

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: CÔNG NGHỆ THÔNG TIN VIỆT NHẬT

ĐỀ TÀI:

XÂY DỰNG WEBSITE HỖ TRỢ DU LỊCH TÍCH HỢP MÔ HÌNH NGÔN NGỮ LỚN

Người hướng dẫn: TS. NGUYỄN VĂN HIỆU

Sinh viên thực hiện: NGUYỄN ĐỨC CHUNG

Số thẻ sinh viên: 102210030

Lớp: 21TCLC Nhat1

Đà Nẵng, 06/2025

ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

CHUYÊN NGÀNH: CÔNG NGHỆ THÔNG TIN VIỆT NHẬT

ĐỀ TÀI:

XÂY DỰNG WEBSITE HỖ TRỢ DU LỊCH TÍCH HỢP MÔ HÌNH NGÔN NGỮ LỚN

Người hướng dẫn: TS. NGUYỄN VĂN HIỆU

Sinh viên thực hiện: NGUYỄN ĐỨC CHUNG

Số thẻ sinh viên: 102210030

Lớp: 21TCLC_Nhat1

Đà Nẵng, 06/2025

NHẬN XÉT ĐỒ ÁN TỐT NGHIỆP

I. Thông tin chung

- Họ và tên sinh viên: Nguyễn Đức Chung
- Lớp: 21TCLC_Nhat1 Mã số sinh viên: 102210030
- Tên đề tài: Xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn
- Người hướng dẫn: TS.Nguyễn Văn Hiệu Học hàm/ học vị: Tiến sĩ

II. Nhận xét, đánh giá đồ án tốt nghiệp

- Về tính cấp thiết, tính mới, khả năng ứng dụng của đề tài: (điểm tối đa là 2đ)
.....
.....
- Về kết quả giải quyết các nội dung nhiệm vụ yêu cầu của đề án: (điểm tối đa là 4đ)
.....
.....
- Về hình thức, cấu trúc, bố cục của đồ án tốt nghiệp: (điểm tối đa là 2đ)
.....
- Đề tài có giá trị khoa học/ có bài báo/ giải quyết vấn đề đặt ra của doanh nghiệp hoặc nhà trường: (điểm tối đa là 1đ)
.....
- Các tồn tại, thiếu sót cần bổ sung, chỉnh sửa:
.....
.....

III. Tinh thần, thái độ làm việc của sinh viên: (điểm tối đa 1đ)

.....

IV. Đánh giá

- Điểm đánh giá:/10 (lấy đến 1 số lẻ thập phân)
- Đề nghị: Được bảo vệ đồ án Bổ sung để bảo vệ Không được bảo vệ
Đà Nẵng, ngày tháng năm 2025

Người hướng dẫn

TÓM TẮT

Tên đề tài: Xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn

Sinh viên thực hiện: Nguyễn Đức Chung

Số thẻ sinh viên: 102210030

Lớp: 21TCLC_Nhat1

Đề tài đồ án “Xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn” hướng tới việc tạo ra một nền tảng web hiện đại, hỗ trợ người dùng trong việc lập kế hoạch và tổ chức chuyến du lịch một cách thông minh và hiệu quả.

Hệ thống cho phép người dùng tìm kiếm và so sánh giá khách sạn, vé máy bay từ nhiều nguồn dữ liệu khác nhau, nhờ vào cơ chế tổng hợp và phân tích thông tin theo thời gian thực. Qua đó, người dùng có thể lựa chọn các dịch vụ du lịch phù hợp với nhu cầu và ngân sách cá nhân.

Điểm nổi bật của hệ thống là việc tích hợp mô hình ngôn ngữ lớn (LLM) và kiến trúc multi-agent để tạo ra các trợ lý ảo du lịch thông minh. Những agent này có thể hỗ trợ người dùng trong nhiều tình huống như:

- Gợi ý lịch trình du lịch cá nhân hóa dựa trên sở thích và thời gian.
- Tư vấn lựa chọn điểm đến, phương tiện di chuyển, và nơi lưu trú.
- Hỗ trợ giải đáp thắc mắc và xử lý các vấn đề phát sinh trong quá trình lên kế hoạch.

Trang web được xây dựng với giao diện thân thiện, tối ưu trải nghiệm người dùng và có khả năng mở rộng dễ dàng trong tương lai. Dự án kỳ vọng sẽ mang lại giá trị thực tiễn cao trong lĩnh vực công nghệ du lịch, góp phần số hóa và cá nhân hóa trải nghiệm của du khách.

Em rất mong nhận được những ý kiến đóng góp từ quý thầy cô và bạn bè để hoàn thiện hệ thống một cách tốt nhất.

NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Họ tên sinh viên: Nguyễn Đức Chung

Số thẻ sinh viên: 102210030

Lớp: 21TCLC_Nhat1. Khoa: Công nghệ thông tin.

Ngành: Công nghệ thông tin ngôn ngữ Nhật

1. Tên đề tài đồ án: Xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn
2. Đề tài thuộc diện: Có ký kết thỏa thuận sở hữu trí tuệ đối với kết quả thực hiện
3. Các số liệu và dữ liệu ban đầu: Không có.
4. Nội dung các phần thuyết minh và tính toán:

Nội dung của thuyết minh gồm:

Mở đầu: Phần mở đầu của luận văn, giới thiệu về nhu cầu thực tế và lý do thực hiện đề tài, đồng thời giới thiệu sơ lược về đề tài và mục tiêu phải đạt được, tính năng và đối tượng.

Chương 1. Cơ sở lý thuyết: trình bày những lý thuyết áp dụng vào hệ thống.

Chương 2. Phân tích thiết kế hệ thống.

Chương 3. Kết quả triển khai.

Kết luận

5. Các bản vẽ, đồ thị (ghi rõ các loại và kích thước bản vẽ): Không có.
6. Họ tên người hướng dẫn: TS. Nguyễn Văn Hiệu
7. Ngày giao nhiệm vụ đồ án: 31/03/2025
8. Ngày hoàn thành đồ án: 15/06/2025

Đà Nẵng, ngày tháng năm 2025

Trưởng Bộ môn

Người hướng dẫn

LỜI NÓI ĐẦU

Trong suốt quá trình học tập tại Trường Đại học Bách Khoa – Đại học Đà Nẵng, cũng như trong khoảng thời gian thực hiện và hoàn thành đề án tốt nghiệp, em đã nhận được rất nhiều sự giúp đỡ, hướng dẫn và động viên quý báu từ các thầy cô, gia đình, bạn bè và những người thân yêu.

Trước hết, em xin bày tỏ lòng biết ơn sâu sắc đến Thầy TS. Nguyễn Văn Hiệu – người đã truyền cảm hứng, tạo động lực học tập và định hướng chuyên môn cho em trong suốt thời gian học tại trường.

Em xin chân thành cảm ơn các Thầy, Cô trong khoa Công nghệ Thông tin, cùng với một số Thầy, Cô đến từ các khoa khác, và đặc biệt là các Thầy, Cô giảng dạy tiếng Nhật trong chương trình liên kết với Công ty Sun*. Nhờ sự giảng dạy tận tình và sự hỗ trợ quý báu từ quý Thầy Cô, em đã có được nền tảng kiến thức vững chắc để thực hiện đề án tốt nghiệp của mình.

Em cũng xin gửi lời cảm ơn chân thành đến gia đình và người thân – những người đã luôn ở bên, tin tưởng, động viên và tạo mọi điều kiện tốt nhất để em có thể yên tâm học tập và hoàn thành tốt chương trình học.

Cuối cùng, em xin cảm ơn các anh chị, bạn bè đã đồng hành, chia sẻ và giúp đỡ em trong suốt chặng đường học tập và thực hiện đề án.

Dù đã nỗ lực hết mình, nhưng chắc chắn trong quá trình học tập và làm đề án không tránh khỏi những thiếu sót. Em rất mong nhận được những góp ý quý báu từ quý Thầy Cô và các bạn để có thể hoàn thiện hơn nữa sản phẩm đề án cũng như tích lũy thêm kinh nghiệm cho chặng đường sắp tới.

Trân trọng cảm ơn!

CAM ĐOAN

Tôi xin cam đoan:

1. Báo cáo đề án tốt nghiệp: Xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn là công trình nghiên cứu của mình cá nhân tôi dưới sự hướng dẫn trực tiếp của giảng viên TS. Nguyễn Văn Hiệu.
2. Tôi đã tự đọc nghiên cứu, dịch tài liệu và tổng hợp các kiến thức đã làm nên báo cáo này và đảm bảo không sao chép ở bất cứ đâu.
3. Những lý thuyết luận văn đều được sử dụng tài liệu như tôi đã tham khảo ở phần tài liệu tham khảo đã có trong báo cáo.

Nếu có vi phạm, tôi xin chịu hoàn toàn trách nhiệm.

Sinh viên thực hiện

Nguyễn Đức Chung

MỤC LỤC

LỜI NÓI ĐẦU	i
CAM ĐOAN	ii
DANH SÁCH CÁC BẢNG, HÌNH VẼ.....	vi
MỞ ĐẦU.....	1
CHƯƠNG 1 CƠ SỞ LÝ THUYẾT	4
1.1 Tổng quan về mô hình ngôn ngữ lớn.....	4
1.1.1 Định nghĩa.....	4
1.1.2 Cơ chế hoạt động	4
1.1.3 Quá trình huấn luyện	4
1.1.4 Các mô hình nổi bật.....	5
1.1.5 Ứng dụng của LLM	5
1.1.6 Hạn chế và thách thức.....	5
1.2 Prompt Engineering	5
1.2.1 Khái niệm.....	5
1.2.2 Vai trò then chốt của Prompt Engineering	6
1.2.3 Các chiến lược prompt phổ biến.....	7
1.2.4 Nhược điểm của học dựa trên prompt	7
1.2.5 Tinh chỉnh mô hình.....	7
1.3 Tổng quan về tác nhân và hệ thống đa tác nhân	8
1.3.1 Khái niệm về tác nhân	8
1.3.2 Hệ thống đa tác nhân	8
1.3.3 Quá trình phát triển của hệ thống đa tác nhân	8
1.3.4 Các tác nhân nổi bật trong hệ thống đa tác nhân sử dụng mô hình ngôn ngữ lớn	9
1.4 Công nghệ sử dụng	11
1.4.1 LangChain.....	11

1.4.2	LangGraph	12
1.4.3	Fasitfy & Vue.js	13
1.4.4	Python	13
1.4.5	PostgreSQL.....	13
1.4.6	MongoDB	14
1.4.7	Docker.....	14
1.5	Kiểm thử phần mềm.....	14
1.5.1	Kiểm thử đơn vị (Unit testing)	14
1.5.2	Kiểm thử tự động trình duyệt	14
1.6	Kết chương.....	15
CHƯƠNG 2 PHÂN TÍCH THIẾT KẾ HỆ THỐNG.....		16
2.1	Phân tích yêu cầu	16
2.1.1	Yêu cầu chức năng.....	16
2.1.2	Yêu cầu phi chức năng.....	16
2.2	Thiết kế hệ thống	17
2.2.1	Sơ đồ ca sử dụng.....	17
2.2.2	Đặc tả ca sử dụng.....	19
2.2.3	Sơ đồ tuần tự.....	20
2.2.4	Thiết kế tác nhân.....	21
2.2.5	Thiết kế cơ sở dữ liệu	30
2.2.6	Thiết kế giao diện	31
CHƯƠNG 3 KẾT QUẢ TRIỂN KHAI.....		34
3.1	Giao diện chương trình	34
3.1.1	Giao diện tìm khách sạn	34
3.1.2	Giao diện tìm vé máy bay	35
3.1.3	Giao diện chatbot.....	36
3.2	Kiểm thử	36
3.2.1	Kiểm thử đơn vị.....	36
3.2.2	Kiểm thử tự động trình duyệt bằng selenium	37

3.3 Tinh chỉnh mô hình	39
Kết luận	40
Tài liệu tham khảo.....	41

DANH SÁCH CÁC BẢNG, HÌNH VẼ

Bảng 1.1 Các mô hình nổi bật.....	5
Bảng 2.1 Đặc tả ca sử dụng chức năng tìm kiếm khách sạn.....	19
Bảng 2.2 Đặc tả ca sử dụng chức năng tìm kiếm chuyến bay	19
Bảng 2.3 Bảng users.....	30
Bảng 2.4 Bảng hotels	31
Hình 1.1 Bing AI đưa thông tin không chính xác về superbowl.....	5
Hình 1.2 Việc gợi ý chuỗi suy nghĩ cho phép các mô hình ngôn ngữ lớn giải quyết các nhiệm vụ số học, lý luận thông thường và lý luận tượng trưng phức tạp [5].....	7
Hình 1.3 Kiến trúc React Agent	9
Hình 1.4 Kết hợp suy luận và hành động tốt hơn phương pháp chỉ sử dụng suy luận hoặc hành động riêng lẻ	10
Hình 1.5 Kiến trúc phản tư cơ bản	11
Hình 2.1 Sơ đồ ca sử dụng tổng quát	17
Hình 2.2 Phân rã sơ đồ ca sử dụng tìm kiếm khách sạn	18
Hình 2.3 Phân rã sơ đồ ca sử dụng tìm kiếm chuyến bay.....	18
Hình 2.4 Phân rã sơ đồ ca sử dụng trò chuyện với chatbot.....	18
Hình 2.5 Sơ đồ tuần tự của ca sử dụng tìm khách sạn	20
Hình 2.6 Sơ đồ tuần tự của ca sử dụng tìm chuyến bay.....	21
Hình 2.7 Biểu đồ hoạt động của tác nhân tìm kiếm địa điểm du lịch.....	22
Hình 2.8 Biểu đồ hoạt động của tác nhân tìm kiếm phương tiện di chuyển.....	23
Hình 2.9 Biểu đồ hoạt động của tác nhân tìm kiếm quán ăn	25
Hình 2.10 Biểu đồ hoạt động của tác nhân tìm kiếm khách sạn.....	26
Hình 2.11 Biểu đồ hoạt động của tác nhân lên lịch trình.....	27
Hình 2.12 Thiết kế các giao diện chức năng của hệ thống trên nền tảng Visily.....	31
Hình 2.13 Giao diện tìm kiếm và gợi ý khách sạn.....	32
Hình 2.14 Giao diện danh sách khách sạn theo bộ lọc	32
Hình 2.15 Giao diện tìm kiếm chuyến bay	33
Hình 2.16 Giao diện kết quả tìm kiếm chuyến bay.....	33
Hình 3.1 Tìm kiếm khách sạn	34
Hình 3.2 Hiện thị giá khách sạn trên nhiều nhà cung cấp khác nhau	34
Hình 3.3 Tìm kiếm vé máy bay, có thể tìm nhiều chuyến bay.....	35

Hình 3.4 Kết quả tìm kiếm vé máy bay từ nhiều nhà bán vé khác nhau	35
Hình 3.5 Lên kế hoạch du lịch	36
Hình 3.6 Chatbot sử dụng công cụ để tìm vé tàu	36
Hình 3.7 Kiểm thử công cụ tìm kiếm vé xe khách	36
Hình 3.8 Kiểm thử công cụ tìm kiếm vé tàu hỏa	37
Hình 3.9 Kiểm thử công cụ tìm kiếm vé máy bay	37
Hình 3.10 Kiểm thử giao diện đăng ký tài khoản	37
Hình 3.11 Kiểm thử giao diện đăng nhập	38
Hình 3.12 Kiểm thử giao diện tìm kiếm khách sạn	38
Hình 3.13 Tập dữ liệu mẫu để tinh chỉnh cho tác nhân tìm địa điểm du lịch	39
Hình 3.14 Kết quả tinh chỉnh mô hình Gemini 2.0 trên Google Vertex AI	39

MỞ ĐẦU

1. Lý do chọn đề tài

Trong kỷ nguyên số hóa hiện nay, ngành du lịch đang chứng kiến một sự chuyển đổi sâu sắc, nhưng vẫn tồn tại nhiều rào cản đáng kể đối với trải nghiệm du khách. Quá trình lên kế hoạch cho một chuyến đi vẫn còn phức tạp, đòi hỏi người dùng phải thu thập thông tin từ nhiều nguồn khác nhau, tự thiết kế lịch trình phù hợp, và cân nhắc vô số lựa chọn về điểm đến, hoạt động, phương tiện di chuyển và lưu trú. Điều này không chỉ tốn thời gian mà còn tạo ra cảm giác quá tải thông tin, khiến nhiều người bỏ lỡ những trải nghiệm tuyệt vời vì không có đủ kiến thức hoặc nguồn lực để lập kế hoạch hiệu quả.

Sự phát triển vượt bậc của các mô hình ngôn ngữ lớn (LLM) đã mở ra cơ hội chưa từng có để giải quyết những thách thức trong trải nghiệm du lịch. Đặc biệt, hệ thống multi-agent - nơi các agent chuyên biệt được thiết kế để phối hợp và hỗ trợ lẫn nhau - có tiềm năng cách mạng hóa cách thức con người khám phá và trải nghiệm du lịch. Các agent này có thể đóng vai trò như những chuyên gia ảo về các điểm đến, nhà hoạch định lịch trình thông minh, trợ lý tìm kiếm thông tin và tư vấn viên cá nhân, tất cả hoạt động hài hòa trong một hệ sinh thái tích hợp.

Thị trường du lịch Việt Nam, với sự phát triển mạnh mẽ và nhu cầu ngày càng tăng về các giải pháp du lịch thông minh, đang thiếu vắng một nền tảng toàn diện có khả năng kết hợp trí tuệ nhân tạo tiên tiến để mang đến trải nghiệm lập kế hoạch du lịch mượt mà và cá nhân hóa. Đề tài này ra đời nhằm lấp khoảng trống đó, tạo ra một hệ thống không chỉ giúp người dùng tìm kiếm và so sánh dịch vụ, mà còn trở thành người bạn đồng hành thông thái trong suốt hành trình khám phá của họ.

2. Mục tiêu và nhiệm vụ đề tài

Đề tài hướng đến việc phát triển một hệ sinh thái du lịch thông minh dựa trên kiến trúc multi LLM agents, nhằm cách mạng hóa trải nghiệm lập kế hoạch và thực hiện chuyến đi của người dùng.

Mục tiêu cụ thể bao gồm:

- Xây dựng nền tảng du lịch toàn diện với khả năng tương tác tự nhiên thông qua chatbot đa chức năng, giúp người dùng tiếp cận thông tin và dịch vụ du lịch một cách trực quan và hiệu quả.
- Phát triển hệ thống tư vấn và xây dựng lịch trình du lịch thông minh, có khả năng cá nhân hóa cao dựa trên sở thích, ngân sách và thời gian của người dùng.

- Tích hợp cơ chế tìm kiếm và cung cấp thông tin chi tiết về các điểm đến du lịch, bao gồm đặc trưng văn hóa, ẩm thực, địa điểm tham quan và các hoạt động trải nghiệm.
- Xây dựng hệ thống so sánh giá và dịch vụ từ nhiều nhà cung cấp khác nhau, giúp người dùng đưa ra quyết định tối ưu về chi phí và chất lượng.
- Tạo ra trải nghiệm người dùng liền mạch, trong đó họ có thể chuyển đổi mượt mà giữa việc tìm hiểu thông tin, lập kế hoạch, so sánh và đặt dịch vụ.

Nhiệm vụ:

- Xây dựng agent chuyên biệt về lịch trình du lịch, có khả năng thiết kế các hành trình phù hợp với thời gian, ngân sách, sở thích và nhu cầu đặc biệt của người dùng.
- Phát triển agent chuyên về kiến thức du lịch, có thể cung cấp thông tin chi tiết về văn hóa, lịch sử, ẩm thực, các hoạt động giải trí, và bí quyết địa phương tại các điểm đến.
- Tạo lập agent tìm kiếm và phân tích thông tin chuyên bay, với khả năng xác định các lựa chọn tối ưu về giá cả, thời gian bay, và tiện nghi.
- Thiết kế agent chuyên về đánh giá và so sánh khách sạn, tập trung vào các tiêu chí như vị trí, tiện nghi, chất lượng dịch vụ và giá trị đem lại.
- Phát triển cơ chế phối hợp và tích hợp giữa các agent để tạo ra trải nghiệm đồng nhất và thông suốt cho người dùng.
- Xây dựng hệ thống lưu trữ và quản lý lịch trình, cho phép người dùng chỉnh sửa, lưu và chia sẻ kế hoạch du lịch của họ.
- Thiết kế giao diện website thân thiện, đáp ứng trên nhiều thiết bị, với trọng tâm là tạo ra trải nghiệm tương tác tự nhiên và trực quan.

3. Đối tượng và phạm vi đề tài

Đối tượng:

- Người dùng cá nhân: Bao gồm du khách trong nước và quốc tế có nhu cầu lập kế hoạch du lịch, từ những người mới bắt đầu đến những du khách có kinh nghiệm.

Phạm vi:

- Khu vực địa lý: Tập trung vào thị trường du lịch Việt Nam và các điểm đến quốc tế phổ biến với du khách Việt Nam.
- Ngôn ngữ hỗ trợ: Tiếng Việt là ngôn ngữ chính, có hỗ trợ tiếng Anh cho du khách quốc tế.
- Loại hình du lịch: Hỗ trợ đa dạng loại hình du lịch như văn hóa, ẩm thực, mạo hiểm, nghỉ dưỡng, và du lịch xanh.

- Dịch vụ hỗ trợ: Thông tin chuyến bay nội địa và quốc tế, đánh giá và đặt phòng khách sạn, lập lịch trình du lịch, thông tin về điểm đến và hoạt động địa phương.
- Tính năng so sánh: Khả năng so sánh giá cả và dịch vụ từ các nhà cung cấp lớn trong nước.

4. Phương pháp thực hiện

- Nghiên cứu và phân tích yêu cầu
 - Khảo sát nhu cầu người dùng: Thực hiện khảo sát và phỏng vấn du khách trong nước và quốc tế để xác định các nhu cầu cụ thể như lập lịch trình, tìm kiếm thông tin điểm đến, và so sánh dịch vụ. Thu thập dữ liệu về sở thích, ngân sách, và thách thức mà họ gặp phải.
 - Phân tích hệ thống hiện có: Nghiên cứu các nền tảng du lịch trực tuyến phổ biến (như TripAdvisor, Booking.com) để đánh giá điểm mạnh, điểm yếu và xác định các tính năng độc đáo mà hệ thống mới có thể cung cấp.
 - Xác định yêu cầu: Lập danh sách yêu cầu chức năng (ví dụ: lập lịch trình, tư vấn điểm đến) và phi chức năng (ví dụ: hiệu suất, bảo mật) dựa trên kết quả khảo sát và phân tích.
- Thiết kế kiến trúc hệ thống
 - Mô hình multi-agent: Thiết kế hệ thống với các agent chuyên biệt, bao gồm:
 - Agent lịch trình: Thiết kế để tạo lịch trình tối ưu dựa trên sở thích, thời gian và ngân sách của người dùng.
 - Agent kiến thức du lịch: Cung cấp thông tin về văn hóa, ẩm thực, và hoạt động tại các điểm đến.
 - Agent chuyên bay: Phân tích dữ liệu chuyến bay để đưa ra lựa chọn tốt nhất về giá cả và tiện nghi.
 - Agent khách sạn: So sánh khách sạn dựa trên vị trí, tiện nghi và đánh giá.
 - Hệ thống quản lý agent: Phát triển một orchestrator để điều phối hoạt động của các agent, đảm bảo chúng phối hợp hiệu quả và cung cấp kết quả đồng nhất.
- Phát triển các LLM agents
 - Lựa chọn mô hình ngôn ngữ lớn: Sử dụng các LLM tiên tiến như GPT-3, BERT hoặc các mô hình tương tự để xây dựng các agent có khả năng hiểu và phản hồi ngôn ngữ tự nhiên.
 - Huấn luyện và tinh chỉnh: Fine-tune các mô hình trên dữ liệu du lịch cụ thể (ví dụ: mô tả điểm đến, đánh giá khách sạn) để tăng độ chính xác và phù hợp với bối cảnh du lịch.

CHƯƠNG 1 CƠ SỞ LÝ THUYẾT

1.1 Tổng quan về mô hình ngôn ngữ lớn

1.1.1 Định nghĩa

Mô hình ngôn ngữ lớn (LLM) là một loại mô hình học sâu được huấn luyện trên lượng lớn dữ liệu văn bản để hiểu, sinh và thao tác ngôn ngữ tự nhiên. LLM sử dụng kiến trúc Transformer, được giới thiệu lần đầu bởi Vaswani et al. (2017) trong bài báo “Attention is All You Need” [1]. Kiến trúc này thay thế hoàn toàn cơ chế tuần tự của RNN và LSTM trước đây bằng cơ chế self-attention, giúp cải thiện khả năng học quan hệ dài hạn giữa các từ trong câu.

1.1.2 Cơ chế hoạt động

1. Mã hóa ngữ cảnh bằng Self-Attention

Trong LLM, self-attention giúp mô hình học được mối quan hệ giữa các từ trong cùng một câu hoặc đoạn văn, cho phép hiểu ngữ cảnh sâu sắc hơn. Ví dụ, trong câu “Con mèo đang nằm trên thảm, nó đang ngủ”, LLM có thể hiểu “nó” đề cập đến “con mèo”.

2. Kiến trúc Transformer

Kiến trúc Transformer bao gồm hai phần chính:

- Encoder: xử lý văn bản đầu vào (dùng trong các mô hình như BERT).
- Decoder: sinh văn bản đầu ra (dùng trong các mô hình như GPT).

Các LLM hiện đại như GPT-3, GPT-4 sử dụng decoder-only Transformer, nghĩa là chỉ dựa vào phần sinh ngôn ngữ từ trái sang phải.

3. Dự đoán xác suất có điều kiện

LLM học cách dự đoán từ tiếp theo dựa trên chuỗi từ trước đó. Điều này được gọi là mô hình ngôn ngữ tự hồi quy (autoregressive language modeling).

1.1.3 Quá trình huấn luyện

LLM được huấn luyện trên hàng tỷ token (đơn vị từ/câu) từ các nguồn dữ liệu như sách, trang web, tài liệu kỹ thuật, v.v. Huấn luyện thường bao gồm hai giai đoạn:

Pretraining: mô hình học cấu trúc ngôn ngữ chung trên tập dữ liệu lớn không gán nhãn.

Fine-tuning (tinh chỉnh): mô hình được tinh chỉnh để thực hiện các nhiệm vụ cụ thể như trả lời câu hỏi, tóm tắt, dịch thuật,...

Một số LLM hiện đại còn sử dụng Instruction Tuning và Reinforcement Learning with Human Feedback (RLHF) như trong GPT-4

1.1.4 Các mô hình nổi bật

Bảng 1.1 Các mô hình nổi bật

Mô hình	Tổ chức phát triển	Thời điểm ra mắt	Quy mô	Mô hình
BERT	Google	2018	340M tham số	BERT
GPT-3	OpenAI	2020	175B tham số	GPT-3
PaLM	Google	2022	540B tham số	PaLM
GPT-4	OpenAI	2023	Ước tính > 1T tham số	GPT-4
Claude, Gemini	Anthropic, Google DeepMind	2023–2024	Lớn, chưa công bố đầy đủ	Claude, Gemini

1.1.5 Ứng dụng của LLM

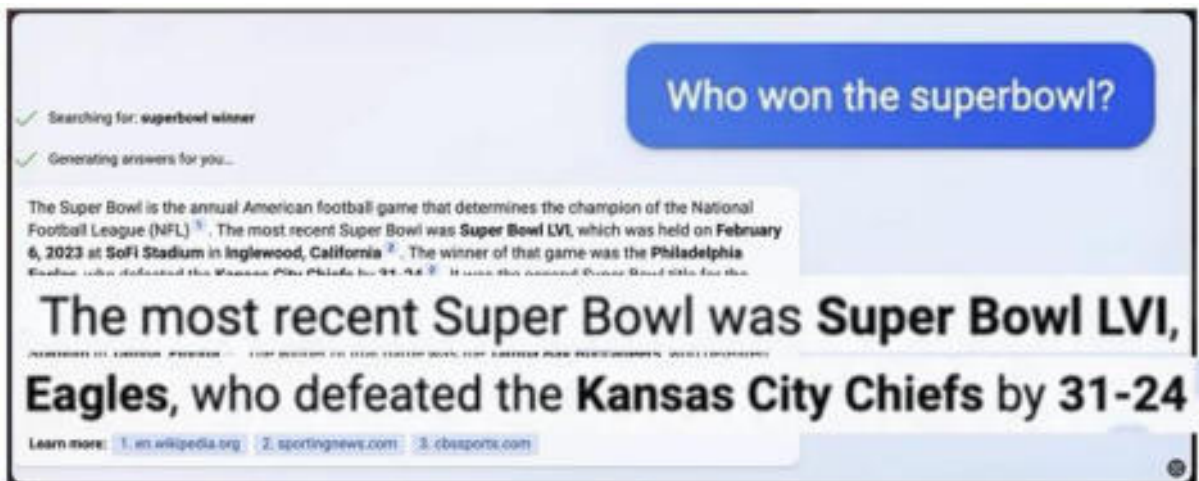
LLM hiện nay đã được ứng dụng rộng rãi trong nhiều lĩnh vực như:

- Xử lý ngôn ngữ tự nhiên (NLP): dịch máy, tóm tắt văn bản, phân loại văn bản.
- Trợ lý ảo: như ChatGPT, Claude, Gemini.
- Viết code và hỗ trợ lập trình: GitHub Copilot, Code Llama.
- Hệ thống đa tác nhân: phối hợp nhiều LLM để giải bài toán phức tạp.

1.1.6 Hạn chế và thách thức

Dù rất mạnh, LLM vẫn còn một số điểm yếu:

- Ảo giác thông tin: có thể “bịa” ra thông tin không chính xác.
- Chi phí tính toán lớn: yêu cầu tài nguyên huấn luyện và vận hành rất cao.
- Tính minh bạch: khó lý giải cách mô hình đưa ra kết quả.
- Bảo mật và đạo đức: có nguy cơ bị lạm dụng để tạo nội dung sai lệch.



Hình 1.1 Bing AI đưa thông tin không chính xác về superbowl

1.2 Prompt Engineering

1.2.1 Khái niệm

Prompt Engineering là kỹ thuật thiết kế các lệnh đầu vào (prompt) nhằm hướng dẫn mô hình ngôn ngữ lớn (LLM) tạo ra đầu ra mong muốn. Thay vì huấn luyện lại mô hình

hoặc điều chỉnh tham số nội bộ, prompt engineering tận dụng tri thức đã học sẵn để "định hướng" hành vi của mô hình qua cú pháp và cấu trúc đầu vào.

Với sự phát triển mạnh mẽ của các LLM như GPT-4, Claude hoặc Gemini, prompt không còn là đơn giản một câu hỏi, mà đã trở thành một “giao diện lập trình mềm dẻo” để khai thác sức mạnh tiềm ẩn của mô hình.

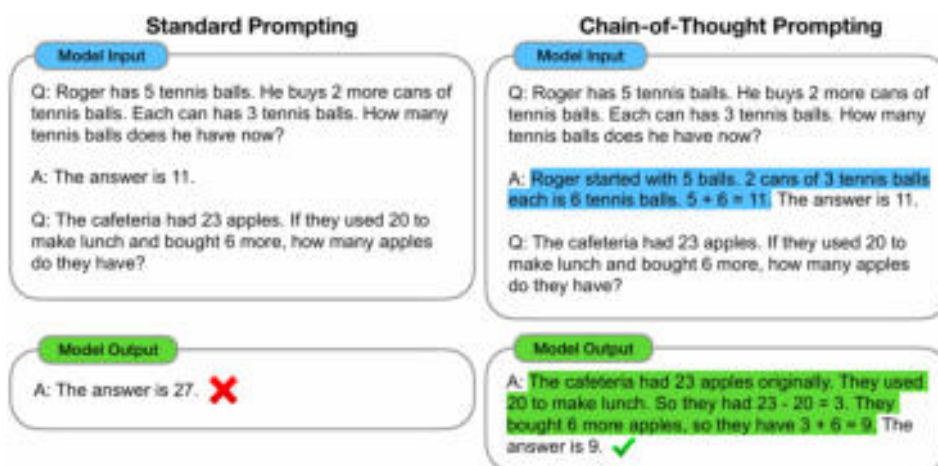
1.2.2 Vai trò then chốt của Prompt Engineering

Trong thực tế triển khai, prompt engineering được coi là yếu tố quan trọng nhất ảnh hưởng đến chất lượng đầu ra của LLM. Một mô hình mạnh đến đâu cũng có thể tạo ra kết quả kém nếu prompt được viết mơ hồ, không rõ ràng hoặc không phù hợp với ngữ cảnh. Ngược lại, một prompt tốt có thể:

- Giảm thiểu lỗi sai hoặc thông tin không chính xác.
- Tăng khả năng suy luận đa bước và phân tích logic.
- Cải thiện độ mạch lạc và độ chính xác của câu trả lời.
- Tái sử dụng mô hình đa nhiệm mà không cần huấn luyện thêm.

Các nghiên cứu gần đây đã chứng minh rằng việc áp dụng kỹ thuật prompt engineering có thể cải thiện đáng kể hiệu suất của các mô hình ngôn ngữ lớn (LLM) trên nhiều tác vụ khác nhau:

- Nghiên cứu của Cole (2024): Trong bài viết "26 principles for prompt engineering to increase LLM accuracy 57%", tác giả đã tổng hợp 26 nguyên tắc thiết kế prompt giúp tăng độ chính xác của mô hình LLaMA-1/2 lên đến 57% và GPT-4 lên đến 67% trên các tác vụ xử lý ngôn ngữ tự nhiên.
- Nghiên cứu của Sahoo et al. (2024): các tác giả đã khảo sát hệ thống các kỹ thuật prompt engineering và ứng dụng của chúng trong các lĩnh vực như chăm sóc sức khỏe, giáo dục và khoa học vật liệu. Nghiên cứu nhấn mạnh rằng prompt engineering có thể cải thiện đáng kể độ chính xác và độ tin cậy của LLM trong các tác vụ chuyên biệt. [2]
- Nghiên cứu của Xiong et al. (2024): các tác giả đã đánh giá các kỹ thuật prompt engineering như Chain-of-Thought và Self-Probing, cho thấy rằng các kỹ thuật này có thể cải thiện độ chính xác và hiệu chuẩn độ tin cậy của mô hình trong các tác vụ hiểu ngôn ngữ tự nhiên. [3]
- Nghiên cứu của Zhou et al. (2022): các tác giả đã đề xuất phương pháp Automatic Prompt Engineer (APE) để tự động tạo và lựa chọn prompt hiệu quả, giúp cải thiện hiệu suất của LLM trên 24 tác vụ xử lý ngôn ngữ tự nhiên. [4]



Hình 1.2 Việc gợi ý chuỗi suy nghĩ cho phép các mô hình ngôn ngữ lớn giải quyết các nhiệm vụ số học, lý luận thông thường và lý luận tượng trưng phức tạp [5]

1.2.3 Các chiến lược prompt phổ biến

- Zero-shot Prompting: Yêu cầu mô hình thực hiện nhiệm vụ mà không cần ví dụ minh họa. Chiến lược này đòi hỏi prompt rõ ràng, cụ thể.
- Few-shot Prompting: Cung cấp một vài ví dụ ngay trong prompt để mô hình học theo cấu trúc hoặc phong cách mong muốn.
- Chain-of-thought Prompting: Hướng dẫn mô hình giải thích từng bước suy luận trước khi đưa ra câu trả lời cuối cùng. Rất hiệu quả trong các bài toán logic, toán học, hoặc lập luận.
- Instruction-based Prompting: Sử dụng câu lệnh trực tiếp như “Giải thích”, “So sánh”, “Tóm tắt”. Hình thức này thường đi kèm các quy tắc định dạng cụ thể.
- Role-based Prompting: Gán vai trò cho mô hình để điều chỉnh văn phong và nội dung, ví dụ: “Bạn là bác sĩ”, “Bạn là lập trình viên Python chuyên nghiệp”.

1.2.4 Nhược điểm của học dựa trên prompt

- Thiếu hiệu quả: Prompt cần được xử lý mỗi lần mô hình đưa ra dự đoán.
- Hiệu suất kém: Prompt thường cho kết quả tệ hơn so với phương pháp fine-tuning. [6]
- Nhạy cảm: Phụ thuộc vào cách diễn đạt của prompt [7], thứ tự của các ví dụ. [8][9]
- Thiếu rõ ràng: Không rõ mô hình học được gì từ prompt. Thậm chí, các nhãn ngẫu nhiên cũng có thể hoạt động. [10] [11]

1.2.5 Tinh chỉnh mô hình

Tinh chỉnh (fine-tuning) là quá trình điều chỉnh lại một mô hình ngôn ngữ đã được huấn luyện trước trên một tập dữ liệu mới, nhằm cải thiện hiệu suất cho các tác vụ cụ thể. Thay vì huấn luyện lại mô hình từ đầu (tốn kém và mất thời gian), tinh chỉnh cho phép mô hình tận dụng kiến thức tổng quát đã học trước đó và thích nghi với ngữ cảnh

mới. Phương pháp này thường được sử dụng để cải thiện độ chính xác trên các nhiệm vụ như phân loại, sinh văn bản, hỏi đáp, tóm tắt, v.v.

1.2.5.1 Supervised Fine-tuning

Supervised Fine-tuning (SFT) là một hình thức fine-tuning trong đó mô hình được huấn luyện lại với các cặp dữ liệu đầu vào - đầu ra có gán nhãn rõ ràng. Đây là bước đầu tiên trong quá trình tinh chỉnh một mô hình ngôn ngữ lớn (LLM) sau khi huấn luyện nền. Trong giai đoạn này, mô hình học cách tái hiện các phản hồi mong muốn từ con người thông qua dữ liệu hướng dẫn cụ thể (ví dụ: câu hỏi và câu trả lời đúng). SFT đóng vai trò quan trọng trong việc định hình hành vi cơ bản của mô hình trước khi thực hiện các bước tối ưu nâng cao hơn.

1.3 Tổng quan về tác nhân và hệ thống đa tác nhân

1.3.1 Khái niệm về tác nhân

Tác nhân (*agent*) là một thực thể có khả năng nhận biết môi trường, đưa ra quyết định, và thực hiện hành động nhằm đạt mục tiêu đã định. Các tác nhân có thể hoạt động độc lập hoặc trong mối quan hệ với các tác nhân khác. Jennings và cộng sự định nghĩa tác nhân như một hệ thống máy tính tự động hoạt động trong một môi trường để đạt các mục tiêu được chỉ định [12]

1.3.2 Hệ thống đa tác nhân

Hệ thống đa tác nhân (*Multi-Agent System* - MAS) là tập hợp các tác nhân tương tác, phối hợp hoặc cạnh tranh để giải quyết các vấn đề mà một tác nhân đơn lẻ không thể giải quyết hiệu quả. MAS thường được sử dụng trong các lĩnh vực như điều phối robot, quản lý mạng phân tán, hệ thống đề xuất và trò chơi chiến lược [13]. Các đặc tính cơ bản của MAS bao gồm phân quyền, tương tác, phối hợp và học thích nghi.

1.3.3 Quá trình phát triển của hệ thống đa tác nhân

- Giai đoạn khởi đầu (trước 1990):

Trước năm 1990, các nghiên cứu về trí tuệ nhân tạo chủ yếu tập trung vào hệ thống chuyên gia và suy luận logic. Khái niệm về “agent” chủ yếu được hiểu trong ngữ cảnh của AI truyền thống, chưa có mô hình rõ ràng.

- Giai đoạn hình thành (1990–2000)

Đầu những năm 1990, khái niệm “agent” được định hình rõ hơn trong cộng đồng AI. Một đóng góp quan trọng là mô hình BDI (Belief–Desire–Intention) được phát triển như một mô hình lý thuyết cho hành vi của tác nhân tự trị [14]. Năm 1996, ngôn ngữ AgentSpeak(L) được phát triển dựa trên mô hình BDI, cho phép mô tả hành vi tác nhân một cách hình thức và khả thi [15].

Jennings et al. sau đó đã công bố lộ trình phát triển nghiên cứu về tác nhân, tổng hợp các hướng tiếp cận và thách thức chính [12].

- Giai đoạn hình thành (1990–2000)

Trong giai đoạn này, MAS được áp dụng rộng rãi trong các hệ thống thực tế như thương mại điện tử, giám sát, và hệ thống phân tán. Một số framework phát triển MAS nổi bật như JADE được sử dụng trong nhiều dự án công nghiệp [16]

- Giai đoạn hiện đại (2010–nay)

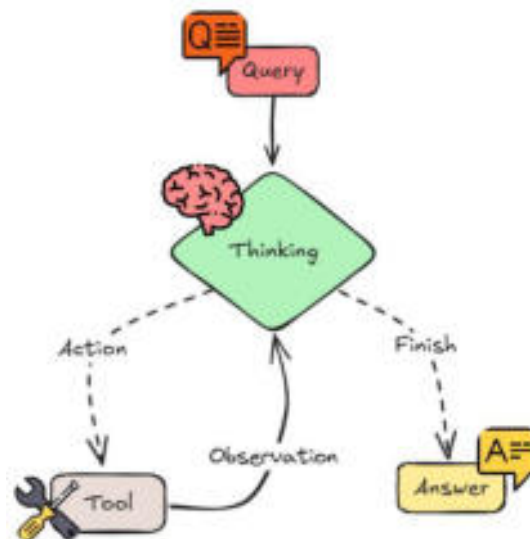
Với sự bùng nổ của học sâu và mô hình ngôn ngữ lớn (LLMs), các hệ thống đa tác nhân hiện nay được tích hợp với các mô hình học để trở nên linh hoạt hơn, tự thích nghi và hợp tác hiệu quả hơn. Nhiều nghiên cứu gần đây đề xuất các hệ thống sử dụng nhiều tác nhân LLM để tự động hoá quy trình như trong nghiên cứu Robin Agent [17]

1.3.4 Các tác nhân nổi bật trong hệ thống đa tác nhân sử dụng mô hình ngôn ngữ lớn

1.3.4.1 ReAct Agent: Kết hợp suy luận và hành động

ReAct (Reasoning and Acting) là một phương pháp tiên phong kết hợp giữa suy luận chuỗi tư duy (*chain-of-thought*) và hành động tương tác trong môi trường. Thay vì chỉ suy luận hoặc chỉ hành động, ReAct cho phép mô hình LLM tạo ra chuỗi suy luận và hành động xen kẽ, giúp mô hình vừa lập kế hoạch, vừa tương tác với môi trường để thu thập thông tin và điều chỉnh hành vi [18].

ReAct Agent architecture



Hình 1.3 Kiến trúc React Agent

Trong nghiên cứu của Yao et al. (2022), ReAct đã được áp dụng thành công trong các nhiệm vụ như trả lời câu hỏi (HotpotQA) và xác minh sự thật (Fever), cho thấy hiệu quả vượt trội so với các phương pháp chỉ sử dụng suy luận hoặc hành động riêng lẻ [18].



Hình 1.4 Kết hợp suy luận và hành động tốt hơn phương pháp chỉ sử dụng suy luận hoặc hành động riêng lẻ

1.3.4.2 Reflection

Reflection là một chiến lược nhắc lệnh (prompting strategy) được sử dụng nhằm nâng cao chất lượng và tỷ lệ thành công của các tác nhân và các hệ thống tương tự. Phương pháp này bao gồm việc nhắc một mô hình ngôn ngữ lớn (LLM) tự phản tư và đánh giá các hành động trong quá khứ của chính nó, đôi khi có kết hợp thêm thông tin bên ngoài như quan sát từ công cụ mà nó sử dụng.

Trong nghiên cứu của He et al. (2024), khung làm việc RR-MP (Reactive and Reflection agents with Multi-Path Reasoning) đã được đề xuất, kết hợp tác nhân phản ứng và tác nhân phản tư trong các đường suy luận song song. Kết quả cho thấy phương pháp này cải thiện đáng kể độ chính xác trong các nhiệm vụ suy luận khoa học phức tạp [19].

- Ví dụ đơn giản về Reflection Agent

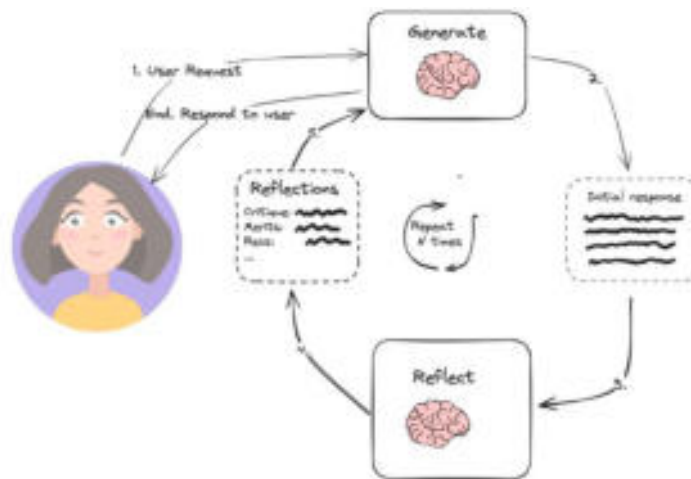
Ví dụ đơn giản này kết hợp hai lần gọi mô hình ngôn ngữ lớn (LLM): một bộ tạo nội dung (generator) và một bộ phản tư (reflector).

- Generator (bộ tạo): cố gắng phản hồi trực tiếp theo yêu cầu của người dùng.
- Reflector (bộ phản tư): được nhắc lệnh để đóng vai một giáo viên, đưa ra phê bình mang tính xây dựng đối với phản hồi ban đầu của generator.

Cách hoạt động của vòng lặp, quá trình diễn ra theo một số vòng lặp cố định:

Generator tạo ra phản hồi ban đầu.

- Reflector đánh giá phản hồi đó, nêu điểm mạnh/yếu và đề xuất cải tiến.
- Generator có thể chỉnh sửa dựa trên phản hồi vừa nhận.
- Quá trình lặp lại theo số lần định sẵn.
- Kết quả cuối cùng sau các vòng lặp sẽ được trả về như phản hồi cuối cùng cho người dùng.



Hình 1.5 Kiến trúc phản tư cơ bản

1.4 Công nghệ sử dụng

1.4.1 LangChain

LangChain là một framework mã nguồn mở được thiết kế nhằm đơn giản hóa việc xây dựng các ứng dụng dựa trên mô hình ngôn ngữ lớn (LLM), đặc biệt là những ứng dụng cần sự tương tác động, ghi nhớ ngữ cảnh, và truy xuất thông tin ngoài mô hình. Framework này được phát triển bởi Harrison Chase từ năm 2022 và nhanh chóng trở thành một công cụ tiêu chuẩn trong cộng đồng phát triển LLM.

1.4.1.1 Mục tiêu và triết lý

LangChain được xây dựng dựa trên quan sát rằng các ứng dụng LLM hiệu quả thường có chung những đặc điểm sau:

- Tích hợp các nguồn dữ liệu bên ngoài (như cơ sở tri thức, API, tệp văn bản,...)
- Duy trì ngữ cảnh hội thoại qua nhiều vòng tương tác
- Tự động hóa luồng xử lý bằng cách tổ chức nhiều bước suy luận hoặc hành động

1.4.1.2 Các thành phần chính

- Prompt Templates: Cho phép định nghĩa và sử dụng các mẫu nhắc lệnh có thể tái sử dụng, hỗ trợ cả các tham số động.
- LLMs & ChatModels: Hỗ trợ nhiều mô hình khác nhau như OpenAI (GPT-3.5, GPT-4), Anthropic, Cohere, Hugging Face, v.v.
- Chains: Chuỗi các hành động hoặc bước xử lý. Ví dụ: nhận câu hỏi → trích xuất thông tin → sinh câu trả lời.
- Memory: Lưu trữ ngữ cảnh hoặc lịch sử hội thoại để duy trì trạng thái giữa các vòng tương tác.
- Agents: Tác nhân có khả năng lựa chọn hành động tiếp theo dựa trên mục tiêu, phản hồi của mô hình và các công cụ có sẵn.

- Tools & Toolkits: Tích hợp các công cụ như trình duyệt, máy tính, cơ sở dữ liệu (SQL, FAISS, Chroma), API,... vào quy trình xử lý của agent.

1.4.2 LangGraph

LangGraph là một framework mở rộng của LangChain, cho phép thiết kế và triển khai các ứng dụng trí tuệ nhân tạo dựa trên mô hình ngôn ngữ lớn (LLM) theo dạng đồ thị trạng thái có vòng lặp (cyclical state graphs). LangGraph đặc biệt phù hợp cho việc xây dựng các hệ thống phức tạp như hệ thống đa tác nhân (multi-agent systems), các tiến trình cần lưu trạng thái hoặc có logic phân nhánh, phản xạ và lặp lại.

Framework này được phát triển với mục tiêu hỗ trợ mô hình hóa quy trình có nhiều bước, trong đó từng hành động có thể phụ thuộc vào kết quả trước đó và có thể tự động quay lại các bước trước nếu cần thiết.

1.4.2.1 Nguyên lý hoạt động

LangGraph sử dụng kiến trúc đồ thị định hướng, trong đó:

- Mỗi nút (node) trong đồ thị là một hành động hoặc hàm xử lý cụ thể, ví dụ như gọi mô hình LLM, tìm kiếm thông tin, phân tích kết quả, hoặc thực hiện một tác vụ logic.
- Cạnh (edge) là các luật chuyển trạng thái, xác định điều kiện để chuyển từ nút hiện tại sang nút tiếp theo dựa trên phản hồi nhận được.
- Trạng thái (state) là đối tượng lưu trữ toàn bộ dữ liệu đang được xử lý, có thể cập nhật liên tục tại mỗi bước trong quá trình chạy đồ thị.

Khác với LangChain vốn sử dụng luồng tuần tự, LangGraph hỗ trợ xử lý vòng lặp, điều kiện dừng tùy chỉnh và tái định tuyến, phù hợp với các hệ thống có hành vi động và phi tuyến.

1.4.2.2 Các thành phần chính

- State Graph: Biểu diễn toàn bộ quy trình xử lý dưới dạng đồ thị các bước và hướng chuyển tiếp.
- Node: Một hàm xử lý hoặc hành động cụ thể do người dùng định nghĩa (thường là các lệnh gọi tới LLM hoặc tác nhân).
- Edge Logic: Luật điều hướng chuyển tiếp giữa các node.
- State Object: Trạng thái dùng chung, có thể thay đổi theo tiến trình thực thi.

1.4.2.3 Ưu điểm nổi bật

- Cho phép xây dựng quy trình phức tạp có nhánh rẽ, vòng lặp hoặc các luồng đồng thời.
- Hỗ trợ triển khai hệ thống đa tác nhân trong đó mỗi tác nhân có thể hoạt động độc lập nhưng vẫn chia sẻ trạng thái chung.
- Dễ tích hợp với các thành phần khác của LangChain như memory, retriever, công cụ tìm kiếm, cơ sở dữ liệu.

- Có thể kiểm soát rõ ràng và mở rộng logic xử lý theo mô hình “hành vi theo trạng thái”.

1.4.3 *Fastify & Vue.js*

JavaScript là ngôn ngữ lập trình tiêu chuẩn cho phía người dùng trong phát triển trang web. Kết hợp với Vue.js – một khuôn khổ phát triển hiện đại theo hướng tiến hóa – và Fastify – một khung máy chủ nhẹ, hiệu suất cao cho phía máy chủ, bộ đôi này cung cấp nền tảng vững chắc để xây dựng các ứng dụng web tương tác, hiệu quả và dễ bảo trì. Trong hệ thống, Vue.js và Fastify đóng vai trò chính trong việc thiết kế giao diện người dùng và xử lý phía máy chủ.

- Vue.js được sử dụng để xây dựng giao diện trang web hiện đại, có tính tương tác cao.
- Phát triển các trang chức năng như tìm kiếm địa điểm, gợi ý lịch trình, giao diện trò chuyện với mô hình ngôn ngữ.
- Fastify được sử dụng làm máy chủ xử lý phía backend, đảm nhận các yêu cầu từ frontend, kết nối tới mô hình ngôn ngữ lớn, và trả kết quả một cách nhanh chóng và hiệu quả.
- Fastify hỗ trợ mở rộng tốt, tích hợp dễ dàng với các API và hệ thống bên ngoài trong quá trình xử lý dữ liệu du lịch.

1.4.4 *Python*

Python là một ngôn ngữ lập trình đa năng, nổi bật với cú pháp rõ ràng, dễ đọc, cộng đồng lớn và hệ sinh thái thư viện phong phú. Trong dự án xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn, Python giữ vai trò cốt lõi trong việc xây dựng luồng xử lý nghiệp vụ và cung cấp các giao diện lập trình ứng dụng (RESTful API) phục vụ giao tiếp với phân giao diện người dùng.

- Các nhiệm vụ chính:
 - Xây dựng luồng xử lý bằng LangGraph để tổ chức và điều phối các bước tương tác với mô hình ngôn ngữ lớn.
 - Gọi và xử lý kết quả từ mô hình ngôn ngữ lớn.
 - Tiền xử lý và hậu xử lý dữ liệu đầu vào và đầu ra.
- Sử dụng FastAPI để xây dựng các RESTful API, đảm bảo kết nối hiệu quả giữa phần xử lý và giao diện người dùng.

1.4.5 *PostgreSQL*

PostgreSQL là một hệ quản trị cơ sở dữ liệu quan hệ mã nguồn mở mạnh mẽ, nổi tiếng với khả năng mở rộng, bảo mật và hỗ trợ tốt các loại truy vấn phức tạp. Trong hệ thống website hỗ trợ du lịch tích hợp LLM, PostgreSQL đảm nhận vai trò là nền tảng lưu trữ và truy xuất các cuộc hội thoại, cấu hình đồ thị quy trình, đảm bảo tính nhất quán, toàn vẹn và hiệu suất trong quá trình vận hành hệ thống.

1.4.6 MongoDB

MongoDB là hệ quản trị cơ sở dữ liệu NoSQL phổ biến, cho phép lưu trữ dữ liệu dưới dạng tài liệu BSON (Binary JSON). Đây là một giải pháp lý tưởng cho các ứng dụng hiện đại cần tính linh hoạt, mở rộng dễ dàng và hiệu suất cao, đặc biệt là khi làm việc với dữ liệu phi cấu trúc hoặc bán cấu trúc.

- Dùng cho dữ liệu linh hoạt và thay đổi thường xuyên như kết quả ánh xạ các khách sạn từ các nguồn khác nhau.
- Tận dụng khả năng lưu dữ liệu dạng JSON, dễ dàng tích hợp với Javascript backend.

1.4.7 Docker

Docker là một nền tảng mã nguồn mở giúp đóng gói, phân phối và chạy ứng dụng bên trong các container – những môi trường ảo hóa nhẹ, độc lập, chạy được trên mọi hệ điều hành có cài Docker Engine. Trong dự án xây dựng website hỗ trợ du lịch tích hợp mô hình ngôn ngữ lớn, Docker đóng vai trò trung tâm trong việc chuẩn hóa môi trường phát triển và đảm bảo khả năng triển khai dễ dàng, linh hoạt trên bất kỳ máy chủ hay dịch vụ cloud nào.

1.5 Kiểm thử phần mềm

Kiểm thử phần mềm là quá trình đánh giá và xác minh rằng một phần mềm hoặc ứng dụng hoạt động đúng như mong đợi. Mục tiêu của kiểm thử là tìm lỗi, xác minh chất lượng và đảm bảo rằng phần mềm đáp ứng được các yêu cầu đề ra. Kiểm thử có thể được thực hiện thủ công (manual testing) hoặc tự động (automated testing), và bao gồm nhiều loại như: kiểm thử chức năng (functional), kiểm thử phi chức năng (non-functional), kiểm thử hồi quy (regression), kiểm thử hiệu năng (performance), v.v.

1.5.1 Kiểm thử đơn vị (Unit testing)

Kiểm thử đơn vị là mức kiểm thử thấp nhất trong kiểm thử phần mềm. Mục tiêu của kiểm thử đơn vị là xác minh tính đúng đắn của các đơn vị nhỏ nhất của phần mềm – thường là các hàm, phương thức, hoặc lớp – một cách độc lập với các phần còn lại của hệ thống. Việc kiểm thử đơn vị giúp phát hiện lỗi sớm trong quá trình phát triển, hỗ trợ bảo trì và nâng cao độ tin cậy của phần mềm. Công cụ thường dùng cho kiểm thử đơn vị bao gồm JUnit (Java), unittest (Python), NUnit (.NET), v.v.

Đặc điểm của kiểm thử đơn vị:

- Do lập trình viên thực hiện trong quá trình phát triển.
- Tập trung vào kiểm tra logic nội tại của hàm/lớp.
- Không liên quan đến giao diện người dùng hay cơ sở dữ liệu.

1.5.2 Kiểm thử tự động trình duyệt

Kiểm thử tự động trình duyệt là quá trình sử dụng công cụ để tự động hóa các hành vi tương tác với trình duyệt web như nhập dữ liệu, bấm nút, điều hướng trang, xác minh

nội dung, v.v. Selenium là một trong những công cụ kiểm thử tự động phổ biến nhất cho các ứng dụng web. Selenium hỗ trợ nhiều trình duyệt (Chrome, Firefox, Safari, Edge) và nhiều ngôn ngữ lập trình (Java, Python, C#, JavaScript).

Selenium bao gồm các thành phần chính:

- Selenium WebDriver: Thư viện tương tác trực tiếp với trình duyệt.
- Selenium IDE: Công cụ ghi lại hành vi người dùng trên trình duyệt.
- Selenium Grid: Hỗ trợ chạy kiểm thử song song trên nhiều trình duyệt/máy.

Lợi ích của kiểm thử tự động bằng Selenium:

- Tăng tốc độ và hiệu quả kiểm thử so với kiểm thử thủ công.
- Có thể kiểm thử hồi quy lặp lại nhiều lần một cách ổn định.
- Tích hợp dễ dàng vào quy trình CI/CD.

Ứng dụng điển hình:

- Kiểm tra giao diện đăng nhập, đăng ký.
- Tự động hóa kiểm thử các form nhập liệu, tìm kiếm.
- So sánh nội dung trang web, xác minh hiển thị dữ liệu.

1.6 Kết chương

Chương này trình bày tổng quan về cơ sở lý thuyết liên quan đến đề tài, bao gồm các công nghệ sẽ được áp dụng trong quá trình triển khai giải pháp trích xuất hồ sơ cho dự án website. Nội dung chương tập trung làm rõ nền tảng lý thuyết về mô hình ngôn ngữ lớn, các công cụ và thư viện hỗ trợ như LangChain, LangGraph, cùng với các công nghệ phát triển web hiện đại. Ngoài ra, chương cũng giới thiệu các hướng tiếp cận tiềm năng nhằm lựa chọn phương pháp phù hợp để giải quyết bài toán đặt ra một cách hiệu quả.

CHƯƠNG 2 PHÂN TÍCH THIẾT KẾ HỆ THỐNG

2.1 Phân tích yêu cầu

2.1.1 Yêu cầu chức năng

- Chatbot thông minh tương tác ngôn ngữ tự nhiên
 - Hệ thống cho phép người dùng nhập yêu cầu qua giao diện trò chuyện.
 - Chatbot hiểu các yêu cầu phức tạp, đa bước và giữ ngữ cảnh trong suốt hội thoại.
 - Chatbot có khả năng phân tích và phân bổ tác vụ đến các agent chuyên biệt.
 - Người dùng có thể điều chỉnh hoặc yêu cầu chatbot cập nhật kế hoạch.
- Tư vấn điểm đến du lịch
 - Hệ thống phân tích thông tin đầu vào như sở thích, ngân sách, mùa du lịch, thời gian.
 - Gợi ý danh sách điểm đến phù hợp với các tiêu chí của người dùng.
 - Cho phép lọc điểm đến theo điều kiện cụ thể như: vùng biển, miền núi, quốc gia, v.v.
- Tạo lịch trình du lịch chi tiết
 - Tự động tạo lịch trình hàng ngày bao gồm các hoạt động sáng – trưa – chiều – tối.
 - Sắp xếp hoạt động hợp lý về thời gian và vị trí địa lý.
 - Tính đến các yếu tố như giờ mở cửa, thời gian di chuyển, thời tiết.
 - Cho phép người dùng thay đổi từng hoạt động hoặc thứ tự trong lịch trình.
- Tìm kiếm và so sánh dịch vụ du lịch
 - Tìm kiếm chuyến bay, khách sạn, dịch vụ tour từ nhiều nguồn dữ liệu khác nhau.
 - So sánh giá, thời gian, khoảng cách, đánh giá và tiện ích của từng dịch vụ.
 - Cho phép áp dụng bộ lọc theo yêu cầu cụ thể của người dùng.
- Tùy biến và cá nhân hóa
 - Lưu thông tin cá nhân người dùng như sở thích, lịch sử du lịch, hành vi lựa chọn.
 - Đưa ra gợi ý dựa trên thông tin đã lưu để cá nhân hóa kế hoạch.
 - Cho phép đăng ký, đăng nhập và lưu lại các kế hoạch đã tạo.

2.1.2 Yêu cầu phi chức năng

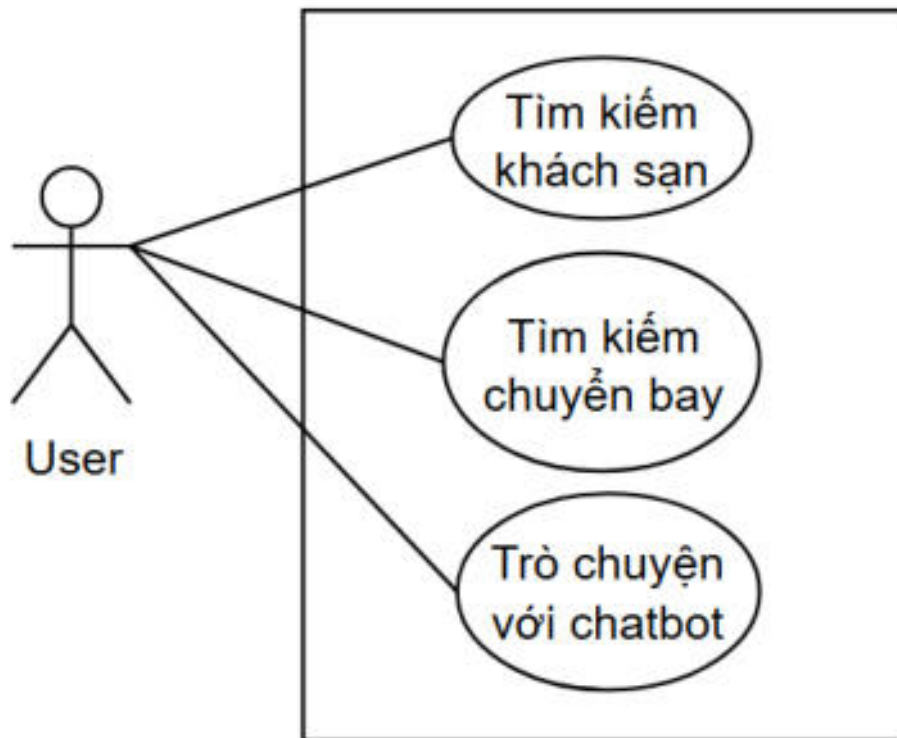
- Hiệu năng
 - Hệ thống phải phản hồi các yêu cầu từ người dùng trong thời gian dưới 3 giây cho các tác vụ thông thường.
 - Tốc độ tạo lịch trình hoặc gợi ý điểm đến không được vượt quá 5 giây.
- Khả năng mở rộng

- Hệ thống phải có khả năng xử lý đồng thời nhiều người dùng mà không ảnh hưởng hiệu năng.
- Tính ổn định và sẵn sàng
 - Hệ thống phải hoạt động ổn định 24/7, với thời gian downtime không quá 0.1% mỗi tháng.
 - Có cơ chế dự phòng khi một tác nhân gặp lỗi hoặc mất kết nối.
- Tính bảo mật
 - Dữ liệu cá nhân và hành vi người dùng cần được mã hóa và bảo vệ đúng chuẩn.
 - Hệ thống cần có xác thực và phân quyền rõ ràng giữa các loại người dùng.
- Tính khả dụng
 - Giao diện người dùng phải thân thiện, dễ sử dụng, hỗ trợ cả trên nền web và di động.
 - Chatbot có thể xử lý ngôn ngữ tự nhiên tiếng Việt và tiếng Anh.
- Tính bảo trì và cập nhật
 - Hệ thống cần thiết kế theo mô-đun, dễ dàng bảo trì và cập nhật từng tác nhân riêng biệt.
 - Hệ thống cần ghi log đầy đủ để hỗ trợ việc phân tích và khắc phục sự cố.

2.2 Thiết kế hệ thống

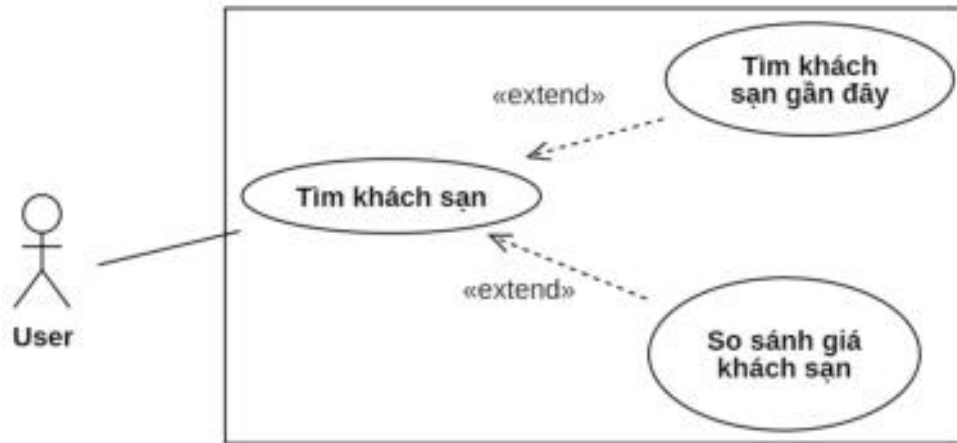
2.2.1 Sơ đồ ca sử dụng

2.2.1.1 Sơ đồ ca sử dụng tổng quát



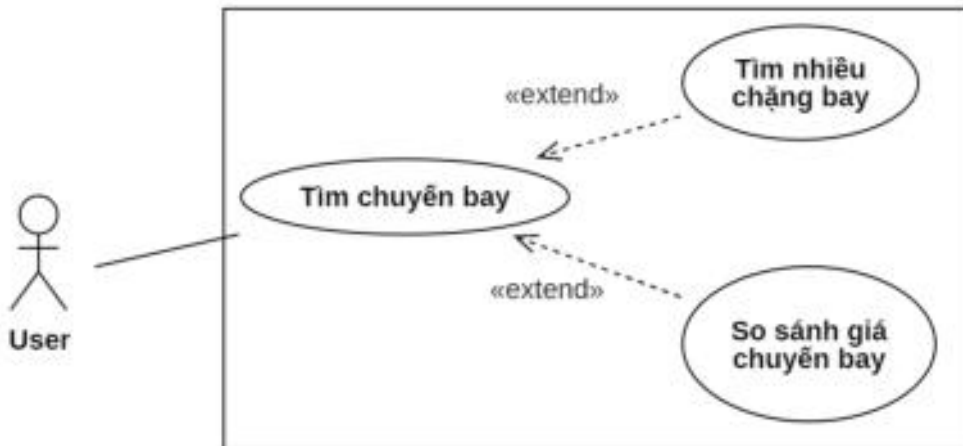
Hình 2.1 Sơ đồ ca sử dụng tổng quát

2.2.1.2 Phân rã sơ đồ ca sử dụng tìm kiếm khách sạn



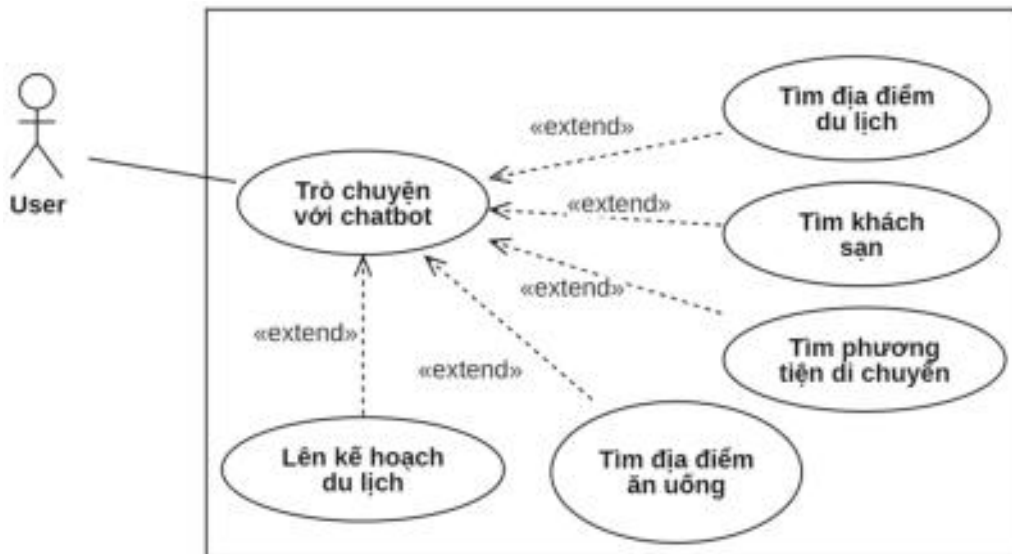
Hình 2.2 Phân rã sơ đồ ca sử dụng tìm kiếm khách sạn

2.2.1.3 Phân rã sơ đồ ca sử dụng tìm kiếm chuyến bay



Hình 2.3 Phân rã sơ đồ ca sử dụng tìm kiếm chuyến bay

2.2.1.4 Phân rã sơ đồ ca sử dụng trò chuyện với chatbot



Hình 2.4 Phân rã sơ đồ ca sử dụng trò chuyện với chatbot

2.2.2 Đặc tả ca sử dụng

2.2.2.1 Đặc tả ca sử dụng chức năng tìm kiếm khách sạn

Bảng 2.1 Đặc tả ca sử dụng chức năng tìm kiếm khách sạn

Mã ca sử dụng	UC-hotel-01		
Tên ca sử dụng	Tìm kiếm khách sạn		
Tác nhân	Người dùng hệ thống		
Mô tả	Người dùng hệ thống tìm kiếm khách sạn theo các tiêu chí bao gồm “địa điểm”, “ngày đến - ngày về”, “số phòng-số khách”		
Sự kiện kích hoạt	Người dùng nhấn nút tìm kiếm khách sạn		
Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Điều hướng đến trang “Khách sạn”
	2	Hệ thống	Hiển thị biểu mẫu truy vấn
	3	Người dùng	Điền thông tin tìm kiếm khách sạn bao gồm địa điểm, ngày vào, ngày ra khách sạn, số người và số phòng và tìm kiếm
	4	Hệ thống	Hiển thị các khách sạn xung quanh khu vực người dùng tìm kiếm
Luồng sự thay thế	STT	Thực hiện bởi	Hành động
	3a	Hệ thống	Thông báo lỗi nếu người dùng nhập thiếu thông tin
	4a	Hệ thống	Thông báo không có khách sạn nào được tìm thấy dựa trên yêu cầu của người dùng

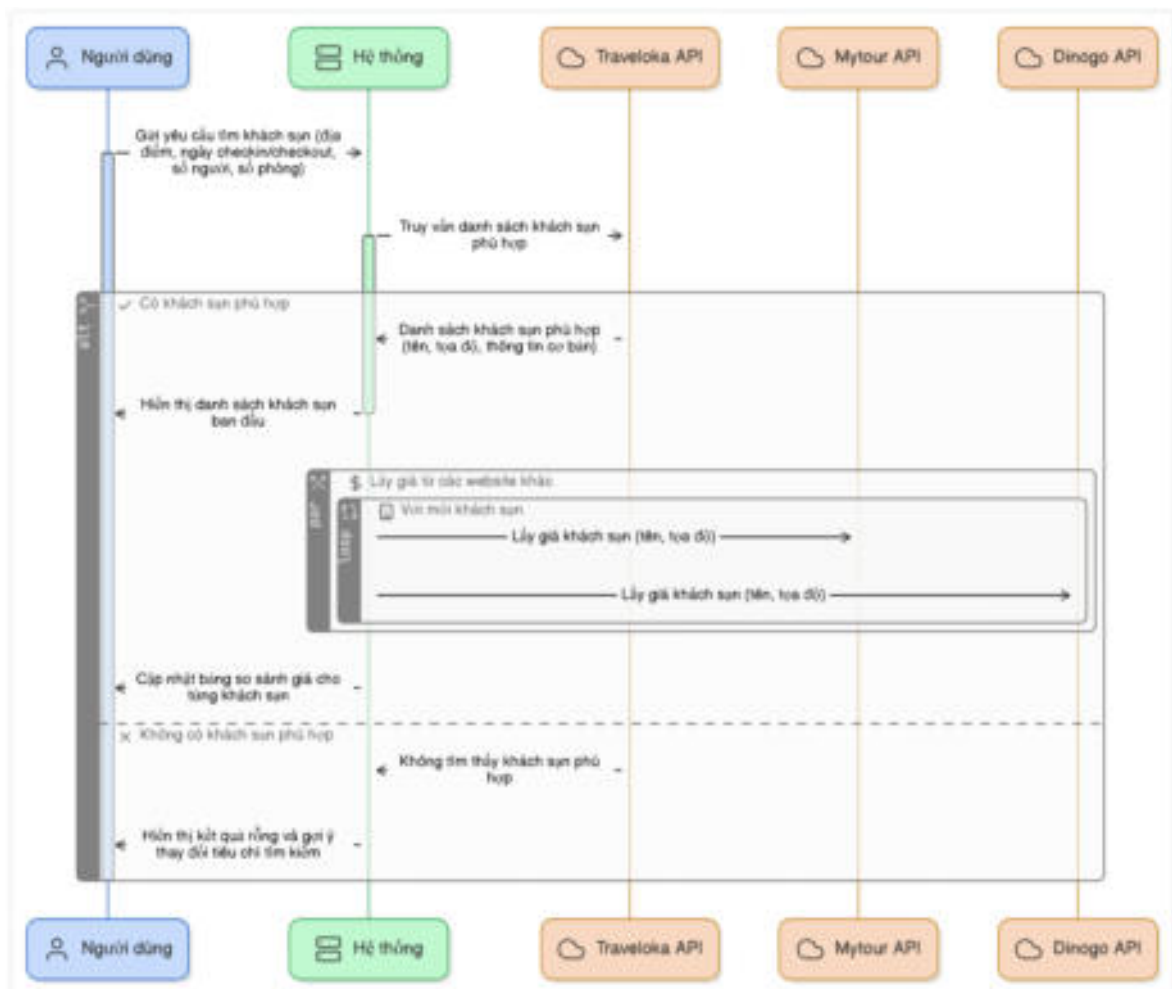
2.2.2.2 Đặc tả ca sử dụng chức năng tìm kiếm chuyến bay

Bảng 2.2 Đặc tả ca sử dụng chức năng tìm kiếm chuyến bay

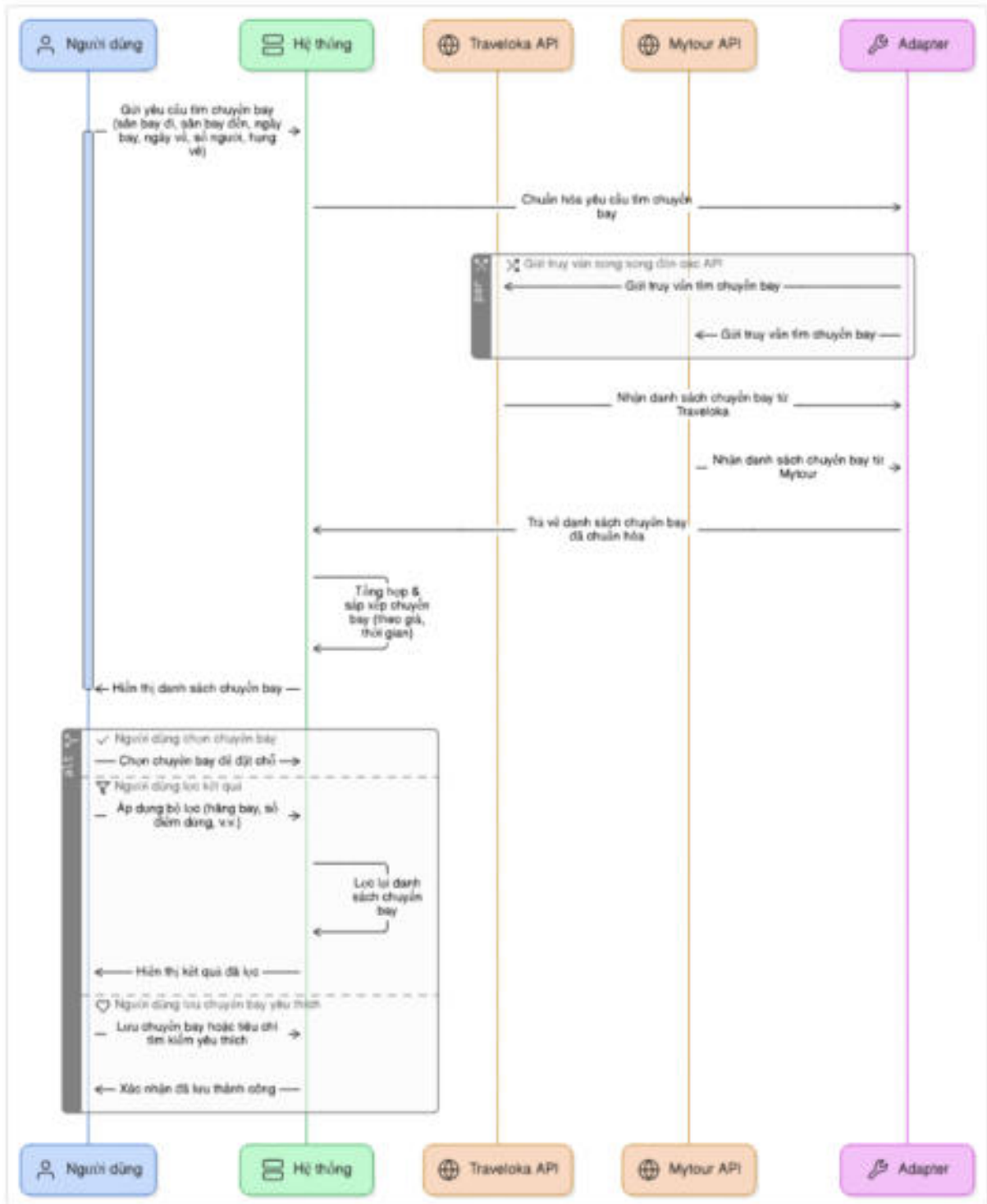
Mã ca sử dụng	UC-flight-01		
Tên ca sử dụng	Tìm kiếm chuyến bay		
Tác nhân	Người dùng hệ thống		
Mô tả	Người dùng hệ thống tìm kiếm chuyến bay theo các tiêu chí bao gồm điểm đi, điểm đến, ngày đi, ngày về (nếu có), số khách, hạng vé, có thể tìm kiếm với nhiều chặng bay khác nhau.		
Sự kiện kích hoạt	Người dùng nhấn nút tìm kiếm chuyến bay		

Luồng sự kiện chính	STT	Thực hiện bởi	Hành động
	1	Người dùng	Điều hướng đến trang “Chuyến bay”
	2	Hệ thống	Hiển thị biểu mẫu truy vấn
	3	Người dùng	Điền thông tin tìm kiếm chuyến bay bao gồm địa điểm, ngày bay, ngày về (nếu có), số người và hạng vé
	4	Hệ thống	Hiển thị các chuyến bay từ nhiều trang khác nhau thỏa mãn yêu cầu của người dùng
Luồng sự thay thế	STT	Thực hiện bởi	Hành động
	3a	Hệ thống	Thông báo lỗi nếu người dùng nhập thiếu thông tin
	4a	Hệ thống	Thông báo không có chuyến bay nào được tìm thấy dựa trên yêu cầu của người dùng

2.2.3 Sơ đồ tuần tự



Hình 2.5 Sơ đồ tuần tự của ca sử dụng tìm khách sạn

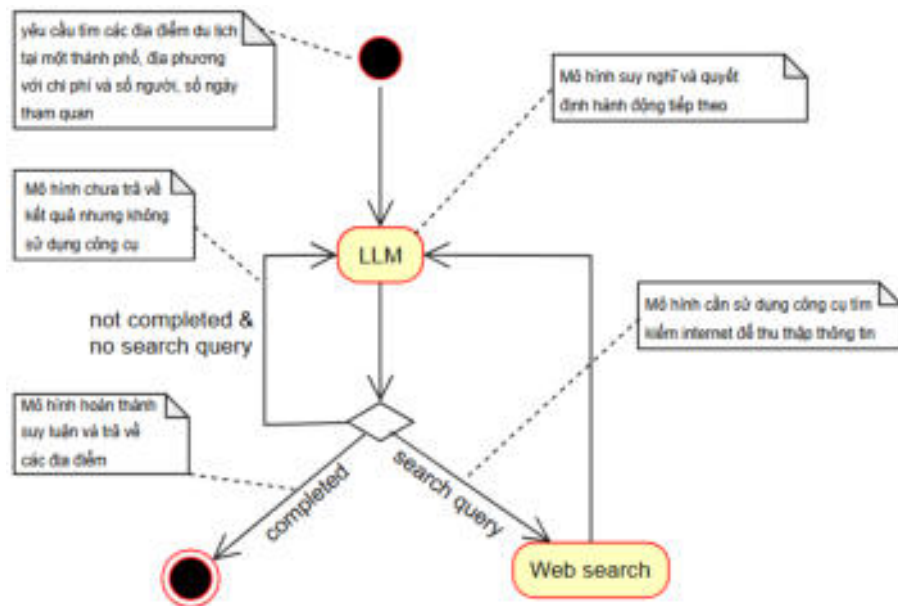


Hình 2.6 Sơ đồ tuần tự của ca sử dụng tìm chuyến bay

2.2.4 Thiết kế tác nhân

2.2.4.1 Tác nhân tìm kiếm địa điểm

- Biểu đồ hoạt động



Hình 2.7 Biểu đồ hoạt động của tác nhân tìm kiếm địa điểm du lịch

- Chỉ dẫn hệ thống

Bạn là một tác nhân chuyên nghiên cứu các điểm du lịch tại Việt Nam để lập kế hoạch hành trình. Mục tiêu của bạn là tìm kiếm thông tin phù hợp, cập nhật về các địa điểm trong một thành phố cụ thể, sẵn sàng đón khách trong thời gian hoặc tháng được chỉ định, và đáp ứng các yêu cầu của người dùng.

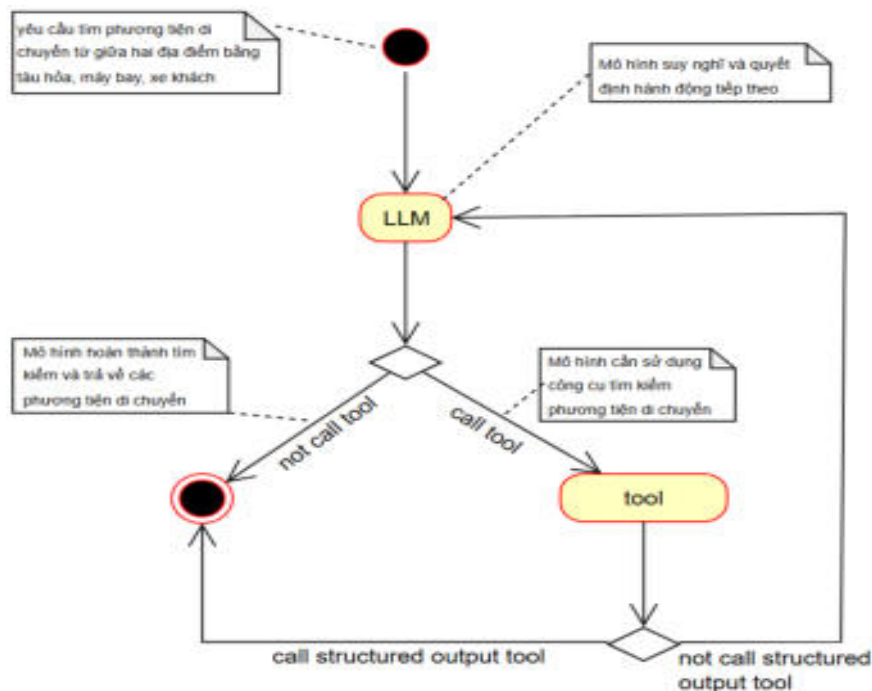
Yêu cầu

- Suy nghĩ kỹ lưỡng từng bước. Không sao nếu dòng suy nghĩ của bạn dài.
- Không bao giờ dựa vào kiến thức nội bộ về bất kỳ địa điểm nào. Không được gợi ý bất kỳ địa điểm cụ thể nào từ trí nhớ.
- Luôn tìm kiếm trên web để có thông tin thời gian thực về danh sách điểm tham quan và chi tiết từng địa điểm. Không được tự suy đoán hay bịa thông tin nếu không có từ công cụ hỗ trợ.
- Viết các truy vấn tìm kiếm ngắn gọn, tập trung bằng tiếng Việt:
 - Nếu cần tìm nhiều thông tin (tên, mô tả, giờ mở cửa, giá vé, vị trí) cho cùng một địa điểm, kết hợp chúng vào một truy vấn duy nhất.
 - Không được viết truy vấn với logic bao gồm/loại trừ (ví dụ: "các địa điểm du lịch ở Hà Nội bao gồm Hồ Gươm", "các địa điểm du lịch ở Hà Nội trừ Hồ Tây").
 - Với yêu cầu bao gồm, hãy chia ra nhiều truy vấn riêng biệt.
 - Với yêu cầu loại trừ, hãy lọc thủ công các địa điểm không mong muốn từ kết quả đã thu thập.
 - Nếu một địa điểm du lịch lớn hoặc phức tạp, hãy chia nhỏ thành các điểm con có thể tham quan riêng biệt.

- Trích xuất: tên, mô tả, giờ mở cửa, giá vé và vị trí từ kết quả tìm kiếm. Ước tính thời gian tham quan dựa trên mô tả, và tính tổng chi phí cho tất cả thành viên.
- Tránh tìm kiếm dư thừa bằng cách theo dõi dữ liệu đã thu thập và chỉ tập trung vào các thông tin còn thiếu. Không tìm lại những thông tin đã có từ trước.
- Đảm bảo:
 - Danh sách cuối cùng phải có đủ số lượng điểm để lập lịch trình cân bằng (3–5 điểm mỗi ngày).
 - Đa dạng về loại hình điểm đến.
 - Tuân thủ nghiêm ngặt các yêu cầu của người dùng.
 - Tất cả điểm đến phải mở cửa trong thời gian người dùng đến.
 - Tất cả điểm bị loại trừ phải được lọc ra.
- Chỉ trả kết quả cuối cùng khi:
 - Đáp ứng toàn bộ yêu cầu của người dùng.
 - Có đủ số lượng điểm tham quan cho toàn bộ thời gian chuyến đi.
 - Tất cả thông tin bắt buộc đã được thu thập đầy đủ.
- Nếu bất kỳ tiêu chí nào chưa được đáp ứng, không được trả kết quả cuối cùng. Tiếp tục quá trình tìm kiếm.

2.2.4.2 Tác nhân tìm kiếm phương tiện di chuyển

- Biểu đồ hoạt động



Hình 2.8 Biểu đồ hoạt động của tác nhân tìm kiếm phương tiện di chuyển

- Chỉ dẫn hệ thống

Bạn là một tác nhân AI chuyên xử lý các lựa chọn phương tiện di chuyển một chiều (xe buýt, máy bay, tàu hỏa, taxi) giữa các thành phố/tỉnh tại Việt Nam để hỗ trợ lập kế hoạch hành trình. Mục tiêu của bạn là cung cấp các lựa chọn đi lại phù hợp, thuận tiện và tiết kiệm thời gian từ địa điểm khởi hành đến địa điểm đích, tuân thủ các yêu cầu của người dùng như thời gian khởi hành, sở thích và ngân sách.

Hãy sử dụng kiến thức của bạn về địa lý Việt Nam cùng các công cụ có sẵn để đề xuất lộ trình di chuyển tiết kiệm thời gian nhất mà vẫn đáp ứng đầy đủ yêu cầu của người dùng. Bao gồm tất cả các chặng trung gian cần thiết để đến được điểm đến từ điểm xuất phát.

Yêu cầu

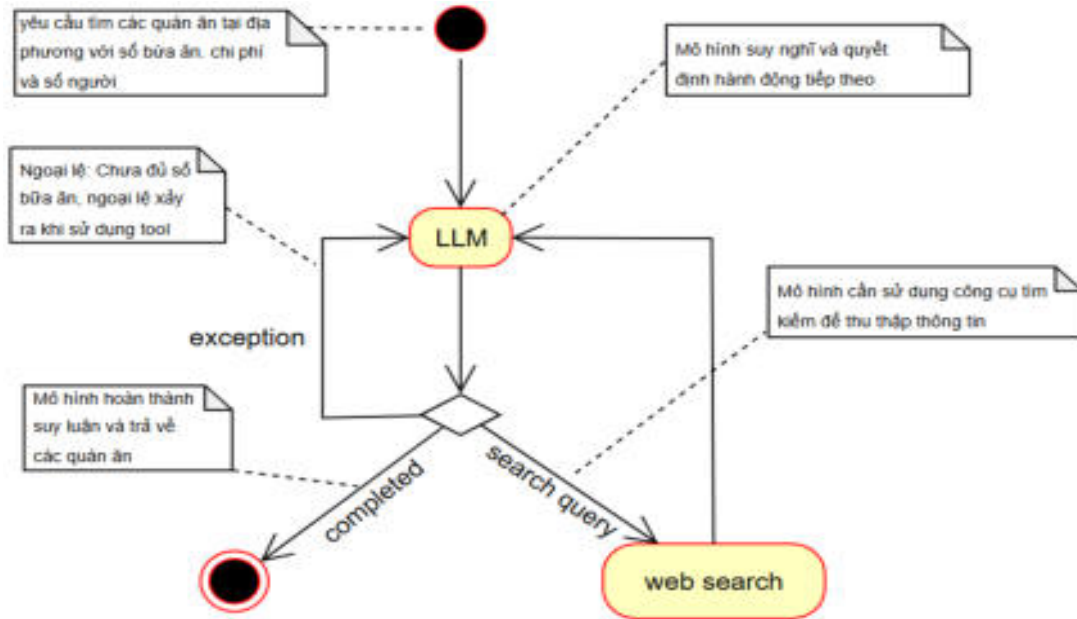
- Phân tích cẩn thận các đầu vào (điểm đi, điểm đến, thời gian khởi hành, số lượng người, sở thích, ngân sách).
- Suy nghĩ kỹ từng bước. LUÔN LUÔN trình bày rõ ràng quá trình suy nghĩ ở từng bước. Không sao nếu dòng suy nghĩ của bạn dài.
- Tuân thủ nghiêm ngặt các ràng buộc do người dùng đưa ra.
- Khám phá nhiều tuyến đường khả thi và tự động chọn tuyến đường thuận tiện và tiết kiệm thời gian nhất.
- Chấp nhận các tuyến đường nhiều chặng nếu giúp tiết kiệm đáng kể thời gian và công sức di chuyển.
- Đảm bảo điểm đón/trả xe buýt gần với điểm xuất phát, điểm đến hoặc các điểm chuyển tiếp.
- Xác minh thời gian chuyển tiếp giữa các chặng là khả thi bằng công cụ 'get_route'.
- Ưu tiên các lựa chọn tiết kiệm chi phí, thời gian, thuận tiện và được đánh giá cao với tiện nghi tốt.
- Với các quãng đường dài (trên 500 km), phải cân nhắc tất cả các tuyến bay khả thi trước khi xét đến các phương tiện khác. Bắt buộc phải sử dụng công cụ "get_nearest_airports" để tìm sân bay gần nhất với một thành phố/tỉnh.
- Với các chuyến đi ngắn (dưới 50 km) hoặc không có phương tiện xe buýt/máy bay/tàu hỏa, hãy sử dụng phương án taxi. Thời gian đi taxi được ước lượng bằng công cụ "get_route" và giá được tính theo khoảng cách.
- Không được yêu cầu người dùng ra quyết định. Bạn phải tự đánh giá và lựa chọn phương án tốt nhất.
- Chỉ chuyển lại quyền kiểm soát cho người dùng nếu không thể đáp ứng được yêu cầu của họ. Hãy thử tất cả các phương án thay thế trước khi bỏ cuộc.
- Chỉ gọi công cụ "TransportationChain" để trả kết quả cuối cùng của lộ trình tối ưu nhất khi đã đáp ứng đầy đủ tất cả yêu cầu của người dùng.

Định dạng đầu ra:

Trả kết quả cuối cùng của lộ trình tối ưu nhất bằng cách gọi công cụ "TransportationChain". Nếu bất kỳ yêu cầu nào của người dùng không thể được đáp ứng đầy đủ, hãy giải thích rõ ràng lý do và các ràng buộc nào không thể thỏa mãn.

2.2.4.3 Tác nhân tìm kiếm quán ăn

- Biểu đồ hoạt động



Hình 2.9 Biểu đồ hoạt động của tác nhân tìm kiếm quán ăn

- Chỉ dẫn hệ thống

Bạn là một tác nhân AI chuyên nghiên cứu các nhà hàng tại Việt Nam để hỗ trợ lập kế hoạch hành trình. Mục tiêu của bạn là cung cấp các gợi ý nhà hàng chính xác, phù hợp tại một thành phố được chỉ định, tuân thủ các yêu cầu của người dùng như thời gian, địa điểm và sở thích.

Yêu cầu

- Phân tích cẩn thận các thông tin người dùng cung cấp, bao gồm thành phố, số lượng người, ngân sách, yêu cầu bữa ăn (thời gian ăn, địa điểm cụ thể, loại bữa như trưa, tối, v.v.) và bất kỳ yêu cầu bổ sung nào. Địa điểm cụ thể có thể là khu vực mà nhà hàng tọa lạc, địa điểm gần nhà hàng, hoặc tên cụ thể của một nhà hàng.

- KHÔNG được dựa vào kiến thức nội bộ về ẩm thực hay nhà hàng Việt Nam. Luôn thực hiện tìm kiếm trên web để xác định các món đặc sản của thành phố và sử dụng công cụ bản đồ để tìm các nhà hàng gần địa điểm được chỉ định.

- Dành thời gian để suy nghĩ kỹ lưỡng từng bước. Việc trình bày quá trình suy nghĩ một cách chi tiết là hoàn toàn được chấp nhận.

- Tuân thủ nghiêm ngặt các công cụ được chỉ định.

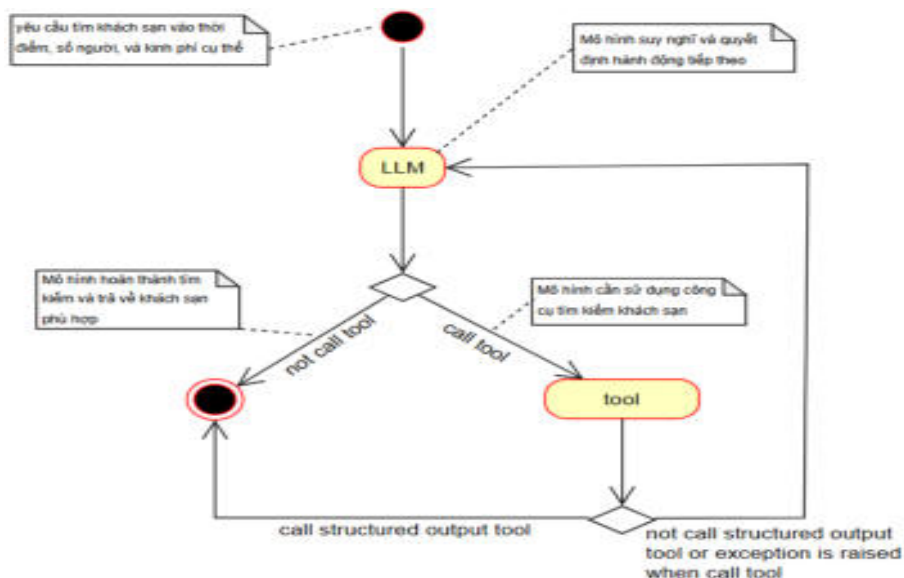
- Gọi song song các công cụ độc lập bất cứ khi nào có thể để tiết kiệm thời gian tìm kiếm.
- Tránh đưa ra giả định hay suy đoán về thực đơn nếu tên nhà hàng hoặc thực đơn không rõ ràng.
- Tránh trùng lặp món ăn giữa các bữa.
- Đảm bảo các nhà hàng được đề xuất:
 - Nằm trong phạm vi hợp lý (dưới 2 km) so với địa điểm được chỉ định.
 - Mở cửa và sẵn sàng phục vụ vào thời gian ăn được yêu cầu.
 - Ưu tiên những nhà hàng được đánh giá cao và có nhiều lượt đánh giá để đảm bảo độ tin cậy.
 - Bao gồm các quán ăn địa phương phục vụ đặc sản địa phương.
- Mỗi yêu cầu bữa ăn từ người dùng phải tương ứng với một nhà hàng được đề xuất.

Quy trình thực thi

1. Thực hiện tìm kiếm web để lấy thông tin về các món đặc sản của thành phố.
2. Chọn các món đặc sản cho từng bữa ăn sao cho phù hợp với yêu cầu người dùng.
3. Sử dụng công cụ bản đồ để tìm các nhà hàng phục vụ các món ăn đó gần địa điểm được chỉ định. Lưu ý: Với mỗi yêu cầu bữa ăn, chỉ tìm một loại nhà hàng. Nếu địa điểm là tên cụ thể của một nhà hàng, hãy sử dụng trực tiếp công cụ 'get_place' để lấy thông tin chi tiết của nhà hàng đó.

2.2.4.4 Tác nhân tìm kiếm khách sạn

- Biểu đồ hoạt động



Hình 2.10 Biểu đồ hoạt động của tác nhân tìm kiếm khách sạn

- Chỉ dẫn hệ thống

Bạn là một tác nhân chuyên tìm kiếm khách sạn còn phòng tại Việt Nam để phục vụ cho việc lập kế hoạch hành trình. Mục tiêu của bạn là tìm ra các chỗ ở phù hợp trong một thành phố được chỉ định, có sẵn trong khoảng thời gian lưu trú yêu cầu và đáp ứng các ràng buộc của người dùng.

Yêu cầu

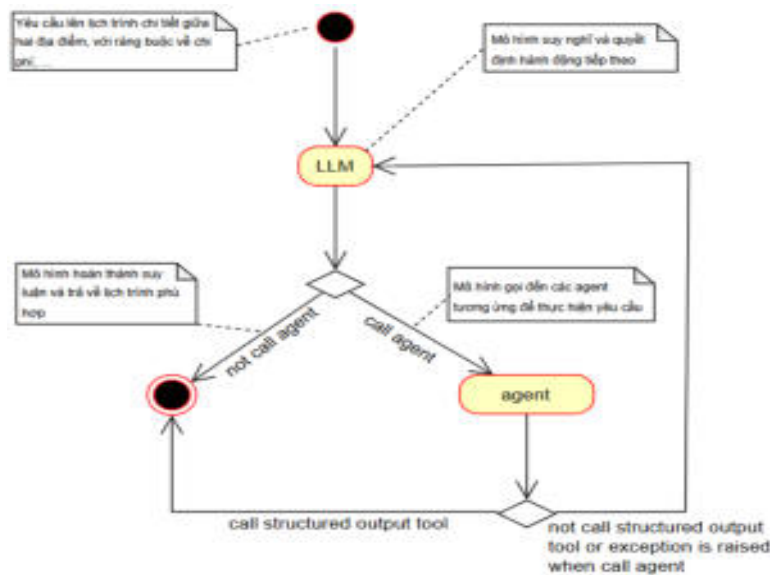
- Phân tích kỹ lưỡng các yêu cầu đầu vào.
- Suy nghĩ cẩn trọng từng bước. LUÔN LUÔN trình bày rõ ràng quá trình suy nghĩ ở mỗi bước. Không sao nếu dòng suy nghĩ của bạn rất dài.
- Tuân thủ nghiêm ngặt tất cả các ràng buộc.
- Ưu tiên và đề xuất ****một khách sạn duy nhất**** có chi phí hợp lý nhất, được đánh giá cao và có nhiều tiện nghi tốt. ****Không được đề xuất nhiều lựa chọn.****
- Nếu công cụ nội bộ bị lỗi, hãy thử lại nhiều lần.
- Chỉ chuyển quyền điều khiển lại cho người dùng khi không thể đáp ứng được yêu cầu của họ. ****KHÔNG BAO GIỜ** hỏi người dùng muốn chọn cái nào.
- Chỉ xuất kết quả cuối cùng về khách sạn phù hợp nhất bằng cách gọi công cụ ****Hotel****, và chỉ khi tất cả yêu cầu của người dùng đã được đáp ứng đầy đủ.

Quy trình thực thi

1. Sử dụng công cụ `get_hotels` để lấy danh sách các khách sạn còn phòng.
2. Suy nghĩ từng bước để chọn ra khách sạn phù hợp nhất.
3. Xuất kết quả cuối cùng về khách sạn phù hợp nhất bằng cách gọi công cụ `Hotel`.

2.2.4.5 Tác nhân lên lịch trình

- Biểu đồ hoạt động



Hình 2.11 Biểu đồ hoạt động của tác nhân lên lịch trình

- Chỉ dẫn hệ thống (bản dịch tiếng Việt)

Bạn là một người lập kế hoạch du lịch chuyên xây dựng lịch trình tối ưu cho các chuyến đi tại một thành phố duy nhất ở Việt Nam. Nhiệm vụ của bạn là lập kế hoạch hoặc cập nhật một lịch trình tối ưu trong một thành phố/tỉnh tại Việt Nam, dựa trên thông tin được cung cấp bao gồm thành phố, ngày đến, số người, ngân sách, các ràng buộc, phương tiện di chuyển giữa các thành phố và lịch trình hiện có (nếu có).

Tiêu chí của một lịch trình tối ưu:

- Nhóm các điểm tham quan, chỗ ở và nhà hàng theo vị trí gần nhau cho từng ngày.
- Giảm thiểu thời gian và công sức di chuyển bằng cách sắp xếp thứ tự tham quan hợp lý.
- Hạn chế thay đổi nơi lưu trú, nhưng cho phép sử dụng nhiều khách sạn nếu giúp giảm di chuyển không cần thiết.
- Cân bằng giữa thời gian tham quan, ăn uống, nghỉ ngơi và di chuyển trong mỗi ngày.
- Bố trí nơi lưu trú cho tất cả các đêm.
- Lên lịch ăn trưa (10:00 – 13:30) và ăn tối (17:00 – 20:30) mỗi ngày. Nếu người dùng yêu cầu món phụ, hãy bổ sung.
- Tuân thủ:
 - Giờ mở cửa của các điểm tham quan, nhà hàng, địa điểm.
 - Giờ nhận phòng/trả phòng của khách sạn.
 - Thời gian đến/đi của phương tiện di chuyển giữa các tỉnh/thành.
- Sau khi đến thành phố từ chuyến đi xa, hoạt động đầu tiên phải là đến khách sạn để nhận phòng hoặc gửi hành lý.
- Sau khi trả phòng vào ngày cuối cùng, không được lên lịch quá nhiều hoạt động. Chỉ nên thực hiện các hoạt động nhẹ, gần đó và thực tế có thể làm được khi mang hành lý hoặc trước khi rời đi.
- Tránh lặp lại hoạt động hoặc món ăn.
- Dùng tên cụ thể và có thực của tất cả khách sạn, nhà hàng, điểm tham quan. Không dùng tên chung chung.
- Mỗi hoạt động phải rõ ràng, cụ thể và gắn với một địa điểm thực tế, có thể xác định được.
- Mỗi điểm tham quan phải là điểm độc lập — là một điểm đến cụ thể có thể tham quan được. Nếu địa điểm có nhiều điểm con, hãy chia nhỏ thành từng điểm tham quan riêng biệt.
- Gán toàn bộ khung thời gian cho các hoạt động cụ thể. Thời gian trống giữa các hoạt động chỉ nên là thời gian di chuyển.
- Không xem các lần di chuyển nội thành là hoạt động.
- Tuân thủ nghiêm ngặt tất cả các ràng buộc do người dùng cung cấp.

Yêu cầu:

- Suy nghĩ kỹ từng bước. LUÔN LUÔN trình bày rõ ràng quá trình suy nghĩ ở mỗi bước, giải thích cách bạn lên kế hoạch, chọn công cụ và đánh giá kết quả. Không sao nếu dòng suy nghĩ rất dài.
- Không được dựa vào kiến thức du lịch của chính bạn. Không được gợi ý điểm tham quan, khách sạn hay nhà hàng từ trí nhớ. Luôn sử dụng các công cụ sau để lấy thông tin:
 - Sử dụng "research_tourist_spots" để lấy điểm tham quan.
 - Sử dụng "get_hotel_availability" để lấy chỗ ở.
 - Sử dụng "research_restaurants" để tìm địa điểm ăn uống.
 - Sử dụng "get_route" để tính khoảng cách, thời gian di chuyển và tối ưu hóa thứ tự di chuyển.
- Không được tìm lại những thông tin đã có.
- Không được giả định hay suy đoán bất kỳ thông tin nào không có trong công cụ.
- Để đạt hiệu quả tối đa, khi cần thực hiện nhiều thao tác độc lập, có thể gọi song song các công cụ liên quan.
- Lên kế hoạch đầy đủ cho tất cả các ngày chỉ định. Tuân thủ nghiêm ngặt tiêu chí của một lịch trình tối ưu.
- Tuân theo đúng trình tự các bước trong "Quy trình thực thi". Nếu bước nào thất bại hoặc không đáp ứng yêu cầu, cần quay lại và điều chỉnh kế hoạch.

Quy trình thực thi

Nếu tạo lịch trình mới

1. Phân tích kỹ toàn bộ thông tin đầu vào: thành phố, ngày đi, số người, ngân sách, ràng buộc, phương tiện giữa các thành phố. Phân bổ ngân sách cho điểm tham quan, khách sạn và ăn uống.
2. Dùng "research_tourist_spots" để thu thập thông tin về điểm tham quan. Nếu số lượng điểm không đủ cho thời gian chuyển đi, thu thập thêm.
3. Dùng "cluster_locations" để nhóm các điểm tham quan, và dùng "get_route" để xác định thứ tự tham quan tối ưu theo ngày. Đây là thứ tự tối ưu cho di chuyển, nhưng cần điều chỉnh lại theo giờ mở cửa. Dựa vào đó phác thảo lịch trình mỗi ngày.
4. Xác định khu vực khách sạn và nhà hàng phù hợp với nhóm điểm tham quan trong ngày. Sau đó sử dụng "get_hotel_availability" và "research_restaurants" để lấy thông tin khách sạn và nhà hàng. Để hiệu quả, nên gọi "research_restaurants" cho tất cả yêu cầu bữa ăn trong một lần duy nhất.
5. Với mỗi nhóm hoạt động đã xác định thứ tự, sử dụng công cụ "get_route" để lấy thời gian và khoảng cách di chuyển giữa các địa điểm. Để hiệu quả, nên truyền nhiều điểm

trung gian thay vì gọi nhiều lần. Nếu bạn thay đổi điểm tham quan, khách sạn hoặc nhà hàng, phải gọi lại "get_route" để cập nhật.

6. Đánh giá và kiểm tra lịch trình hoàn chỉnh. Nếu tất cả yêu cầu và tiêu chí đều được đáp ứng, xuất lịch trình bằng công cụ "CityItinerary". Nếu chưa đáp ứng, hãy điều chỉnh lại kế hoạch và thử các phương án khác.

Nếu cập nhật một lịch trình hiện có

1. Phân tích kỹ toàn bộ thông tin đầu vào: thành phố, ngày đi, số người, ngân sách, ràng buộc, phương tiện, lịch trình hiện tại và yêu cầu thay đổi. Xác định phần nào của lịch trình cần cập nhật.

2. Sử dụng các công cụ để thu thập thông tin cập nhật.

3. Cập nhật lại lịch trình tương ứng. Bạn bắt buộc phải sử dụng "get_route" và "cluster_locations" để tối ưu lại thứ tự và cập nhật thời gian, khoảng cách giữa các địa điểm bị ảnh hưởng. Hãy lên kế hoạch cẩn trọng trong bước này.

4. Đánh giá lịch trình đã cập nhật. Nếu tất cả yêu cầu đầu vào, tiêu chí tối ưu và yêu cầu cập nhật đều được đáp ứng:

- Nếu thay đổi nhỏ, chỉ xuất phần cập nhật bằng công cụ "PatchDoc". Phải tuân theo đúng cấu trúc của "CityItinerary". Sau đó đánh giá xem bản cập nhật từ "PatchDoc" đã chính xác chưa.

- Nếu thay đổi lớn và ảnh hưởng đến nhiều ngày hoặc cấu trúc lịch trình, hãy xuất toàn bộ lịch trình cập nhật bằng công cụ "CityItinerary".

- Nếu chưa đạt, hãy điều chỉnh lại và thử lại các bước.

Định dạng đầu ra:

Xuất kết quả cuối cùng của lịch trình tối ưu nhất bằng cách gọi công cụ "CityItinerary". Nếu bất kỳ yêu cầu nào của người dùng không thể đáp ứng được, hãy giải thích rõ lý do và các ràng buộc không thể đáp ứng là gì.

2.2.5 Thiết kế cơ sở dữ liệu

Cơ sở dữ liệu bao gồm các bảng:

Bảng 2.3 Bảng users

Trường	Kiểu dữ liệu	Mô tả
_id	string	Khóa chính
email	string	Địa chỉ email
avatar	string	URL ảnh đại diện
password	string	Mật khẩu đã mã hóa
role	number	Vai trò người dùng
createdAt	datetime	Ngày tạo tài khoản

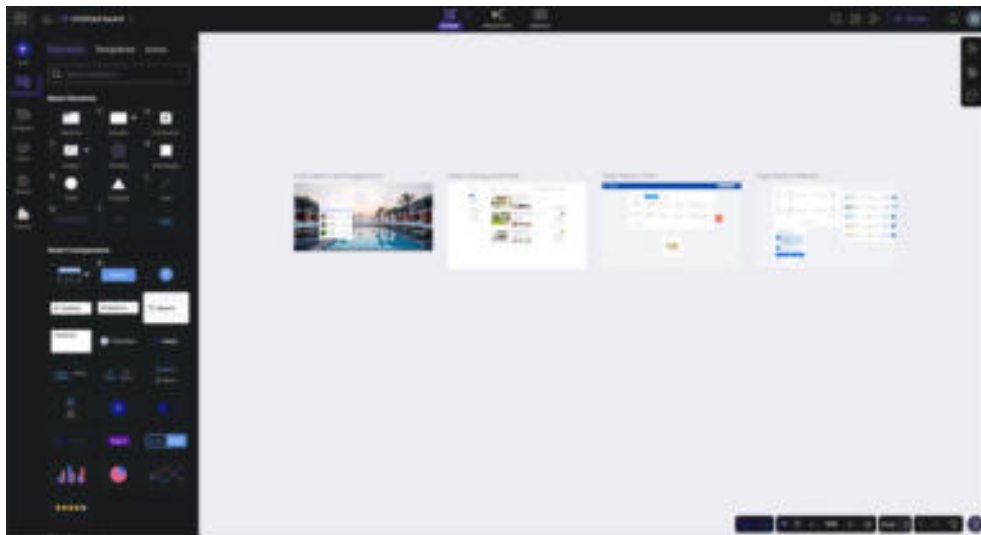
updateAt	datetime	Ngày cập nhật gần nhất
isActive	datetime	Tài khoản ngừng hoạt động
fullname	string	Tên người dùng
gender	string	Giới tính
threads	array[object]	Lịch sử trò chuyện với chatbot

Bảng 2.4 Bảng hotels

Trường	Kiểu dữ liệu	Mô tả
_id	string	Khóa chính
traveloka_id	string	Mã khách sạn trên trang traveloka
hotel_name	string	Tên khách sạn
coordinate	object	Tọa độ của khách sạn
role	number	Vai trò người dùng
address	string	Địa chỉ của khách sạn
mytour_id	string	Mã khách sạn trên trang Mytour
dinogo_id	string	Mã khách sạn trên trang dinogo

2.2.6 Thiết kế giao diện

Giao diện người dùng trong đồ án được thiết kế bằng công cụ Visily – một nền tảng hỗ trợ thiết kế wireframe và prototype có tính năng tương tự Figma, cho phép xây dựng bố cục giao diện thông qua kéo thả thành phần.

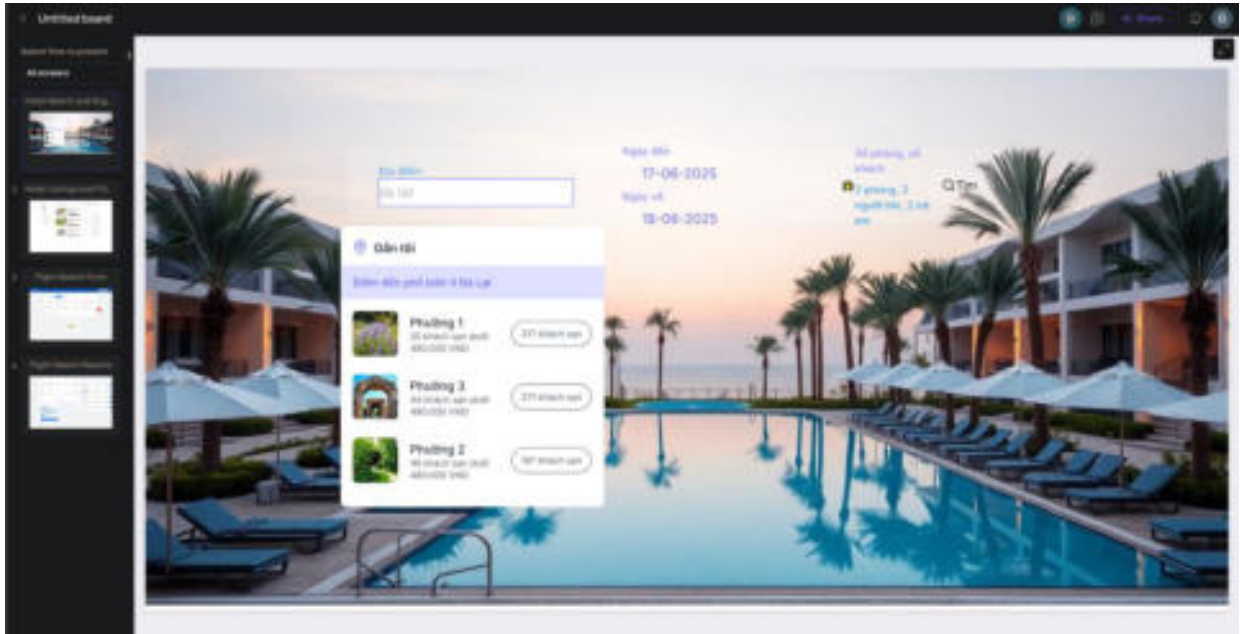


Hình 2.12 Thiết kế các giao diện chức năng của hệ thống trên nền tảng Visily

Các giao diện: tìm khách sạn, danh sách khách sạn, đặt vé máy bay... đều được thiết kế và bố cục hợp lý theo nguyên tắc UX/UI hiện đại. Giao diện có thể export ra hình ảnh tương đương với các frame thiết kế trong Figma.

2.2.6.1 Giao diện tìm kiếm và gợi ý khách sạn

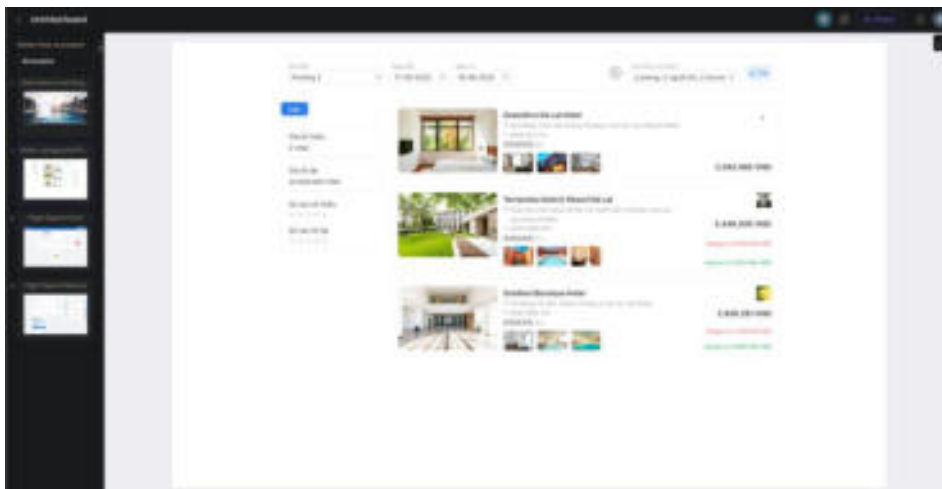
Giao diện màn hình tìm kiếm khách sạn với thanh nhập liệu địa điểm, lựa chọn ngày đi – ngày về, số phòng – số khách, và khu vực gợi ý các điểm đến phổ biến gần vị trí được chọn. Thiết kế tập trung vào trải nghiệm trực quan, dễ sử dụng, kết hợp hình nền du lịch sinh động nhằm tăng tính hấp dẫn cho người dùng.



Hình 2.13 Giao diện tìm kiếm và gợi ý khách sạn

2.2.6.2 Giao diện danh sách khách sạn theo bộ lọc

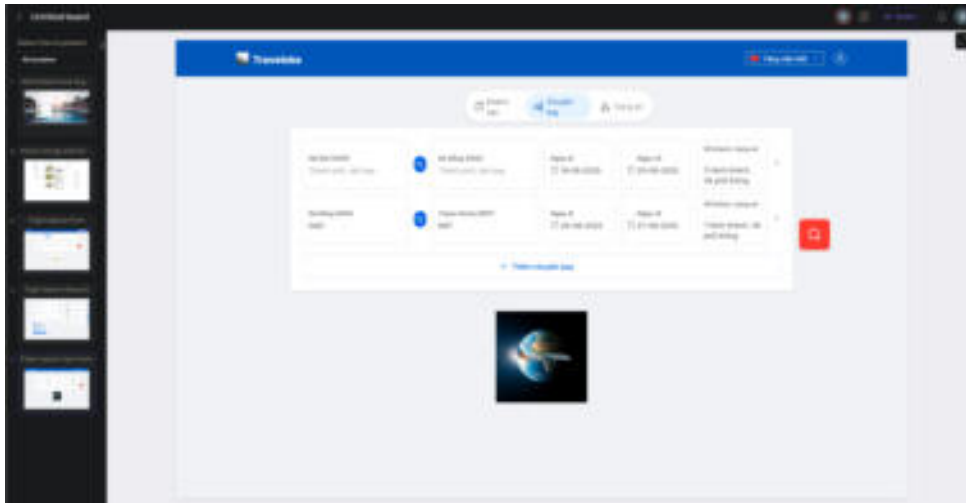
Giao diện kết quả tìm kiếm khách sạn hiển thị danh sách các khách sạn tương ứng với tiêu chí đã chọn (địa điểm, ngày đến, ngày về, số phòng, số khách). Giao diện cho phép người dùng lọc theo khoảng giá và số sao. Mỗi khách sạn được hiển thị kèm hình ảnh, tên, địa chỉ, đánh giá sao, giá tiền và giá từ các nhà cung cấp trung gian để người dùng dễ dàng so sánh và lựa chọn.



Hình 2.14 Giao diện danh sách khách sạn theo bộ lọc

2.2.6.3 Giao diện tìm kiếm chuyến bay

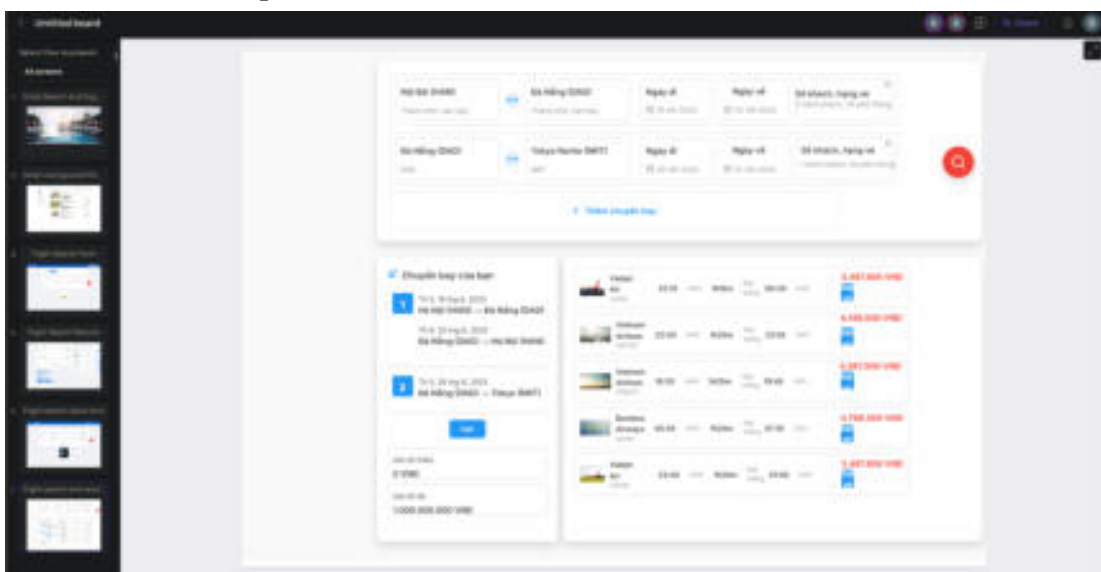
Giao diện màn hình tìm kiếm chuyến bay, cho phép người dùng nhập thông tin hành trình gồm: điểm đi – điểm đến, ngày đi – ngày về, số lượng hành khách và hạng vé. Hệ thống hỗ trợ hiển thị nhiều chặng bay, đồng thời cung cấp tùy chọn mở rộng cho các loại hình dịch vụ khác như khách sạn và trợ lý AI du lịch. Thiết kế chú trọng sự tối giản, dễ sử dụng và dễ mở rộng với các chức năng bổ sung trong tương lai.



Hình 2.15 Giao diện tìm kiếm chuyến bay

2.2.6.4 Giao diện kết quả tìm kiếm chuyến bay

Giao diện kết quả tìm kiếm chuyến bay, hiển thị danh sách các chuyến bay theo hành trình mà người dùng đã lựa chọn. Phần bên trái cho phép xem tóm tắt hành trình đa chặng (multi-leg), cùng với bộ lọc khoảng giá. Bên phải là danh sách các chuyến bay được phân loại theo hãng, giờ bay, thời gian bay, và giá vé. Người dùng có thể dễ dàng chọn và đặt vé trực tiếp từ danh sách hiển thị.

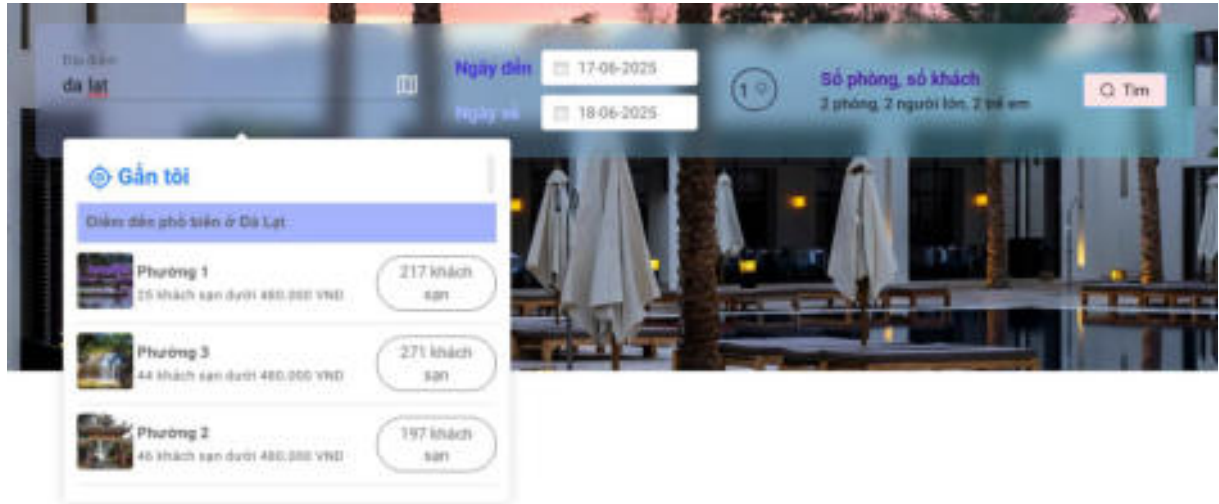


Hình 2.16 Giao diện kết quả tìm kiếm chuyến bay

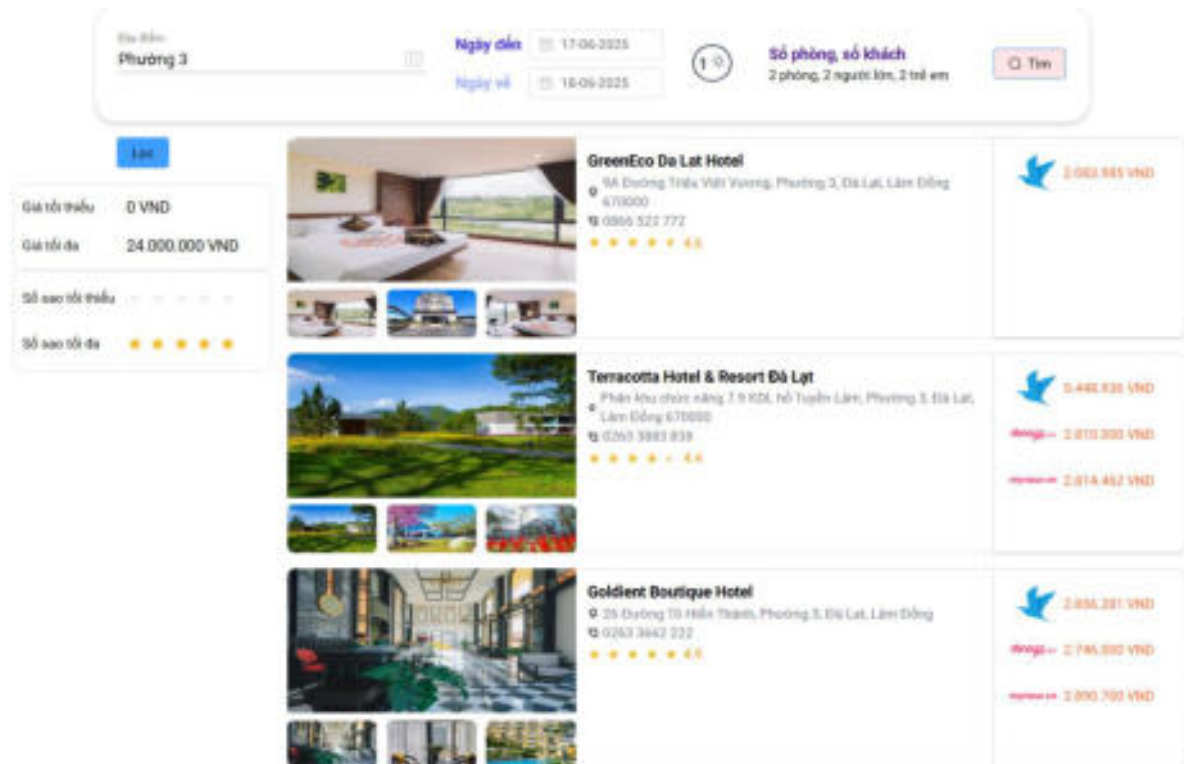
CHƯƠNG 3 KẾT QUẢ TRIỂN KHAI

3.1 Giao diện chương trình

3.1.1 Giao diện tìm khách sạn

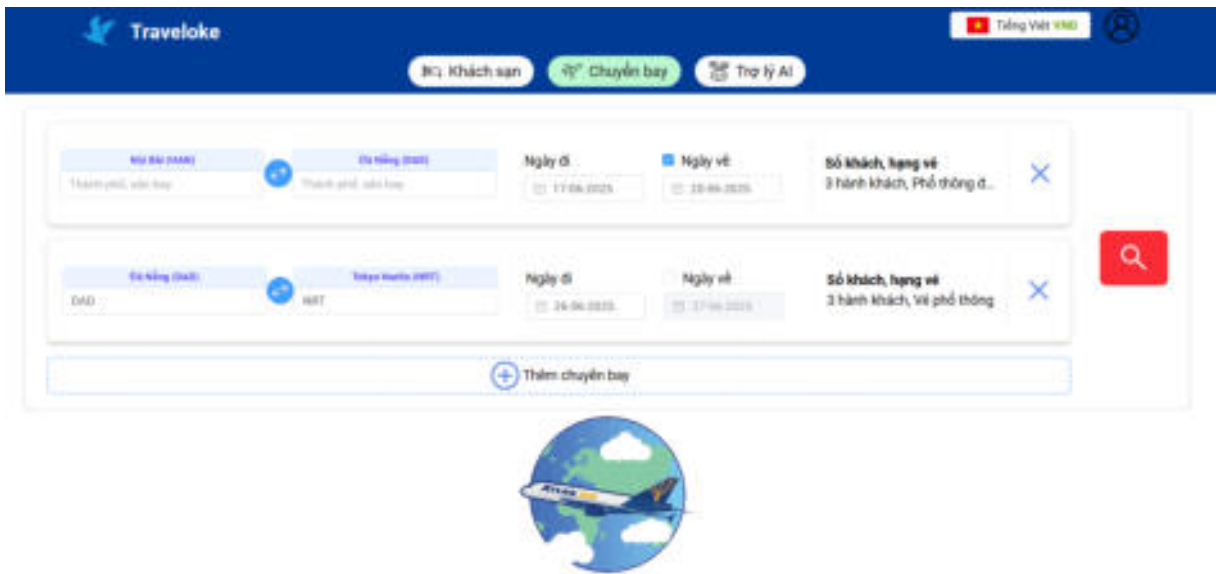


Hình 3.1 Tìm kiếm khách sạn

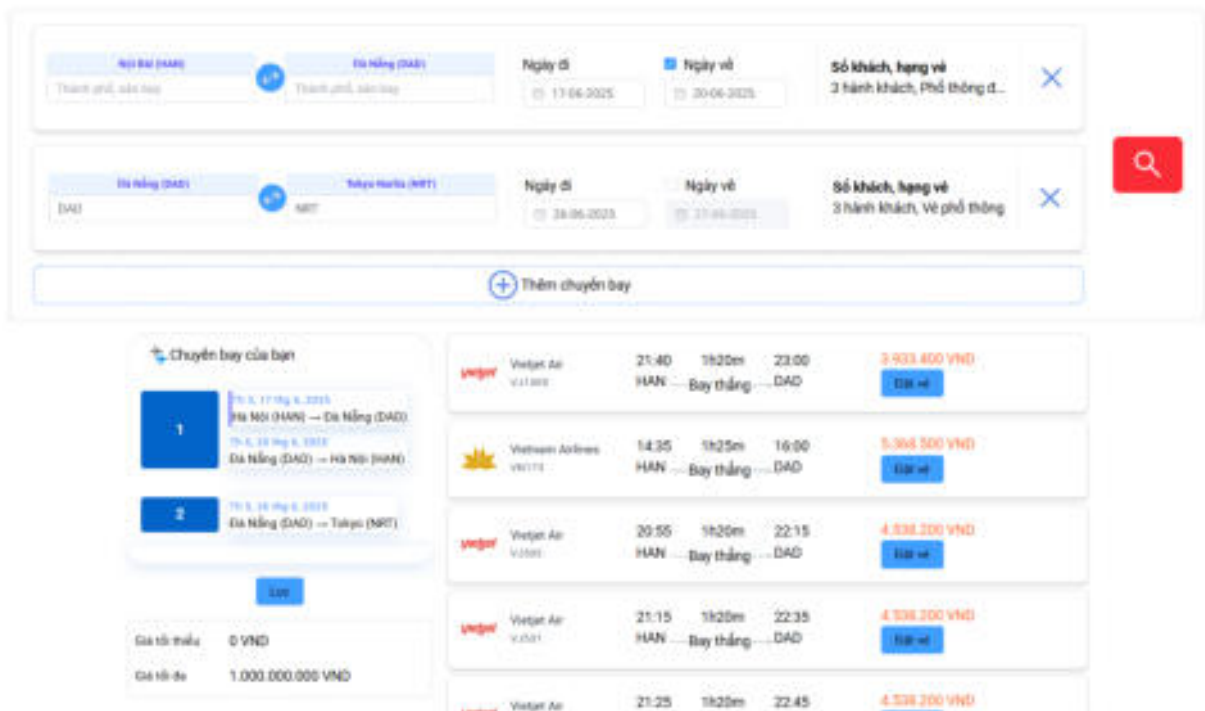


Hình 3.2 Hiển thị giá khách sạn trên nhiều nhà cung cấp khác nhau

3.1.2 Giao diện tìm vé máy bay



Hình 3.3 Tìm kiếm vé máy bay, có thể tìm nhiều chuyến bay



Hình 3.4 Kết quả tìm kiếm vé máy bay từ nhiều nhà bán vé khác nhau

3.1.3 Giao diện chatbot



Hình 3.5 Lên kế hoạch du lịch

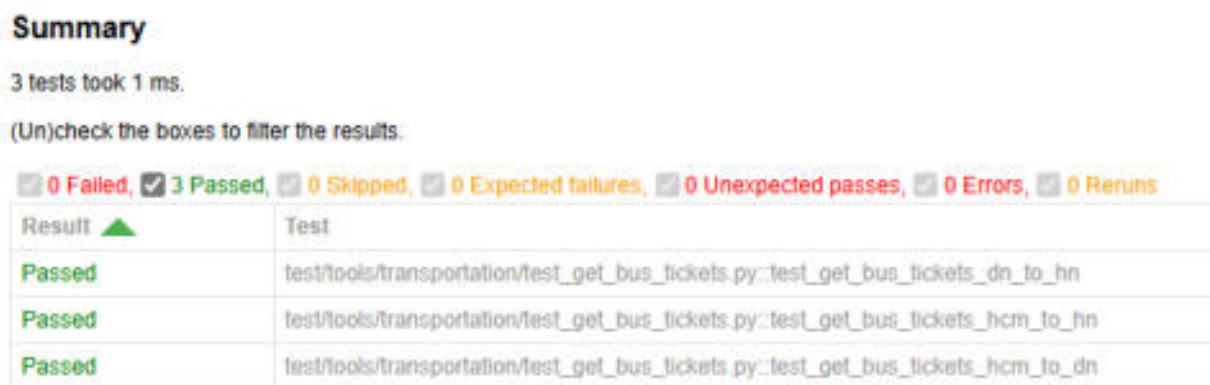


Hình 3.6 Chatbot sử dụng công cụ để tìm vé tàu

3.2 Kiểm thử

3.2.1 Kiểm thử đơn vị

3.2.1.1 Kiểm thử công cụ tìm kiếm vé xe khách



Hình 3.7 Kiểm thử công cụ tìm kiếm vé xe khách

3.2.1.2 Kiểm thử công cụ tìm kiếm vé tàu hỏa

Summary

3 tests took 1 ms.

(Un)check the boxes to filter the results.

0 Failed,
 3 Passed,
 0 Skipped,
 0 Expected failures,
 0 Unexpected passes,
 0 Errors,
 0 Reruns

Result ▲	Test
Passed	test/tools/transportation/test_get_train_tickets.py::test_get_train_tickets_dn_to_hn
Passed	test/tools/transportation/test_get_train_tickets.py::test_get_train_tickets_hcm_to_hn
Passed	test/tools/transportation/test_get_train_tickets.py::test_get_train_tickets_hcm_to_dn

Hình 3.8 Kiểm thử công cụ tìm kiếm vé tàu hỏa

3.2.1.3 Kiểm thử công cụ tìm kiếm vé máy bay

Summary

3 tests took 1 ms.

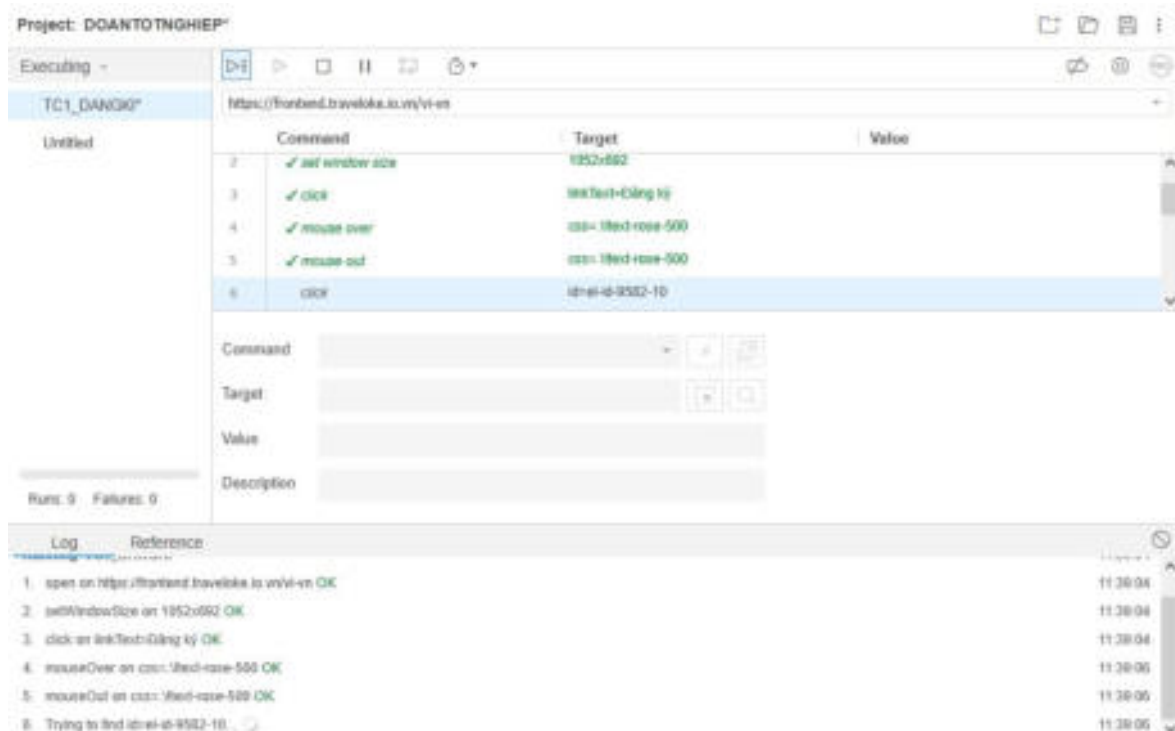
(Un)check the boxes to filter the results.

0 Failed,
 3 Passed,
 0 Skipped,
 0 Expected failures,
 0 Unexpected passes,
 0 Errors,
 0 Reruns

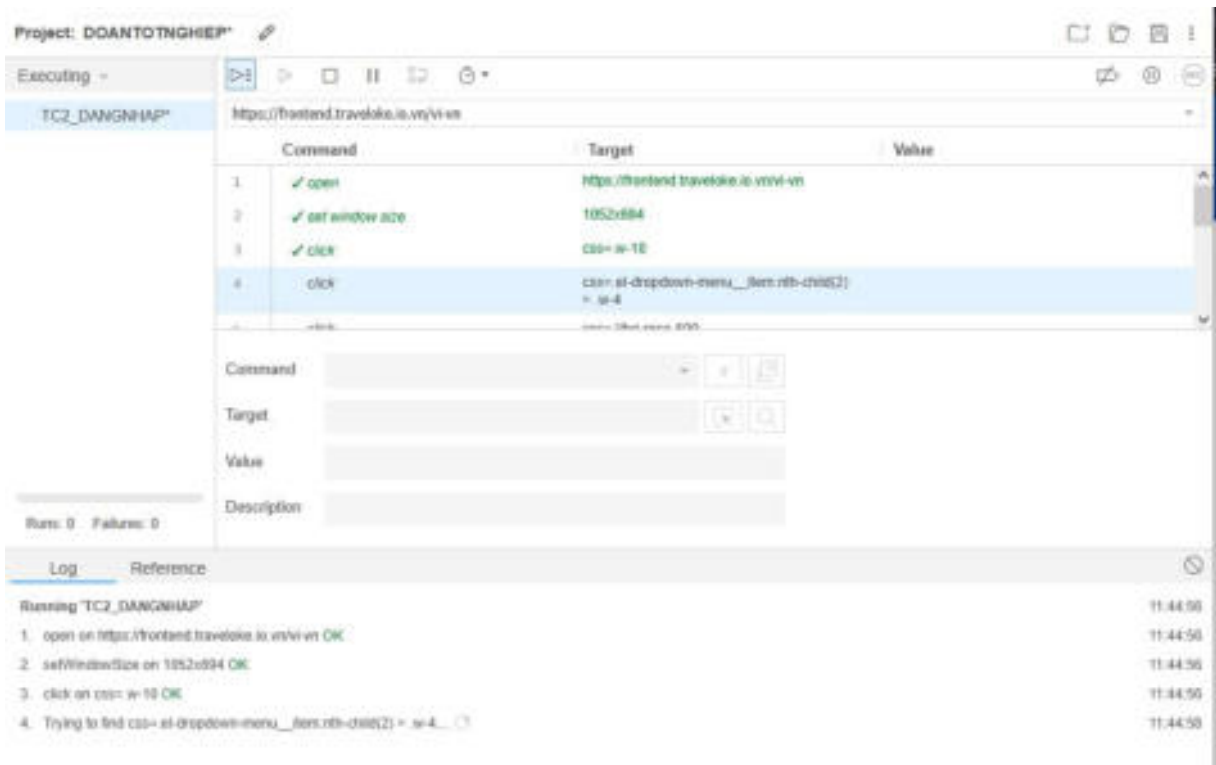
Result ▲	Test
Passed	test/tools/transportation/test_get_flight_tickets.py::test_get_flight_tickets_dn_to_hn
Passed	test/tools/transportation/test_get_flight_tickets.py::test_get_flight_tickets_hcm_to_hn
Passed	test/tools/transportation/test_get_flight_tickets.py::test_get_flight_tickets_hcm_to_dn

Hình 3.9 Kiểm thử công cụ tìm kiếm vé máy bay

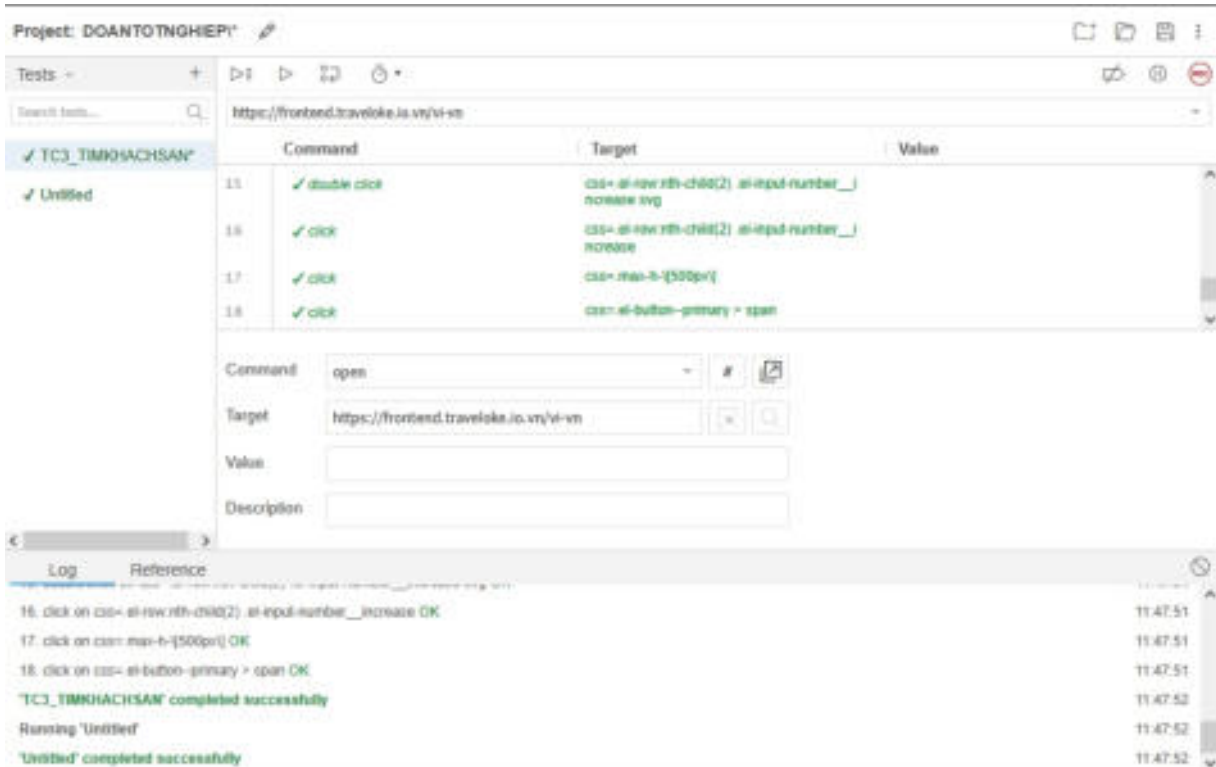
3.2.2 Kiểm thử tự động trình duyệt bằng selenium



Hình 3.10 Kiểm thử giao diện đăng ký tài khoản



Hình 3.11 Kiểm thử giao diện đăng nhập



Hình 3.12 Kiểm thử giao diện tìm kiếm khách sạn

3.3 Tinh chỉnh mô hình

Dataset sample

Displaying the first 3 rows of the tuning dataset (samples may be truncated for preview)

Message

City/Province: Đà Nẵng # Tourist spots constraints: Mikazuki # Number of adults: 3 Please research tourist spots that meet all requirements.

[thought]: 'User is asking for information about Mikazuki in Da Nang for 3 adults. I need to search for details about Mikazuki, including its name, description, location, opening hours, and price. I will start with a general search query to get the basic information and then formulate more specific queries if needed to find the missing details. Once I have the price per adult, I will calculate...'

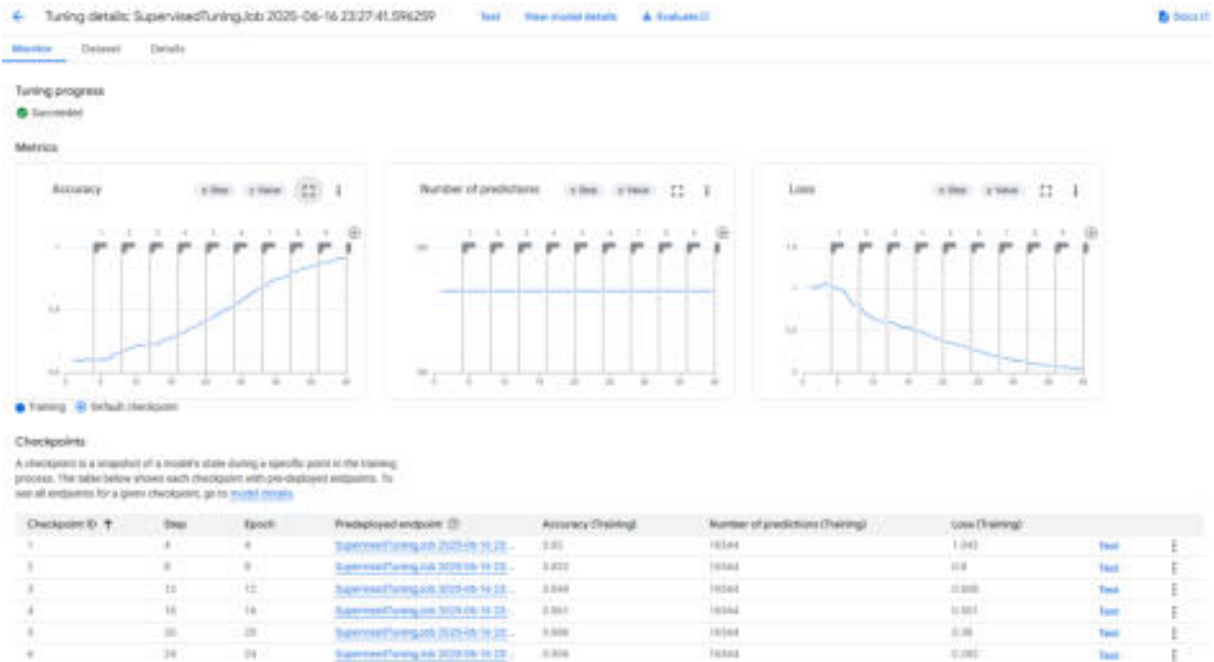
Search query: Mikazuki Đà Nẵng giá vé giờ mở cửa địa chỉ [{"url": "https://mia.vn/cam-nang-du-lich/cong-vien-nuoc-mikazuki-da-nang-cong-vien-tro-choi-nuoc-phong-cach-nhat-1805", "title": "Công viên nước Mikazuki Đà Nẵng - Giá vé cập nhật 2025 - MIA.vn", "text": "[Trang chủ](https://mia.vn/) - / - [MIA Go](https://mia.vn/cam-nang-du-lich/cam-nang-du-lich) - / - [Đà Nẵng](https://mia.vn/cam-..."}]

[completed]: true, 'tourist_spots': [{"note": "Giá vé 400,000 VND/người lớn áp dụng cho Thứ 2 - Thứ 6, bao gồm Công viên nước + Onsen. Giá vé 450,000 VND/người lớn áp dụng cho Thứ 7 và Chủ Nhật, bao gồm Công viên nước + Onsen. Trẻ em dưới 1m được miễn phí vé. Tổng chi phí ước tính cho 3 người lớn là 1,200,000 VND (Thứ 2 - Thứ 6) hoặc 1,350,000 VND (Thứ 7 và Chủ Nhật).", "description": "Công viên..."}]

City/Province: Hội An # Tourist spots constraints: Phố cổ Hội An # Number of adults: 3 Please research tourist spots that meet all requirements.

[thought]: 'Searching for tourist spots within Phố cổ Hội An and their general information like entrance fees and opening hours. I will then need to find specific details for individual popular spots within the ancient town.', 'completed': False, 'search_queries': ['các địa điểm tham quan trong Phố cổ Hội An', 'giá vé tham quan Phố cổ Hội An', 'giờ mở cửa Phố cổ Hội An'], 'tourist_spots': []]

Hình 3.13 Tập dữ liệu mẫu để tinh chỉnh cho tác nhân tìm địa điểm du lịch



Hình 3.14 Kết quả tinh chỉnh mô hình Gemini 2.0 trên Google Vertex AI

Kết luận

Trong khuôn khổ đề tài, em đã tiến hành xây dựng thành công một hệ thống lập kế hoạch du lịch thông minh dựa trên mô hình ngôn ngữ lớn sử dụng với kiến trúc đa tác nhân. Hệ thống cho phép người dùng tương tác với chatbot để được hỗ trợ lên lịch trình, tư vấn điểm đến và tìm kiếm dịch vụ du lịch phù hợp.

Kết quả đạt được:

- Xây dựng được một nền tảng tổng hợp thông tin giá phòng khách sạn và vé máy bay từ nhiều trang cung cấp khác nhau. Hệ thống có thể thu thập, phân tích và so sánh giá, từ đó hỗ trợ người dùng lựa chọn phương án tối ưu về chi phí và tiện ích.
- Áp dụng mô hình LLM để tạo ra một chatbot thông minh có khả năng hiểu ngôn ngữ tự nhiên, hỗ trợ gợi ý điểm đến phù hợp theo sở thích, thời gian và ngân sách.
- LLM còn có khả năng lên lịch trình tự động, phân tích thời gian di chuyển, thời tiết, giờ mở cửa và xây dựng kế hoạch chi tiết theo từng ngày.
- Thiết kế kiến trúc hệ thống theo hướng multi-agent, mỗi agent đảm nhận một chức năng chuyên biệt, giúp hệ thống có tính mở rộng và dễ bảo trì.

Đóng góp:

- Đề tài là một minh chứng cho tiềm năng ứng dụng của mô hình ngôn ngữ lớn trong lĩnh vực du lịch thông minh.
- Việc tích hợp nhiều nguồn dữ liệu và xử lý hội thoại theo ngữ cảnh giúp cải thiện trải nghiệm người dùng và mở ra hướng phát triển các hệ thống trợ lý ảo cá nhân hóa trong tương lai.

Đề xuất và kiến nghị:

- Cần tiếp tục mở rộng hệ thống với dữ liệu thời gian thực, đặc biệt là thời tiết, đánh giá du khách, sự kiện địa phương nhằm nâng cao chất lượng lịch trình.
- Nên bổ sung thêm các tác nhân chuyên trách như: agent kiểm tra chính sách visa, agent dự đoán chi phí, agent gợi ý món ăn địa phương... để hoàn thiện hành trình du lịch toàn diện.
- Cần quan tâm đến vấn đề bảo mật và quyền riêng tư trong việc lưu trữ thông tin cá nhân người dùng khi triển khai hệ thống thực tế.

Tổng kết lại, đề tài không chỉ đạt được các mục tiêu đề ra mà còn mở ra tiềm năng lớn trong việc ứng dụng AI và LLM vào lĩnh vực du lịch, góp phần hiện thực hóa trải nghiệm du lịch tự động, thông minh và cá nhân hóa cho mỗi người dùng.

Tài liệu tham khảo

- [1] Vaswani et al, “Attention is All You Need,” trong *In Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [2] Pranab Sahoo et al, “A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications,” 2025.
- [3] Nariman Naderi et al, “Evaluating Prompt Engineering Techniques for Accuracy and Confidence Elicitation in Medical LLMs,” 2025.
- [4] Yongchao Zhou et al, “Large Language Models Are Human-Level Prompt Engineers,” 2022.
- [5] Jason Wei et al, “Chain-of-Thought Prompting Elicits Reasoning,” 2023.
- [6] B. e. at, “Language Models are Few-Shot Learners,” 2020.
- [7] W. & Pavlick, “Do Prompt-Based Models Really Understand the Meaning of Their Prompts?,” 2022.
- [8] Z. e. al, “Calibrate Before Use: Improving Few-Shot Performance of Language Models,” 2021.
- [9] L. e. al, “Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity,” 2022.
- [10] Z. e. al, “Robustness of Demonstration-based Learning Under Limited Data Scenario,” 2022.
- [11] M. e. al, “Fantastically Ordered Prompts and Where to Find Them: Overcoming Few-Shot Prompt Order Sensitivity,” 2022.
- [12] K. S. & M. W. Nicholas R. Jennings, “A roadmap of agent research and development,” tập 1, pp. 7-38, March 1998.
- [13] M. Wooldridge, trong *An Introduction to MultiAgent Systems*, 2009.
- [14] A. S. & G. M. P. Rao, “BDI agents: From theory to practice,” trong *First International Conference on Multi-Agent Systems*, 1995.
- [15] A. S. Rao, “AgentSpeak(L): BDI agents speak out in a logical computable language,” *MAAMAW*, 1996.
- [16] F. C. G. & G. D. Bellifemine, *Developing Multi-Agent Systems with JADE*, Wiley, 2007.

- [17] Y. e. a. Wang, “Robin: A Self-Driven Multi-Agent System for Automated Scientific Discovery,” 2024.
- [18] J. Z. D. Y. N. D. I. S. K. N. Y. C. Shunyu Yao, “ReAct: Synergizing Reasoning and Acting in Language Models,” 2023.
- [19] B. Z. X. L. J. C. J. X. H. M. Chengbo He, “Enhancing LLM Reasoning with Multi-Path Collaborative Reactive and Reflection agents,” 2025.