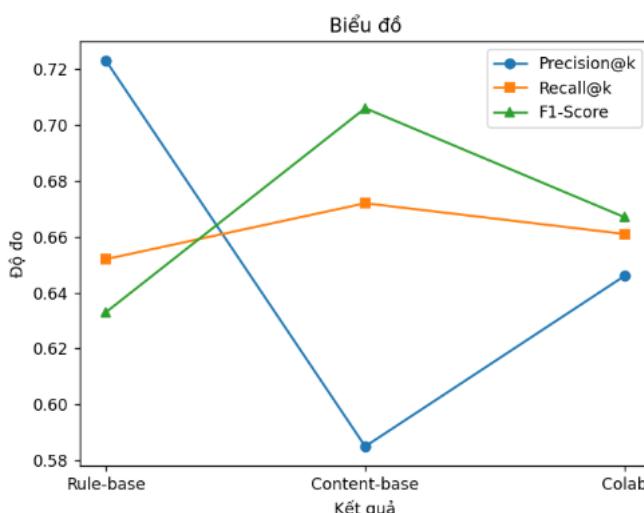
- Cosine Similarity: Sử dụng độ tương đồng cosine giữa vector embedding của truy vấn và embedding của công việc để xác định mức độ phù hợp. Phương pháp này không xét đến thông tin hồ sơ người dùng hay bối cảnh hội thoại.
- Rule-based Filtering: Lọc công việc dựa trên các quy tắc cố định như vị trí địa lý, mức giá và loại dịch vụ.
- Content-base Filtering: Lọc công việc dựa trên giữa hồ sơ nhân viên và yêu cầu công việc dựa trên đặc trưng nội dung.
- Collaborative Filtering: Lọc công việc dựa trên câu hỏi người dùng, kinh nghiệm người dùng, mức độ yêu thích với loại công việc, mức độ đánh giá của khách hàng và độ tương đồng của công việc với công việc đã làm trước đó.

Các phương pháp trên được sử dụng làm đường cơ sở để so sánh với mô hình đề xuất nhằm chứng minh tính hiệu quả của cách tiếp cận ARAG kết hợp hybrid recommendation.

3.3. Kết quả thực nghiệm.

3.3.1. RAG với Hybrid Recommend

a. Kết quả:



Ảnh 46. Biểu đồ kết quả thực nghiệm

top_k = 300	Precision@k	Recall@k	F1-Score
Rule-based Filtering	0,723	0,585	0,646
Content-based Filtering	0,652	0,672	0,661
Collaborative Filtering	0,633	0,706	0,667

Bảng 15. Kết quả đánh giá thực nghiệm độ đo theo topk=300

Suy ra, Collaborative Filtering có độ cân bằng tốt nhất về cả độ đúng đắn, chính xác và không bỏ sót tạo hiệu quả gợi ý công việc tố hơn.

- Với Rule-based Filtering: được thực hiện bằng cách trả về danh sách công việc (top_k) phù hợp với câu hỏi người dùng nhập vào.
- Với Content-based Filtering: được thực hiện dựa vào câu hỏi người dùng nhập vào và thông tin của người dùng bao gồm: vị trí, loại công việc được đánh giá cao bởi người dùng khác và kinh nghiệm của người dùng.
- Với Collaborative Filtering: được thực hiện bằng cách tổng hợp các điều kiện: câu hỏi người dùng, kinh nghiệm người dùng, mức độ yêu thích với loại công việc, mức độ đánh giá của khách hàng và độ tương đồng của công việc với công việc đã làm trước đó.

b. Điều kiện chọn công việc

Công việc sẽ được ưu tiên đúng, phù hợp với câu hỏi của người dùng: vị trí, thời gian, giá, Hơn nữa, công việc thỏa mãn: phù hợp năng lực hoặc độ yêu thích của người dùng với công việc hoặc mức độ đánh giá của khách hàng dành cho người dùng.

c. Hybrid Recommend

Đối với *người dùng mới* sẽ tập trung tìm công việc xoay quanh câu hỏi người dùng và thông tin của người dùng, ví dụ: vị trí nơi ở, mức độ yêu thích công việc (Content-base)

Đối với *người dùng lâu* sẽ tập trung tìm công việc dựa trên câu hỏi người dùng, thông tin cá nhân, vị trí, mức độ yêu thích công việc, mức độ đánh giá của khách hàng và các công việc tương tự công việc mà người dùng đã ứng tuyển. (Collab)

3.3.2. ARAG

- Trong tổng số 180 câu hỏi với 100 câu hỏi về công việc, 20 câu hỏi trò chuyện, 30 câu hỏi về thông tin ứng dụng và 30 câu hỏi về chính sách.
- Độ chính xác đạt: 0,985

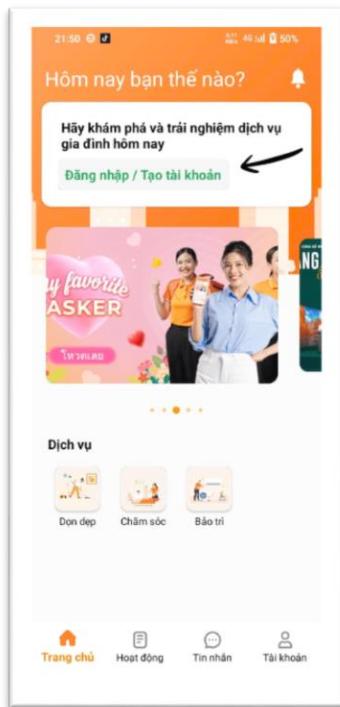
3.4. Thử nghiệm ứng dụng Helpo tối ưu hóa kết nối khách hàng và người cung cấp dịch vụ chăm sóc tại gia

3.4.1. Giao diện cho khách hàng

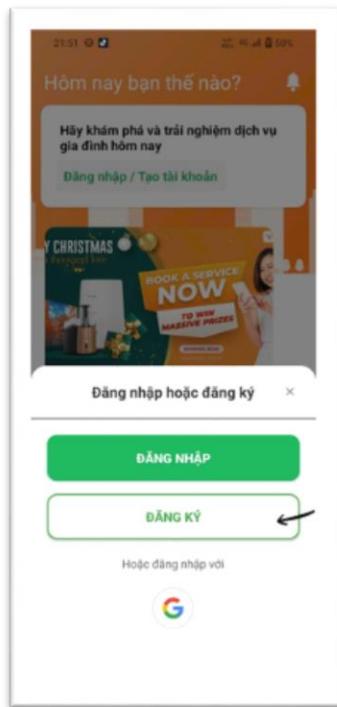
1. Chức năng đăng ký

Khi vào trang chủ, chọn “Đăng nhập/ Tạo tài khoản”. Cửa sổ lựa chọn đăng nhập hoặc đăng ký xuất hiện, chọn “Đăng ký”

Đồ án tốt nghiệp Đại học

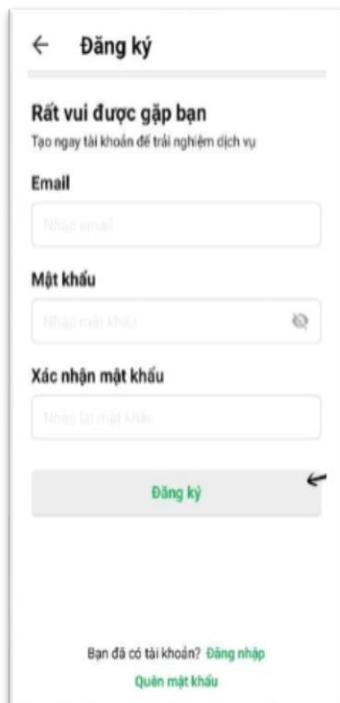


Ảnh 47. GD trang chủ của khách hàng chưa đăng nhập

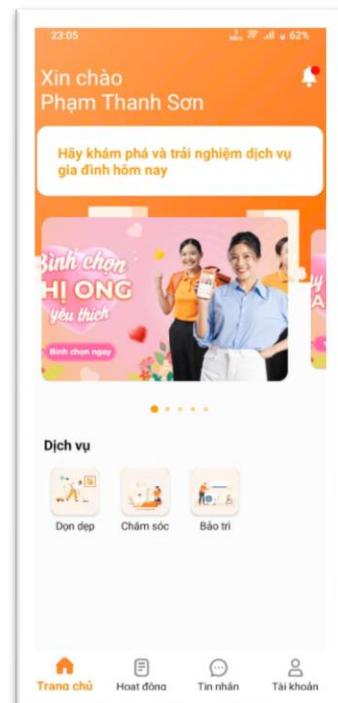


Ảnh 48. GD lựa chọn đăng nhập/ đăng ký

Giao diện đăng ký hiện ra, nhập thông tin đăng ký và chọn “Đăng ký”. Sau khi đăng ký thành công sẽ quay trở về màn trang chủ,



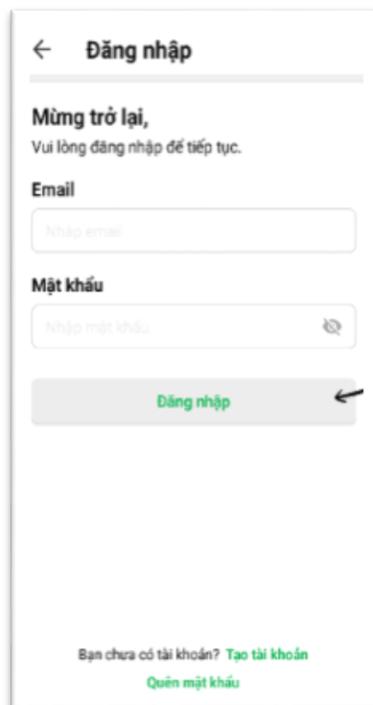
Ảnh 49. GD đăng ký của khách hàng



Ảnh 50. GD trang chủ của khách hàng đã đăng nhập

2. Chức năng đăng nhập

Khi vào trang chủ, chọn “Đăng nhập/ Tạo tài khoản”. Cửa sổ lựa chọn đăng nhập hoặc đăng ký xuất hiện, chọn “Đăng nhập”.



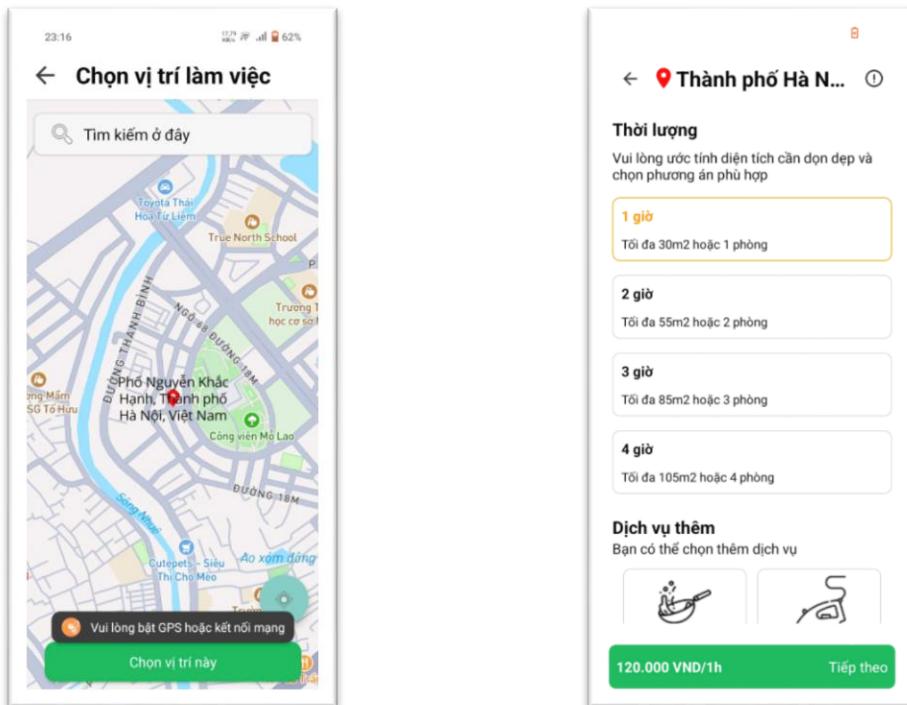
Ảnh 51. GD đăng nhập của khách hàng

Giao diện đăng nhập hiện ra, nhập thông tin đăng nhập và chọn “Đăng nhập”, Sau khi đăng ký thành công sẽ quay trở về màn trang chủ.

3. Chức năng đăng công việc

Sau khi đăng nhập thành công, chọn dịch vụ “Dọn dẹp”. Giao diện bัน đồ hiện ra yêu cầu người dùng chọn vị trí đăng công việc. Sau khi chọn xong vị trí, chọn “Chọn vị trí này”. Giao diện đăng ký công việc dọn dẹp hiện ra, lựa chọn thời lượng, dịch vụ thêm và chọn “Tiếp theo”

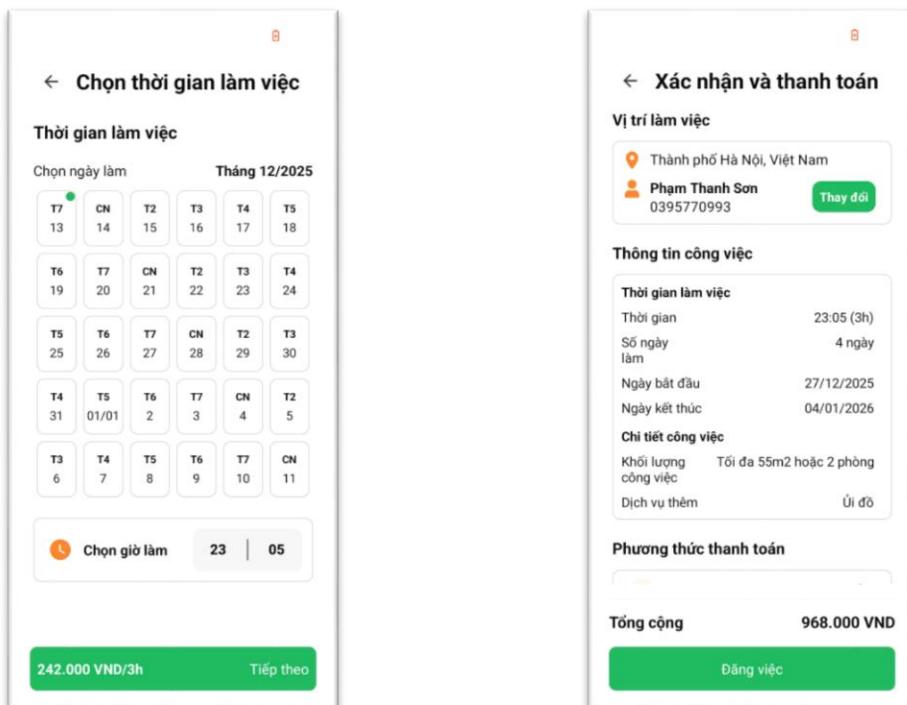
Đồ án tốt nghiệp Đại học



Ảnh 52. GD chọn vị trí làm việc

Ảnh 53. GD đăng ký dịch vụ dọn dẹp

Giao diện đăng ký thời gian làm việc hiện ra, lựa chọn những ngày, giờ mong muốn sau đó chọn “Tiếp theo”. Giao diện xác nhận thanh toán hiện ra, kiểm tra thông tin và chọn “Đăng việc”.



Ảnh 54. GD chọn thời gian làm việc

Ảnh 55. GD xác nhận và thanh toán

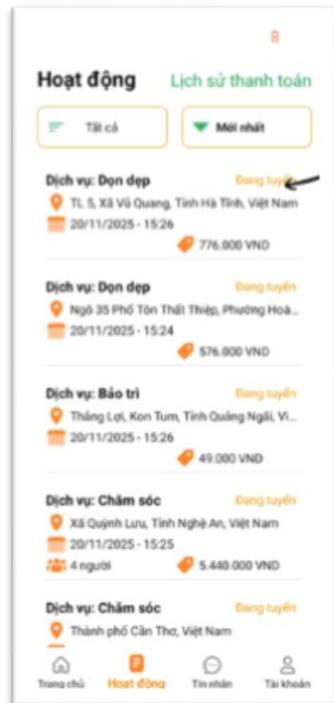
Giao diện mã QR hiện ra, hệ thống thông báo “Đăng việc thành công”. Chọn “Lưu mã QR về máy” và chọn nút “x” để thoát khỏi giao diện QR. Sau đó giao diện quay về màn trang chủ.



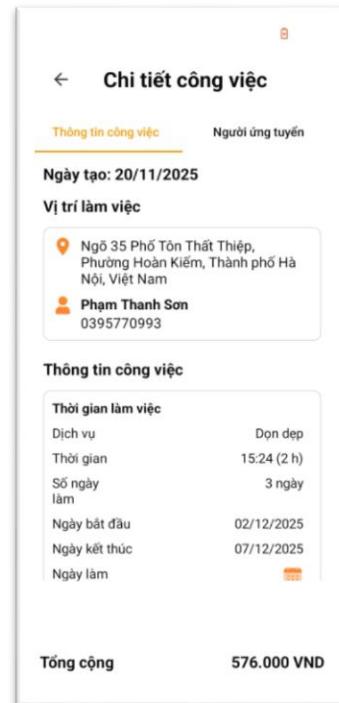
Ảnh 56. GD mã thanh toán QR

4. Chức năng quản lý người ứng tuyển

Sau khi đăng nhập thành công, chọn “Hoạt động”. Giao diện danh sách công việc đã đăng hiện ra, chọn dịch vụ dọn dẹp. Giao diện chi tiết công việc hiện ra, chọn “Người ứng tuyển”

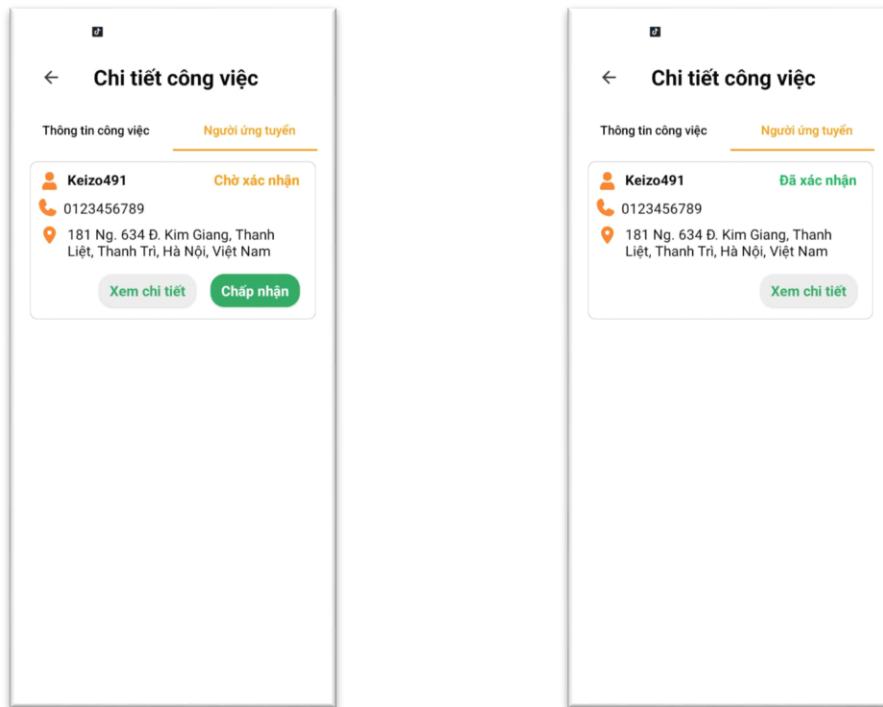


Ảnh 57. GD công việc đã đăng



Ảnh 58. GD chi tiết công việc

Giao diện danh sách ứng tuyển hiện ra, chọn “Chấp nhận” để xác nhận ứng viên

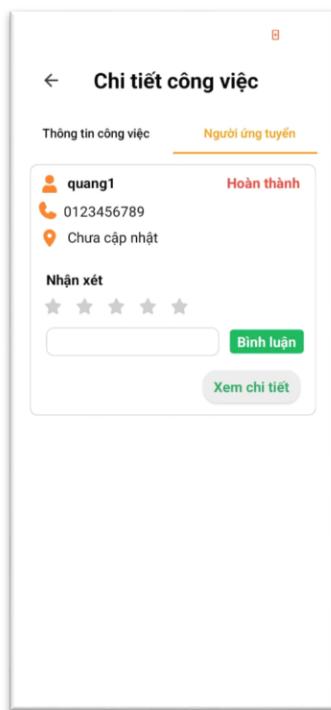


Ảnh 59. GD danh sách ứng viên

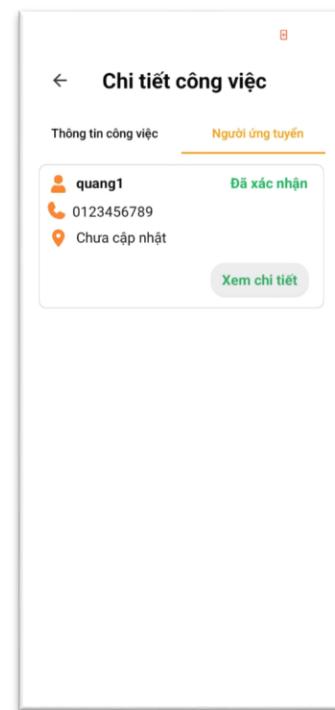
Ảnh 60. GD chấp nhận ứng viên

5. Chức năng đánh giá nhân viên

Sau khi đăng nhập thành công, tại giao diện trang chủ, chọn “Hoạt động”. Hệ thống hiển thị danh sách công việc đã đăng, chọn công việc đã hoàn thành. Hệ thống hiển thị thông tin chi tiết về công việc đã đăng, chọn “Người ứng tuyển”. Hệ thống hiển thị danh sách người làm đang ở trạng thái hoàn thành, ta nhập bình luận và đánh giá số sao sau đó chọn “Bình luận”.

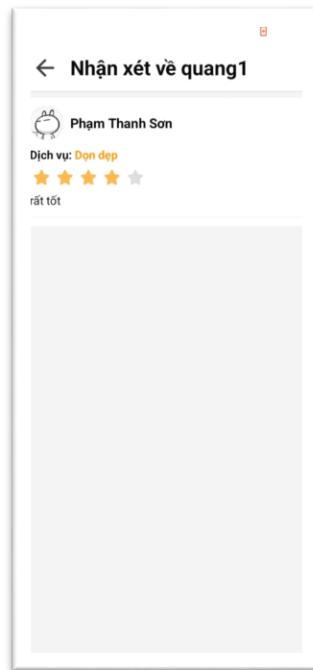


Ảnh 61. GD đánh giá nhân viên



Ảnh 62. GD sau khi đánh giá nhân viên

Ta cũng có thể xem các bài đánh giá về nhân viên đó bằng cách chọn “Xem chi tiết”, giao diện danh sách bài đánh giá hiện ra.



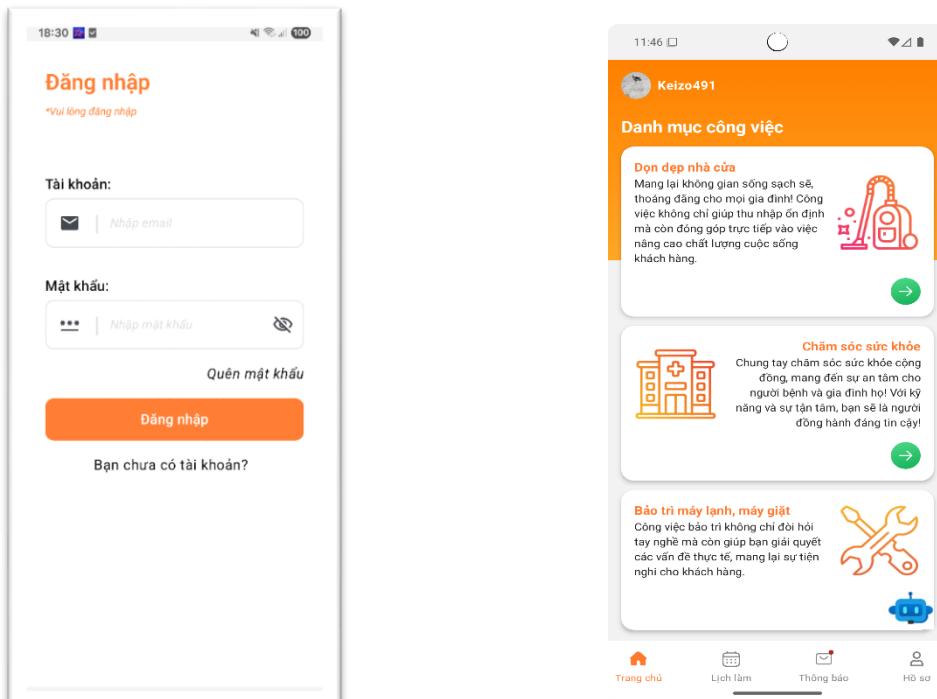
Ảnh 63. GD bài đánh giá

3.4.2. Giao diện cho nhân viên

- Chức năng đăng nhập

Đồ án tốt nghiệp Đại học

Nhân viên lần đầu vào ứng dụng, cần đăng nhập bằng thông tin tài khoản, mật khẩu. Sau đó, nếu thành công, giao diện “Trang chủ” hiện ra.



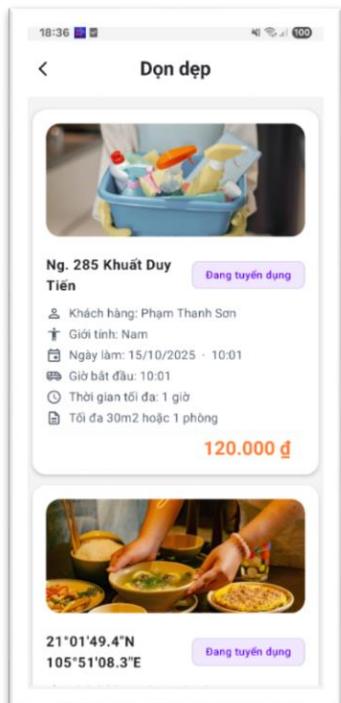
Ảnh 65. GD trang chủ cho nhân viên

Ảnh 64. GD nhân viên đăng nhập

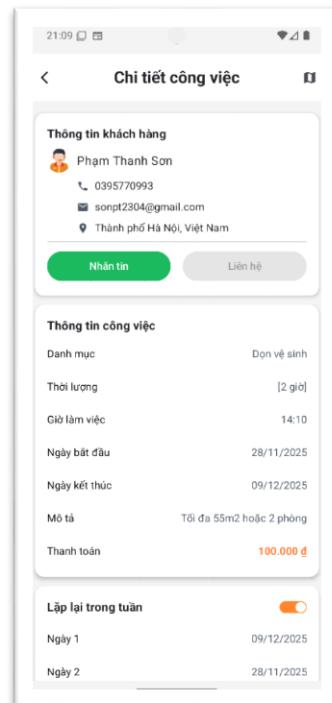
– Chức năng xem chi tiết công việc

Để có thể xem chi tiết một công việc, đầu tiên hãy chọn danh mục công việc. Bạn có thể chọn dọn dẹp/ chăm sóc sức khoẻ/ bảo trì như *Ảnh 63*. Sau đó danh sách công việc theo danh mục sẽ hiện ra và hãy chọn một công việc cụ thể. Ở đây, chúng tôi đang minh họa luồng xem chi tiết công việc dọn dẹp nào đó.

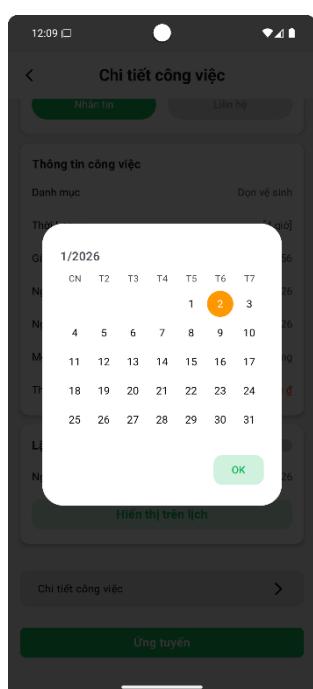
Đồ án tốt nghiệp Đại học



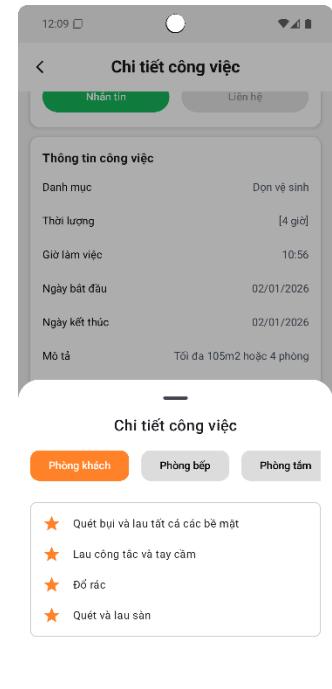
Ảnh 66. Danh sách công việc dọn dẹp



Ảnh 67. Chi tiết công việc dọn dẹp



Ảnh 68. Lịch cần thực hiện của việc

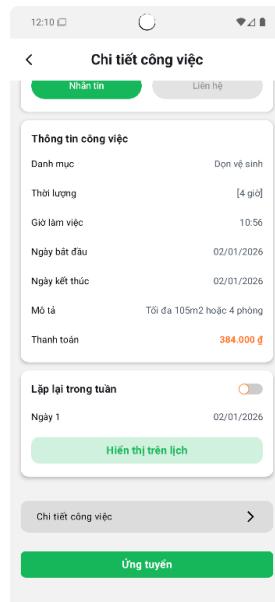


Ảnh 69. Phạm vi công việc dọn dẹp

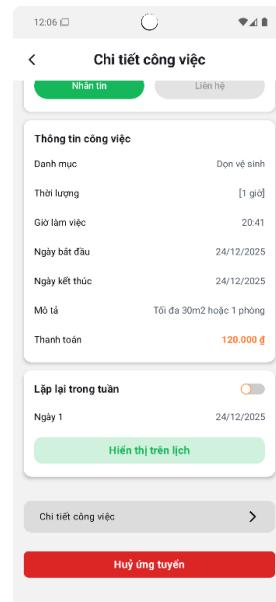
- Chức năng ứng tuyển/ huỷ ứng tuyển công việc

Sau khi xem chi tiết công việc, bạn có 2 tuỳ chọn. Nếu chưa ứng tuyển, bạn có thể chọn “Ứng tuyển” để ghi danh cho khách hàng biết rằng có người nhận uỷ thác công việc này. Nếu đã từng ứng tuyển, chọn “Huỷ ứng tuyển” trước khi công việc bắt đầu nếu như có lý do.

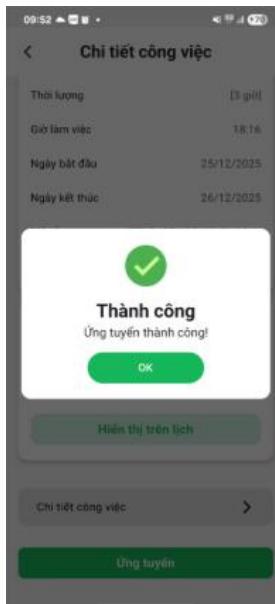
Đồ án tốt nghiệp Đại học



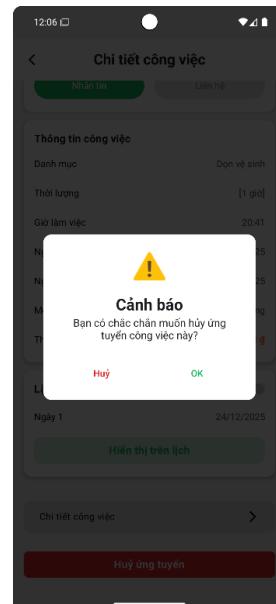
Ảnh 70. Chức năng ứng tuyển



Ảnh 71. Chức năng huỷ ứng tuyển

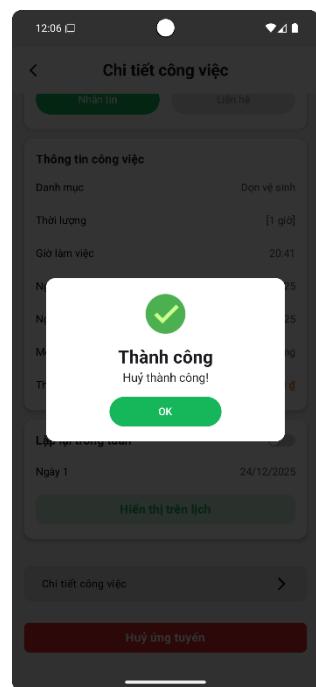


Ảnh 72. Thông báo ứng tuyển thành công



Ảnh 73. Hiển thị thông báo cảnh báo

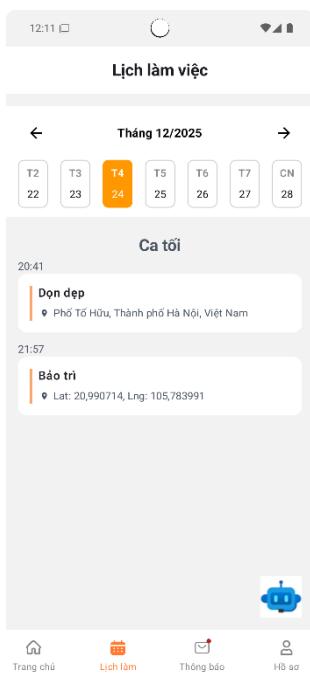
Đồ án tốt nghiệp Đại học



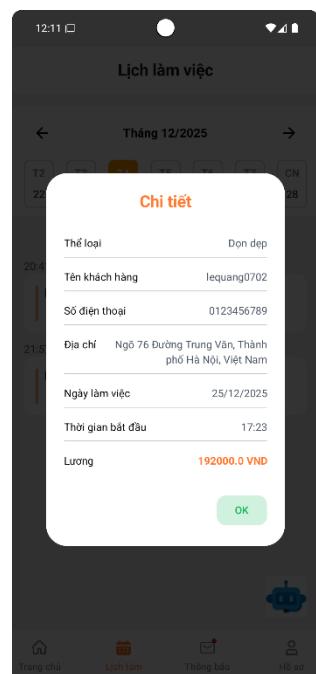
Ảnh 74. Hiển thị thông báo huỷ thành công

– Chức năng xem công việc theo ngày

Nhân viên có thể xem chi tiết công việc theo ngày với bộ lọc thời gian rất thuận tiện, dễ sử dụng.



Ảnh 75. Lịch làm việc

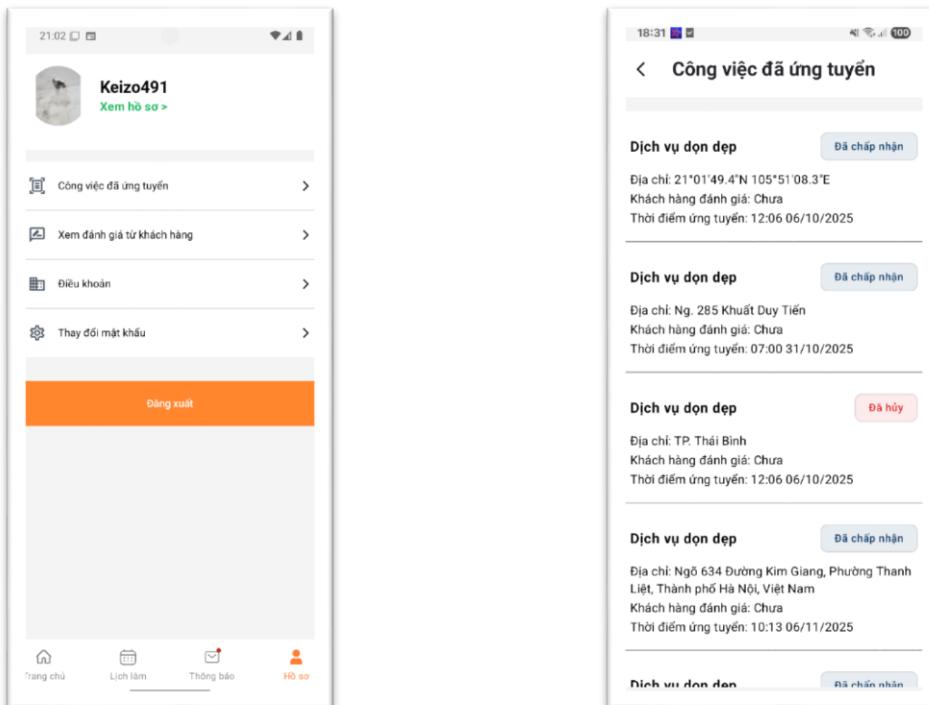


Ảnh 76. Chi tiết lịch một công việc

– Chức năng xem công việc đã ứng tuyển

Đồ án tốt nghiệp Đại học

Nhân viên có thể dễ dàng xem lại những công việc đã ứng tuyển. Danh sách dưới đây là lịch sử đơn ứng tuyển, cùng với trạng thái của công việc đó (*Ảnh dưới*).

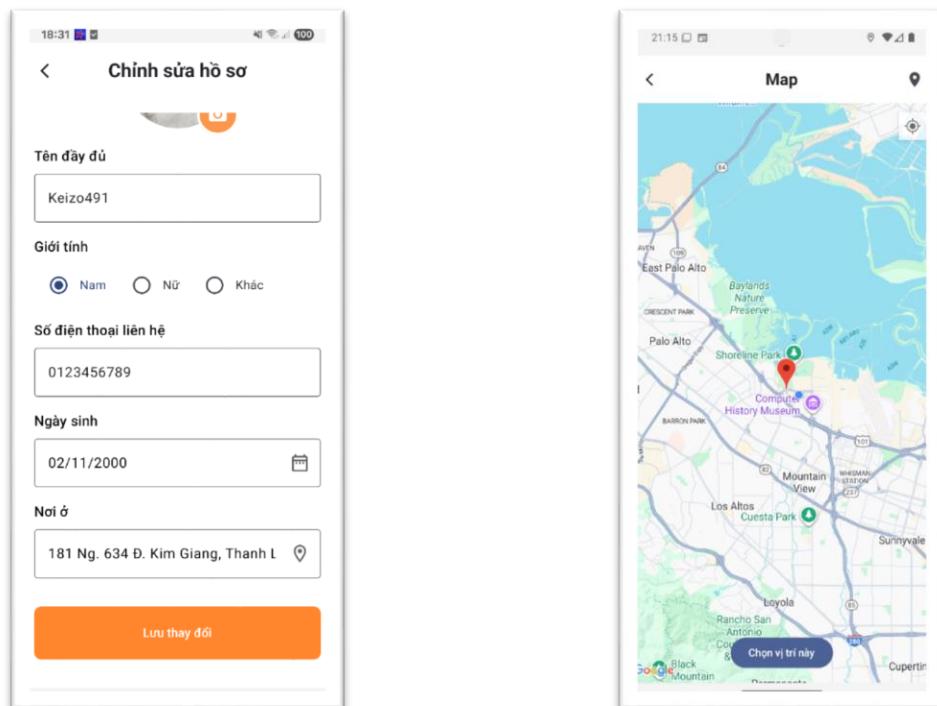


Ảnh 77. GD cài đặt

Ảnh 78. Danh sách công việc đã ứng tuyển

- Chức năng chỉnh sửa hồ sơ nhân viên

Dưới đây là giao diện nhân viên cập nhật hồ sơ.

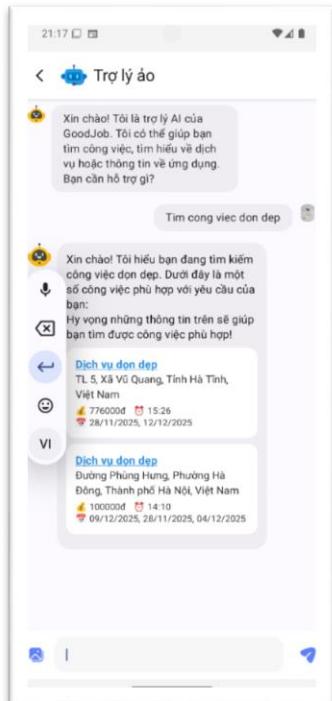


Ảnh 79. Cập nhật hồ sơ nhân viên

Ảnh 80. Chọn địa chỉ

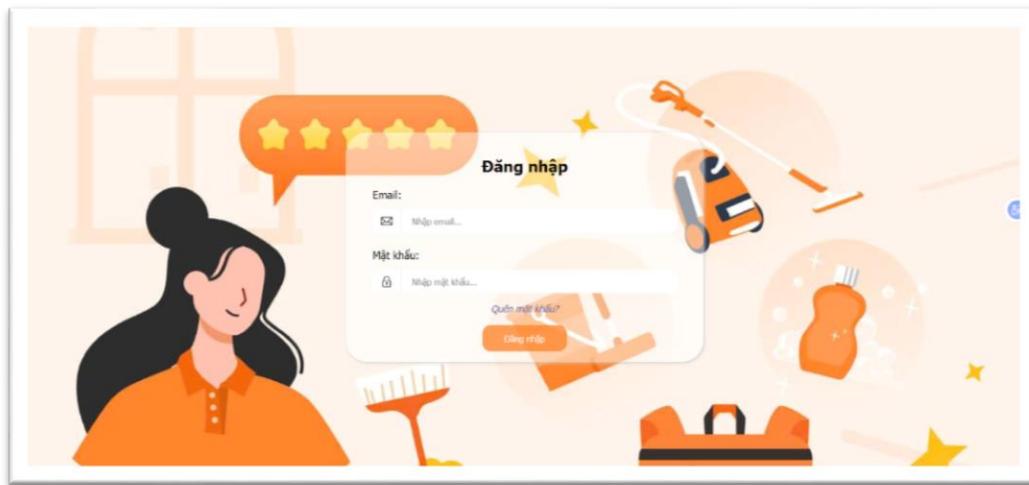
- Chức năng tư vấn bởi chatbot

Nếu gặp khó khăn trong việc lựa chọn công việc phù hợp với bản thân. Ứng dụng đã phát triển chatbot hỗ trợ tìm việc. Chatbot này sẽ tìm những công việc gần nhân viên, gợi ý những công việc tương tự trong lịch sử ứng tuyển trước đó. Ngoài ra, nó còn có thể hỗ trợ nhân viên hiểu hơn về chính sách của ứng dụng.



Ảnh 81. GD tư vấn với chatbot

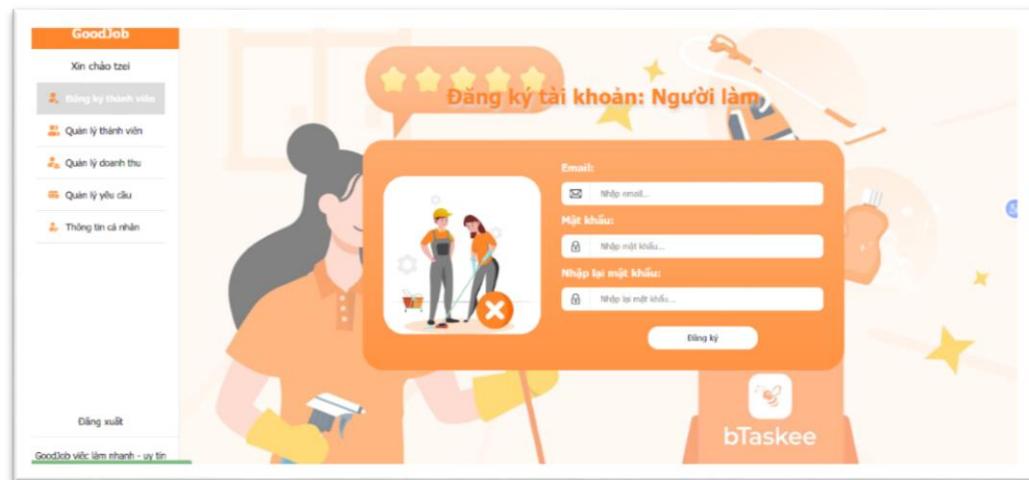
3.4.3. Giao diện cho quản trị viên



Ảnh 82. GD đăng nhập Admin



Ảnh 83. GD màn hình chính Admin



Ảnh 84. GD đăng ký nhân viên

Đồ án tốt nghiệp Đại học



Ảnh 85. GD quản lý nhân viên

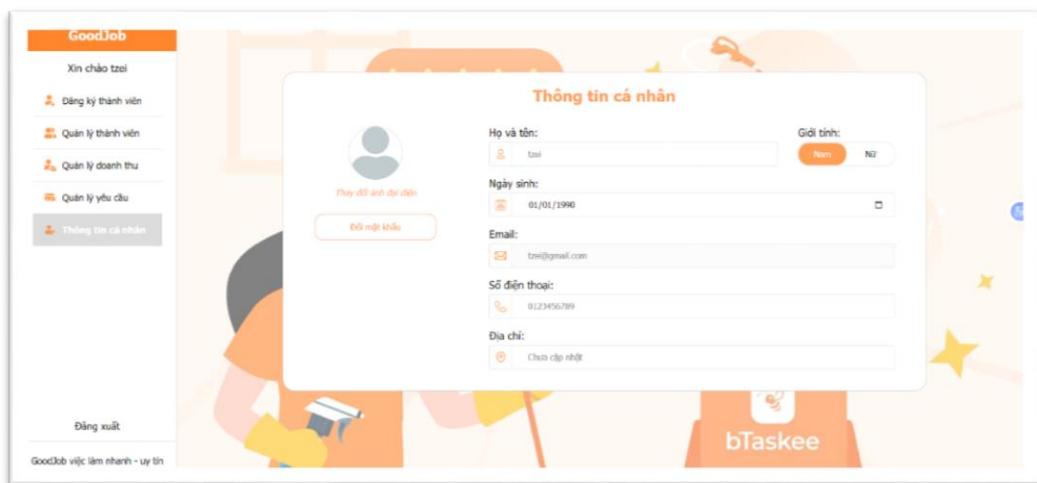


Ảnh 86. GD quản lý doanh thu



Ảnh 87. GD quản lý yêu cầu

Đồ án tốt nghiệp Đại học



Ảnh 88. GD thông tin cá nhân (Admin)

Tiểu kết chương 3

Chương 3 đã trình bày quá trình thử nghiệm và đánh giá hệ thống Helpo, bao gồm việc chuẩn bị dữ liệu thực nghiệm, triển khai cài đặt và phân tích kết quả. Hệ thống được kiểm thử với tập dữ liệu thử nghiệm gồm 1.000 công việc thuộc ba nhóm dịch vụ chính (dọn dẹp, chăm sóc sức khỏe, bảo trì thiết bị), cùng với dữ liệu người dùng và nhân viên thử nghiệm. Các phương pháp đánh giá được áp dụng gồm Precision@k, Recall@k và F1-Score để đo độ chính xác của gợi ý công việc và công thức Haversine để tính khoảng cách địa lý, đảm bảo tính thực tiễn trong việc kết nối khách hàng với người cung cấp dịch vụ.

Nhìn chung, chương 3 đã xác nhận tính khả thi và hiệu quả của giải pháp đề xuất, đồng thời tạo tiền đề cho việc mở rộng và cải tiến hệ thống trong tương lai.

PHỤ LỤC CÀI ĐẶT VÀ TRIỂN KHAI

1. Môi trường triển khai

- a. Yêu cầu phần cứng
 - CPU: Intel Core i5 trở lên hoặc tương đương
 - RAM: Tối thiểu 8GB (khuyến nghị 16GB)
 - Ổ cứng: Tối thiểu 20GB dung lượng trống
 - Kết nối mạng: Tốc độ tối thiểu 10Mbps
- b. Yêu cầu phần mềm

CÔNG CỤ/ FRAMEWORK	PHIÊN BẢN	MỤC ĐÍCH
NodeJS	v18.x trở lên	Runtime cho Backend
npm/yarn	v8.x / v1.22.x	Quản lý package
Android Studio	2023.x	Phát triển Android
Visual Studio Code	v1.x trở lên	Phát triển Backend
Postman	v11.x trở lên	Kiểm thử API
Git		Chứa nguồn code

Bảng 16. Yêu cầu phần mềm khi triển khai

2. Cài đặt và cấu hình

2.1. Cài đặt môi trường

a. Cài đặt NodeJS

Thực hiện tải NodeJS trên trang chủ, chọn Window Installer

Download Node.js®

Get Node.js® v24.12.0 (LTS) for Windows using Docker with npm

Info Want new features sooner? Get the [latest Node.js version](#) instead and try the latest improvements!

```
1 # Docker has specific installation instructions for each operating system.
2 # Please refer to the official documentation at https://docker.com/get-started/
3
4 # Pull the Node.js Docker image:
5 docker pull node:24-alpine
6
7 # Create a Node.js container and start a Shell session:
8 docker run -it --rm --entrypoint sh node:24-alpine
9
10 # Verify the Node.js version:
11 node -v # Should print "v24.12.0".
12
13 # Verify npm version:
14 npm -v # Should print "11.6.2".
```

PowerShell

 Copy to clipboard

Docker is a containerization platform. If you encounter any issues please visit [Docker's website](#)

Or get a prebuilt Node.js® for Windows running a x64 architecture.

 Windows Installer (.msi)

 Standalone Binary (.zip)

b. Cài đặt Visual Studio Code

Thực hiện tải IDE Visual Studio Code, chọn tải bản Window



The screenshot shows the Visual Studio Code download page. At the top, there is a navigation bar with links to Docs, Updates, Blog, API, Extensions, MCP, and FAQ. On the right, there is a search bar labeled 'Search Docs' and a 'Download' button. Below the navigation bar, a message says 'Version 1.107 is now available! Read about the new features and fixes from November.' A large heading 'Download Visual Studio Code' is centered, with the subtext 'Free and built on open source. Integrated Git, debugging and extensions.' Below the heading, there are three main download sections: 'Windows', 'Linux', and 'Mac'. Each section includes icons for the respective operating system, download links, and supported architectures (e.g., x64, Arm64). The 'Windows' section also lists User Installer, System Installer, .zip, and CLI options. The 'Linux' section lists .deb, .rpm, .tar.gz, Snap, and CLI options. The 'Mac' section lists .zip, Intel chip, Apple silicon, and Universal options.

c. Cài đặt Android Studio

Thực hiện tải IDE Android Studio android-studio-2025.2.2.8-windows.exe

Đồ án tốt nghiệp Đại học

Nền tảng	Gói Android Studio	Kích thước	Giá trị tổng kiểm SHA-256
Windows (64 bit)	android-studio-2025.2.2.8-windows.exe Để xuất	1,4 GB	3acc0587882f81281714098e440673f44de2ecaf75e9f005b334ec5fd2d29da8
Windows (64 bit)	android-studio-2025.2.2.8-windows.zip Không có trình cài đặt .exe	1,5 GB	d63603d6bbc433484b4059060319cc116a50c9bdf0d37610dd83ac68d77a1294
Mac (64 bit)	android-studio-2025.2.2.8-mac.dmg	1,5 GB	aa1bb1027fe0999b3154a9f017c30eaafe56b59faf7640e9ff72a361ab047f
Mac (64 bit, ARM)	android-studio-2025.2.2.8-mac_arm.dmg	1,5 GB	2e6c6eb3e911b42d08a3340bae6ad8759ddc894b4c3199272252f2343535496f
Linux (64 bit)	android-studio-2025.2.2.8-linux.tar.gz	1,5 GB	c6cf40050f5ff3f82dc5ca226647ffc44b664ce8faadbe13cbecc3bd3ff82e12b
ChromeOS	android-studio-2025.2.2.8-cros.deb	1,2 GB	aa6315049c158d488a970a7fd36592d21c7f7f535eb20e9eb2277e84e9444293

Bạn có thể tải thêm dữ liệu xuống trong [tệp lưu trữ có thể tải xuống](#). Đối với nội dung tải xuống Trình mô phỏng Android, hãy xem [Bản lưu trữ nội dung tải xuống cho trình mô phỏng](#).

d. Cài đặt Python

Công cụ hỗ trợ phát triển ứng dụng trên android

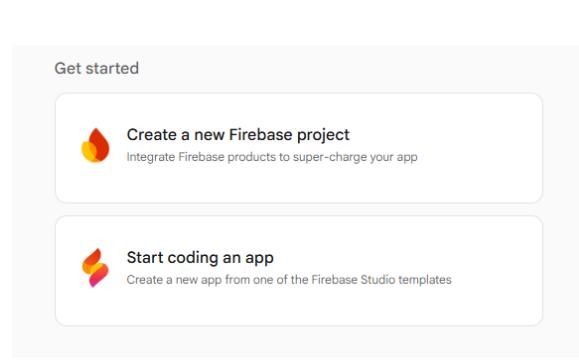


2.2. Cấu hình các công cụ khác

a. Firebase:

Thực hiện cấu hình firebase trên <https://firebase.google.com>

- Chọn “Create a new Firebase Project”, sau đó đặt tên dự án
- Sau đó cấu hình kết nối với firebase
- Thực hiện config theo mẫu dưới



```
1 const admin = require('firebase-admin');
2 const dotenv = require('dotenv');
3 dotenv.config();
4
5 const serviceAccount = JSON.parse(process.env.FB);
6 // const serviceAccount = require('../jobs-4c9e3.firebaseio-admin');
7
8 if (!admin.apps.length) {
9     admin.initializeApp({
10         credential: admin.credential.cert(serviceAccount),
11     });
12 }
13
14 const auth = admin.auth();
15 const db = admin.firestore();
16 const Timestamp = admin.firestore.Timestamp;
17
18 module.exports = { db, auth, admin, Timestamp };
```

b. Pinecone:

Thực hiện cấu hình Pinecone trên <https://www.pinecone.io>

1. Sign up / Login
2. Vào console
3. Tạo API key và Environment / Host
4. Thực hiện config theo bên dưới

The left side shows the Pinecone sign-up page with fields for email and password, and options to continue with Google, GitHub, or Microsoft. The right side shows a Python code snippet for initializing a Pinecone service with environment variables.

```
load_dotenv()

class PineconeService:

    def __init__(self):

        self.PINECONE_API_KEY = os.getenv("PINECONE_API_KEY")
        self.PINECONE_HOST = os.getenv("PINECONE_HOST")

        self.pc = Pinecone(api_key=self.PINECONE_API_KEY)
        self.index = self.pc.Index(name="demo-pinecone", host=self.PINECONE_HOST)

        self.namespace = "jobs-area"
```

c. Cloudinary:

Thực hiện cài đặt Cloudinary qua link: <https://cloudinary.com>

1. Sign up
2. Vào dashboard lấy: cloud_name, api_key, api_secret
3. Thực hiện config như hướng dẫn

The left side shows the Cloudinary sign-up page with options to sign up with email, Google, or GitHub. The right side shows a Node.js code snippet for configuring Cloudinary with environment variables.

```
1 const dotenv = require('dotenv');
2 dotenv.config();
3 const cloudinary = require('cloudinary').v2;
4
5 cloudinary.config({
6   cloud_name: process.env.CLOUD_NAME,
7   api_key: process.env.CLOUD_API_KEY,
8   api_secret: process.env.CLOUD_API_SECRET
9 });
10
11 module.exports = cloudinary;
```

2.3. Thiết lập và khởi tạo Backend

- a. Cấu hình môi trường
 - Di chuyển vào thư mục dự án backend.
 - Tạo file .env tại thư mục gốc và cấu hình các biến môi trường.
- b. Cài đặt thư viện
 - Mở terminal tại thư mục dự án, chạy lệnh cài đặt các gói phụ thuộc: npm install
 - Các thư viện chính sẽ được cài đặt:
 - + express
 - + firebase-admin
 - + cloudinary
- c. Khởi chạy Server
 - Chạy server ở chế độ development (tự động reload khi sửa code): npm start
 - Server sẽ khởi chạy tại cổng 5000 (mặc định) và kết nối tới Firebase
 - Kiểm tra log terminal để đảm bảo kết nối Database thành công

2.4. Thiết lập và khởi tạo frontend (Mobile App)

- a. Cấu hình Android Studio
 - Mở dự án ClientApp và WorkApp bằng Android Studio.
 - Đợi Gradle sync hoàn tất để tải các thư viện cần thiết.
- b. Cấu hình Gradle & Dependencies
 - Dự án sử dụng Kotlin DSL (build.gradle.kts).
 - Đảm bảo cấu hình SDK tương thích:
 - + compileSdk = 35
 - + minSdk = 31 (Android 12)
 - + targetSdk = 34
 - + jvmTarget = "11"
 - Các thư viện chính được tích hợp:
 - + Networking: Retrofit 2, OkHttp 3.
 - + Database: Room Database, Firebase Firestore/Realtime.
 - + UI/Image: Glide (load ảnh), Lottie (animation), Material Design 3.

- + Map: Mapbox Android SDK.
- c. Kết nối API Backend
 - Cấu hình địa chỉ IP của backend server trong file cấu hình retrofit (RetrofitInstance hoặc Constants).
 - Đảm bảo thiết bị chạy máy ảo (Emulator) hoặc máy thật cùng mạng Wifi với server backend.
 - API Base URL ví dụ: http://192.168.1.X:5000/api/
- d. Build và Run
 - Kết nối máy ảo hoặc thiết bị thật qua USB debugging.
 - Nhấn nút Run (Shift + F10) trên Android Studio để cài đặt APK

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết quả đạt được

- Hệ thống đã đáp ứng được các chức năng cơ bản của một ứng dụng tìm việc và đăng việc trên thiết bị di động Android
- Phát triển trang quản trị viên trên nền tảng website
- Giao diện được thiết kế để tối ưu trải nghiệm, thân thiện với người sử dụng
- Phát triển chatbot có khả năng giải đáp thắc mắc, tư vấn công việc cho người dùng

2. Những hạn chế

- Hệ thống còn đơn giản, cần phát triển thêm nhiều tính năng hỗ trợ đáp ứng với nhu cầu thực tế
- Cần cải thiện tính năng về chính sách, tiếp nhận phản hồi khi người dùng có vấn đề thắc mắc, khiếu nại
- Chatbot còn khá sơ khai, chưa đáp ứng được nhiều phản hồi người dùng trong cùng một thời điểm

3. Hướng phát triển tiếp theo

- Mở rộng hệ sinh thái dịch vụ như chăm sóc thú cưng, vận chuyển hàng, sửa chữa đồ điện tử
- Nâng cấp Chatbot nhằm tăng khả năng hiểu ngữ cảnh hội thoại, hỗ trợ đa ngôn ngữ và tích hợp tính năng gợi ý thông minh dựa trên lịch sử sử dụng.
- Tích hợp thanh toán trực tuyến đa dạng

TÀI LIỆU THAM KHẢO

- [1] "Get started with Kotlin," [Online]. Available:
kotlinlang.org/docs/getting-started.html.
- [2] "Introduction to Node.js," [Online]. Available:
<https://nodejs.org/en/learn/getting-started/introduction-to-nodejs>.
- [3] "Python Introduction," [Online]. Available:
https://www.w3schools.com/python/python_intro.asp.
- [4] "JupViec.vn," [Online]. Available: <https://jupvieu.vn>.
- [5] "RAG là gì?," AWS, [Online]. Available:
<https://aws.amazon.com/what-is/retrieval-augmented-generation/#:~:text=With%20RAG%2C%20an%20information%20retrieval,data%20to%20create%20better%20responses..>

BẢNG PHÂN CHIA CÔNG VIỆC

THÀNH VIÊN	CÔNG VIỆC THỰC HIỆN
Lê Minh Quang	- Thiết kế giao diện mobile app (dành cho nhân viên) - Phân tích thiết kế hệ thống
Đỗ Đức Thiện	- Xây dựng Backend - Recommandation System - ARAG
Phạm Thanh Sơn	- Thiết kế giao diện mobile app (dành cho khách hàng) - ARAG

Bảng 17. Phân chia công việc cho các thành viên trong nhóm