

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

**ĐỀ TÀI: XÂY DỰNG HỆ THỐNG GỢI Ý NHỮNG ĐỊA ĐIỂM
NÊN ĐẾN KHI ĐI DU LỊCH**

Giảng viên hướng dẫn: TS. NGUYỄN ĐÌNH HIỀN

Sinh viên thực hiện: VĂN CÔNG HÀO

Lớp : CÔNG NGHỆ THÔNG TIN

Khoá : 61

Tp. Hồ Chí Minh, năm 2024

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN**



BÁO CÁO ĐỒ ÁN TỐT NGHIỆP

**ĐỀ TÀI: XÂY DỰNG HỆ THỐNG GỢI Ý NHỮNG ĐỊA ĐIỂM
NÊN ĐẾN KHI ĐI DU LỊCH**

Giảng viên hướng dẫn: TS. NGUYỄN ĐÌNH HIỀN

Sinh viên thực hiện: VĂN CÔNG HÀO

Lớp : CÔNG NGHỆ THÔNG TIN

Khoá : 61

Tp. Hồ Chí Minh, năm 2024

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP
BỘ MÔN: CÔNG NGHỆ THÔNG TIN

-----***-----

Mã sinh viên: 6151071045

Họ tên SV: Văn Công Hào

Khóa: 61

Lớp: CQ.61.CNTT

1. Tên đề tài

Xây dựng hệ thống gợi ý những địa điểm nên đến khi đi du lịch.

2. Mục đích, yêu cầu

2.1 Mục đích

Xây dựng một hệ thống gợi ý các địa điểm nên đến khi đi du lịch tại một khu vực bất kỳ, từ những gợi ý đó giúp du khách tiết kiệm thời gian tìm kiếm thông tin địa điểm, từ đó lập nên kế hoạch cho chuyến đi một cách hiệu quả, tối ưu. Bên cạnh đó, góp phần thúc đẩy sự phát triển mạnh mẽ và bền vững của ngành du lịch Việt Nam.

2.2 Yêu cầu

- Phân tích nhu cầu và ngân sách của người dùng: Hệ thống cần có khả năng hiểu rõ nhu cầu và ngân sách của người dùng để đưa ra các gợi ý phù hợp. Điều này có thể đạt được thông qua việc thu thập và phân tích dữ liệu từ người dùng.
- Gợi ý địa điểm du lịch phù hợp: Dựa trên nhu cầu và ngân sách người dùng, hệ thống cần đưa ra gợi ý về địa điểm du lịch phù hợp. Các gợi ý này nên bao gồm thông tin mô tả địa điểm như vị trí, chi phí, trang web giới thiệu (nếu có),.... và các hoạt động có thể tham gia.

- Lập kế hoạch cho chuyến đi: Ngoài việc gợi ý các địa điểm, hệ thống cũng cần hỗ trợ người dùng trong việc lập kế hoạch cho chuyến đi của họ. Điều này có thể bao gồm việc: cung cấp lộ trình di chuyển sao cho quãng đường đi là ngắn nhất, đề xuất nơi lưu trú, ăn uống, vui chơi/tham quan, quán cà phê và ước tính chi phí cho từng hạng mục.
- Dễ sử dụng và thân thiện với người dùng: Hệ thống cần được thiết kế một cách trực quan và dễ sử dụng, giúp du khách dễ dàng sử dụng các tính năng của hệ thống.
- Cập nhật thông tin liên tục: Ngành du lịch luôn thay đổi và phát triển, do đó hệ thống cần có khả năng cập nhật thông tin một cách liên tục để đảm bảo các gợi ý luôn cập nhật và phù hợp.

3. Nội dung và phạm vi đề tài

3.1 Nội dung của đề tài

- Phân tích nhu cầu và ngân sách của người dùng.
- Xây dựng hệ thống có khả năng gợi ý các địa điểm du lịch phù hợp dựa trên nhu cầu và ngân sách của người dùng.
- Lập kế hoạch cho chuyến đi.
- Thiết kế hệ thống dễ sử dụng và thân thiện với người dùng.
- Cập nhật thông tin liên tục.

3.2 Phạm vi đề tài

- Hệ thống sẽ tập trung vào việc gợi ý các địa điểm du lịch trong nước, đặc biệt là các khu vực tập trung nhiều khách du lịch như: Hồ Chí Minh, Hà Nội.
- Hệ thống sẽ hỗ trợ người dùng lên danh sách những địa điểm cho chuyến đi với loại hình du lịch bụi – một loại hình du lịch với chi phí không quá cao - thời gian đi là hai ngày một đêm và đi trong ngày (một ngày).

4. Công nghệ, công cụ và ngôn ngữ lập trình

4.1 Công cụ

- SQL Server: Là hệ thống quản lý cơ sở dữ liệu quan hệ (Relational Database Management System) của Microsoft, giúp bạn lưu trữ và quản lý dữ liệu cho hệ thống của mình.
- Microsoft Visual Studio: Là môi trường phát triển tích hợp (IDE) của Microsoft, hỗ trợ nhiều ngôn ngữ lập trình và công cụ, giúp bạn dễ dàng viết, kiểm tra và debug code.
- Postman: Là một công cụ phổ biến trong việc thử nghiệm các API. Với nó, ta có thể gọi REST API mà không cần viết bất kỳ một lệnh code nào.
- Bing Maps API: Là một công cụ mạnh mẽ có thể được sử dụng để tạo ra các ứng dụng và trang web bản đồ sáng tạo và hữu ích, do Microsoft phát triển.

4.2 Ngôn ngữ lập trình

C#: Là ngôn ngữ lập trình đa mục đích, hướng đối tượng, được Microsoft phát triển. C# rất phù hợp để xây dựng các ứng dụng Windows và Web.

5. Các kết quả chính dự kiến sẽ đạt được và ứng dụng

5.1 Các kết quả chính dự kiến

- Hệ thống gợi ý địa điểm du lịch thông minh: Hệ thống này sẽ giúp du khách tiết kiệm thời gian và công sức trong việc tìm kiếm thông tin và lập kế hoạch cho chuyến đi. Nó sẽ phân tích nhu cầu và ngân sách của du khách, gợi ý các địa điểm phù hợp và đề xuất cách di chuyển giữa các địa điểm trong chuyến đi tốt nhất.
- Cơ sở dữ liệu về địa điểm du lịch: Cơ sở dữ liệu này sẽ chứa thông tin chi tiết về các địa điểm du lịch, bao gồm vị trí, chi phí, thông tin giới thiệu, hình ảnh, và các hoạt động có thể tham gia.

5.2 Ứng dụng của đề tài

- Hỗ trợ du khách: Hệ thống gợi ý địa điểm du lịch sẽ giúp du khách lên kế hoạch cho chuyến đi một cách hiệu quả và tối ưu, giúp họ tận hưởng chuyến đi một cách trọn vẹn nhất.
- Thúc đẩy ngành du lịch: Bằng cách giới thiệu những điểm đến mới và hấp dẫn, hệ thống có thể góp phần thúc đẩy du lịch địa phương và đa dạng hóa lựa chọn cho du khách.
- Nghiên cứu và phát triển: Kết quả của đề tài có thể được sử dụng như một nền tảng cho các nghiên cứu và ứng dụng tiếp theo trong lĩnh vực du lịch và công nghệ thông tin.

6. Giáo viên và cán bộ hướng dẫn

Họ tên: TS.Nguyễn Đình Hiền.

Đơn vị công tác: Trường Đại Học Công Nghệ Thông Tin, Trường Đại Học Quốc Gia Thành Phố Hồ Chí Minh.

Điện thoại: 0918 735 299

Email: hiennd@uit.edu.vn

Ngày tháng năm 2024
Trưởng BM Công nghệ Thông tin

Đã giao nhiệm vụ TKTN
Giáo viên hướng dẫn

ThS. Trần Phong Nhã

TS.Nguyễn Đình Hiền

Đã nhận nhiệm vụ TKTN

Sinh viên: Văn Công Hào

Điện thoại: 0397841034

Ký tên:

Email: 6151071045@st.utc2.edu.vn

LỜI CẢM ƠN

Trong suốt quá trình thực hiện đồ án tốt nghiệp với đề tài “Xây dựng hệ thống gợi ý những địa điểm nên đến khi đi du lịch”, đây là kết quả của sự phấn đấu không ngừng nghỉ của bản thân. Bên cạnh đó để có thể hoàn thành đồ án một cách tốt nhất, em xin bày tỏ lòng biết ơn chân thành đến thầy giáo hướng dẫn – TS.Nguyễn Đình Hiền – người đã hướng dẫn em một cách tận tâm trong suốt quá trình làm đồ án. Thầy đã dành thời gian để giải đáp những thắc mắc của em, đưa ra những gợi ý và giải pháp thiết thực để hoàn thành đồ án.

Em cũng muốn gửi lời cảm ơn của bản thân đến các thầy cô giáo Trường Đại Học Giao Thông Vận Tải Phân Hiệu Tại Thành Phố Hồ Chí Minh nói chung, cũng như các thầy cô giáo trong Bộ môn Công Nghệ Thông Tin nói riêng. Những người đã truyền đạt cho em những kiến thức về các môn đại cương và chuyên ngành, giúp em có những nền tảng lý thuyết vững vàng và tạo điều kiện cho em trong suốt quá trình học tập.

Cuối cùng, em xin chân thành cảm ơn gia đình và bạn bè, những người luôn sát cánh, tạo điều kiện, quan tâm, giúp đỡ và động viên em trong suốt quá trình học tập và hoàn thành đồ án tốt nghiệp.

Mặc dù đã cố gắng hết sức, nhưng do điều kiện về thời gian hạn hẹp, lượng kiến thức về đề tài rộng lớn và kinh nghiệm còn hạn chế của bản thân, đồ án này không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự chỉ bảo, góp ý của các thầy cô để em có thể bổ sung, hoàn thiện và nâng cao chất lượng đồ án. Những bài học và kinh nghiệm mà em học hỏi được từ thầy cô sẽ là hành trang quý giá cho công việc thực tế sau này.

Em xin chân thành cảm ơn!

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp. Hồ Chí Minh, ngày tháng năm
Giáo viên hướng dẫn

TS.Nguyễn Đình Hiến

MỤC LỤC

NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP	ii
DANH MỤC CHỮ VIẾT TẮT	x
DANH MỤC BẢNG BIỂU	xi
DANH MỤC HÌNH ẢNH	xii
CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI	1
1.1 Lý do chọn đề tài	1
1.1.1 Khái quát ngành du lịch hiện nay.....	1
1.1.2 Lý do lựa chọn đề tài.....	3
1.2 Mục tiêu của đề tài.....	5
1.3 Phạm vi và đối tượng nghiên cứu	6
1.4 Phương pháp nghiên cứu	7
1.5 Đóng góp của đề tài	8
1.6 Lịch trình nghiên cứu.....	9
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	10
2.1 Khái niệm về hệ thống tư vấn du lịch.....	10
2.2 Khảo sát các hệ thống tư vấn du lịch	10
2.3 Các giải pháp thiết kế hệ thống tư vấn du lịch	11
2.4 Các thuật toán liên quan.....	12
2.4.1 Thuật toán quay lui (Backtracking)	12
2.4.2 Thuật toán đàn kiến (Ant Algorithm)	13
CHƯƠNG 3: THIẾT KẾ HỆ THỐNG TƯ VẤN DU LỊCH	18
3.1 Phân tích hệ thống.....	18
3.1.1 Yêu cầu hệ thống.....	18
3.1.2 Kiến trúc của hệ thống	19
3.2 Thiết kế cơ sở dữ liệu	20
3.2.1 Thu thập dữ liệu	20
3.2.2 Mô hình ER.....	23

3.2.3 Mô hình quan hệ.....	24
3.3 Thiết kế các chức năng của hệ thống.....	26
3.3.1 Thuật giải giải quyết vấn đề thứ nhất.....	27
3.3.2 Thuật giải giải quyết vấn đề thứ hai.....	28
3.3.3 Tối ưu hóa lộ trình.....	31
CHƯƠNG 4: THỬ NGHIỆM HỆ THỐNG.....	42
4.1 Các công cụ cài đặt	42
4.1.1 Cơ sở dữ liệu SQL Server	42
4.1.2 Công cụ Microsoft Visual Studio.....	44
4.1.3 Công cụ Postman.....	45
4.1.4 Bing Maps API.....	46
4.2 Giao diện hệ thống và thử nghiệm các chức năng.....	49
4.3 Kết quả thử nghiệm.....	51
KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	53
1. Kết quả đạt được	53
2. Hạn chế	53
3. Hướng phát triển	54
TÀI LIỆU THAM KHẢO.....	55

DANH MỤC CHỮ VIẾT TẮT

STT	Mô tả	Ý nghĩa	Ghi chú
1	RDBMS	Relational Database Management System: hệ quản trị cơ sở dữ liệu quan hệ	
2	SQL	Structured Query Language	
3	SSAS	SQL Server Analysis Services	
4	SSRS	SQL Server Reporting Services	
5	ETL	Extract – Transform – Load	
6	SSIS	SQL Server Integration Services	
7	OOP	Object-oriented programming	
8	CLR	Common Language Runtime	
9	IDE	Integrated Development Environment	
10	ER	Entity – Relationship	
11	API	Application Programming Interface	
12	REST API	Representational State Transfer API	
13	HTTP	HyperText Transfer Protocol	
14	SOAP	Simple Object Access Protocol	
15	XML	Extensible Markup Language	
16	TSP	Travelling Salesman Problem	
17	JSON	JavaScript Object Notation	
18	IDE	Integrated Development Environment	
19	UWP	Universal Windows Platform	

DANH MỤC BẢNG BIỂU

Bảng 1. 1: Tổng thu từ khách du lịch giai đoạn 2008 – 2023	3
Bảng 1. 2: Lịch trình nghiên cứu	9
Bảng 3. 1: Area	25
Bảng 3. 2: TypeLocation	25
Bảng 3. 3: Location	25

DANH MỤC HÌNH ẢNH

Hình 1. 1: Biểu đồ lượt khách quốc tế đến qua các năm 2008 – 2024.....	2
Hình 1. 2: Số lượt khách du lịch nội địa năm 2024 (tháng 1 đến tháng 3)	2
Hình 2. 1: Thí nghiệm “Chiếc cầu đôi” (Double Bridge Experiment).....	14
Hình 2. 2: Lottery Wheel.....	16
Hình 3. 1: Kiến trúc hệ thống	19
Hình 3. 2: Mô hình ER	24
Hình 3. 3: Mô hình diagram	24
Hình 3. 4: Độ dài dây cung.....	32
Hình 3. 5: Kết quả JSON	37
Hình 3. 6: Kết quả JSON	38
Hình 4. 1: SQL Server	42
Hình 4. 2: Microsoft Visual Studio	44
Hình 4. 3: Công cụ Postman.....	45
Hình 4. 4: GraphQL API	47
Hình 4. 5: Bing Maps	47
Hình 4. 6: Dữ liệu đầu vào	49
Hình 4. 7: Cần nơi lưu trữ không ?.....	50
Hình 4. 8: Danh sách tổ hợp các địa điểm.....	50
Hình 4. 9: Không thể khởi tạo	50
Hình 4. 10: Thứ tự di chuyển	51
Hình 4. 11: Kết quả tối ưu chi phí	51
Hình 4. 12: Thông tin lộ trình.....	52
Hình 4. 13: Hiệu suất chạy	52

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

1.1 Lý do chọn đề tài

1.1.1 Khái quát ngành du lịch hiện nay

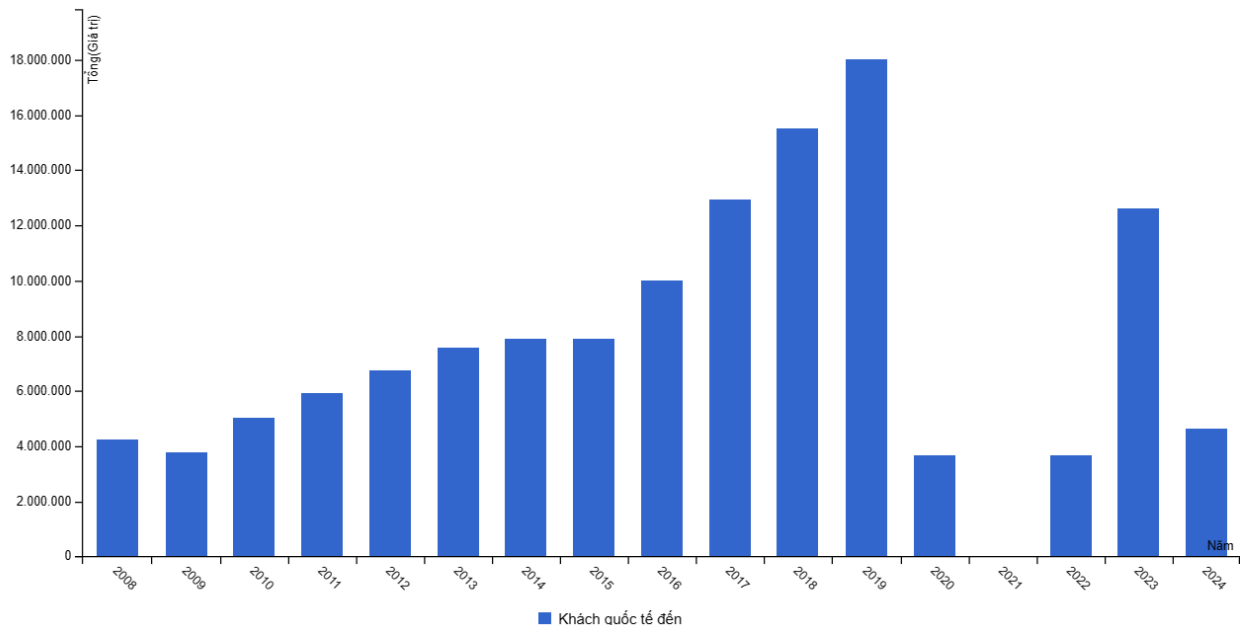
Ngành du lịch Việt Nam đã và đang khẳng định vị thế quan trọng trong nền kinh tế quốc dân, dự kiến ngành du lịch sẽ đóng góp hơn 6% GDP vào năm 2024 và hứa hẹn sẽ tiếp tục tăng trưởng vào thời gian tới. Mặc dù bị ảnh hưởng bởi đại dịch COVID-19, ngành du lịch nước ta đã cho thấy những dấu hiệu hồi phục lạc quan, nhờ vào việc mở cửa cho các du khách quốc tế vào tháng 03 năm 2022. Kết quả là, theo số liệu do Cục Du lịch Quốc gia Việt Nam thông tin, năm 2023, du lịch Việt Nam đã có sự phục hồi ngoạn mục, vượt ra ngoài chỉ tiêu ban đầu đề ra khi ngành đã đón 12.602.434 lượt khách quốc tế tăng 245,3% so với năm 2022. Khách nội địa đạt 108.200 lượt, tăng 6,8% so với năm 2022, tổng thu của ngành du lịch đạt 678,3 nghìn tỷ đồng, tăng 37,0% so với năm 2022.

Sự đa dạng về cảnh quan thiên nhiên và văn hóa là điểm nhấn độc đáo của du lịch Việt Nam. Thủ đô Hà Nội níu chân du khách bởi nét đẹp cổ kính hòa quyện với hiện đại, với những món ăn đường phố hấp dẫn như phở, bún chả, bún thang và những quán cà phê ấm cúng ẩn mình trong những con ngõ nhỏ. Thành phố Hồ Chí Minh, trung tâm thương mại của Việt Nam, với những di tích lịch sử như Dinh Độc Lập, Chợ Bến Thành,... và nhịp sống đô thị sôi động của một thành phố trẻ. Vịnh Hạ Long, di sản thiên nhiên thế giới, ẩn chứa vẻ đẹp hùng vĩ của núi non và hang động kỳ bí, được ví như "viên ngọc bích" của ngành du lịch Việt Nam. Phố cổ Hội An thu hút du khách bởi kiến trúc cổ kính đã nhuộm màu thời gian và nghệ thuật thủ công tinh tế, như một bức tranh cổ kính giữa lòng phố thị. Bên cạnh đó, Việt Nam còn sở hữu nhiều điểm đến hấp dẫn không kém phần nổi tiếng khác như: Đà Nẵng với những bãi biển thơ mộng, Huế mộng mơ với di sản văn hóa Cố đô, Sapa ẩn chứa vẻ đẹp hoang sơ của núi rừng Tây Bắc, Phú Quốc với những hòn đảo nguyên sơ và thơ mộng.

Ngành du lịch Việt Nam không chỉ đóng góp vào sự phát triển kinh tế mà còn tạo ra nhiều cơ hội việc làm, thúc đẩy các ngành công nghiệp liên quan và quảng bá hình ảnh đất nước, con người Việt Nam ra thế giới. Với tiềm năng to lớn và sự nỗ lực của Chính phủ

cùng các doanh nghiệp du lịch, Việt Nam hứa hẹn sẽ trở thành điểm đến du lịch hấp dẫn du khách quốc tế trong tương lai.

Sau đây là một vài số liệu của Cục Du lịch Quốc gia Việt Nam thông tin:



Hình 1. 1: Biểu đồ lượt khách quốc tế đến qua các năm 2008 – 2024

(Nguồn: Tổng cục Thống kê, Tổng cục Du Lịch)

Chỉ tiêu	Tháng 1	Tháng 2	Tháng 3	Tổng
Số lượt khách du lịch nội địa (Nghìn lượt người)				
Lượt khách	7.500	14.000	8.500	30.000
Phân theo nhóm khách (Nghìn lượt người)				
Khách tham quan trong ngày	2.500	9.100	5.500	17.100
Khách có nghỉ qua đêm tại cơ sở lưu trú	5.000	4.900	3.000	12.900

Hình 1. 2: Số lượt khách du lịch nội địa năm 2024 (tháng 1 đến tháng 3)

(Nguồn: vietnamtourism.gov.vn)

Bảng 1. 1: Tổng thu từ khách du lịch giai đoạn 2008 – 2023

Năm	Tổng thu từ khách du lịch (nghìn tỷ đồng)	Tốc độ tăng trưởng (%)
2023	678.30	37,0
2022	495.00	175,0
2021	180.00	-42,3
2020	312.00	-57,0
2019	726.00	14,0
2018	637.00	17,7
2017	541.00	29,7
2016	417.27	17,5
2015	355.55	*
2014	322.86	11,4
2013	289.84	80,6
2012	160.00	23,1
2011	130.00	35,4
2010	96.00	41,2
2009	68.00	13,3
2008	60.00	

(Nguồn: Tổng cục Du lịch, Báo cáo TSA 2013 - 2015)

1.1.2 Lý do lựa chọn đề tài

Thông qua khái quát tình hình ngành du lịch nước ta hiện nay, ta thấy rằng con người đang có xu hướng dành nhiều thời gian đi du lịch. “Tại sao bạn lại đi du lịch?”, khi hỏi mọi người chúng ta sẽ nhận được nhiều câu trả lời: “Tôi đi đến đó để biết ở đó người dân sống thế nào, món ăn ở đó như thế nào!”, “Hang động đó trông như thế nào!”, “Muốn làm quen

những người bạn mới, ở những vùng đất mới!”,.... Nhìn chung, ta có thể hình dung các lý do mà ngay nay con người lại đi du lịch thường xuyên hơn:

Họ muốn khám phá và trải nghiệm: Du lịch cho phép con người khám phá những địa điểm mới, trải nghiệm những nền văn hóa khác nhau và thử những món ăn độc đáo. Đó là những điều mà bạn không thể biết nếu không đi du lịch, đồng thời du lịch giúp bạn mở rộng tầm nhìn và tạo ra những kỷ niệm đáng nhớ.

Hiểu rõ hơn nơi mình sinh ra và lớn lên: Đi du lịch khám phá nhiều nơi giúp cho con người ta có sự mở rộng hiểu biết về thế giới, về lịch sử, văn hóa và phong tục, tập quán ở nhiều nơi, sẽ tạo điều kiện cho bạn hiểu hơn về quê hương, đất nước của mình. Hiểu được tại sao mặc dù cũng sống trong cùng một đất nước, hay cùng một châu lục với nhau lại có suy nghĩ, lối sống và sinh hoạt khác nhau. Việc này giúp chúng ta cảm thấy trân trọng, yêu quê hương đất nước mình hơn.

Du lịch giúp con người ta hiểu về bản thân hơn: Như cách mà David Mitchell - tiểu thuyết gia người Anh, từng nói “*Travel enough, you meet yourself.*” (tạm dịch: “*Một khi đã đi du lịch đủ nhiều, bạn sẽ hiểu được bạn là ai.*”). Bằng cách đi du lịch bạn sẽ phải học cách tự chăm sóc bản thân mình, học các kỹ năng sinh tồn trong những hoàn cảnh và môi trường khác biệt luôn thay đổi không ngừng. Việc này không những giúp bạn phát triển toàn diện và hoàn thành kỹ năng sống, mà còn giúp bạn khám phá những sở thích mới, niềm vui mới. Bạn sẽ hiểu được rằng mình phù hợp với môi trường nào, những tình huống nào sẽ gây trở ngại cho bạn khi gặp phải, cũng như biết được cách khắc phục.

Giúp con người ta có những mối quan hệ mới và bồi đắp tình cảm cho các mối quan hệ hiện tại: Trong suốt cuộc hành trình, bạn sẽ được tiếp xúc và gặp gỡ nhiều người trong nhiều tình huống xã hội đa dạng. Đây vừa là cơ hội để bạn rèn luyện kỹ năng giao tiếp, kỹ năng ngoại ngữ và cũng vừa là dịp để bạn có thể hiểu thêm những người bạn mới này thông qua trò chuyện. Ngoài ra việc đi du lịch với gia đình và bạn bè là cơ hội để mọi người gắn kết tình cảm, hiểu rõ nhau hơn, biết được những câu chuyện trong cuộc sống của nhau, cùng đưa ra hướng giải quyết. Cùng nhau lưu lại những khoảnh khắc vui vẻ và tạo ra những kỷ niệm đẹp.

Du lịch giúp bạn giải tỏa căng thẳng và tìm kiếm cảm hứng: Những chuyến du lịch giúp tạo ra những “khoảng nghỉ” cho bạn trong công việc. Cho dù đó có là chuyến du lịch

ngắn ngày, ví dụ như vào cuối tuần, một buổi chiều muộn hay sáng sớm bình minh, thì bản thân việc đi du lịch đến những nơi không đông người, ít khói bụi và tiếng ồn, gần gũi với thiên nhiên đã có phần nào giúp bạn giải tỏa căng thẳng, khởi dậy trong con người bạn cảm hứng sáng tạo, ý tưởng mới, độc đáo cho công việc, cuộc sống và nghệ thuật.

Cải thiện, tăng cường sức khỏe và trân trọng cuộc sống hơn: Du lịch thường đi kèm với các hoạt động thể chất như: đi bộ, leo núi, bơi lội, đạp xe,... Những hoạt động này sẽ giúp con người tăng cường sức khỏe, cải thiện thể chất và tinh thần. Giúp con người ta trân trọng cuộc sống hiện tại, biết ơn những gì mà mình đang có và sống một cuộc sống trọn vẹn hơn.

Vì vậy, để có được những cuộc hành trình một cách trọn vẹn thì cần có một kế hoạch chi tiết và chính chu bao gồm: nơi nghỉ ngơi – lưu trú – trong suốt quá trình du lịch, những địa điểm thăm quan không thể bỏ qua, các món ngon, đặc sản cần phải thử, những quán cà phê nhẹ nhàng, phong cảnh đẹp để gặp gỡ, trò chuyện với bạn bè và quan trọng hơn hết đảm bảo chi phí cho chuyến đi trong phạm vi cho phép. Để có thể làm điều đó du khách cần phải bỏ ra khá nhiều thời gian để có thể tìm kiếm những địa điểm đó, tính toán chi tiêu cần phải trả khi đến đó, cũng như tính toán chi phí đi lại. Công việc đó có thể mất đến vài ngày hoặc thậm chí là một tuần, thấy được khó khăn đó mà bản thân cũng đã gặp phải khi lập kế hoạch đi chơi với bạn bè nên em muốn xây dựng một hệ thống giúp cho bản thân em cũng như các du khách khác tiết kiệm thời gian, rút ngắn quá trình lên kế hoạch bằng cách hệ thống sẽ thực hiện gợi ý các địa điểm lưu trú, tham quan, vui chơi, ăn uống, cafe,... sao cho phù hợp với ngân sách của người dùng. Hệ thống gợi ý địa điểm nên đến khi đi du lịch sẽ giúp thu hút du khách đến với các địa điểm mới, ít được biết đến, qua đó góp phần phân tán lượng khách du lịch, giảm tải cho các điểm du lịch nổi tiếng, đồng thời thúc đẩy phát triển du lịch ở các địa phương còn nhiều tiềm năng.

1.2 Mục tiêu của đề tài

Hệ thống này sẽ được xây dựng dựa trên các thuật toán tiên tiến, có khả năng phân tích nhu cầu và ngân sách của du khách một cách chính xác và hiệu quả. Nó sẽ gợi ý những điểm đến phù hợp với nhu cầu và ngân sách của du khách, dựa trên dữ liệu du lịch toàn diện và được cập nhật liên tục.

Đây sẽ là hệ thống có khả năng:

- Phân tích nhu cầu và ngân sách của du khách dựa trên các dữ liệu của du khách, đảm bảo tính chính xác và hiệu quả.
- Dựa trên các thuật toán tiên tiến gợi ý những điểm đến phù hợp với nhu cầu và ngân sách của du khách, dựa trên dữ liệu du lịch toàn diện và cập nhật liên tục.
- Lập kế hoạch chi tiết cho chuyến đi, bao gồm:
 - Lịch trình di chuyển tối ưu hóa thời gian và chi phí, kết hợp các phương tiện di chuyển phù hợp.
 - Gợi ý nơi lưu trú, ăn uống, vui chơi/tham quan, cà phê đa dạng, phù hợp với nhu cầu và ngân sách của du khách.
 - Ước tính chi phí cụ thể cho từng hạng mục, giúp du khách dễ dàng quản lý ngân sách.

Cung cấp cho du khách:

- Tiết kiệm thời gian và công sức trong việc lên kế hoạch cho chuyến đi, giải phóng du khách khỏi những căng thẳng, mệt mỏi trong công việc và cuộc sống.
- Tận hưởng chuyến đi một cách trọn vẹn nhất, an tâm nhất và thoải mái khám phá những điểm đến mới.

1.3 Phạm vi và đối tượng nghiên cứu

Về phạm vi nghiên cứu:

Hệ thống gợi ý những địa điểm nên đến khi đi du lịch được đề cập vào trong đề án này sẽ là các địa điểm ở những khu vực thành phố với lượng khách du lịch tập trung thường xuyên và được biết tới nhiều trong cả nước gồm: thành phố Hồ Chí Minh và thủ đô Hà Nội. Đây là hai địa điểm có nhiều điểm vui chơi, ăn uống,.. để mọi người có thể trải nghiệm, giải trí.

Hệ thống này sẽ dựa vào mức chi phí cho chuyến đi du lịch của người dùng mà lên danh sách các địa điểm có thể đến với mức chi phí đó. Số lượng địa điểm gợi ý sẽ tùy thuộc vào thời gian mà người dùng muốn đi, trong đề tài này sẽ tập trung vào hai hình thức thời gian: đi trong ngày (một ngày) và đi hai ngày một đêm. Các địa điểm nào được gợi ý sẽ được hệ thống lựa chọn thông qua việc sử dụng dữ liệu sau khi tính toán đó là chi phí cần

phải trả của một người khi đến các địa điểm đó. Song song đó, hệ thống sẽ tiến hành sắp xếp các địa điểm sao cho quãng đường đi hết tất cả các địa điểm là ngắn nhất.

Về đối tượng nghiên cứu:

Hệ thống này được xây dựng dành cho các cá nhân, nhóm người (bạn bè, gia đình, đồng nghiệp hay các cặp đôi) với mong muốn trải nghiệm du lịch một mình, riêng tư, với chi phí không quá cao nhưng vẫn có thể được trải nghiệm chuyến đi du lịch thật trọn vẹn, vui vẻ. Kiểu du lịch như thế thường được gọi là du lịch ‘bụi’. Các đối tượng này là những người có thu nhập thấp hoặc trung bình khá, muốn được đi du lịch, trải nghiệm những điều mới, tiếp thu những kiến thức mới và thưởng thức các món ăn đặc sản của các địa điểm, khu vực đó.

1.4 Phương pháp nghiên cứu

Phương pháp phân tích tổng hợp:

Mục đích: Thu thập thông tin toàn diện về du lịch trong nước, tập trung vào các khu vực, thành phố thu hút nhiều khách du lịch: thành phố Hồ Chí Minh và thủ đô Hà Nội.

Cách thực hiện:

- Thu thập, phân tích dữ liệu từ các nguồn như: các trang mạng xã hội (Facebook, TikTok, Youtube, Instagram,...), website du lịch (Traveloka, Booking,...),... phổ biến có nhiều người biết đến và sử dụng.
- Thu thập thông tin các địa điểm tham quan, vui chơi, nơi lưu trú, các quán ăn, tiệm cà phê, phương tiện di chuyển,...
- Phân tích nhu cầu và ngân sách được sử dụng cho chuyến đi của du khách trong ngày (một ngày) và hai ngày một đêm.

Phương pháp mô phỏng:

Mục đích: Lập danh sách chi tiết những địa điểm cho chuyến đi, bao gồm thông tin khoảng cách giữa các địa điểm, địa chỉ của từng địa điểm, link giới thiệu địa điểm và ước tính chi phí phát sinh cho từng địa điểm.

Cách thực hiện:

- Sử dụng các thuật toán tiên tiến và phân tích nhu cầu, ngân sách của du khách mà từ đó đưa ra những địa điểm du lịch phù hợp.

- Ước tính chi phí cho từng hạng mục của chuyến đi dựa trên dữ liệu du lịch và giá cả thị trường thông qua việc tìm hiểu ở các trang web, mạng xã hội.
- Từ những địa điểm thoả mãn các yêu cầu về ngân sách và nhu cầu của du khách, tiến hành đo lường khoảng cách giữa các địa điểm và gợi ý cho du khách lộ trình đi sao cho quãng đường đi tất cả các địa điểm là ngắn nhất.

1.5 Đóng góp của đề tài

Đối với lĩnh vực nghiên cứu:

Đề xuất phương pháp mới: Đề tài này đề xuất một phương pháp mới kết hợp các phương pháp nghiên cứu và thuật toán tiên tiến. Phương pháp này cung cấp một cách tiếp cận mới để giải quyết vấn đề gợi ý địa điểm du lịch, mở ra hướng nghiên cứu mới trong lĩnh vực này.

Cung cấp kiến thức mới về nhu cầu du khách: Đề tài này cung cấp kiến thức mới về nhu cầu du khách khi đi du lịch một mình hoặc riêng tư. Điều này giúp cải thiện hiểu biết về thị trường du lịch trong nước và hỗ trợ việc phát triển các dịch vụ du lịch phù hợp.

Đối với thực tiễn:

Phát triển hệ thống gợi ý địa điểm du lịch dễ sử dụng, hỗ trợ đa nền tảng: Hệ thống được phát triển trong đề tài này giúp giải quyết các vấn đề cụ thể mà du khách thường gặp phải khi lựa chọn địa điểm du lịch như việc tìm kiếm thông tin, lựa chọn địa điểm phù hợp với ngân sách và thời gian,....

Thúc đẩy du lịch trong nước, thu hút du khách, nâng cao doanh thu ngành du lịch: Đề tài này góp phần thúc đẩy du lịch trong nước bằng cách cung cấp một hệ thống gợi ý địa điểm du lịch hiệu quả, giúp thu hút du khách và nâng cao doanh thu cho ngành du lịch.

Nâng cao chất lượng dịch vụ du lịch, tiết kiệm thời gian và chi phí cho du khách: Hệ thống gợi ý địa điểm du lịch được phát triển trong đề tài này giúp nâng cao chất lượng dịch vụ du lịch bằng cách cung cấp thông tin chính xác và kịp thời, giúp du khách tiết kiệm thời gian và chi phí trong quá trình lựa chọn và lên kế hoạch cho chuyến đi của mình.

1.6 Lịch trình nghiên cứu

Bảng 1. 2: Lịch trình nghiên cứu

Thời gian	Nội dung công việc	Ghi chú
Tuần 1 (01/04 - 07/04)	Chọn đề tài, xây dựng đề cương. Báo cáo GVHD	
Tuần 2 (08/04 - 14/04)	Tìm hiểu, kiểm thử thuật toán và so sánh với các thuật toán khác. Báo cáo GVHD	
Tuần 3 (15/04 - 21/04)	Tìm dữ liệu và thiết kế cơ sở dữ liệu. Báo cáo GVHD	
Tuần 4 (22/04 - 28/05)	Xây dựng module tối ưu hóa chi phí GA. Báo cáo GVHD	
Tuần 5 (29/04 - 05/05)	Chạy và kiểm tra module tối ưu hóa chi phí với dữ liệu trong cơ sở dữ liệu. Chỉnh sửa cơ sở dữ liệu cho phù hợp. Báo cáo GVHD	
Tuần 6 (06/05 - 12/05)	Xây dựng module tối ưu hóa lộ trình. Báo cáo GVHD	
Tuần 7 (13/05 - 19/05)	Chạy và kiểm tra module tối ưu hóa chi phí với dữ liệu trong cơ sở dữ liệu. Chỉnh sửa cơ sở dữ liệu cho phù hợp. Báo cáo GVHD	
Tuần 8 (20/05 - 26/05)	Chạy và chỉnh sửa các lỗi phát sinh khi kết hợp hai module với nhau. Báo cáo GVHD	
Tuần 9 (27/05 - 02/06)	Tiến hành chạy và kiểm tra lỗi với dữ liệu trong cơ sở dữ liệu. Thiết kế giao diện người dùng. Báo cáo GVHD	
Tuần 10 (03/06 - 09/06)	Chạy, kiểm tra và fix lỗi chương trình. Chỉnh sửa và hoàn thiện báo cáo. Báo cáo GVHD	
Tuần 11 (10/06 - 16/06)	

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Khái niệm về hệ thống tư vấn du lịch

Du lịch là một trải nghiệm tuyệt vời, nhưng việc lên kế hoạch cho chuyến đi có thể tốn nhiều thời gian và công sức. Nên ở đâu ?, Nên ăn gì và ở đâu ?, Nên chơi gì ở đâu ?, Nên đi đâu uống cafe ?, đó là những câu hỏi cần được trả lời mà mọi chuyến đi nào cũng phải biết. Vì vậy, hệ thống gợi ý địa điểm du lịch sẽ là một trợ thủ đắc lực, giúp bạn tìm được câu trả lời và biến việc du lịch trở nên dễ dàng hơn bao giờ hết.

Hệ thống tư vấn du lịch là một hệ thống hoạt động dựa trên dữ liệu và thuật toán, được thiết kế để giúp người dùng lên kế hoạch, lựa chọn các địa điểm du lịch phù hợp với nhu cầu và ngân sách của bản thân. Hệ thống này hoạt động bằng cách phân tích dữ liệu đã được thu thập từ nhiều nguồn khác nhau, bao gồm thông tin về các địa điểm du lịch, thông tin về người dùng (nhu cầu, ngân sách,...).

Sau đó hệ thống sẽ sử dụng các thuật toán tiên tiến để xử lý và phân tích dữ liệu này, tạo ra các gợi ý địa điểm du lịch phù hợp. Các gợi ý này bao gồm thông tin về địa điểm như: vị trí, chi phí dự kiến, link giới thiệu,... Cũng như gợi ý lộ trình đi tất cả các địa điểm một cách tối ưu và quãng đường đi là ngắn nhất.

Mục tiêu của hệ thống tư vấn du lịch là giúp người dùng tiết kiệm thời gian và công sức trong việc tìm kiếm, lựa chọn địa điểm du lịch và tạo trải nghiệm du lịch tốt nhất cho họ. Đồng thời, hệ thống cũng giúp thúc đẩy ngành du lịch bằng cách giúp người dùng khám phá và tận hưởng các địa điểm du lịch mới và hấp dẫn.

2.2 Khảo sát các hệ thống tư vấn du lịch

Hiện nay có nhiều hệ thống tư vấn du lịch phổ biến, được nhiều người biết tới như:

- TripAdvisor (www.tripadvisor.com.vn): Đây là một nền tảng cung cấp các thông tin về các địa điểm du lịch, nhà hàng, khách sạn, chuyến bay. Người dùng có thể thực hiện việc đánh giá, đặt biệt là có thể tạo hành trình đi và chia sẻ kinh nghiệm du lịch cho mọi người.
- Google Trips (www.google.com/travel): Cung cấp gợi ý địa điểm khách sạn, nhà nghỉ, các chuyến bay. Nó sử dụng dữ liệu từ Google Maps và các nguồn khác

để đề xuất các điểm tham quan, ăn uống và hoạt động dựa trên vị trí của người dùng.

- Booking (www.booking.com): Là trang web đặt phòng khách sạn trực tuyến, cung cấp các thông tin về khách sạn, các chuyến bay, địa điểm tham quan. Người dùng có thể so sánh giá cả trước khi quyết định.
- Traveloka (www.traveloka.com): Tương tự như Booking cho phép người dùng đặt khách sạn trực tiếp, cung cấp thông tin vé máy bay, các hoạt động vui chơi.

Nhìn chung, có thể thấy rằng tất cả các hệ thống đều gợi ý các địa điểm du lịch theo từng hạng mục, cho phép người dùng thực hiện đánh giá, cung cấp thông tin chi tiết của địa điểm cũng như giá cả, chi phí. Nếu đặt biệt hơn thì cho phép người dùng tự tạo thủ công hành trình của mình bằng cách lấy lịch sử đánh giá địa điểm, như trong TripAdvisor. Nhưng có một vài điều mà các hệ thống trên chưa thực hiện:

1. Gợi ý danh sách tổ hợp các địa điểm có đầy đủ các hạng mục lưu trú, ăn uống, vui chơi/tham quan, cà phê. Cũng như số lượng các địa điểm của từng hạng mục.
2. Nếu như có danh sách thì chưa đảm bảo tổng chi phí đi hết tất cả địa điểm đó trong phạm vi ngân sách của người dùng.
3. Nếu có, thì cũng không đảm bảo khoảng cách giữa các địa điểm trong danh sách không quá xa nhau ảnh hưởng đến thời gian di chuyển của cả chuyến đi.
4. Gợi ý thứ tự di chuyển qua tất cả các địa điểm sao cho quãng đường di chuyển là ngắn nhất.

2.3 Các giải pháp thiết kế hệ thống tư vấn du lịch

Giữ nguyên các tính năng mà các hệ thống tư vấn du lịch hiện nay đang có, và bổ sung các tính năng chưa thực hiện được đã đề cập trong mục 2.1. Sau đây là một số các giải pháp thiết kế:

a) Gợi ý danh sách địa điểm

- Xây dựng danh sách các địa điểm thuộc từng hạng mục (lưu trú, ăn uống, vui chơi/tham quan, cà phê) từ dữ liệu có sẵn.
- Tạo danh sách tổ hợp các địa điểm có đầy đủ các hạng mục và số lượng địa điểm của từng hạng mục.

b) Tối ưu chi phí

- Tính toán và lựa chọn các tổ hợp địa điểm sao cho tổng chi phí không vượt qua ngân sách người dùng đã cung cấp.

c) Khoảng cách giữa các địa điểm

- Sử dụng các công thức tính toán để tính khoảng cách giữa các địa điểm.
- Loại bỏ các tổ hợp có các địa điểm cách xa nhau một ngưỡng nào đó.

d) Thứ tự di chuyển tối ưu

- Áp dụng các thuật toán tiên tiến để tìm đường đi ngắn nhất qua tất cả các địa điểm trong tổ hợp, hay nói cách khác là giải quyết bài toán TSP.

2.4 Các thuật toán liên quan

2.4.1 Thuật toán quay lui (Backtracking)

Thuật toán quay lui là giải thuật được thiết kế dựa trên đệ quy. Ý tưởng của quay lui là tìm lời giải từng bước, mỗi bước chọn một trong số các lựa chọn khả dĩ và đệ quy. Người đầu tiên đề ra thuật ngữ này (backtrack) là nhà toán học người Mỹ D. H. Lehmer vào những năm 1950.

Trong hệ thống tư vấn du lịch, thuật toán quay lui đóng vai trò quan trọng trong việc tạo ra các giải pháp tối ưu chi phí và phù hợp với nhu cầu của người dùng. Thuật toán này được sử dụng trong hệ thống tư vấn du lịch để tạo ra danh sách các tổ hợp địa điểm với số lượng địa điểm phù hợp với từng hạng mục (lưu trú, ăn uống, vui chơi/tham quan, quán cà phê). Mỗi danh sách tổ hợp sẽ bao gồm số lượng địa điểm nhất định, được xác định trước dựa trên loại hình thời gian du lịch (1 ngày hoặc 2 ngày 1 đêm).

Thuật toán quay lui hoạt động theo các bước sau:

- Bước 1: Tạo một danh sách rỗng (empty) và danh sách chứa tổ hợp (result) .
- Bước 2 : Xét từng địa điểm trong danh sách địa điểm của một hạng mục.
- Bước 3: Thêm địa điểm đang xét vào empty.
- Bước 4: Tiếp tục xét các địa điểm khác và thêm chúng vào empty.
- Bước 5: Nếu số lượng phần tử trong empty đã thỏa mãn yêu cầu, thì thêm empty vào result, làm trống empty, quay lại bước 2 và xét phần tử tiếp theo.

- Bước 6: Sau khi xét hết tất cả các địa điểm trong danh sách. Trả về kết quả result, chứa danh sách các tổ hợp địa điểm với số lượng địa điểm mỗi tổ hợp phù hợp với hạng mục đang xét.

Ưu điểm:

- Đơn giản và dễ hiểu: Thuật toán quay lui có cấu trúc rõ ràng và dễ hiểu, dễ dàng triển khai khi giải quyết các vấn đề kết hợp.
- Hiệu quả: Đối với các bài toán có số lượng giải pháp hữu hạn và cấu trúc rõ ràng, thuật toán quay lui có thể tìm ra tất cả các giải pháp một cách hiệu quả.
- Dễ dàng triển khai: Thuật toán quay lui có thể được triển khai tương đối dễ dàng bằng nhiều ngôn ngữ lập trình khác nhau.
- Linh hoạt: Có thể dễ dàng thay đổi để giải quyết các vấn đề khác nhau bằng cách thay đổi hàm đệ quy và điều kiện dừng.

Nhược điểm:

- Độ phức tạp thời gian: Đối với các bài toán có số lượng giải pháp lớn hoặc cấu trúc phức tạp, thuật toán quay lui có thể tốn nhiều thời gian và tài nguyên tính toán.
- Nguy cơ thiếu bộ nhớ: Việc sử dụng đệ quy trong thuật toán quay lui có thể dẫn đến nguy cơ thiếu bộ nhớ, đặc biệt là khi xử lý các bài toán có nhiều cấp độ đệ quy.
- Không hiệu quả trong các trường hợp tồi tệ: Khi không có giới hạn rõ ràng về số lượng lời giải hoặc khi không thể cắt tỉa các nhánh không cần thiết, thuật toán quay lui có thể trở nên không hiệu quả.
- Khó tối ưu hóa: Việc tối ưu hóa thuật toán quay lui cho các bài toán cụ thể có thể phức tạp và đòi hỏi nhiều kỹ thuật nâng cao.

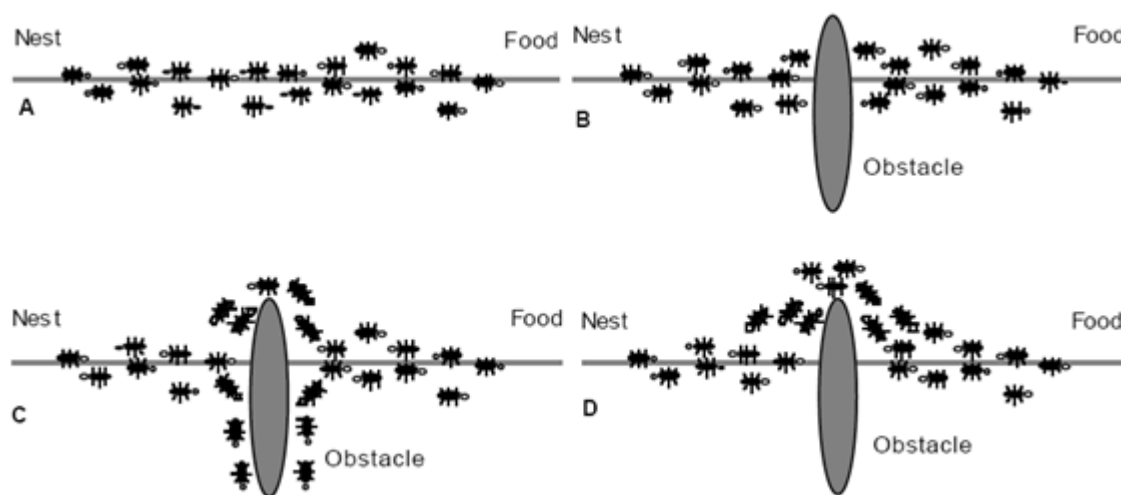
2.4.2 Thuật toán đàn kiến (Ant Algorithm)

a) Nguồn gốc của thuật toán

Năm 1989, nhà bác học người Đan Mạch Deneubourg và các cộng sự công bố kết quả nghiên cứu về thí nghiệm trên đàn kiến Argentina (một loài kiến hiếm trên thế giới), gọi là thí nghiệm “Chiếc cầu đôi” (Double Bridge Experiment). Cụ thể, họ đã đặt một vật cản gồm hai nhánh (nhánh dài hơn bằng hai lần nhánh ngắn hơn) nối với tổ của một đàn

kiến với nguồn thức ăn, sau đó họ thả một đàn kiến và bắt đầu quan sát chuyển động của chúng trong một khoảng thời gian đủ lớn. Kết quả là ban đầu các con kiến đi theo hai nhánh của chiếc cầu với số lượng gần như ngang nhau, nhưng càng về khoảng thời gian cuối người ta quan sát thấy rằng các con kiến có xu hướng chọn nhánh ngắn hơn (80-100% số lượng).

Các nhà sinh học đã giải thích kết quả đó như sau: Do đặc tính tự nhiên và đặc tính hóa học, mỗi con kiến khi di chuyển luôn để lại một lượng hóa chất trên đường đi của nó gọi là các vết mùi (pheromone trail) và thường chúng sẽ đi con đường có mùi đậm đặc hơn. Các vết mùi này là những hóa chất bay hơi theo thời gian, do vậy lượng mùi ban đầu ở hai nhánh là xấp xỉ như nhau, nhưng sau một khoảng thời gian nhất định nhánh ngắn hơn sẽ có lượng mùi đậm đặc hơn so với nhánh dài do cũng một lượng mùi gần xấp xỉ như nhau khi phân bố ở nhánh dài mật độ phân bố mùi ở nhánh này không dày bằng nhánh có độ dài ngắn hơn, thêm nữa cũng do lượng mùi trên nhánh dài hơn cũng sẽ bị bay hơi nhanh hơn trong một khoảng thời gian.



Hình 2. 1: Thí nghiệm “Chiếc cầu đôi” (Double Bridge Experiment)

(Nguồn: <https://www.researchgate.net/>)

Chú thích:

A: Đàn kiến đi từ tổ đến chỗ thức ăn.

B: Đặt một vật cản ở giữa đường đi, vật cản gồm hai nhánh (nhánh dài có độ dài bằng hai lần nhánh ngắn hơn).

C: Ban đầu số lượng các con kiến đi qua cả hai đường gần như bằng nhau, nên lượng mùi tỏa ra là như nhau.

D. Do đường đi ngắn hơn, nên nồng độ mùi của các con kiến ở con đường phía trên cao hơn, nên những con kiến có xu hướng chọn đường đi phía trên.

Năm 1991, với cơ sở là kết quả thí nghiệm nổi tiếng trên, nhà khoa học người Bỉ Marco Dorigo đã xây dựng thuật toán đàn kiến (Ant Algorithm, hay còn gọi là Hệ kiến, Ant System) đầu tiên ứng dụng vào bài toán người du lịch (TSP), và công bố trong luận án tiến sĩ của ông.

b) Áp dụng vào hệ thống tư vấn du lịch

Trong hệ thống tư vấn du lịch, thuật toán đàn kiến được sử dụng để tìm ra đường đi ngắn nhất qua tất cả các địa điểm trong một tổ hợp và tổ hợp này nằm trong danh sách các tổ hợp thỏa mãn các điều kiện sau:

- Có đầy đủ các hạng mục, số lượng địa điểm của từng hạng mục và tổng chi phí của mỗi tổ hợp không vượt quá ngân sách cho phép.
- Khoảng cách giữa các địa điểm trong một tổ hợp không lớn hơn một ngưỡng cho phép nào đó.

Để bắt chước hành vi của các con kiến thực, ta cần tạo ra con kiến nhân tạo có các đặc trưng sản sinh ra vết mùi để lại trên đường đi và khả năng lần vết theo nồng độ mùi để lựa chọn con đường có nồng độ cao hơn để đi. Từ tổ hợp các địa điểm, ta sẽ tính toán khoảng cách giữa các địa điểm để tạo ra ma trận khoảng cách với mỗi đỉnh là số thứ tự của địa điểm đó trong tổ hợp đang xét. Gọi trọng số $c_{i,j}$ là khoảng cách giữa địa điểm i và địa điểm j trong tổ hợp và gọi $\tau_{i,j}$ nồng độ mùi trên quãng đường di chuyển giữa địa điểm i và địa điểm j . Ban đầu, nồng độ mùi trên mỗi cạnh được khởi tạo bằng một hằng số c nào đó.

i. Phương pháp tìm đường đi theo hành vi con kiến

Các con kiến sẽ tiến hành tìm đường đi từ một địa điểm xuất phát (trong hệ thống mặc định là địa điểm đầu tiên trong tổ hợp) qua một loạt các địa điểm khác trong tổ hợp và quay trở về địa điểm ban đầu, tại địa điểm u một con kiến sẽ chọn địa điểm v chưa đi qua trong tập các địa điểm láng giềng của u theo xác suất p sau:

$$p_{u,v} = \begin{cases} \frac{(\tau_{u,v})^\alpha (\eta_{u,v})^\beta}{\sum_{w \in UV(u)} [(\tau_{u,w})^\alpha (\eta_{u,w})^\beta]} & , v \in UV(u) \\ 0 & , v \notin UV(u) \end{cases}$$

Trong đó:

$UV(u)$: Là tập các địa điểm láng giềng của u chưa được con kiến hiện tại đi qua.

$\eta_{u,v} = \frac{1}{c_{u,v}}$: Gọi là thông tin heuristic giúp đánh giá chính xác hơn sự lựa chọn của con kiến khi đi từ địa điểm u qua địa điểm v .

Ta có thể hiểu đơn giản như sau: Việc quyết định lựa chọn địa điểm tiếp theo để đi của con kiến sẽ được chọn ngẫu nhiên theo xác suất (tức địa điểm nào có xác suất cao hơn sẽ có khả năng chọn cao hơn, nhưng không có nghĩa các địa điểm có xác suất thấp sẽ không được chọn mà nó được chọn với cơ hội thấp hơn. Và xác suất này tỷ lệ thuận vết mùi trên đường đi được chọn và tỉ lệ nghịch với khoảng cách giữa các địa điểm; α, β là những hệ số điều khiển việc lựa chọn của con kiến nghiêng về phía nào (nếu $\alpha < \beta$ thì chọn đường đi ngắn hơn).

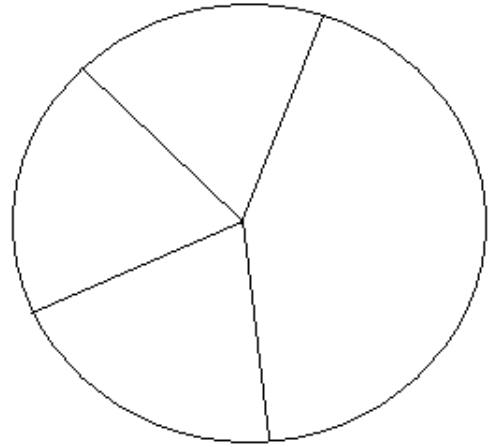
ii. Kỹ thuật bánh xe xổ số - (Lottery Wheel)

Đây là kỹ thuật được sử dụng cho ý tưởng chọn địa điểm tiếp theo dựa trên xác suất đã nêu ở trên. Cụ thể kỹ thuật như sau:

Giả sử $UV(u) = \{v_1, v_2, \dots, v_m\}$ là tập các địa điểm láng giềng của u và p_1, p_2, \dots, p_m là xác suất lựa chọn địa điểm tiếp theo từ u tương ứng đến v_1, v_2, \dots, v_m .

Ta có: $\sum_{i=1}^m p_i = 1$ tức là chắc chắn chọn một trong các địa điểm trên để đi tiếp. Để đảm bảo ưu thế cho những đỉnh có xác suất lớn, nhưng vẫn đảm bảo của các địa điểm có xác suất thấp hơn người ta sinh ra một số ngẫu nhiên $k \in (0; \sum]$ rồi chọn j nhỏ nhất sao cho $\sum_{i=1}^j p_i \geq k$.

Như vậy, các con kiến sẽ xuất phát từ một đỉnh, lần lượt tới thăm các đỉnh tiếp theo như quy tắc trên (đánh dấu các địa điểm đã tới) cho đến khi đến địa điểm cuối cùng rồi quay về địa điểm ban đầu. Quá trình này được lặp đi lặp lại, quãng đường đi ngắn hơn sẽ được cập nhật cho đến một khoảng thời gian đủ tốt (thông thường với số địa điểm ≤ 200 chỉ cần lặp 500 lần là đủ để tìm kết quả tối ưu, tùy vào bộ dữ liệu mà có số lần lặp khác nhau).



Hình 2. 2: Lottery Wheel

Trong quá trình các con kiến tìm đường đi các vết mùi ($\tau_{i,j}$) sẽ được cập nhật lại, vì chúng bị biến đổi do quá trình bay hơi và do quá trình tích lũy của các con kiến khác trên quãng đường đó. Vết mùi sẽ được cập nhật theo công thức sau:

$$\tau_{i,j} \leftarrow p\tau_{i,j} + \Delta_{i,j}$$

Trong đó $p \in (0, 1)$ gọi là tham số bay hơi (sau mỗi lần cập nhật lượng mùi trên cạnh (i, j) - quãng đường từ địa điểm i đến địa điểm j - sẽ mất đi một lượng là $(1 - p) \tau_{i,j}$), thường được chọn là 0.8 trong cài đặt và chạy chương trình. Ngoài lượng bay hơi, mỗi cạnh (i, j) còn được tích tụ thêm một lượng $\Delta_{i,j}$ nhất định tùy thuộc vào từng con kiến đi qua, cụ thể được tính như sau:

$$\Delta_{i,j} \leftarrow \Delta_{i,j} + \frac{Q}{L_k}$$

Trong đó: Q là hằng số.

L_k là độ dài hành trình của con kiến thứ k .

Nhờ việc cập nhật mùi này, mà có ảnh hưởng đến quyết định của các con kiến, có thể ở bước lặp này chọn một cạnh để đi nhưng đến bước lặp khác vẫn con kiến đó lại không đi quãng đường đó nữa. Cũng nhờ vậy thuật toán có khả năng tìm được lời giải tốt trong những trường hợp dữ liệu cực lớn.

Sau n lần thực hiện lặp đi lặp các công việc lựa chọn địa điểm để đi và cập nhật nồng độ mùi trên quãng đường đi của mỗi con kiến, thuật toán trả về một danh sách các đỉnh trong ma trận khoảng cách có thứ tự. Dựa vào danh sách có thứ tự này hệ thống tiến hành sắp xếp lại các địa điểm trong tổ hợp đang xét, thứ tự sắp xếp đó sẽ là lộ trình di chuyển sao cho quãng đường đi qua tất cả các địa điểm trong tổ hợp là nhỏ nhất.

Nhận xét: Thuật toán có hiệu quả làm việc rất cao, do những vòng lặp của thuật toán đều đã có liên hệ mật thiết với nhau (thông qua nồng độ mùi để lại sau mỗi lần lặp). Tuy nhiên, nó không thể tránh khỏi những khuyết điểm của một giải thuật heuristic, đó là không thể đưa ra lời giải tốt nhất, mà chỉ đưa ra lời giải chấp nhận được. Ngoài ra, tuy ý tưởng thuật toán dễ hiểu, nhưng việc cài đặt lại khá phức tạp.

CHƯƠNG 3: THIẾT KẾ HỆ THỐNG TƯ VẤN DU LỊCH

3.1 Phân tích hệ thống

3.1.1 Yêu cầu hệ thống

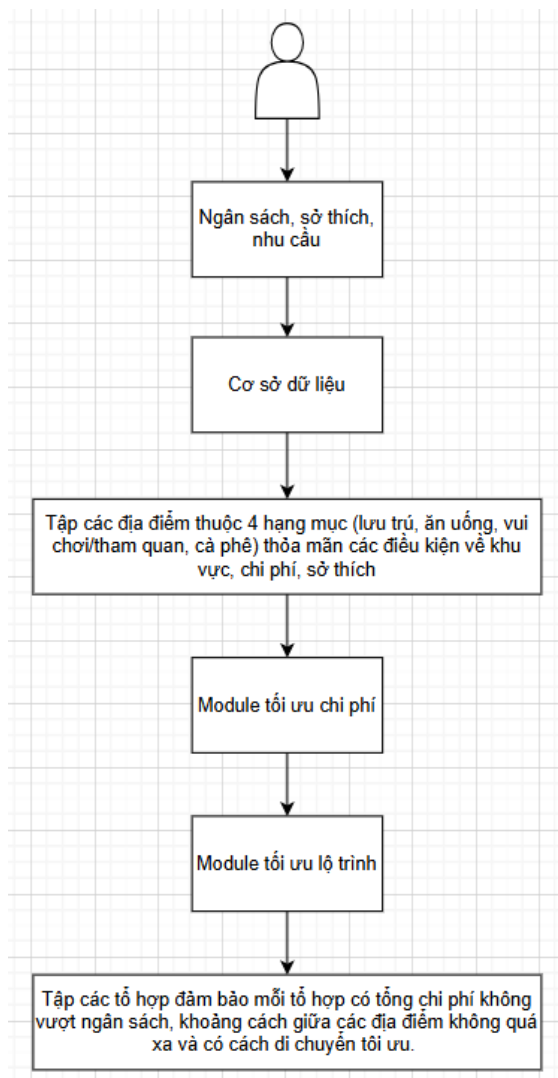
Yêu cầu chức năng:

- Gợi ý địa điểm: Hệ thống cần có khả năng gợi ý các địa điểm du lịch phù hợp với ngân sách và sở thích của người dùng.
- Lọc địa điểm: Hệ thống cần có khả năng lọc các địa điểm dựa trên các tiêu chí như ngân sách, loại hình du lịch,....
- Hiện thị thông tin địa điểm: Hệ thống cần có khả năng hiển thị thông tin chi tiết về mỗi địa điểm được gợi ý, bao gồm địa chỉ, mô tả, hình ảnh, chi phí dự kiến, và link liên kết.
- Tính toán tổng chi phí: Hệ thống cần có khả năng tính toán tổng chi phí cho mỗi lựa chọn địa điểm dựa trên thông tin về chi phí từ cơ sở dữ liệu.

Yêu cầu phi chức năng:

- Hiệu suất: Hệ thống cần phải đáp ứng nhanh chóng đối với các yêu cầu từ người dùng, đặc biệt là khi lọc và gợi ý các địa điểm.
- Tính sẵn sàng: Hệ thống cần phải luôn sẵn sàng để phục vụ người dùng, với thời gian downtime tối thiểu.
- Mở rộng: Hệ thống cần được thiết kế để dễ dàng mở rộng, cho phép thêm các chức năng mới hoặc xử lý nhiều người dùng hơn mà không ảnh hưởng đến hiệu suất.
- Dễ sử dụng: Giao diện người dùng của hệ thống cần phải thân thiện và dễ sử dụng, giúp người dùng dễ dàng nhập thông tin và hiểu các gợi ý của hệ thống.

3.1.2 Kiến trúc của hệ thống



Hình 3. 1: Kiến trúc hệ thống

Dữ liệu đầu vào:

- Ngân sách dùng cho chuyến đi.
- Nhu cầu và sở thích của người dùng.

Cơ sở dữ liệu:

- Truy xuất các địa điểm thuộc các hạng mục: lưu trú, ăn uống, vui chơi/tham quan, cà phê.
- Các địa điểm trong danh sách truy xuất đã thỏa mãn các điều kiện sau:
 - Chi phí dừng chân tại địa điểm không vượt quá ngân sách.

- Địa điểm nằm trong khu vực mà người dùng muốn đến: Hồ Chí Minh, Hà Nội.
- Phù hợp với sở thích và nhu cầu của người dùng (có thể có hoặc không).

Module tối ưu chi phí:

- Thực hiện các công việc xử lý liên quan đến chi phí và tạo tổ hợp.
- Dựa vào loại hình thức thời gian (trong ngày và hai ngày một đêm), hệ thống xác định số lượng địa điểm cho từng hạng mục.
- Tạo tổ hợp, mỗi tổ hợp phải có đầy đủ hạng mục cũng như số lượng địa điểm của mỗi hạng mục.
- Lọc và loại bỏ các tổ hợp có tổng chi phí vượt quá ngân sách.

➔ Kết quả: Danh sách tổ hợp đủ các hạng mục, số lượng địa điểm mỗi hạng mục và tổng chi phí mỗi tổ hợp không vượt quá ngân sách.

Module tối ưu lộ trình:

- Thực hiện các công việc liên quan đến lộ trình, đường đi.
- Các địa điểm trong mỗi tổ hợp từ kết quả trả về của module tối ưu chi phí có thể nằm xa nhau. Vì vậy, module này sẽ tiến hành tính toán và loại bỏ các tổ hợp có chứa những địa điểm khoảng cách với nhau vượt quá một ngưỡng maxDistance cho phép nào đó.

➔ Có được danh sách các tổ hợp có khoảng giữa các địa điểm trong tổ hợp không xa nhau.

➔ Từ mỗi tổ hợp, module sẽ xử lý và đưa ra thứ tự di chuyển qua tất cả các địa điểm sao cho quãng đường đi là ngắn nhất.

3.2 Thiết kế cơ sở dữ liệu

3.2.1 Thu thập dữ liệu

Mục đích thu thập dữ liệu: Hệ thống gợi ý này cần dữ liệu chi tiết về các địa điểm du lịch để cung cấp các gợi ý phù hợp với nhu cầu của người dùng. Các địa điểm này thuộc các hạng mục: lưu trú, quán ăn, vui chơi/tham quan, quán cà phê, đồng thời các địa điểm này sẽ nằm ở thành phố Hồ Chí Minh và thủ đô Hà Nội.

Các dữ liệu sẽ được thu thập từ nhiều nguồn khác nhau, bao gồm các trang web du lịch phổ biến (traveloka, booking,...), các trang mạng xã hội với các bài đăng, các video review địa điểm thu thập (Tiktok, Instagram,...).

Loại dữ liệu được thu thập: tên địa điểm, địa chỉ, hình ảnh, mô tả địa điểm, link, vĩ độ, kinh độ và tổng chi phí dự kiến cho địa điểm (chi phí này ước tính dành cho một người). Ngoài ra, mỗi hạng mục sẽ có dữ liệu riêng để có thể giúp hệ thống có hệ gợi ý địa điểm phù hợp với người dùng hơn:

- Địa điểm lưu trú: Loại hình (khách sạn, nhà nghỉ, homestay,...).
- Địa điểm quán ăn: Loại ẩm thực (bình dân, Hàn Quốc, Buffet,...).
- Địa điểm vui chơi/tham quan: Loại hoạt động (leo núi, tham quan, đi bộ, bơi lội, bắn súng,...).
- Địa điểm cà phê: Phong cách (thiên nhiên, vintage, cổ điển,...).

Cách tính toán tổng chi phí dự kiến của địa điểm cho từng hạng mục:

- Đối với lưu trú:
 - Lấy trung bình cộng tất cả các giá phòng (giá gốc, không phải giá khuyến mãi) không giống nhau của địa điểm đó và cộng thêm 100.000 VND là phí phát sinh.
 - Ví dụ: Khách sạn đó có 3 phòng với các mệnh giá: 350.000 VND, 200.000 VND, 350.000 VND.
Thì tính tổng chi phí như sau: $(350.000 + 200.000)/2 + 100.000$ (phí phát sinh) = 375.000 VND chi phí/người.
- Đối với quán ăn:
 - Dựa vào giá các món ăn và loại món ăn gì mà dự kiến giá sao cho phù hợp sức ăn của một người trung bình. Tùy vào món ăn mà có thể sẽ cộng thêm chênh lệch trong khoảng 10.000 VND → 30.000 VND.
 - Ví dụ: Đối với các món như: bún, hủ tiếu, cơm tấm, bún đậu mắm tôm,... thì lấy giá món cao nhất được bán trong quán (chẳng hạn quán bún bò có bán món giá 30.000 VND và bún bò đặc biệt giá 40.000 VND thì tổng chi phí là 40.000 VND).

- Ví dụ: Đối với các món ăn vặt: chân gà, cá viên chiên, chè,... thì lấy giá các món ăn có số lượng giống nhau nhiều nhất (chẳng hạn, quán chè: có các giá 15.000 VND, 10.000 VND, 25.000 VND trong đó có nhiều món có giá 15.000 VND thì tổng chi phí tính như sau: 15.000×2 (vì 1 người có thể ăn 2 phần chè) + 10.000 (chênh lệch) = 40.000 VND/người).
- Ví dụ: Đối với các quán ăn có nhiều món như quán cơm gia đình, nhà hàng,... thì lấy giá xuất hiện nhiều nhất trong thực đơn quán và quy định rằng lượng ăn trung bình của một người có thể gọi 3 món (chẳng hạn, quán cơm gia đình có giá các món là 30.000 VND, 40.000 VND và 50.000 VND trong đó phần lớn món đều có giá 30.000 VND thì tổng chi phí = $30.000 \times 3 = 90.000$ VND/người).
- Đối với vui chơi/tham quan:
 - Lấy giá vé cao nhất của địa điểm đó (nếu có và lấy giá gốc, không khuyến mãi), và không tính tiền phát sinh khác tiền ăn, uống, mua đồ lưu niệm khi đến địa điểm đó.
 - Ví dụ: Địa điểm có các giá vé: người lớn: 120.000 VND, trẻ em: 50.000 VND. Thì tổng chi phí là 120.000 VND.
- Đối với cà phê:
 - Mặc định là 1 người sẽ gọi một đồ uống, cho nên sẽ lấy giá đồ uống cao nhất của quán và có thể cộng thêm giá các loại bánh ngọt tùy quán trong quá trình thu thập. Sẽ không tính phí có thể phát sinh: phí gửi xe, phí chụp hình,... Nếu trong trường hợp trong quá trình thu thập không thể biết được quán đó bán với giá bao nhiêu thì sẽ dự kiến tổng chi phí là 80.000 VND (vì nhìn tổng thể các quán cà phê có bán các loại nước uống dưới hoặc bằng 80.000 VND)
 - Ví dụ: Quán bán: cà phê đen: 25.000 VND, cacao 30.000 VND, milo đậm 25.000 VND thì tổng chi phí là 30.000 VND.

→ **Lưu ý:** Dữ liệu trên được thu thập và tính toán một cách thủ công không thể tránh khỏi sai sót. Trên hết, các số liệu về giá được dùng để tính toán như trên được thu thập từ

03/04/2024 → 22/04/2024 nên xét về mặt hiện tại thì giá có thể thay đổi nhiều so với lúc tính toán.

Dữ liệu sau khi thu thập sẽ được lưu trữ vào cơ sở dữ liệu SQL, cơ sở này có cấu trúc thiết kế sẽ được nêu trong mục 3.3.

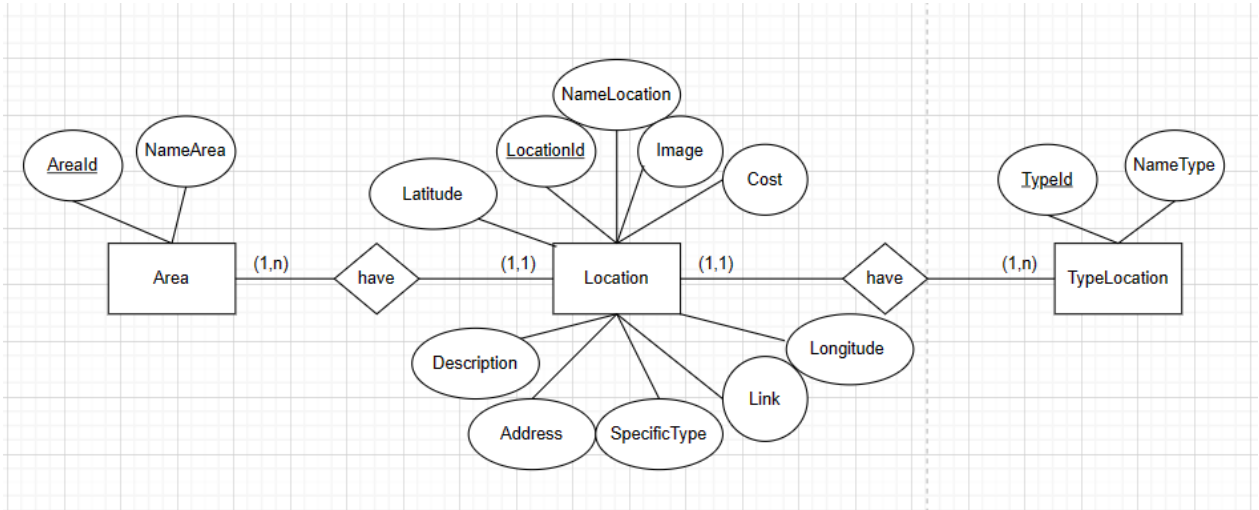
3.2.2 Mô hình ER

Theo cách thu thập dữ liệu như trong mục 3.2, ta có thể xác định được số lượng thực thể và thuộc tính của từng thực thể như sau:

- Thực thể *Khu vực (Area)* có các thuộc tính sau: Id (thuộc tính khóa chính), tên khu vực (Hồ Chí Minh, Hà Nội).
- Thực thể *Lưu trú*: Id (thuộc tính khóa chính), tên địa điểm, địa chỉ, hình ảnh, mô tả địa điểm, link, vĩ độ, kinh độ và tổng chi phí dự kiến cho địa điểm và loại hình.
- Thực thể *Quán ăn*: Id (thuộc tính khóa chính), tên địa điểm, địa chỉ, hình ảnh, mô tả địa điểm, link, vĩ độ, kinh độ và tổng chi phí dự kiến cho địa điểm và loại ẩm thực.
- Thực thể *Giải trí*: Id (thuộc tính khóa chính), tên địa điểm, địa chỉ, hình ảnh, mô tả địa điểm, link, vĩ độ, kinh độ và tổng chi phí dự kiến cho địa điểm và loại hình hoạt động.
- Thực thể *Cà phê*: Id (thuộc tính khóa chính), tên địa điểm, địa chỉ, hình ảnh, mô tả địa điểm, link, vĩ độ, kinh độ và tổng chi phí dự kiến cho địa điểm và phong cách.

Có thể thấy rằng, các thực thể lưu trú, quán ăn, giải trí, và cà phê có các thuộc tính phần lớn là giống nhau. Vì vậy, ta có thể gộp các thực thể này thành một thực thể duy nhất là *Địa điểm (Location)* có đầy đủ các thuộc tính đó và một thêm một thuộc tính *Loại cụ thể (SpecificType)* - thay thế cho thuộc tính: loại hình, loại ẩm thực, loại hình hoạt động và phong cách). Để có thể phân biệt được thực thể *Địa điểm (Location)* thuộc hạng mục lưu trú, quán ăn, vui chơi/tham quan hay cà phê ta sẽ bổ sung thêm thực thể *Loại địa điểm (TypeLocation)* có thuộc tính: id (thuộc tính khóa chính) và tên loại địa điểm).

Sau đây là một hình ER hoàn chỉnh:



Hình 3. 2: Mô hình ER

3.2.3 Mô hình quan hệ

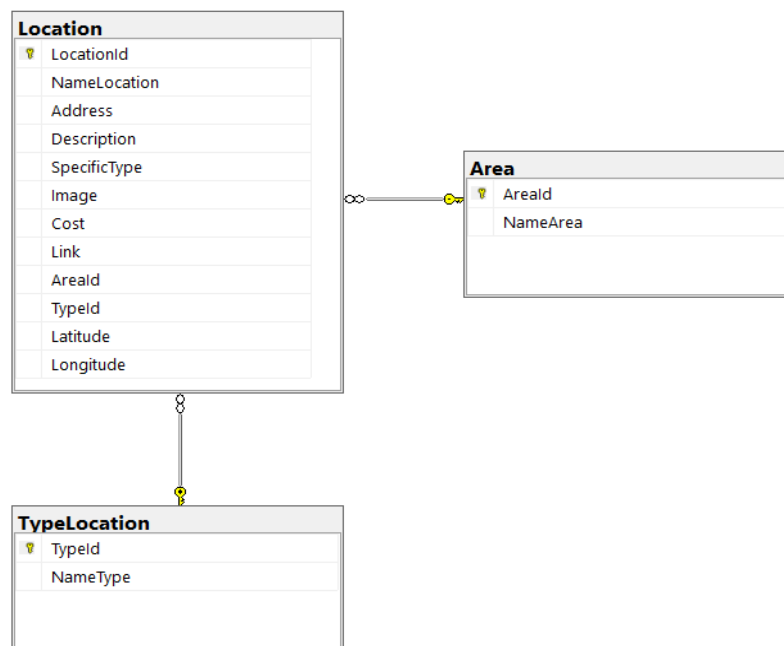
Từ mô hình ER, ta có mô hình quan hệ sau:

Area (**AreaId**, NameArea).

TypeLocation (**TypeId**, NameType).

Location (**LocationId**, NameLocation, Address, Description, SpecificType, Image, Cost, Link, AreaId, TypeId, Latitude, Longitude).

Cuối cùng mà mô hình diagram được xây dựng từ mô hình quan hệ trên:



Hình 3. 3: Mô hình diagram

Giải thích các bảng:

Bảng 3. 1: Area

Tên thuộc tính	Kiểu dữ liệu	Ý nghĩa
AreaId	int	Xác định khu vực duy nhất
NameArea	nvarchar(100)	Tên khu vực (Hồ Chí Minh, Hà Nội)

➤ Khóa chính: AreaId.

Bảng 3. 2: TypeLocation

Tên thuộc tính	Kiểu dữ liệu	Ý nghĩa
TypeId	int	Xác định hạng mục duy nhất
NameType	nvarchar(100)	Tên hạng mục (lưu trú, quán ăn,...)

➤ Khóa chính: TypeId.

Bảng 3. 3: Location

Tên thuộc tính	Kiểu dữ liệu	Ý nghĩa
LocationId	int	Xác định địa điểm duy nhất
NameLocation	nvarchar(250)	Tên địa điểm
Address	nvarchar(250)	Địa chỉ đến địa điểm
Description	nvarchar(MAX)	Mô tả địa điểm
SpecificType	nvarchar(200)	Loại cụ thể của địa điểm
Image	varbinary(MAX)	Ảnh của địa điểm
Cost	decimal	Chi phí dự kiến đến địa điểm

Link	nvarchar(MAX)	Đường dẫn giới thiệu địa điểm
AreaId	int	Xác định khu vực địa điểm thuộc
TypeId	int	Xác định hạng mục của địa điểm
Latitude	float	Vĩ độ của địa điểm
Longitude	float	Kinh độ của địa điểm

- Khóa chính: LocationId.
- Khóa ngoại: AreaId (khóa chính bảng Area), TypeId (khóa chính bảng TypeLocation).

3.3 Thiết kế các chức năng của hệ thống

Các chức năng chính của hệ thống:

- Tạo danh sách tổ hợp các địa điểm gợi ý cho người dùng, có đầy đủ hạng mục, số lượng địa điểm và chi phí trong phạm vi ngân sách.
- Đưa ra cách di chuyển tối ưu, sao cho quãng đường di chuyển là ngắn nhất.

Để hệ thống có thể thực hiện các chức năng trên cần giải quyết các vấn đề sau:

- Vấn đề thứ nhất: Tạo ra tổ hợp các địa điểm của từng hạng mục, với số lượng địa điểm mỗi tổ hợp tùy theo hình thức thời gian.
- Vấn đề thứ hai: Từ các tổ hợp từng hạng mục, tạo ra các tổ hợp lớn hơn chứa các tổ hợp con đảm bảo đầy đủ hạng mục và tổng chi phí mỗi tổ hợp không vượt quá ngân sách.
- Tối ưu hóa lộ trình di chuyển: Làm thế nào để có quãng đường di chuyển là ngắn nhất ?.

Để có thể dễ dàng làm việc ta sẽ khởi tạo một *class ItemLoc* như sau:

```
public class ItemLoc
{
    public int Id { get; set; }
```

```

        public decimal Cost { get; set; }
        public double Latitude { get; set; }
        public double Longitude { get; set; }
    }

```

Trong đó: Id: Id của địa điểm.

Cost: Chi phí dự kiến của địa điểm.

Latitude: Vĩ độ của địa điểm.

Longitude: Kinh độ của địa điểm.

➤ Mỗi *ItemLoc* sẽ đại diện cho một địa điểm.

3.3.1 Thuật giải giải quyết vấn đề thứ nhất

Đầu tiên ta cần phải xác định số lượng địa điểm theo hình thức thời gian du lịch:

- Trong ngày (một ngày): 1 địa điểm lưu trú (hoặc 0), 4 địa điểm ăn uống, 3 vui chơi/tham quan, 2 địa điểm cà phê.
- Hai ngày một đêm: 1 địa điểm lưu trú, 5 địa điểm ăn uống, 4 địa điểm vui chơi/tham quan, 3 địa điểm cà phê.

Thuật giải:

```

public List<List<ItemLoc>> GetCombinations(List<ItemLoc> list, int length)
{
    var result = new List<List<Item>>>();
    if (length == 1)
    {
        foreach (var item in list)
        {
            result.Add(new List<ItemLoc> { item });
        }
        return result;
    }
    else
    {
        for (int i = 0; i < list.Count; i++)

```



```

    {
        var smallerCombinations = GetCombinations(list.Skip(i + 1).ToList(),
length - 1);
        foreach (var values in smallerCombinations)
        {
            var combination = new List<ItemLoc> { list[i] };
            combination.AddRange(values);
            result.Add(combination);
        }
    }
    return result;
}
}

```

Đầu vào của thuật giải là một danh sách các địa điểm của một hạng mục và chiều dài hay số lượng địa điểm trong mỗi tổ hợp được tạo ra. Kết quả cuối cùng là một danh sách các danh sách *ItemLoc* mỗi danh sách con sẽ là một tổ hợp với độ dài là *length* (số lượng phần tử trong mỗi tổ hợp). Để tránh việc tạo ra các tổ hợp giống nhau như *[1, 2]*, *[2, 1]*, thì thuật giải *GetCombinations* đã sử dụng câu lệnh *list.Skip(i + 1).ToList()* để tránh các phần tử đã được xử lý và xem xét các phần tử chưa được xử lý.

Ví dụ: *var list = new List<int> { 1, 2, 3 };*

var combinations = GetCombinations(list, 2); → *[1, 2]*, *[1, 3]* và *[2, 3]*

3.3.2 Thuật giải giải quyết vấn đề thứ hai

Sau khi đã có danh sách các các tổ hợp của từng hạng mục, bước thực hiện tiếp theo là kết hợp các tổ hợp này tạo thành tổ hợp lớn hoàn chỉnh đảm bảo đầy đủ hạng mục và tổng chi phí không vượt ngân sách.

Sau đây là thuật giải thực hiện:

```

public static List<List<ItemLoc>> FindCombinations(List<ItemLoc> hotels,
List<ItemLoc> foods, List<ItemLoc> entertains, List<ItemLoc> coffees, int
numHotel, int numFood, int numEntertain, int numCoffee, decimal maxCost)
{

```

```

List<List<ItemLoc>> combinations = new List<List<ItemLoc>>();
if (numHotel != 0)
{
    hotels = hotels.OrderBy(h => h.Cost).ToList();
    foods = foods.OrderBy(h => h.Cost).ToList();
    entertains = entertains.OrderBy(h => h.Cost).ToList();
    coffees = coffees.OrderBy(h => h.Cost).ToList();
    List<List<ItemLoc>> hotelsCombination = GetCombinations(hotels,
numHotel).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    List<List<ItemLoc>> foodsCombination = GetCombinations(foods,
numFood).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    List<List<ItemLoc>> entertainCombination = GetCombinations(entertains,
numEntertain).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    List<List<ItemLoc>> coffeesCombination = GetCombinations(coffees,
numCoffee).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    foreach (var hotel in hotelsCombination)
    {
        foreach (var food in foodsCombination)
        {
            foreach (var entertain in entertainCombination)
            {
                foreach (var coffee in coffeesCombination)
                {
                    var combination = new List<ItemLoc>();
                    combination.AddRange(hotel);
                    combination.AddRange(food);
                    combination.AddRange(entertain);
                    combination.AddRange(coffee);
                    if (combination.Sum(item => item.Cost) <= maxCost)
                    {

```

```

        combinations.Add(combination);
    }
}
}
}
}
}
else
{
    foods = foods.OrderBy(h => h.Cost).ToList();
    entertains = entertains.OrderBy(h => h.Cost).ToList();
    coffees = coffees.OrderBy(h => h.Cost).ToList();
    List<List<ItemLoc>> foodsCombination = GetCombinations(foods,
numFood).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    List<List<ItemLoc>> entertainCombination = GetCombinations(entertains,
numEntertain).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    List<List<ItemLoc>> coffeesCombination = GetCombinations(coffees,
numCoffee).OrderBy(x => Guid.NewGuid()).Take(20).ToList();
    foreach (var food in foodsCombination)
    {
        foreach (var entertain in entertainCombination)
        {
            foreach (var coffee in coffeesCombination)
            {
                var combination = new List<ItemLoc>();
                combination.AddRange(food);
                combination.AddRange(entertain);
                combination.AddRange(coffee);
                if (combination.Sum(item => item.Cost) <= maxCost)
                {

```

```

        combinations.Add(combination);
    }
}
}
}
}
return combinations;
}

```

Đầu vào của thuật giải là bốn danh sách các địa điểm tương ứng với các hạng mục lưu trú, ăn uống, vui chơi/tham quan, cà phê. Ngoài ra còn có số lượng phần tử của các tổ hợp được tạo ra tương ứng với từng hạng mục và ngân sách được dùng trong chuyến đi du lịch. Đầu tiên, phương thức sẽ gọi *GetCombinations()* để khởi tạo danh sách các tổ hợp cho từng hạng mục. Tiếp theo, duyệt qua danh sách tổ hợp của của các hạng mục và kết hợp với chúng với nhau để tạo ra tổ hợp mới, đồng thời kiểm tra và loại bỏ các tổ hợp mới được tạo có tổng chi phí vượt quá ngân sách cho phép.

Trong phương thức này đã chia ra làm hai trường hợp:

- Trường hợp 1 ($\text{numHotel} \neq 0$): Các hạng mục đều có phần tử, tức là tổ hợp được tạo ra có đầy đủ các hạng mục.
- Trường hợp 2 ($\text{numHotel} == 0$): Hạng mục lưu trú không có bất kỳ phần tử nào, tức là dành cho hình thức thời gian trong ngày hay người dùng không muốn gợi ý địa điểm lưu trú.

Kết quả cuối cùng của phương thức *FindCombinations()* là một danh sách các danh sách *ItemLoc* thỏa mãn cả hai mục tiêu sau:

- Các tổ hợp chứa đầy đủ các hạng mục và số lượng địa điểm của từng hạng mục.
- Tổng chi phí các tổ hợp không vượt quá ngân sách.

→ Dựa vào danh sách này, hiển thị ra cho người dùng theo từng tổ hợp – một tổ hợp như là một lộ trình đi (ở đây chỉ nói đến các địa điểm sẽ di chuyển trong chuyến đi).

3.3.3 Tối ưu hóa lộ trình

a) Khoảng cách giữa các địa điểm

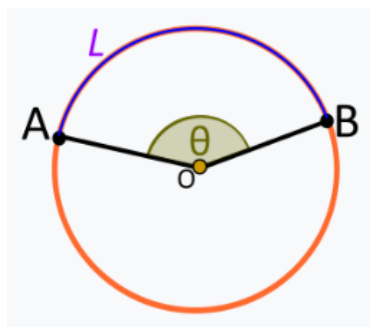
Sau khi có được danh sách tổ hợp các địa điểm, ta nhận thấy rằng khoảng cách giữa các địa điểm trong một số tổ hợp khá xa nhau điều này có thể sẽ làm ảnh hưởng tới thời gian và quãng đường di chuyển giữa các địa điểm cũng như cả cuộc hành trình. Vì vậy, để khắc phục nhược điểm này hệ thống sẽ tiến hành loại bỏ các tổ hợp có khoảng cách giữa các địa điểm xa nhau bằng cách sử dụng một công thức để tính toán khoảng cách giữa hai địa điểm được gọi là công thức Haversine.

Công thức Haversine là một công thức toán học được sử dụng vĩ độ và kinh độ để tính toán khoảng cách giữa hai điểm trên bề mặt cầu, ví dụ như Trái Đất. Công thức này được đặt theo tên của hàm Haversine trong toán học, một hàm liên quan đến các hàm lượng giác. Hàm Haversine của một góc θ là bình phương của hàm \sin của một nửa góc đó.

$$hav(\theta) = \sin^2\left(\frac{\theta}{2}\right) \quad (1)$$

Giả sử ta có hai điểm A và B lần lượt có vĩ độ và kinh độ là φ_A, φ_B và λ_A, λ_B . Gọi $R \approx 6371 \text{ Km}$ là bán kính của Trái Đất, tính khoảng cách d giữa A và B trên bề mặt Trái Đất.

Ta có công thức tính độ dài dây cung sau:



Hình 3. 4: Độ dài dây cung

(Nguồn: en.wikipedia.org/wiki/Central_angle)

$$d = R * \theta \Rightarrow \theta = \frac{d}{R} \quad (2)$$

Trong đó: θ : góc tạo bởi dây cung.

d hoặc L : độ dài dây cung.

R : bán kính đường tròn.

Thay (2) vào (1) ta được:

$$hav(\theta) = \sin^2\left(\frac{d}{2R}\right)$$

Mặc khác ta có:

$$hav(\theta) = hav(\varphi_1 - \varphi_2) + \cos(\varphi_1) \cos(\varphi_2) hav(\lambda_1 - \lambda_2)$$

Trong đó: φ_1 và λ_1 : Là vĩ độ và kinh độ của điểm thứ nhất.

φ_2 và λ_2 : Là vĩ độ và kinh độ của điểm thứ hai.

$$\Rightarrow \sin^2\left(\frac{d}{2R}\right) = hav(\varphi_A - \varphi_B) + \cos(\varphi_A) \cos(\varphi_B) hav(\lambda_A - \lambda_B)$$

$$\Leftrightarrow \sin^2\left(\frac{d}{2R}\right) = \sin^2\left(\frac{\varphi_A - \varphi_B}{2}\right) + \cos(\varphi_A) \cos(\varphi_B) \sin^2\left(\frac{\lambda_A - \lambda_B}{2}\right)$$

$$\Leftrightarrow \sin\left(\frac{d}{2R}\right) = \sqrt{\sin^2\left(\frac{\varphi_A - \varphi_B}{2}\right) + \cos(\varphi_A) \cos(\varphi_B) \sin^2\left(\frac{\lambda_A - \lambda_B}{2}\right)}$$

(Áp dụng hàm nghịch đảo hàm \sin - \arcsin)

$$\Leftrightarrow \frac{d}{2R} = \sin^{-1} \sqrt{\sin^2\left(\frac{\varphi_A - \varphi_B}{2}\right) + \cos(\varphi_A) \cos(\varphi_B) \sin^2\left(\frac{\lambda_A - \lambda_B}{2}\right)}$$

$$\Rightarrow d = 2R \sin^{-1} \sqrt{\sin^2\left(\frac{\varphi_A - \varphi_B}{2}\right) + \cos(\varphi_A) \cos(\varphi_B) \sin^2\left(\frac{\lambda_A - \lambda_B}{2}\right)} \quad (3)$$

Lưu ý: Đổi vĩ độ và kinh độ sang radian để tính. Công thức Haversine tính toán dựa trên vĩ độ và kinh độ nên giá trị trả về là khoảng cách theo đường chim bay, vì vậy khi so với khoảng cách trên thực tế khi di chuyển có thể chênh lệch 1 đến 2 kilomet.

Ví dụ: Điểm A có vĩ độ: 51.5007, kinh độ: 0.1246, điểm B có vĩ độ: 40.6892, kinh độ: 74.0445. Biết $R = 6371$ kilomet là bán kính Trái Đất. Tính khoảng cách d từ A đến B.

Bài làm

- Chuyển đổi sang radian:

- Vĩ độ và kinh độ của điểm A:

$$\varphi_A = 51.5007 \times \frac{\pi}{180} \approx 0.8985 \text{ radian}$$

$$\lambda_A = 00.1246 \times \frac{\pi}{180} \approx 0.0022 \text{ radian}$$

- Vĩ độ và kinh độ của điểm B:

$$\varphi_B = 40.6892 \times \frac{\pi}{180} \approx 0.7101 \text{ radian}$$

$$\lambda_B = 74.0445 \times \frac{\pi}{180} \approx 1.2923 \text{ radian}$$

- Khoảng cách d từ A đến B là:

$$d = 2R \sin^{-1} \sqrt{\sin^2\left(\frac{\varphi_A - \varphi_B}{2}\right) + \cos(\varphi_A) \cos(\varphi_B) \sin^2\left(\frac{\lambda_A - \lambda_B}{2}\right)}$$

Thay thế các giá trị vào biểu thức: $\Rightarrow d = 5574.8405 \text{ kilomet}$

Vậy khoảng cách từ A đến B là 5574.8405 kilomet (theo đường chim bay).

Thuật giải áp dụng công thức Haversine:

Phương thức chuyển đổi các giá trị sang radian:

```
public static double ToRadians(double numb)
{
    return numb * (Math.PI / 180.0);
}
```

Phương thức thực hiện tính toán khoảng cách giữa hai địa điểm *loc1* và *loc2* bằng công thức Haversine:

```
const double R = 6371; // Earth's radius
public static double CalculateDistance_Haversine(ItemLoc loc1, ItemLoc loc2)
{
    double dLat = ToRadians(loc1.Latitude - loc2.Latitude);
    double dLon = ToRadians(loc1.Longitude - loc2.Longitude);
    double lat1 = ToRadians(loc1.Latitude);
    double lat2 = ToRadians(loc2.Latitude);
    //Apply Haversine Formula
    double a = Math.Pow(Math.Sin(dLat / 2), 2) + Math.Cos(lat1) * Math.Cos(lat2)
    * Math.Pow(Math.Sin(dLon / 2), 2);
    double c = 2 * Math.Asin(Math.Sqrt(a));
    return Math.Round(R * c, 2);
}
```

Kết quả trả về là khoảng cách giữa hai điểm được làm tròn đến hai chữ số phần thập phân.

Sau khi có được danh sách tổ hợp các địa điểm *combinations* từ phương thức *FindCombinations()* ta sẽ tiến hành loại bỏ các tổ hợp không thỏa mãn điều kiện về khoảng cách giữa các địa điểm (khoảng cách giữa các địa điểm > *maxDistance*) bằng phương thức sau:

```
public static List<List<ItemLoc>> CombinationFilter(List<List<ItemLoc>>
combinations, double maxDistance)
{
    List<List<ItemLoc>> result = new List<List<ItemLoc>>();
    bool isValid;
    foreach (var item in combinations)
    {
        isValid = true;
        for (int i = 0; i < item.Count; i++)
        {
            for (int j = i + 1; j < item.Count; j++)
            {
                if (CalculateDistance_Haversine(item[i], item[j]) > maxDistance)
                {
                    isValid = false;
                    break;
                }
            }
            if (isValid == false)
            {
                break;
            }
        }
        if (isValid == true)
        {
            result.Add(item);
        }
    }
}
```



```

    }
}
return result;
}

```

Phương thức *CombinationFilter()* sẽ duyệt qua từng tổ hợp trong danh sách. Tiếp theo nó sử dụng phương thức *CalculateDistance_Haversine()* để tính toán khoảng cách giữa các địa điểm trong tổ hợp. Nếu trong trường hợp kết quả trả về lớn hơn *maxDistance* nó sẽ bỏ qua tổ hợp hiện tại và xét tổ hợp tiếp theo, ngược lại nó sẽ xét tiếp các địa điểm khác trong tổ hợp. Tổ hợp chứa tất cả các địa điểm thỏa mãn sẽ được thêm vào *result*.

Kết quả cuối cùng sau khi thực hiện ta sẽ thu được danh sách các tổ hợp đảm bảo khoảng cách giữa các địa điểm trong mỗi tổ hợp không quá xa ($> maxDistance$).

b) Đường đi ngắn nhất qua tất cả các địa điểm

Bên cạnh việc loại bỏ các tổ hợp có khoảng cách giữa các địa điểm xa nhau (theo tiêu chí cụ thể), hệ thống còn giúp đưa ra đường đi ngắn nhất hay thứ tự đi qua tất cả địa điểm sao cho tổng quan đường di chuyển là ngắn nhất.

Tạo ma trận khoảng cách giữa các địa điểm:

Để có thể lấy được khoảng cách thực giữa các địa điểm ta sẽ sử dụng một API do Microsoft phát triển đó là Distance Matrix.

Cấu trúc API Distance Matrix có dạng sau:

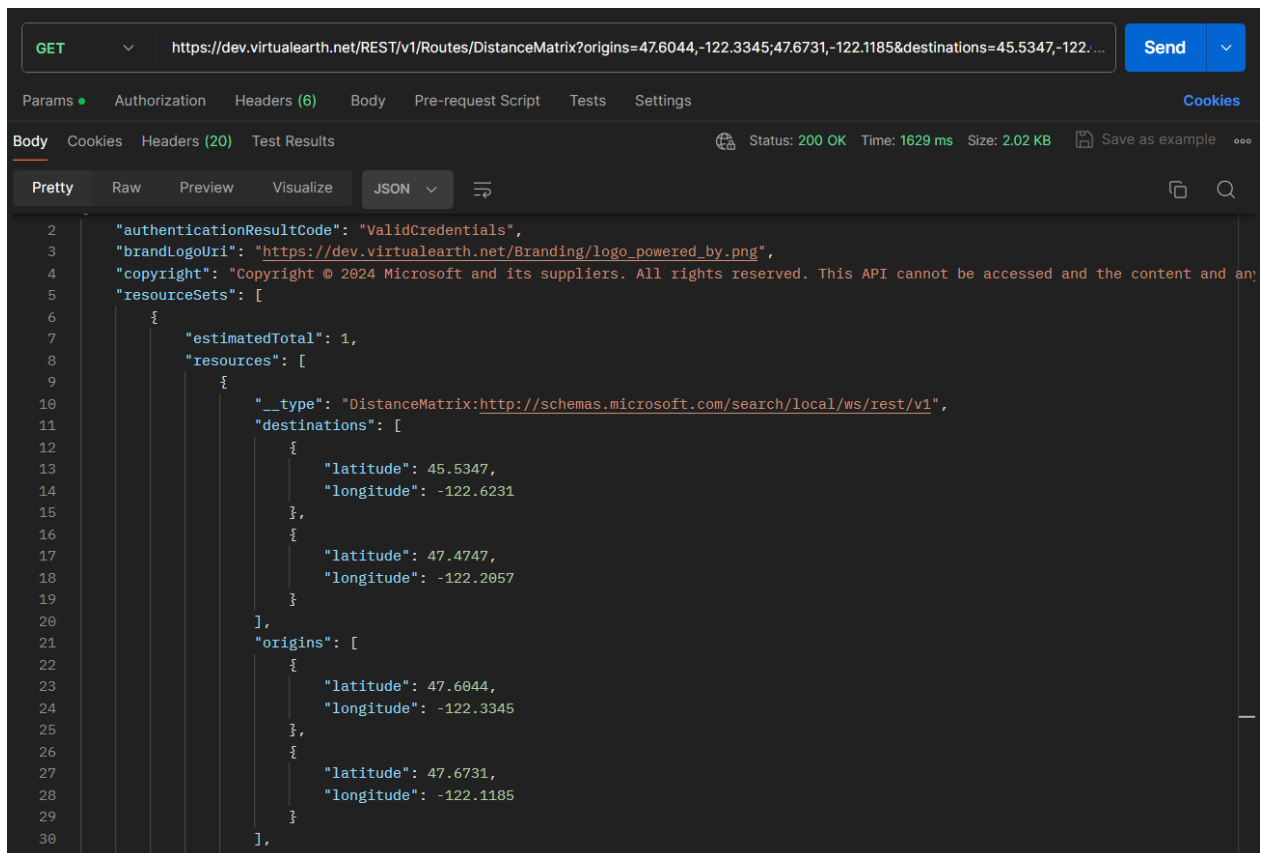
<https://dev.virtualearth.net/REST/v1/Routes/DistanceMatrix?origins={lat0,long0;lat1,lon1;latM,lonM}&destinations={lat0,lon0;lat1,lon1;latN,longN}&travelMode={travelMode}&startTime={startTime}&timeUnit={timeUnit}&key={BingMapsKey}>

Trong đó:

- *origins={lat0,long0;lat1,lon1;latM,lonM}*: Đây là danh sách các điểm bắt đầu, mỗi điểm được xác định bởi vĩ độ (lat) và kinh độ (lon).
- *destinations={lat0,lon0;lat1,lon1;latN,longN}*: Đây là danh sách các điểm đích, mỗi điểm cũng được xác định bởi vĩ độ và kinh độ.
- *travelMode={travelMode}*: Chế độ di chuyển được sử dụng để tính toán khoảng cách, ví dụ: đi bộ, lái xe, ...

- *startTime={startTime}*: Thời gian bắt đầu di chuyển, dùng để tính toán khoảng cách dựa trên tình hình giao thông thực tế.
- *timeUnit={timeUnit}*: Đơn vị thời gian được sử dụng trong kết quả, ví dụ: giây, phút, giờ.
- *key={BingMapsKey}*: Key để xác thực yêu cầu API của bạn với Bing Maps.

Sau đây là kết quả dưới dạng JSON được trả về khi sử dụng công cụ Postman để kiểm thử API trên (kết quả sau đây sẽ bỏ qua tham số *startTime*, *timeUnit*):



```

2  "authenticationResultCode": "ValidCredentials",
3  "brandLogoUri": "https://dev.virtualearth.net/Branding/logo_powered_by.png",
4  "copyright": "Copyright © 2024 Microsoft and its suppliers. All rights reserved. This API cannot be accessed and the content and an
5  "resourceSets": [
6    {
7      "estimatedTotal": 1,
8      "resources": [
9        {
10         "__type": "DistanceMatrix:http://schemas.microsoft.com/search/local/ws/rest/v1",
11         "destinations": [
12           {
13             "latitude": 45.5347,
14             "longitude": -122.6231
15           },
16           {
17             "latitude": 47.4747,
18             "longitude": -122.2057
19           }
20         ],
21         "origins": [
22           {
23             "latitude": 47.6044,
24             "longitude": -122.3345
25           },
26           {
27             "latitude": 47.6731,
28             "longitude": -122.1185
29           }
30         ]

```

Hình 3. 5: Kết quả JSON

```

31      "results": [
32      {
33          "destinationIndex": 0,
34          "originIndex": 0,
35          "totalWalkDuration": 0,
36          "travelDistance": 281.447,
37          "travelDuration": 156.8667
38      },
39      {
40          "destinationIndex": 1,
41          "originIndex": 0,
42          "totalWalkDuration": 0,
43          "travelDistance": 23.622,
44          "travelDuration": 19.4167
45      },
46      {
47          "destinationIndex": 0,
48          "originIndex": 1,
49          "totalWalkDuration": 0,
50          "travelDistance": 298.316,
51          "travelDuration": 165.6333
52      },
53      {
54          "destinationIndex": 1,
55          "originIndex": 1,
56          "totalWalkDuration": 0,
57          "travelDistance": 30.553,
58          "travelDuration": 21.3
59      }

```

Hình 3. 6: Kết quả JSON

Dựa trên kết quả JSON đó ta khởi tạo các class tương ứng sau:

```

internal class DistanceMatrix
{
    public string AuthenticationResultCode { get; set; }
    public string BrandLogoUri { get; set; }
    public string Copyright { get; set; }
    public List<ResourceSet> ResourceSets { get; set; }
    public int StatusCode { get; set; }
    public string StatusDescription { get; set; }
    public string TraceId { get; set; }
}

public class ResourceSet
{
    public int EstimatedTotal { get; set; }
    public List<Resource> Resources { get; set; }
}

public class Resource
{

```

```

    public string __Type { get; set; }
    public List<Item> Destinations { get; set; }
    public List<Item> Origins { get; set; }
    public List<Result> Results { get; set; }
}

public class Item
{
    public double Latitude { get; set; }
    public double Longitude { get; set; }
}

public class Result
{
    public int DestinationIndex { get; set; }
    public int OriginIndex { get; set; }
    public int TotalWalkDuration { get; set; }
    public double TravelDistance { get; set; }
    public double TravelDuration { get; set; }
}

```

Phương thức *ConnectionAPI_DisatanceMatrix()* sẽ lấy các thông tin về vĩ độ và kinh độ của các địa điểm trong tổ hợp *locations*, kết nối API của BingMap và đồng thời sẽ chuyển đổi dữ liệu JSON được trả về thành ma trận khoảng cách mà hệ thống có thể xử lý được với số hàng và số cột là số lượng địa điểm của tổ hợp *locations*:

```

public async Task<double[,]> ConnectionAPI_DisatanceMatrix(List<ItemLoc>
locations)
{
    string origins = string.Join(";", locations.Select(loc =>
    ${loc.Latitude},{loc.Longitude}"));
    Debug.WriteLine(origins);
    string destinations = origins; // Assuming you want a distance matrix between all
locations

```

```

    string requestUri =
        $"https://dev.virtualearth.net/REST/v1/Routes/DistanceMatrix?origins={origins}&
        destinations={destinations}&travelMode=driving&key={BingKey}";
    HttpResponseMessage response = await client.GetAsync(requestUri);
    string responseBody = "";
    if (response.IsSuccessStatusCode)
    {
        responseBody = await response.Content.ReadAsStringAsync();
        DistanceMatrix distanceMatrix =
            JsonConvert.DeserializeObject<DistanceMatrix>(responseBody);
        double[,] matrix = new double[locations.Count, locations.Count];
        if (distanceMatrix != null)
        {
            foreach (var resourceset in distanceMatrix.ResourceSets)
            {
                foreach (var resource in resourceset.Resources)
                {
                    for (int i = 0; i < locations.Count; i++)
                    {
                        int j = 0;
                        foreach (var result in resource.Results.Where(x => x.OriginIndex
== i).ToList())
                        {
                            matrix[i, j] = result.TravelDistance;
                            j++;
                        }
                    }
                }
            }
        }
    }
}

```

```

        else
        {
            Debug.WriteLine("Data connect API return --> NULL");
        }
        return matrix;
    }
    else
    {
        Debug.WriteLine($"Error: {response.StatusCode}");
    }
    return null;
}

```

Từ ma trận khoảng cách ta áp dụng thuật toán đàn kiến để tìm đường đi ngắn nhất:

```

public async Task<List<ItemLoc>> GetLocations_AntAlgorithm(List<ItemLoc>
locations)
{
    int[] item = new int[locations.Count];
    double[,] matrix = await ConnectionAPI_DisatanceMatrix(locations);
    item = AntAlgorithm.Ant_ACO(matrix);
    List<ItemLoc> result = new List<ItemLoc>();
    for (int k = 0; k < item.Length; k++)
    {
        result.Add(locations[k]);
    }
    return result;
}

```

GetLocations_AntAlgorithm() sẽ trả về một danh sách các địa điểm có thứ tự, dựa vào thứ tự này hệ thống sẽ hiển thị lộ trình đi sao cho tổng quãng đường di chuyển là ngắn nhất.

CHƯƠNG 4: THỬ NGHIỆM HỆ THỐNG

4.1 Các công cụ cài đặt

4.1.1 Cơ sở dữ liệu SQL Server



Hình 4. 1: SQL Server

(Nguồn: vn.got-it.ai)

Đây là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) được phát triển bởi Microsoft. Là một sản phẩm phần mềm có chức năng chính là lưu trữ và truy xuất dữ liệu theo yêu cầu của các ứng dụng phần mềm khác. Có thể chạy trên cùng một máy tính hoặc trên một máy tính khác trên mạng (bao gồm cả Internet). Cũng giống như các hệ quản trị cơ sở dữ liệu qua hệ khác, SQL Server được xây dựng trên lớp SQL – là ngôn ngữ lập trình tiêu chuẩn hoá được quản trị viên cơ sở dữ liệu (DBAs) và các chuyên gia IT sử dụng để quản lý cơ sở dữ liệu và truy vấn các dữ liệu nằm bên trong.

Một số phiên bản của SQL Server:

- SQL Server 2016: được phát triển như một phần trong chiến lược công nghệ “mobile first, cloud first” với những tính năng mới như: điều chỉnh hiệu suất, phân tích hoạt động thời gian thực, trực quan hóa dữ liệu và báo cáo trên thiết bị di động và sự hỗ trợ của hybrid cloud.

- SQL Server 2017: hỗ trợ chạy trên Linux, điều này làm SQL Server chuyển từ nền tảng cơ sở dữ liệu sang một hệ điều hành mã nguồn mở thường thấy trong các doanh nghiệp.
- SQL Server 2019: cho phép người dùng kết hợp các vùng chứa SQL Server, HDFS và Spark với nhau bằng cách sử dụng tính năng Big Data Cluster mới. Thêm vào đó, một tính năng mới khác là khả năng phục hồi dữ liệu được tăng tốc nhanh hơn.

Ưu điểm:

- Bảo mật dữ liệu: Cung cấp các tính năng bảo mật dữ liệu như mã hóa dữ liệu, kiểm soát quyền truy cập và xác thực người dùng để đảm bảo an toàn cho dữ liệu của bạn.
- Dễ sử dụng: SQL Server có giao diện đồ họa thân thiện với người dùng. Bên cạnh đó, nó cung cấp các công cụ quản lý dữ liệu giúp cho người dùng quản lý cơ sở dữ liệu một cách hiệu quả.
- Khả năng mở rộng: Có khả năng mở rộng tốt và có thể được sử dụng để quản lý các cơ sở dữ liệu lớn.
- Hiệu suất cao: SQL Server có khả năng xử lý các truy vấn phức tạp và có thể xử lý hàng triệu bản ghi trong một giây.

Nhược điểm:

- Giá thành cao: SQL Server là một sản phẩm phần mềm có giá thành cao hơn so với phần mềm quản lý cơ sở dữ liệu khác.
- Khó khăn trong việc triển khai: Việc triển khai có thể gặp nhiều trở ngại khó khăn và yêu cầu kỹ thuật viên có nhiều kinh nghiệm để triển khai thành công.

SQL Server có nhiều ứng dụng sau:

- Được dùng để tạo, duy trì, quản lý và triển khai hệ thống RDBMS.
- Nó có thể được sử dụng để phân tích dữ liệu bằng SSAS - SQL Server Analysis Services.
- Tạo báo cáo bằng SSRS – SQL Server Reporting Services và thực hiện quá trình ETL (Extract – Transform – Load) bằng SSIS – SQL Server Integration Services.

4.1.2 Công cụ Microsoft Visual Studio

Visual Studio là một môi trường phát triển tích hợp (IDE) được phát triển bởi Microsoft. Nó cung cấp một loạt các công cụ và tính năng cho việc phát triển ứng dụng, bao gồm phát triển ứng dụng máy tính, ứng dụng web, ứng dụng di động, ứng dụng trò chơi và nhiều ứng dụng khác. Đồng thời nó là IDE tốt nhất để phát triển các ứng dụng viết bằng ngôn ngữ C#.



Hình 4. 2: Microsoft Visual Studio

(Nguồn: hanixdiy.blogspot.com)

Microsoft Visual Studio là một công cụ đa chức năng được sử dụng với nhiều mục đích khác nhau:

- Phát triển ứng dụng Windows: Visual Studio là một công cụ phát triển chính cho phát triển ứng dụng Windows. Bạn có thể phát triển ứng dụng desktop truyền thống, ứng dụng UWP, ứng dụng Windows Forms, và nhiều ứng dụng khác trên nền tảng Windows.
- Phát triển ứng dụng web: Visual Studio hỗ trợ phát triển ứng dụng web bằng cách cung cấp tích hợp với Asp.Net, Asp.Net Core, và các framework phát triển web như Angular, React, và Vue.js.
- Hỗ trợ đa nền tảng: Visual Studio đã tích hợp .NET Core để hỗ trợ phát triển ứng dụng đa nền tảng trên Windows, Linux và macOS.
- Hỗ trợ đa ngôn ngữ: Visual Studio hỗ trợ nhiều ngôn ngữ lập trình, cho phép bạn phát triển ứng dụng bằng nhiều ngôn ngữ khác nhau.
- Quản lý phiên bản tích hợp: Visual Studio tích hợp với các hệ thống quản lý phiên bản như Git và Azure DevOps, giúp bạn theo dõi và quản lý sự thay đổi trong mã nguồn dự án.

Ưu điểm:

- Phát triển nhanh: Giúp phát triển ứng dụng web và ứng dụng Xamarin cho nhu cầu của khách hàng.

- Tính năng điều hướng tốt: Hỗ trợ tìm kiếm, lọc và xem trước mã mà bạn đang làm việc.
- Tùy chỉnh tốt: Cho phép tùy chỉnh và bao gồm cả tiện ích mở rộng của bên thứ ba.
- Hỗ trợ IntelliSense: Giúp tự động hoàn thành mã.

Nhược điểm:

- Debugging: Visual Studio giỏi về debugging, tuy nhiên đôi khi nó có thể bị treo, yêu cầu bạn dừng debugging hoặc khởi động lại Visual Studio.
- Ứng dụng nặng: Visual Studio là một ứng dụng nặng và mất thời gian để tải.
- Giao diện người dùng: Giao diện người dùng của Visual Studio tuy đẹp nhưng có thể hơi khó, đặc biệt là đối với người mới.
- Thiết kế ứng dụng: Sử dụng các thành phần thiết kế của Visual Studio có thể phá vỡ logic thiết kế ứng dụng lớn.

4.1.3 Công cụ Postman



Hình 4. 3: Công cụ Postman

(Nguồn: *testmentor.vn*)

Postman hiện là một trong những công cụ phổ biến để phát triển API và kiểm thử API. Với Postman, ta có thể gọi REST API mà không cần viết bất kỳ dòng lệnh nào. Nó hỗ trợ tất cả các phương thức HTTP (GET, POST, PUT, DELETE,...), Postman còn giúp lưu lại lịch sử các lần request rất có ích cho việc sử dụng lại khi cần.

Postman là một công cụ dành cho các nhà phát triển phần mềm, nhà kiểm thử phần mềm và các chuyên gia trong lĩnh vực API.

Các lợi ích khi sử dụng Postman:

- Kiểm thử API dễ dàng: Cung cấp giao diện thân thiện, dễ sử dụng để tạo và gửi các request. Có thể chủ động xác định các tham số, nội dung yêu cầu, tiêu đề và phân loại một cách rõ ràng.
- Sử dụng Collections: Cho phép người dùng các bộ sưu tập cho lệnh API.
- Tương tác linh hoạt: Cho phép với API thông qua các yêu cầu HTTP: GET, POST, PUT, DELETE,...Có thể yêu cầu đến địa chỉ Url cụ thể, xử lý và hiển thị kết quả trả về, từ đó có thể kiểm tra tính chính xác và hiệu suất của API.
- Quản lý chức năng và biến: Cho phép quản lý nhiều môi trường khác nhau như: môi trường phát triển, môi trường thử nghiệm và sử dụng các biến để điều chỉnh các giá trị yêu cầu. Với điều này sẽ giúp việc chuyển đổi giữa các môi trường và quá trình triển khai và kiểm thử trở nên một cách dễ dàng.
- Tài liệu hóa API: Cho phép dễ dàng tạo và chia sẻ dữ liệu API.

Một số chức năng chính của Postman:

- Cho phép gửi HTTP Request với các method GET, POST, PUT, DELETE.
- Cho phép post dữ liệu dưới dạng form (key-value), text, json.
- Hiện kết quả trả về dạng text, hình ảnh, XML, JSON.
- Hỗ trợ authorization (Oauth1, 2).
- Cho phép thay đổi header của các request.

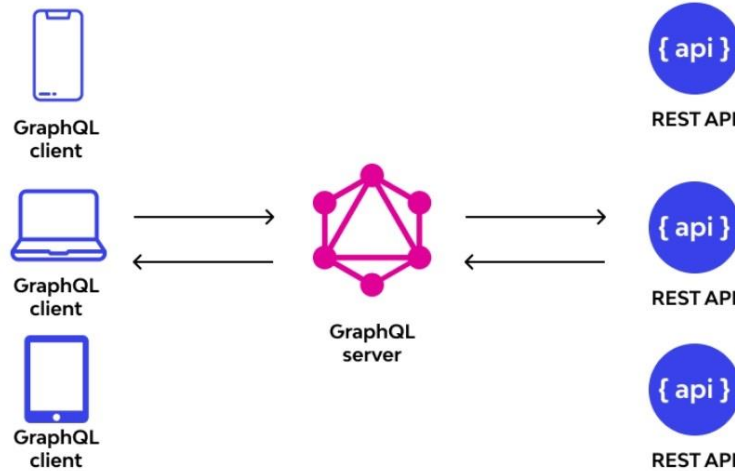
4.1.4 Bing Maps API

API là các phương thức, các giao thức kết nối các thư viện và ứng dụng khác với nhau. Viết tắt của từ Application Programming Interface – giao diện lập trình ứng dụng. Nó cung cấp khả năng truy xuất đến một tập các hàm hay dùng, và từ đó có thể thực hiện việc trao đổi dữ liệu giữa các ứng dụng với nhau.

Có nhiều loại API khác nhau, sau đây là một số loại phổ biến:

- REST API: Là một kiểu API được thiết kế dựa trên các nguyên tắc của giao thức HTTP.
- SOAP API: Là một tiêu chuẩn giao tiếp dựa trên XML, cho phép các ứng dụng trao đổi thông tin qua mạng internet.

- GraphQL API: Cung cấp khả năng truy vấn linh hoạt và mạnh mẽ, cho phép các ứng dụng truy cập dữ liệu một cách hiệu quả hơn. Nó đặc biệt hữu ích trong các ứng dụng lớn và phức tạp, nơi cần tối ưu hóa việc truy cập và sử dụng dữ liệu.



Hình 4. 4: GraphQL API

(Nguồn: interdata.vn)

Bing Maps API là một bộ dịch vụ web cung cấp dữ liệu bản đồ và các chức năng liên quan cho các nhà phát triển web và ứng dụng. Nó cho phép bạn tích hợp bản đồ, hình ảnh vệ tinh, dữ liệu giao thông, thông tin địa điểm và nhiều tính năng khác vào các ứng dụng và trang web của bạn.



Hình 4. 5: Bing Maps

(Nguồn: www.androidheadlines.com)

Sau đây là một số dịch vụ của REST Services do Bing maps API cung cấp:

- Locations: Tìm một vị trí dựa trên địa chỉ, điểm hoặc truy vấn.
- Elevations: Lấy các độ cao cho một tập hợp các vị trí, một đường dẫn hoặc một khu vực trên Trái đất.
- Imagery: Lấy một bản đồ tĩnh, lấy một bản đồ tĩnh hiển thị một tuyến đường và lấy metadata hình ảnh.
- Routes: Tìm một tuyến đường đi bộ, lái xe hoặc sử dụng giao thông công cộng. Tìm các tuyến đường từ các tuyến đường chính đến một vị trí.
- Traffic: Lấy thông tin giao thông cho một khu vực địa lý.
- Autosuggest: Lấy danh sách các thực thể được đề xuất tự động từ truy vấn của người dùng.
- Time Zone: Lấy một múi giờ theo điểm hoặc truy vấn. Chuyển đổi thời gian UTC thành một múi giờ. Lấy thông tin về các tiêu chuẩn múi giờ Windows và IANA.

Bing Maps API cung cấp nhiều mức độ quyền truy cập khác nhau, bao gồm:

- Miễn phí: Cung cấp quyền truy cập vào các tính năng cơ bản như bản đồ, hình ảnh vệ tinh và tìm kiếm.
- Trả phí: Cung cấp quyền truy cập vào các tính năng nâng cao hơn như dữ liệu giao thông, lộ trình và phân tích.

Bing Maps API tương thích với nhiều nền tảng phát triển phổ biến, bao gồm:

- Web: JavaScript, HTML5, CSS3.
- Di động: iOS, Android, Windows Phone.
- Máy tính để bàn: Windows, macOS, Linux.

Bing Maps API được sử dụng bởi nhiều tổ chức và doanh nghiệp trên toàn thế giới, bao gồm:

- Chính phủ: Các cơ quan chính phủ sử dụng Bing Maps API để cung cấp các dịch vụ bản đồ công cộng, chẳng hạn như bản đồ thuế và bản đồ quy hoạch.

- Doanh nghiệp: Các doanh nghiệp sử dụng Bing Maps API để tích hợp bản đồ vào trang web và ứng dụng của họ, để cung cấp thông tin địa điểm cho khách hàng và để phân tích dữ liệu vị trí.
- Nhà phát triển: Các nhà phát triển sử dụng Bing Maps API để tạo các ứng dụng bản đồ sáng tạo và hữu ích.

4.2 Giao diện hệ thống và thử nghiệm các chức năng

Đầu tiên hệ thống nhận dữ liệu đầu vào từ người dùng:

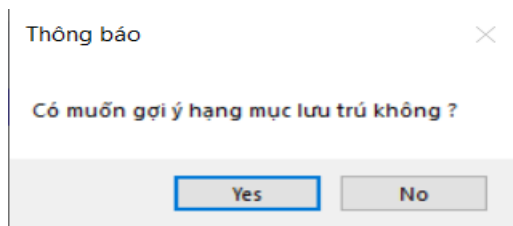
The screenshot shows a web application interface for finding travel destinations. At the top, there are input fields for 'Thành phố' (City) set to 'Hồ Chí Minh', 'Chi phí' (Cost) set to '2000000', 'Số người' (Number of people) set to '1', and 'Thời gian' (Time) set to '1 ngày'. A 'Tìm kiếm' (Search) button is on the right. Below these is a section titled 'Select according to need' with four filter categories: 'Lưu trú' (Accommodation) with options like homestay, khách sạn, and nhà nghỉ; 'Đồ ăn' (Food) with options like ăn vặt, bình dân, buffet, bún, cháo, đồ nướng, and hải sản; 'Vui chơi/tham quan' (Entertainment/Excursion) with options like bắn súng, bơi, giải trí, nghệ thuật, tham quan, and trải nghiệm; and 'Cà phê' (Coffee) with options like Hàn Quốc, hiện đại, thiên nhiên, tối giản, truyền thống, vintage, and xưa cũ. An 'Áp dụng' (Apply) button is to the right of these filters. Below the filters is a table titled 'Địa điểm' (Locations) with columns: STT, Hàng mục, Loại, Ảnh, Tên địa điểm, Địa chỉ, Mô tả, Link, and Chi phí. The table lists three items: 1. Lưu trú, khách sạn, Anh Duy Hotel, 103 Nguyễn Công Trứ, Phu..., Anh Duy Hotel (Near Ben T..., https://www.traveloka.c..., 1.192.000 VND; 2. Lưu trú, khách sạn, Khách sạn Meraki Boutique, 178 Bùi Viện, Phạm Ngũ Lão..., Dành cho những du khách..., https://www.traveloka.c..., 661.000 VND; 3. Lưu trú, khách sạn, Khách sạn DDA Quận 1, 183 Đê Thám, Phường Phá..., Khi lưu trú tại khách sạn th..., https://www.traveloka.c..., 1.134.000 VND. A 'Khởi tạo' (Reset) button is at the bottom of the table.

STT	Hàng mục	Loại	Ảnh	Tên địa điểm	Địa chỉ	Mô tả	Link	Chi phí
1	Lưu trú	khách sạn		Anh Duy Hotel	103 Nguyễn Công Trứ, Phu...	Anh Duy Hotel (Near Ben T...	https://www.traveloka.c...	1.192.000 VND
2	Lưu trú	khách sạn		Khách sạn Meraki Boutique	178 Bùi Viện, Phạm Ngũ Lão...	Dành cho những du khách...	https://www.traveloka.c...	661.000 VND
3	Lưu trú	khách sạn		Khách sạn DDA Quận 1	183 Đê Thám, Phường Phá...	Khi lưu trú tại khách sạn th...	https://www.traveloka.c...	1.134.000 VND

Hình 4. 6: Dữ liệu đầu vào

Như trong hình đầu vào sẽ là: Khu vực muốn du lịch (Hồ Chí Minh), chi phí cho chuyến đi (2.000.000 VND), số lượng người (1 người), hình thức thời gian (1 ngày). Sau đó ấn nút tìm kiếm, hệ thống sẽ truy xuất dữ liệu địa điểm từ cơ sở dữ liệu các địa điểm và đưa ra danh sách các địa điểm thỏa như trong hình (đây chính là dữ liệu mà hệ thống sử dụng để tiến hành xử lý). Ngoài ra, còn có thể chọn lọc đặc trưng của các địa điểm ở từng hạng mục theo sở thích.

Trong trường hợp hình thức thời gian là một ngày thì sẽ hiện ra thông báo sau:



Hình 4. 7: Cần nơi lưu trú không ?

Khi nhấn vào nút khởi tạo, hệ thống tiến hành gợi ý một danh sách 20 tổ hợp các địa điểm cho chuyến đi và được sắp xếp theo chi phí tăng dần như sau:

The screenshot shows a web application interface with a "Recommendation" header and a "Tạo mới" button. It displays three recommendation tables, each with a "Đường đi" button. The first table is for "Gợi ý 1 - Tổng chi phí 1.265.000 VND". The second table is for "Gợi ý 2 - Tổng chi phí 1.321.000 VND". The third table is for "Gợi ý 3 - Tổng chi phí 1.400.000 VND". Each table has columns: STT, Hạng mục, Loại, Tên địa điểm, Địa chỉ, Link, and Chi phí.

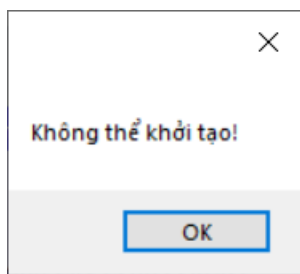
STT	Hạng mục	Loại	Tên địa điểm	Địa chỉ	Link	Chi phí
1	Lưu trú	khách sạn	Khách sạn Meraki Boutique	178 Bùi Viện, Phạm Ngũ Lão, Phạm ...	https://www.traveloka.com/vi-vn/...	661.000 VND
2	Ăn uống	bún	Bún Riêu	219 Đ. Nguyễn Trãi, Phường Nguyễ...	https://www.tiktok.com/@janwitht...	35.000 VND
3	Ăn uống	ăn vặt	Chè Hiến Khánh	718 Nguyễn Đình Chiểu, phường 1, ...	https://www.foody.vn/ho-chi-minh...	99.000 VND
4	Ăn uống	bún	Bún đậu Cỏ Khàn	102/1B Cống Quỳnh, Q1		120.000 VND
5	Ăn uống	ăn vặt	Khoai tây chiên mix - Fries	4 Rạch Bùng Binh, Phường 10, Quậ...	https://www.tiktok.com/@diantho...	150.000 VND

STT	Hạng mục	Loại	Tên địa điểm	Địa chỉ	Link	Chi phí
1	Lưu trú	khách sạn	Khách sạn Meraki Boutique	178 Bùi Viện, Phạm Ngũ Lão, Phạm ...	https://www.traveloka.com/vi-vn/...	661.000 VND
2	Ăn uống	ăn vặt	Bò nê Hi-Ushi	56 Cao Bá Nha, P Nguyễn Cư Trinh, ...		60.000 VND
3	Ăn uống	ăn vặt	Chân Gà - Gấu Food	193 Bà Hạt, Phường 9, Quận 10, Thà...	https://www.tiktok.com/@gaufo...	100.000 VND
4	Ăn uống	bình dân	Nhà hàng Ngõ 89	Ngõ 89 - 89 Nguyễn Du, Quận 1, TP...		150.000 VND
5	Ăn uống	Hàn Quốc	Quán Hàn - Hẻm Fast Food	75 Đ. Nguyễn Cư Trinh, Phường Ng...	https://www.tiktok.com/@diantho...	150.000 VND

STT	Hạng mục	Loại	Tên địa điểm	Địa chỉ	Link	Chi phí
1	Lưu trú	khách sạn	Khách sạn Meraki Boutique	178 Bùi Viện, Phạm Ngũ Lão, Phạm ...	https://www.traveloka.com/vi-vn/...	661.000 VND
2	Ăn uống	ăn vặt	Bột chiên Đức Hoa	1 Lố X chung cư Ngõ Gia Tự, phườn...		100.000 VND

Hình 4. 8: Danh sách tổ hợp các địa điểm

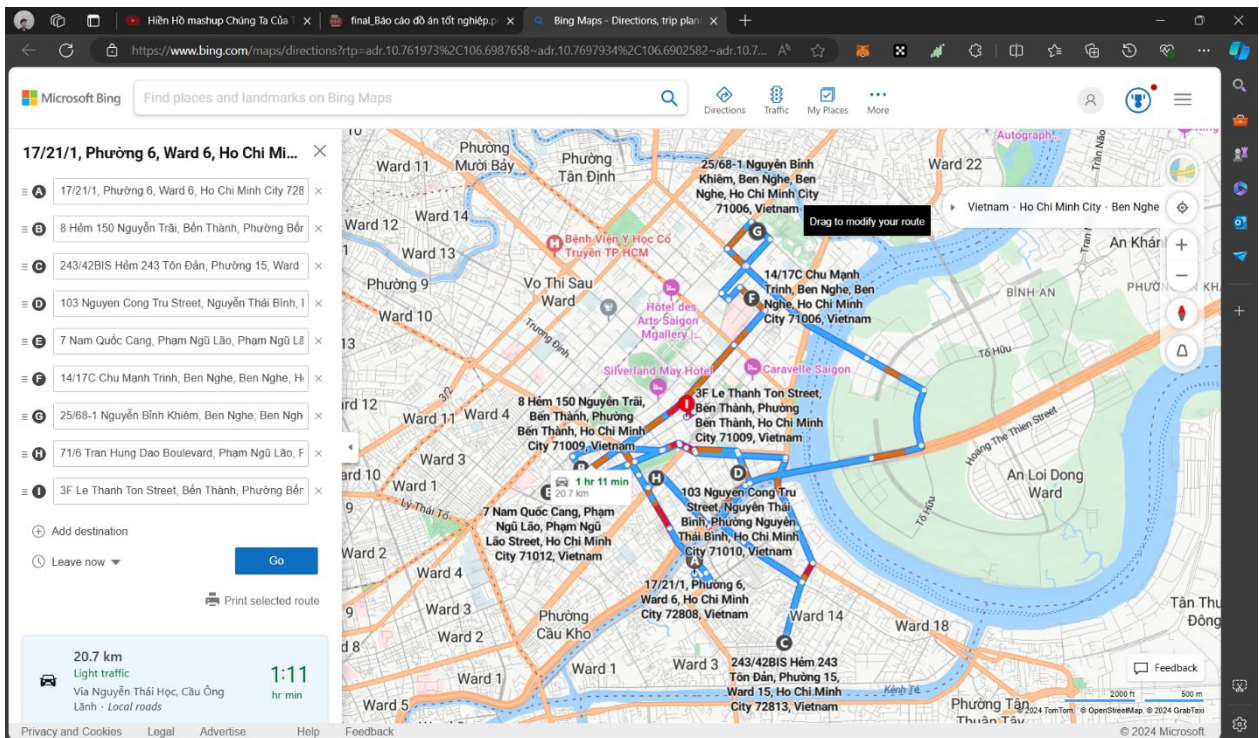
Nếu hiển thị thông báo sau:



Hình 4. 9: Không thể khởi tạo

Thì có nghĩa là cơ sở dữ liệu không đủ để có thể tiến hành khởi tạo danh sách gợi ý.

Nút đường đi có chức năng chuyển hướng đến trang Bing maps, ở đó người dùng có thể xem được thứ tự di chuyển sao cho quãng đường đi ngắn nhất:



Hình 4. 10: Thứ tự di chuyển

4.3 Kết quả thử nghiệm

Tối ưu chi phí: Đã tạo được tổ hợp các địa điểm, đầy đủ các hạng mục, số lượng các địa điểm và trên hết tổng chi phí không vượt quá ngân sách.

Tổ hợp địa điểm (Số lượng tổ hợp - 104):	
Tổng chi chi của tổ hợp là: 974000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 130 - Cost: 0, Id: 121 - Cost: 60000, Id: 104 - Cost: 170000, Id: 141 - Cost: 70000, Id: 152 - Cost: 100000	
Tổng chi chi của tổ hợp là: 954000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 130 - Cost: 0, Id: 121 - Cost: 60000, Id: 104 - Cost: 170000, Id: 141 - Cost: 70000, Id: 181 - Cost: 80000	
Tổng chi chi của tổ hợp là: 1384000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 130 - Cost: 0, Id: 121 - Cost: 60000, Id: 104 - Cost: 170000, Id: 168 - Cost: 80000, Id: 154 - Cost: 500000	
Tổng chi chi của tổ hợp là: 1024000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 130 - Cost: 0, Id: 121 - Cost: 60000, Id: 104 - Cost: 170000, Id: 157 - Cost: 100000, Id: 155 - Cost: 120000	
Tổng chi chi của tổ hợp là: 964000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 130 - Cost: 0, Id: 121 - Cost: 60000, Id: 104 - Cost: 170000, Id: 149 - Cost: 80000, Id: 181 - Cost: 80000	
Tổng chi chi của tổ hợp là: 774000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 111 - Cost: 0, Id: 120 - Cost: 0, Id: 116 - Cost: 30000, Id: 141 - Cost: 70000, Id: 152 - Cost: 100000	
Tổng chi chi của tổ hợp là: 754000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 111 - Cost: 0, Id: 120 - Cost: 0, Id: 116 - Cost: 30000, Id: 141 - Cost: 70000, Id: 181 - Cost: 80000	
Tổng chi chi của tổ hợp là: 1184000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 111 - Cost: 0, Id: 120 - Cost: 0, Id: 116 - Cost: 30000, Id: 168 - Cost: 80000, Id: 154 - Cost: 500000	
Tổng chi chi của tổ hợp là: 824000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 111 - Cost: 0, Id: 120 - Cost: 0, Id: 116 - Cost: 30000, Id: 157 - Cost: 100000, Id: 155 - Cost: 120000	
Tổng chi chi của tổ hợp là: 764000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 111 - Cost: 0, Id: 120 - Cost: 0, Id: 116 - Cost: 30000, Id: 149 - Cost: 80000, Id: 181 - Cost: 80000	
Tổng chi chi của tổ hợp là: 934000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 131 - Cost: 0, Id: 118 - Cost: 20000, Id: 104 - Cost: 170000, Id: 141 - Cost: 70000, Id: 152 - Cost: 100000	
Tổng chi chi của tổ hợp là: 914000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 131 - Cost: 0, Id: 118 - Cost: 20000, Id: 104 - Cost: 170000, Id: 141 - Cost: 70000, Id: 181 - Cost: 80000	
Tổng chi chi của tổ hợp là: 1344000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 131 - Cost: 0, Id: 118 - Cost: 20000, Id: 104 - Cost: 170000, Id: 168 - Cost: 80000, Id: 154 - Cost: 500000	
Tổng chi chi của tổ hợp là: 984000	
Id: 86 - Cost: 35000, Id: 57 - Cost: 139000, Id: 87 - Cost: 150000, Id: 56 - Cost: 250000, Id: 131 - Cost: 0, Id: 118 - Cost: 20000, Id: 104 - Cost: 170000, Id: 157 - Cost: 100000, Id: 155 - Cost: 120000	

Hình 4. 11: Kết quả tối ưu chi phí

Thông tin về lộ trình:

- Khoảng cách giữa các địa điểm trong lộ trình nằm trong phạm vi cho phép.

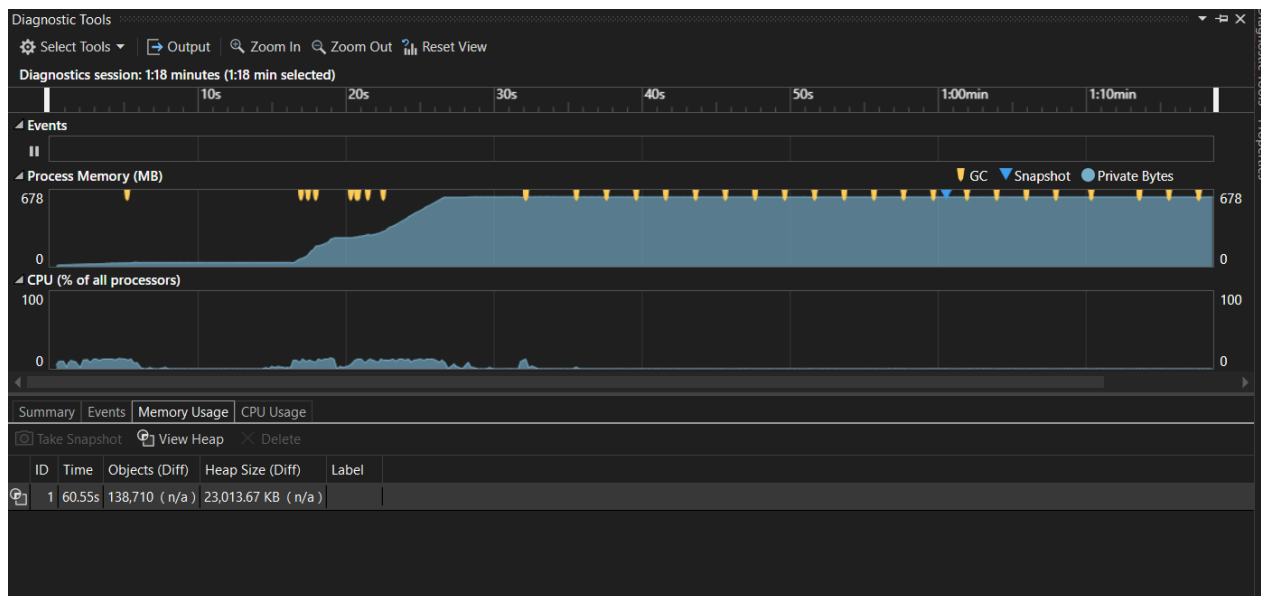
- Các địa điểm đã được sắp xếp theo thứ tự di chuyển.

```
Khoang cách giữa các địa điểm:
Khoang cách từ 0 -> 6: 3.258 Km
Khoang cách từ 6 -> 8: 1.687 Km
Khoang cách từ 8 -> 4: 0.677 Km
Khoang cách từ 4 -> 5: 1.289 Km
Khoang cách từ 5 -> 7: 4.174 Km
Khoang cách từ 7 -> 2: 0.347 Km
Khoang cách từ 2 -> 3: 3.274 Km
Khoang cách từ 3 -> 1: 1.108 Km
Khoang cách từ 1 -> 0: 5.537 Km
-> Tổng quãng đường di chuyển là: 21.351
Trip: 64 -> 94 -> 57 -> 68 -> 114 -> 131 -> 135 -> 149 -> 162 -> 64
```

Hình 4. 12: Thông tin lộ trình

So với thực tế thì với tổng quãng đường 21.351 km để đi qua tất cả 9 địa điểm là không quá nhiều, quãng đường di chuyển giữa các địa điểm có thể chấp nhận được và có thể điều chỉnh trong hệ thống để có kết quả tốt hơn.

Hiệu suất của hệ thống khi chạy:



Hình 4. 13: Hiệu suất chạy

Biểu đồ Process Memory (MB) cho thấy việc sử dụng bộ nhớ của quá trình ổn định với những biến động nhỏ không đáng kể. Biểu đồ đạt trạng thái cao nhất với giá trị là 614 MB. Biểu đồ CPU thấp và ổn định. Tổng quan, hệ thống đang hoạt động ổn định trên số lượng dữ liệu đang xử lý hiện tại.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết quả đạt được

Hệ thống đã thực hiện được các mục tiêu đề ra ban đầu. Phân tích được dữ liệu người dùng gồm: nhu cầu, sở thích và ngân sách của người dùng, từ đó đưa ra được các địa điểm phù hợp. Các địa điểm được gợi ý có đầy đủ các thông tin cơ bản: tên địa điểm, địa chỉ, mô tả thông tin chi tiết, đường link giới thiệu, hình ảnh và cung như là chi phí dừng chân tại địa điểm đó.

Từ danh sách các địa điểm đó, hệ thống đã tạo ra được các tổ hợp địa điểm thỏa mãn các yêu cầu: có đầy đủ các hạng mục lưu trú, ăn uống, vui chơi/tham quan và cà phê, số lượng địa điểm của từng hạng mục phù hợp với hình thức thời gian du lịch (trong ngày và hai ngày một đêm) theo nhu cầu của người dùng, đảm bảo tổng chi phí sử dụng để đi đến tất cả các địa điểm trong tổ hợp không vượt quá ngân sách cho chuyến đi hoặc thậm chí giúp người dùng tiết kiệm được tiền.

Hệ thống cũng đã giúp cho người dùng tiết kiệm thời gian di chuyển từ địa điểm này sang địa điểm khác bằng cách loại bỏ các tổ hợp chứa các địa điểm mà khoảng cách của chúng vượt qua một ngưỡng nào đó (ví dụ: vượt quá 4 km). Đồng thời, hệ thống còn đưa ra được cách thức di chuyển đi tất cả các địa điểm sao cho tổng quãng đường cần đi là ngắn nhất, giúp cho người dùng có trải nghiệm du lịch toàn vẹn nhất.

Ngoài ra, đây là một hệ thống được xây dựng một cách logic, dễ hiểu thông qua việc đã chia ra làm hai module để xử lý các công việc khác nhau: trong khi module tối ưu chi phí chịu trách nhiệm xử lý các công việc sinh tổ hợp và tính toán chi phí tổ hợp, thì module tối ưu lộ trình thực hiện các công việc về tính toán khoảng cách, tìm lộ trình di chuyển ngắn nhất. Cả hai module đều độc lập với nhau nên có thể tùy chỉnh theo nhu cầu. Hệ thống có thể dễ dàng nâng cấp, phát triển thêm chức năng mới.

2. Hạn chế

Tại vì hệ thống được phát triển và hoạt động trên một lượng dữ liệu đủ để thấy được cách mà hệ thống sẽ hoạt động cho nên hiệu suất hoạt động có thể sẽ tốt. Tuy nhiên, trong tương lai nếu cần xử lý một lượng lớn dữ liệu, hay có thêm các chức năng mới có thể làm cho hệ thống bị sụt giảm hiệu suất, thời gian xử lý chậm ảnh hưởng đến trải nghiệm người

dùng. Các thuật toán được sử dụng và xây dựng trong hệ thống chưa phải là tối ưu nhất nên cũng sẽ ảnh hưởng không nhỏ đến hiệu suất của hệ thống và kết quả trả về từ hệ thống chỉ ở mức chấp nhận và tham khảo.

3. Hướng phát triển

Kết hợp hệ thống với một hệ thống gợi ý khác như: Content-Based Recommendation System, Collaborative Filtering Recommendation System để có thể nâng cao trải nghiệm cho người dùng, cũng như các địa điểm được gợi ý có mức độ phù hợp cao hơn, chính xác hơn. Kết hợp chúng với nhau bằng cách sau:

- Lưu lại lịch sử danh sách các tổ hợp địa điểm mà người dùng chọn và đánh giá.
- Dựa vào dữ liệu lịch sử này đưa ra các gợi ý tổ hợp phù hợp như là một đề xuất.
- Dựa vào lịch sử hoạt động tương tác các địa điểm của người dùng mà hệ thống sẽ ưu tiên tìm và chọn các địa điểm có đặc điểm tương tự để tạo tổ hợp.

Ngoài ra, có thể tìm hiểu và nghiên cứu cách tối ưu các thuật toán dùng trong hệ thống hoặc thay đổi lại bằng một thuật toán tốt hơn tối ưu hơn. Hoặc cũng có thể phát triển thêm tính năng cho thuật toán đàn kiến như sau:

- Lưu lại lịch sử đánh giá từng địa điểm của mỗi người dùng và tổng hợp lại.
- Bên cạnh việc viết đưa ra cách đi qua tất cả các địa điểm sao cho là ngắn nhất, thì sẽ ưu tiên đi qua trước các địa điểm có mức độ đánh giá cao hơn.

Cuối cùng là xây dựng một chương trình, ứng dụng hoàn chỉnh: web, android app, ios app,... Với giao diện đẹp mắt, phù hợp với xu hướng, có đầy đủ các chức năng và áp dụng hệ thống vào ứng dụng đó.

TÀI LIỆU THAM KHẢO

- [1] Tổng cục Du lịch, “Tổng thu từ khách du lịch giai đoạn 2008 - 2023,” [Trực tuyến]. Available: <https://vietnamtourism.gov.vn/statistic/receipts>. [Đã truy cập 03/04/2024].
- [2] Microsoft, “What's new in C#,” Microsoft, [Trực tuyến]. Available: <https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-12>. [Đã truy cập 04/04/2024].
- [3] Lucie, “SQL Server là gì? SQL Server giúp bạn làm việc dễ dàng hơn?,” TopDev, [Trực tuyến]. Available: <https://topdev.vn/blog/sql-server-la-gi/#sql-server-la-gi>. [Đã truy cập 04/04/2024].
- [4] Microsoft, “What is Visual Studio?,” Microsoft, [Trực tuyến]. Available: <https://learn.microsoft.com/en-us/visualstudio/get-started/visual-studio-ide?view=vs-2022>. [Đã truy cập 04/04/2024].
- [5] Microsoft, “What is SQL Server?,” Microsoft, [Trực tuyến]. Available: <https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver16>. [Đã truy cập 04/04/2024].
- [6] wikipedia, “C Sharp (ngôn ngữ lập trình),” [Trực tuyến]. Available: [https://vi.wikipedia.org/wiki/C_Sharp_\(ngôn_ngữ_lập_trình\)](https://vi.wikipedia.org/wiki/C_Sharp_(ng%C3%B4n_ng%C3%B9_l%C3%A0p_tr%C3%ACnh)). [Đã truy cập 04/04/2024].
- [7] TopDev, “SQL Server là gì? SQL Server giúp bạn làm việc dễ dàng hơn?,” TopDev, [Trực tuyến]. Available: <https://topdev.vn/blog/sql-server-la-gi/#sql-server-la-gi>. [Đã truy cập 04/04/2024].
- [8] Dương Hồng, “10 Lợi Ích Của Việc Đi Du Lịch Chưa Ai Kể Bạn Nghe,” 03/21/2024. [Trực tuyến]. Available: <https://www.oreka.vn/blog/loi-ich-cua-viec-di-du-lich/>. [Đã truy cập 08/04/2024].
- [9] TopDev, “Postman là gì? API Testing với Postman,” TopDev, [Trực tuyến]. Available: <https://topdev.vn/blog/postman-la-gi/>. [Đã truy cập 07/05/2024].

- [10] Nguyen Van Truong, “Thuật toán quay lui (Backtracking),” 27/07/2017. [Trực tuyến]. Available: <https://viblo.asia/p/thuat-toan-quay-lui-backtracking-bJzKmLbD59N>. [Đã truy cập 07/05/2024].
- [11] Microsoft, “Getting Started with Bing Maps,” Microsoft, 23/04/2024. [Trực tuyến]. Available: <https://learn.microsoft.com/en-us/bingmaps/getting-started/>. [Đã truy cập 09/05/2024].
- [12] Microsoft, “Calculate a Distance Matrix,” Microsoft, 25/04/2024. [Trực tuyến]. Available: <https://learn.microsoft.com/en-us/bingmaps/rest-services/routes/calculate-a-distance-matrix>. [Đã truy cập 09/05/2024].
- [13] TopDev, “API là gì? Tại sao API được sử dụng nhiều hiện nay?,” TopDev, [Trực tuyến]. Available: <https://topdev.vn/blog/api-la-gi/>. [Đã truy cập 09/05/2024].
- [14] Trương Trường Thịnh, “API là gì? Giải đáp A – Z về giao diện lập trình ứng dụng,” 22/03/2024. [Trực tuyến]. Available: <https://interdata.vn/blog/api-la-gi/>. [Đã truy cập 09/05/2024].
- [15] wikipedia, “Haversine formula,” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Haversine_formula. [Đã truy cập 13/05/2024].
- [16] E. Galois, “Hàm số bị lãng quên: haversin,” 09/02/2017. [Trực tuyến]. Available: <https://diendantoanhoc.org/topic/170005-hàm-số-bị-lãng-quên-haversin/>. [Đã truy cập 13/05/2024].
- [17] wikipedia, “Ant colony optimization algorithms,” [Trực tuyến]. Available: https://en.wikipedia.org/wiki/Ant_colony_optimization_algorithms. [Đã truy cập 20/05/2024].
- [18] Công Hào, “Link source code,” Github, 06/05/2024. [Trực tuyến]. Available: https://github.com/hori2012/LocationTravel_DANTN_2024.