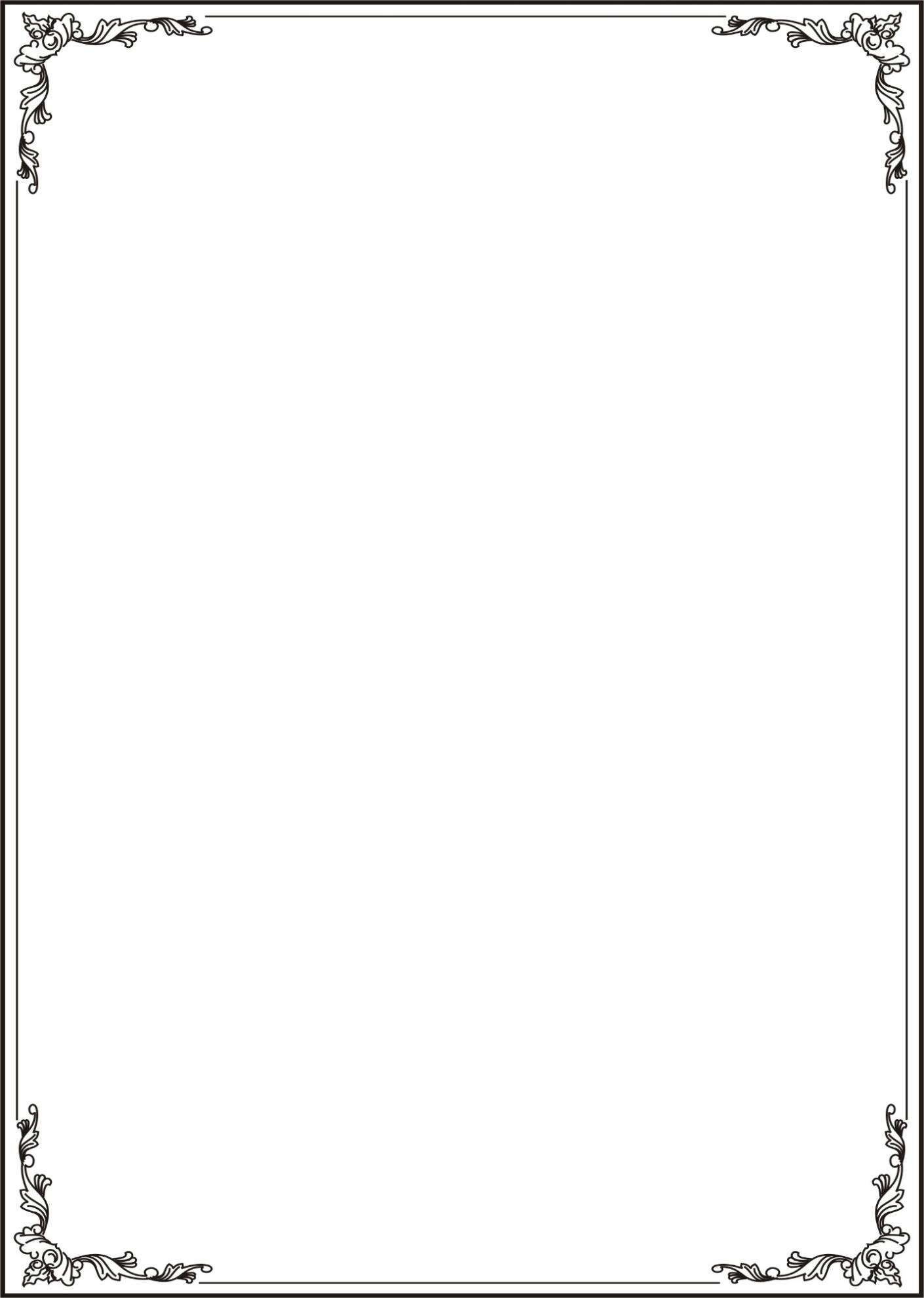
****

**KHOA CÔNG NGHỆ THÔNG TIN 1**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**



**Phương pháp luận và nghiên cứu khoa học**

**ĐỀ TÀI: XÂY DỰNG ỨNG DỤNG ĐẶT ĐỒ ĂN VỚI CẢI TIẾN TỐI ƯU HÓA TRẢI NGHIỆM NGƯỜI DÙNG**

Nhóm môn học :

Sinh viên thực hiện : Đặng Minh Đức

Mã sinh viên : B21DCCN236

Giảng viên hướng dẫn : Nguyễn Thị Kim Chi

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |

Hà Nội

# LỜI CẢM ƠN

Lời đầu tiên, em xin được gửi lời cảm ơn chân thành nhất đến thầy PTIT. Trong quá trình học tập và tìm hiểu môn Phát triển các hệ thống thông minh, em đã nhận được rất nhiều sự quan tâm, giúp đỡ, hướng dẫn tâm huyết và tận tình của thầy.

Em xin chân thành cảm ơn những người bạn đã luôn đồng hành, hỗ trợ và động viên trong suốt quá trình học tập vừa qua.

Em xin chân thành cảm ơn gia đình đã luôn bên cạnh, tạo điều kiện và khích lệ em trong quá trình thực hiện bài tập lớn này. Sự quan tâm và động viên từ người thân là nguồn động lực to lớn giúp em hoàn thành tốt nhiệm vụ.

Contents

[LỜI CẢM ƠN ii](#_Toc182666352)

[DANH SÁCH CÁC HÌNH VẼ, BẢNG BIỂU iv](#_Toc182666353)

[DANH SÁCH CÁC TỪ VIẾT TẮT v](#_Toc182666354)

[MỞ ĐẦU 1](#_Toc182666355)

[1. Giới thiệu 1](#_Toc182666356)

[2. Lý do chọn đề tài 1](#_Toc182666357)

[3. Mục tiêu nghiên cứu 2](#_Toc182666358)

[4. Đối tượng và phạm vi nghiên cứu 2](#_Toc182666359)

[Chương 1. Học sâu 3](#_Toc182666360)

[1.1 Giới thiệu về học sâu 3](#_Toc182666361)

[1.1.1 Lịch sử phát triển và tầm quan trọng của học sâu 3](#_Toc182666362)

[1.1.2. Ứng dụng học sâu trong dịch ngôn ngữ tự nhiên 4](#_Toc182666363)

[1.1.3. Sơ lược về các mô hình học sâu phổ biến 4](#_Toc182666364)

[1.2. Mạng nơ-ron hồi tiếp (RNN) và các biến thể 5](#_Toc182666365)

[1.2.1. Tổng quan về mạng nơ-ron hồi tiếp (Recurrent Neural Network - RNN) 5](#_Toc182666366)

[**1.2.2. Vấn đề trong RNN: Hiện tượng biến mất hoặc bùng nổ gradient** 7](#_Toc182666367)

[1.2.3. Các biến thể của RNN: GRU và LSTM 9](#_Toc182666368)

[1.3. LSTM 10](#_Toc182666369)

[1.3.1 Định nghĩa và khái niệm cơ bản: 10](#_Toc182666370)

[1.3.2. Cấu trúc và cơ chế hoạt động của LSTM 11](#_Toc182666371)

[1.3.3. Quy trình hoạt động chi tiết của LSTM 12](#_Toc182666372)

[1.3.4. Ưu điểm và tính năng nổi bật của LSTM 13](#_Toc182666373)

[1.3.5. Ứng dụng cụ thể của LSTM trong thực tế 13](#_Toc182666374)

[1.3.6. Những thách thức và hướng phát triển 14](#_Toc182666375)

[1.4. Ứng dụng của LSTM trong dịch ngôn ngữ tự nhiên 14](#_Toc182666376)

[1.4.1. Tổng quan về dịch ngôn ngữ tự nhiên 14](#_Toc182666377)

[1.4.2. Vai trò của LSTM trong dịch ngôn ngữ tự nhiên 15](#_Toc182666378)

[1.4.3. Cách LSTM cải thiện chất lượng dịch 16](#_Toc182666379)

[1.4.4. Các ứng dụng thực tế của LSTM trong dịch ngôn ngữ tự nhiên 17](#_Toc182666380)

[1.4.5. Những thách thức và hạn chế của LSTM trong dịch ngôn ngữ tự nhiên 17](#_Toc182666381)

[1.4.6. Hướng phát triển và cải tiến trong tương lai 17](#_Toc182666382)

[1.4. Kết luận 18](#_Toc182666383)

[Chương 2. Dữ liệu và EDA 19](#_Toc182666384)

[2.1. Bộ dữ liệu 19](#_Toc182666385)

[2.1.1. Giới thiệu về bộ dữ liệu 19](#_Toc182666386)

[2.1.2. Mô tả các biến trong bộ dữ liệu 19](#_Toc182666387)

[2.1.3. Kết quả 19](#_Toc182666388)

[2.1.4. Thống kê mô tả 19](#_Toc182666389)

[2.2. Hình vẽ và phân tích các thông số. 20](#_Toc182666390)

[2.2.1. Phân tích số lượng câu 20](#_Toc182666391)

[2.2.2. Phân tích độ dài câu 20](#_Toc182666392)

[2.2.3. Phân tích tần suất từ vựng 21](#_Toc182666393)

[2.3. Phân tích EDA: 23](#_Toc182666394)

[2.3.1. Độ đa dạng của cụm từ dịch 23](#_Toc182666395)

[2.3.2. Phân tích độ dài câu 23](#_Toc182666396)

[2.3.3. Phân tích mối quan hệ giữa độ dài câu tiếng Anh và tiếng Việt 25](#_Toc182666397)

[2.3.4. Phát hiện giá trị ngoại lai 26](#_Toc182666398)

[2.3.5. Kết luận EDA 27](#_Toc182666399)

[2.4. Kết luận: 27](#_Toc182666400)

[Chương 3. Thực nghiệm 28](#_Toc182666401)

[3.1. Mô hình 28](#_Toc182666402)

[3.1.1. Mô hình LSTM 28](#_Toc182666403)

[3.1.2. Xử lý dữ liệu 28](#_Toc182666404)

[3.1.3. Huấn luyện mô hình 30](#_Toc182666405)

[3.2. Thực nghiệm và kết quả. 31](#_Toc182666406)

[3.2.1. Xử lý dữ liệu 31](#_Toc182666407)

[3.2.2. Huấn luyện mô hình: 34](#_Toc182666408)

[3.2.3. Kết quả thực nghiệm: 37](#_Toc182666409)

[3.2.4. Đánh giá mô hình: 38](#_Toc182666410)

[3.3 Kết luận 38](#_Toc182666411)

[KẾT LUẬN 40](#_Toc182666412)

[1. Kết quả đạt được: 40](#_Toc182666413)

[2. Phương hướng phát triển: 40](#_Toc182666414)

[Tài liệu tham khảo 42](#_Toc182666415)

# DANH SÁCH CÁC HÌNH VẼ, BẢNG BIỂU

[Hình 1.1. Sơ đồ mạng RNN 6](#_Toc181749307)

[Hình 1.2. Hình ảnh minh họa sự biến mất và bùng nổ gradient 8](#_Toc181749308)

[Hình 1.3. Hình ảnh tổng quan về LSTM 11](#_Toc181749309)

[Hình 1.4. Hình ảnh minh họa cơ chế hoạt động của LSTM 12](#_Toc181749310)

[Hình 1.5. Sơ đồ kiến trúc encoder – decoder của LSTM 16](#_Toc181749311)

[Hình 2.1. Biểu đồ tần suất của 10 từ tiếng việt phổ biến nhất 22](#_Toc181749312)

[Hình 2.2. Biểu đồ tần suất của 10 từ tiếng anh phổ biến nhất 23](#_Toc181749313)

[Hình 2.3. Biểu đồ phân phối độ dài câu tiếng anh 24](#_Toc181749314)

[Hình 2.4. Biểu đồ phân phối độ dài câu tiếng việt 25](#_Toc181749315)

[Hình 2.5. Biểu đồ sự tương quan giữa độ dài câu tiếng việt và tiếng anh 26](#_Toc181749316)

[Bảng 3.1. Bảng kết quả thực nghiệm của mô hình 37](#_Toc181749317)

# DANH SÁCH CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Tiếng Anh** | **Ý nghĩa** |
| AI | Artificial Intelligence | Trí tuệ nhân tạo |
| BERT | Bidirectional Encoder Representations from Transformers | Mô hình biến thể mã hóa hai chiều từ Transformer |
| CNN | Convolutional Neural Networks | Mạng nơ-ron tích chập |
| EDA | Exploratory Data Analyst | Khai phá dữ liệu |
| GAN | Generative Adversarial Networks | Mạng chống đối tạo sinh |
| GPT | Generative Pre-trained Transformer | Mô hình biến thể sinh tạo được huấn luyện trước |
| GPU | Graphics Processing Unit | Bộ xử lý đồ họa |
| GRU | Gated Recurrent Unit | Mạng nơ-ron với nút hồi tiếp có cổng |
| LSTM | Long Short-Term Memory | Bộ nhớ dài ngắn hạn |
| NLP | **Natural Language Translation** | Dịch ngôn ngữ tự nhiên |
| RNN | Recurrent Neural Networks | Mạng hồi quy |
| SGD | Stochastic Gradient Descent | Giảm độ dốc ngẫu nhiên |
| SMT | Statistical Machine Translation | Dịch máy thông kê |

# MỞ ĐẦU

## Giới thiệu

Cùng với sự phát triển vượt bậc của công nghệ và sự thay đổi trong thói quen tiêu dùng, các ứng dụng đặt đồ ăn trực tuyến đã trở thành công cụ thiết yếu, mang lại sự tiện lợi và tiết kiệm thời gian cho hàng triệu người dùng trên toàn thế giới. Những ứng dụng này không chỉ đóng vai trò là cầu nối giữa khách hàng và nhà hàng mà còn tạo nên một hệ sinh thái phong phú giúp các nhà hàng tối ưu hóa hoạt động kinh doanh.

Tuy nhiên, thị trường ứng dụng đặt đồ ăn hiện nay đang đối mặt với nhiều thách thức. Khách hàng ngày càng có yêu cầu cao hơn về tính cá nhân hóa và sự phù hợp trong gợi ý món ăn. Đồng thời, các nhà hàng cần có những công cụ hỗ trợ hiệu quả hơn để quản lý doanh thu và ra quyết định chiến lược. Việc ứng dụng các công nghệ tiên tiến như *Collaborative Filtering* và tích hợp các API về thời tiết sẽ mang lại giải pháp tối ưu, cải thiện trải nghiệm người dùng và nâng cao hiệu quả kinh doanh.

Đề tài này tập trung xây dựng một ứng dụng đặt đồ ăn với các tính năng mới, bao gồm gợi ý sản phẩm dựa trên sở thích của từng người dùng, tích hợp điều kiện thời tiết trong quá trình gợi ý món ăn, và cung cấp giao diện thống kê doanh thu trực quan cho các nhà hàng đối tác.

## 2. Lý do chọn đề tài

### 2.1. Xu hướng phát triển của ứng dụng đặt đồ ăn

Trong những năm gần đây, ứng dụng đặt đồ ăn trực tuyến đã phát triển mạnh mẽ với các tên tuổi lớn như GrabFood, Baemin, ShopeeFood. Những ứng dụng này không chỉ cung cấp dịch vụ giao đồ ăn mà còn mang lại sự tiện lợi vượt trội trong đời sống hàng ngày. Tuy nhiên, cạnh tranh trong lĩnh vực này ngày càng khốc liệt, và để tạo được sự khác biệt, các ứng dụng cần phải không ngừng cải tiến và đổi mới. Một trong những hướng đi quan trọng là tập trung vào trải nghiệm người dùng cá nhân hóa, giúp đáp ứng chính xác hơn nhu cầu của từng khách hàng.

### 2.2. Nhu cầu cá nhân hóa trải nghiệm người dùng

Các nghiên cứu cho thấy, người dùng có xu hướng quay lại các nền tảng cung cấp trải nghiệm phù hợp với sở thích và hành vi cá nhân. Tuy nhiên, phần lớn các ứng dụng đặt đồ ăn hiện nay vẫn chỉ dựa vào các danh mục tĩnh hoặc đề xuất không chính xác. Việc áp dụng Collaborative Filtering có thể giúp giải quyết vấn đề này, thông qua phân tích dữ liệu và đưa ra các gợi ý dựa trên thói quen mua hàng của người dùng tương tự.

### 2.3. Ảnh hưởng của điều kiện thời tiết đến thói quen ăn uống

Thời tiết là một yếu tố ảnh hưởng lớn đến quyết định đặt món ăn của người dùng. Ví dụ, trong ngày mưa lạnh, các món ăn nóng như lẩu, mì có xu hướng được ưa chuộng hơn. Ngược lại, vào những ngày hè nóng bức, các món giải nhiệt như kem, trà sữa thường được đặt nhiều hơn. Tích hợp API thời tiết để điều chỉnh gợi ý sản phẩm theo thời gian thực không chỉ làm tăng sự hài lòng của người dùng mà còn tạo nên sự khác biệt so với các đối thủ.

### 2.4. Hỗ trợ nhà hàng tối ưu hóa doanh thu

Ngoài việc tối ưu hóa trải nghiệm cho khách hàng, việc cung cấp các công cụ hỗ trợ nhà hàng quản lý hiệu quả hơn cũng là một yếu tố quan trọng. Thống kê doanh thu bằng biểu đồ không chỉ giúp nhà hàng theo dõi tình hình kinh doanh mà còn đưa ra các dự báo hữu ích, hỗ trợ việc ra quyết định.

### 2.5. Ý nghĩa thực tiễn

Đề tài này không chỉ mang lại lợi ích cho người dùng cá nhân mà còn hỗ trợ nhà hàng cải thiện hoạt động kinh doanh, từ đó góp phần vào sự phát triển chung của ngành dịch vụ giao nhận thực phẩm trực tuyến. Với sự kết hợp giữa công nghệ tiên tiến và phân tích dữ liệu, đề tài có tiềm năng ứng dụng cao trong thực tế.

## 3. Mục đích nghiên cứu

Đề tài tập trung vào việc xây dựng một ứng dụng đặt đồ ăn trực tuyến với những cải tiến nhằm tối ưu hóa trải nghiệm người dùng và hỗ trợ nhà hàng kinh doanh hiệu quả. Mục đích cụ thể bao gồm:

* Tối ưu hóa trải nghiệm người dùng cá nhân hóa: Ứng dụng sử dụng Collaborative Filtering để phân tích hành vi người dùng, từ đó gợi ý các món ăn hoặc nhà hàng phù hợp với sở thích và thói quen của từng cá nhân.
* Khai thác dữ liệu thời tiết để tăng tính thông minh trong đề xuất: Tích hợp API thời tiết để đưa ra các gợi ý món ăn phù hợp với điều kiện thời tiết hiện tại, nâng cao sự hài lòng và tính tiện ích.
* Cải thiện công cụ hỗ trợ nhà hàng: Phát triển tính năng thống kê doanh thu trực quan qua biểu đồ, giúp nhà hàng theo dõi hiệu quả kinh doanh, nhận biết xu hướng đặt món, và đưa ra quyết định chiến lược.

## 4. Đối tượng và phạm vi nghiên cứu

### 4.1. Đối tượng nghiên cứu

Bao gồm:

* Người dùng: Là những cá nhân sử dụng ứng dụng đặt đồ ăn trực tuyến, bao gồm các nhóm người dùng phổ biến như sinh viên, nhân viên văn phòng, hoặc các hộ gia đình bận rộn.
* Dữ liệu: Dữ liệu về hành vi người dùng (như lịch sử đặt món, lượt đánh giá các món ăn), thông tin món ăn, và dữ liệu thời tiết theo thời gian thực.
* Nhà hàng: Các nhà hàng sử dụng ứng dụng để cung cấp dịch vụ và mong muốn tăng hiệu quả kinh doanh thông qua các công cụ hỗ trợ.

### 4.2. Phạm vi nghiên cứu

Phạm vi nội dung:

* Xây dựng một ứng dụng mẫu hỗ trợ đặt đồ ăn trực tuyến.
* Tích hợp các thuật toán gợi ý sản phẩm dựa trên Collaborative Filtering.
* Kết nối API thời tiết để gợi ý món ăn theo điều kiện thời tiết.
* Phát triển tính năng thống kê doanh thu qua biểu đồ cho nhà hàng.

Phạm vi kỹ thuật:

* Nghiên cứu và áp dụng các thuật toán Collaborative Filtering.
* Sử dụng công nghệ API thời tiết (như OpenWeather hoặc AccuWeather) để thu thập dữ liệu.
* Phát triển giao diện trực quan và trải nghiệm người dùng dựa trên Flutter.

Phạm vi ứng dụng thực tiễn:

* Ứng dụng thử nghiệm sẽ tập trung vào nhóm người dùng tại một thành phố hoặc khu vực cụ thể để kiểm tra tính hiệu quả.
* Dữ liệu được thu thập giới hạn trong khoảng thời gian từ 3-6 tháng.

# CHƯƠNG 1. TỔNG QUAN VỀ ỨNG DỤNG

## Tổng quan về ứng dụng đặt đồ ăn

### 1.1.1. Khái niệm và vai trò của ứng dụng đặt đồ ăn

Ứng dụng đặt đồ ăn là một nền tảng công nghệ cho phép người dùng lựa chọn, đặt món ăn và nhận giao hàng từ các nhà hàng, quán ăn thông qua thiết bị di động hoặc website. Đây là một phần quan trọng trong ngành công nghệ dịch vụ ăn uống, kết hợp giữa thương mại điện tử và logistics.

Ứng dụng đặt đồ ăn là một giải pháp công nghệ số hóa việc đặt món, giúp kết nối người dùng với nhà hàng một cách nhanh chóng, tiện lợi. Ứng dụng thường cung cấp các tính năng như xem món, đặt món online , theo dõi đơn hàng, và đánh giá nhà hàng.

Vai trò:

1. Đối với người dùng:

* Tiết kiệm thời gian và công sức khi không cần đến tận nơi để mua món ăn.
* Cung cấp sự lựa chọn đa dạng về nhà hàng, món ăn phù hợp với sở thích cá nhân.
* Đem lại trải nghiệm tiện ích qua các tính năng như gợi ý món ăn, ưu đãi giảm giá, theo dõi thời gian giao hàng.

1. Đối với nhà hàng:

* Mở rộng thị trường tiếp cận đến lượng khách hàng lớn hơn, bao gồm cả những khách hàng không thể đến quán trực tiếp.
* Giảm tải chi phí vận hành bằng cách tích hợp với hệ thống giao hàng của bên thứ ba.
* Thu thập dữ liệu khách hàng để cải thiện dịch vụ và xây dựng chiến lược kinh doanh.

1. Đối với xã hội:

* Tăng cường sự linh hoạt trong việc cung cấp dịch vụ ăn uống, đặc biệt hữu ích trong các tình huống như đại dịch hoặc phong tỏa.
* Thúc đẩy sự phát triển của các công nghệ liên quan như thanh toán trực tuyến, AI trong đề xuất món ăn, và quản lý chuỗi cung ứng.

### 1.1.2. Cấu trúc tổng quát của ứng dụng đặt đồ ăn

Ứng dụng đặt đồ ăn thường bao gồm 3 thành phần chính:

1. Thành phần phía người dùng :

* Giao diện người dùng (UI): Thiết kế thân thiện, dễ sử dụng, hỗ trợ các chức năng như tìm kiếm nhà hàng, chọn món, và đặt hàng.
* Quản lý tài khoản: Bao gồm đăng nhập, đăng ký, quản lý thông tin cá nhân, và địa chỉ giao hàng.
* Đặt hàng và thanh toán: Tích hợp các phương thức thanh toán như thẻ tín dụng, ví điện tử, hoặc thanh toán khi nhận hàng.
* Theo dõi đơn hàng: Hiển thị trạng thái đơn hàng, thời gian dự kiến giao, và thông tin người giao hàng.

1. Thành phần phía nhà hàng:

* Quản lý menu: Hỗ trợ nhà hàng thêm, xóa, cập nhật món ăn và giá cả.
* Quản lý đánh giá: Cung cấp thông tin phản hồi từ khách hàng để cải thiện dịch vụ.
* Thống kê về doanh thu theo ngày, tháng, năm.

1. Thành phần phía người giao hàng:

* Quản lý đơn hàng: Tích hợp Google Map đề xuất các đơn hàng thuận lợi đối với người giao hàng. Hỗ trợ quản lý thông tin đơn hàng và giao tiếp với người đặt hàng.
* Quản lý đánh giá: Cung cấp thông tin phản hồi từ khách hàng để cải thiện dịch vụ.

1. Thành phần phía máy chủ:

* Quản lý dữ liệu: Xử lý thông tin từ người dùng, nhà hàng và hệ thống giao hàng.
* Quản lý dữ liệu với trang Admin: Quản lý dữ liệu với giao diện dễ học và sử dụng cho những người quản trị viên hệ thống.
* Hệ thống giao hàng: Kết nối với các dịch vụ vận chuyển để đảm bảo thời gian giao hàng nhanh chóng.
* Hệ thống đề xuất sản phẩm: Tích hợp trí Collaborative Filtering để gợi ý món ăn dựa trên thói quen của người dùng.
* Bảo mật: Đảm bảo an toàn dữ liệu người dùng và giao dịch tài chính thông qua các giao thức mã hóa.

## 1.2. Tổng quan về trải nghiệm người dùng và đề xuất sản phẩm

Trải nghiệm người dùng (UX) trong ứng dụng đặt đồ ăn:

* Khái niệm: Trải nghiệm người dùng tập trung vào cảm giác, sự hài lòng và sự tiện ích mà người dùng có được khi sử dụng ứng dụng.
* Yếu tố quan trọng:

1. Hiệu năng: Ứng dụng cần hoạt động nhanh, mượt mà, không bị gián đoạn trong các thao tác đặt hàng.
2. Sự trực quan: Thiết kế giao diện phải dễ hiểu, phù hợp với mọi lứa tuổi.
3. Cá nhân hóa: Hiển thị nội dung phù hợp với sở thích và lịch sử mua sắm của người dùng.
4. Tương tác đa kênh: Hỗ trợ đồng bộ giữa ứng dụng di động, website, và các nền tảng khác.

Đề xuất sản phẩm:

* Khái niệm: Đề xuất sản phẩm là việc gợi ý món ăn, nhà hàng hoặc các ưu đãi phù hợp với sở thích của từng cá nhân dựa trên dữ liệu thu thập được.
* Các kỹ thuật đề xuất sản phẩm:

1. Rule-based recommendation: Gợi ý dựa trên các quy tắc cố định, ví dụ như món ăn phổ biến nhất hoặc ưu đãi hiện hành.
2. Collaborative Filtering:
   * User-based: Dựa trên hành vi của những người dùng có sở thích tương tự.
   * Item-based: Dựa trên các món ăn có tính chất tương đồng như được đánh giá qua các lần đặt hàng.
3. Content-based recommendation: Gợi ý dựa trên đặc điểm món ăn mà người dùng đã chọn hoặc đánh giá tốt.
4. Hybrid recommendation: Kết hợp nhiều phương pháp để tăng độ chính xác và sự linh hoạt.

Vai trò của đề xuất sản phẩm trong trải nghiệm người dùng:

* Cải thiện tính cá nhân hóa, giúp người dùng dễ dàng tìm thấy món ăn yêu thích mà không cần mất nhiều thời gian tìm kiếm.
* Tăng doanh thu cho nhà hàng thông qua việc gợi ý các món ăn mới hoặc các combo khuyến mãi hấp dẫn.
* Giữ chân người dùng bằng cách tạo trải nghiệm thú vị và tiện ích hơn.

## 1.3. Collaborative Filtering và ứng dụng trong đề xuất sản phẩm

### 1.3.1. Phương pháp lọc cộng tác User-based và Item-based

Collaborative Filtering (CF) là một trong những kỹ thuật phổ biến nhất trong hệ thống gợi ý, sử dụng dữ liệu lịch sử của người dùng để đưa ra các đề xuất cá nhân hóa. CF hoạt động dựa trên ý tưởng rằng nếu người dùng có hành vi hoặc sở thích giống nhau, họ sẽ có khả năng quan tâm đến các sản phẩm tương tự.

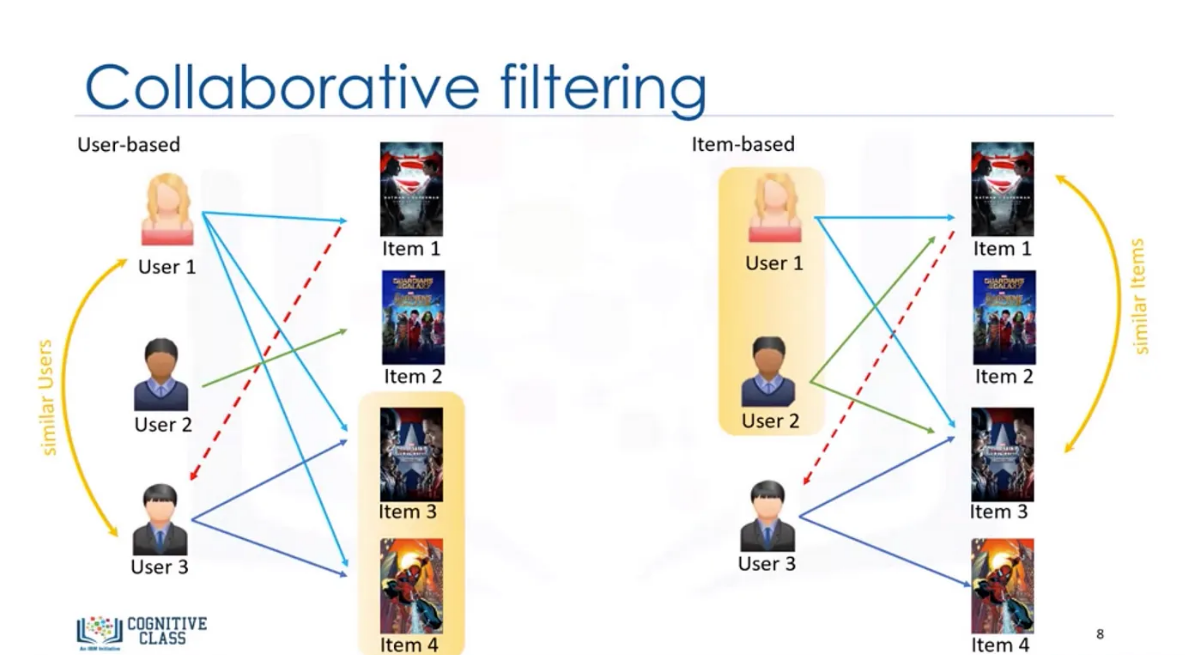
Phương pháp lọc cộng tác được chia thành hai loại chính:

1. User-based Collaborative Filtering:

* Dựa trên hành vi của người dùng. Nếu hai người dùng có lịch sử tương tác giống nhau (ví dụ: cùng đánh giá cao một món ăn), hệ thống sẽ đề xuất các sản phẩm mà một trong hai người chưa khám phá.
* Ví dụ: Người dùng A và B đều thích món "Pizza Pepperoni". Nếu A đã thử món "Burger BBQ" và đánh giá cao, thì hệ thống sẽ gợi ý món này cho B.
* Công thức phổ biến:  
  Hệ thống sử dụng các thuật toán như Cosine Similarity hoặc Pearson Correlation để đo độ tương đồng giữa các người dùng dựa trên ma trận đánh giá.

1. Item-based Collaborative Filtering:

* Dựa trên sự tương đồng giữa các sản phẩm. Nếu hai sản phẩm được đánh giá tích cực bởi cùng nhóm người dùng, chúng sẽ được xem là tương tự nhau.
* Ví dụ:
* Món "Mì Ý sốt bò bằm" và "Mì Ý sốt hải sản" được nhiều người dùng đánh giá cao cùng lúc. Hệ thống sẽ đề xuất món còn lại khi người dùng chọn một trong hai món này.
* Công thức phổ biến:  
  Hệ thống dựa vào ma trận tương tác giữa sản phẩm và người dùng, sau đó tính toán độ tương đồng giữa các sản phẩm bằng Jaccard Similarity hoặc Adjusted Cosine Similarity.



#### Hình 1.0.1 Mô tả về CF

***1.3.2.*** ***Minh họa quy trình Collaborative Filtering***

Trong Collaborative Filtering dựa trên người dùng, ta có một người dùng mục tiêu (active user), là người mà ta muốn tạo gợi ý. Hệ thống Collaborative Filtering sẽ tìm kiếm những người dùng có mức độ tương đồng cao với người dùng mục tiêu, tức là những người có mô hình đánh giá sản phẩm giống nhau. Mức độ tương đồng này được tính toán dựa trên lịch sử, sở thích, và lựa chọn mà người dùng thực hiện khi mua sắm, xem, hoặc sử dụng một sản phẩm nào đó.

Ví dụ: Nếu một người dùng A và người dùng B đều thích các món ăn X, Y, thì nếu A thích món ăn Z, hệ thống có thể gợi ý món ăn Z cho B.

1. Đối với User-based:

Cách hoạt động

* Bước 1: Giả sử ta có một ma trận người dùng - sản phẩm (user-item matrix), biểu diễn điểm đánh giá của 4 người dùng cho 5 món ăn. Người dùng mục tiêu chỉ đánh giá 3 trong số 5 món ăn này.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| User/Món ăn | Món ăn A | Món ăn B | Món ăn C | Món ăn D | Món ăn E |
| User 1 | 5 | 3 | 0 | 0 | 1 |
| User 2 | 4 | 0 | 0 | 2 | 5 |
| User 3 | 0 | 2 | 4 | 5 | 0 |
| User 4 | 1 | 1 | 0 | 0 | 4 |

* Bước 2: Tìm mức độ tương đồng giữa người dùng mục tiêu và những người khác bằng cách sử dụng các kỹ thuật như:
  + Khoảng cách Euclidean (Euclidean Distance).
  + Hệ số tương quan Pearson (Pearson Correlation).
  + Độ tương tự Cosine (Cosine Similarity)..
  + Ví dụ tính toán dựa trên bảng trên ta được:

|  |  |  |  |
| --- | --- | --- | --- |
| User | | Độ Tương Đồng với User 1 | | --- |  |  | | --- | |  | |
| User 2 | 0.8 |
| User 3 | 0.7 |
| User 4 | 0.6 |

* Bước 3: Dự đoán điểm đánh giá cho các món ăn chưa thử. Sau khi có độ tương đồng, hệ thống sẽ tính toán điểm dự đoán cho Món Ăn C và Món Ăn D mà User 1 chưa đánh giá.
* Dự đoán cho Món Ăn C: Cộng điểm đánh giá của các người dùng tương tự, có trọng số theo độ tương đồng:
* Dự đoán cho Món Ăn D: Tương tự, ta tính toán cho Món Ăn D:
* Bước 4: Gợi ý món ăn
* Dựa trên các dự đoán này, hệ thống sẽ gợi ý cho User 1 các món ăn mà người này chưa thử, nhưng có thể sẽ thích, ví dụ như:
  + Món Ăn D (dự đoán điểm: 3.5)
  + Món Ăn C (dự đoán điểm: 2.8)
* Như vậy, User 1 có thể được gợi ý thử Món Ăn D vì dự đoán điểm đánh giá khá cao.

1. Đối với Item-based:

Gần như tương đồng với User-based khác ở bước 2 với ma trận tương đồng sẽ là giữa món ăn với món ăn:

|  |  |  |  |
| --- | --- | --- | --- |
| Món ăn | | Độ Tương Đồng với Món ăn 1 | | --- |  |  | | --- | |  | |
| Món ăn 1 | … |
| Món ăn 2 | … |
| Món ăn 3 | … |

***1.3.3. Ưu điểm và hạn chế của Collaborative Filtering***

Ưu điểm:

1. Không yêu cầu kiến thức về sản phẩm:

* Hệ thống không cần thông tin chi tiết về món ăn (ví dụ: thành phần hoặc cách chế biến). Thay vào đó, nó chỉ dựa vào dữ liệu tương tác như đánh giá, lịch sử đặt hàng.
* Điều này giúp CF dễ dàng mở rộng ra các lĩnh vực khác nhau, từ sách, phim , món ăn, ảnh đến thời trang và thực phẩm.

1. Khả năng cá nhân hóa cao:

* Đề xuất dựa trên hành vi thực tế của người dùng, giúp tăng khả năng cung cấp các gợi ý phù hợp.

1. Khả năng phát hiện sở thích tiềm ẩn:

* CF có thể gợi ý các món ăn mà người dùng chưa từng thử nhưng phù hợp với sở thích của họ.

Hạn chế:

1. Vấn đề Cold Start:

* Đối với người dùng mới (chưa có lịch sử tương tác) hoặc món ăn mới (chưa có đánh giá), hệ thống gặp khó khăn trong việc đưa ra gợi ý.
* Giải pháp: Kết hợp với Content-based Filtering hoặc thu thập thêm thông tin từ người dùng thông qua bảng khảo sát sở thích.
* Dữ liệu thưa thớt (Data Sparsity):
* Trong trường hợp có quá nhiều sản phẩm và người dùng, ma trận tương tác trở nên rất lớn nhưng lại chứa nhiều giá trị rỗng (người dùng không đánh giá hầu hết sản phẩm). Điều này làm giảm hiệu quả của thuật toán CF.

1. Vấn đề Popularity Bias:

* Hệ thống có xu hướng ưu tiên các món ăn phổ biến, khiến các món mới hoặc ít được đặt trở nên khó tiếp cận hơn.

1. Hiệu suất tính toán:

* Khi số lượng người dùng và món ăn tăng lên, thuật toán CF đòi hỏi tài nguyên tính toán lớn để tìm kiếm và so sánh độ tương đồng.

## 1.4. Tổng quan về tích hợp API thời tiết

API thời tiết cung cấp dữ liệu thời tiết theo thời gian thực từ các nguồn đáng tin cậy (như OpenWeatherMap, WeatherStack). API này thường bao gồm các thông tin như:

* Nhiệt độ.
* Độ ẩm.
* Dự báo thời tiết (mưa, nắng, gió).
* Điều kiện khí hậu theo từng khu vực cụ thể.

Ứng dụng API thời tiết trong hệ thống Food-Delivery-App

1. Cá nhân hóa đề xuất món ăn:

* Dữ liệu thời tiết có thể được sử dụng để gợi ý món ăn phù hợp với điều kiện hiện tại.
* Ví dụ:
  + Trời lạnh: Gợi ý các món ăn nóng như lẩu, súp.
  + Trời nóng: Gợi ý đồ uống giải khát hoặc món ăn nhẹ.

1. Tăng cường trải nghiệm người dùng:

* Thời tiết thường ảnh hưởng đến tâm trạng và lựa chọn của khách hàng. Bằng cách tích hợp thời tiết vào hệ thống, ứng dụng có thể cung cấp trải nghiệm "thấu hiểu" người dùng hơn.

Quy trình tích hợp API thời tiết

1. Lựa chọn nhà cung cấp API:

* Các nhà cung cấp phổ biến: OpenWeatherMap, WeatherStack, AccuWeather.
* Tiêu chí lựa chọn: Độ chính xác, khả năng mở rộng, chi phí, và tài liệu hỗ trợ.

1. Kết nối API:

* Tích hợp API thông qua các giao thức HTTP/HTTPS.
* Gửi yêu cầu (request) chứa thông tin địa điểm của người dùng (ví dụ: tọa độ GPS hoặc mã zip).

1. Xử lý dữ liệu:

* Hệ thống nhận phản hồi (response) từ API dưới dạng JSON/XML, sau đó trích xuất thông tin cần thiết (ví dụ: nhiệt độ, thời tiết hiện tại).

1. Ap dụng:

* Dựa vào dữ liệu thời tiết của từng khu vực đối với từng người dùng từ đó tích hợp với món ăn để đưa ra đề xuất đối với người dùng.

Ưu và nhược điểm của tích hợp API thời tiết

Ưu điểm:

* Cập nhật theo thời gian thực: Đảm bảo dữ liệu luôn chính xác.
* Tăng tính cá nhân hóa: Gợi ý sản phẩm theo điều kiện thời tiết của từng khu vực.

Nhược điểm:

* Phụ thuộc vào dịch vụ bên thứ ba: Hệ thống có thể gặp gián đoạn nếu API gặp lỗi hoặc bị giới hạn tài nguyên.
* Chi phí: Một số nhà cung cấp API yêu cầu phí cao cho các tính năng nâng cao hoặc dung lượng lớn.

# Chương 2. XÂY DỰNG ỨNG DỤNG

## 2.1. Thiết kế hệ thống

Việc thiết kế một hệ thống cho ứng dụng đặt đồ ăn yêu cầu sự kết hợp giữa các yếu tố kỹ thuật để đảm bảo hiệu suất, khả năng mở rộng, và tối ưu hóa trải nghiệm người dùng. Trong chương này, chúng tôi sẽ trình bày chi tiết về thiết kế hệ thống của ứng dụng đặt đồ ăn, sử dụng các công nghệ tiên tiến để đảm bảo hiệu quả và tính linh hoạt của hệ thống. Các công nghệ chính bao gồm Dart, Flutter, Python, Django Rest Framework, cùng với các thư viện và plugin như Flutter Google Maps, GetX, và các công cụ hỗ trợ khác.

### 2.1.1. Kiến trúc hệ thống

Hệ thống ứng dụng Food-Delivery-App được thiết kế theo mô hình Client-Server, trong đó:

* Client (Ứng dụng di động): Được phát triển bằng Flutter, sử dụng Dart làm ngôn ngữ lập trình chính. Ứng dụng hoạt động Android, giúp người dùng có thể dễ dàng truy cập vào dịch vụ đặt đồ ăn từ điện thoại di động của mình.
* Server (Backend): Dựa trên Python và Django Rest Framework để phát triển các API phục vụ cho việc xử lý dữ liệu, quản lý người dùng, xử lý đơn hàng và tích hợp với các mô hình học máy (machine learning). Python được sử dụng để huấn luyện và triển khai mô hình Collaborative Filtering, giúp đưa ra các gợi ý về món ăn cho người dùng dựa trên thói quen và sở thích.

Cả client và server sẽ giao tiếp thông qua RESTful API, giúp kết nối các phần tử trong hệ thống với nhau. Dữ liệu được lưu trữ trong cơ sở dữ liệu PostgreSQL, một lựa chọn phổ biến cho các ứng dụng web vì tính ổn định và khả năng mở rộng.

### 2.1.2. Công nghệ sử dụng

Flutter & Dart: Được chọn làm công nghệ chính cho phần client vì khả năng phát triển ứng dụng cross-platform nhanh chóng và dễ dàng. Flutter cung cấp một framework mạnh mẽ để xây dựng giao diện người dùng đẹp mắt, với khả năng tương thích cao trên các thiết bị di động. Dart là ngôn ngữ lập trình chính, mang lại hiệu suất cao và khả năng lập trình bất đồng bộ, điều này rất hữu ích khi xử lý các tác vụ như tải dữ liệu từ API hay hiển thị bản đồ.

Google Maps & Direction API: Google Maps được tích hợp để cung cấp bản đồ cho người dùng và các nhà hàng, cho phép người dùng tìm kiếm và xác định vị trí các nhà hàng xung quanh. Google Directions API giúp tính toán và hiển thị lộ trình di chuyển của người giao hàng từ nhà hàng đến người nhận.

GetX: Là một thư viện quản lý trạng thái cho Flutter, GetX được sử dụng để cải thiện hiệu suất và đơn giản hóa quá trình quản lý trạng thái của ứng dụng. Thư viện này giúp việc xây dựng các tính năng như điều hướng, xử lý sự kiện và quản lý trạng thái người dùng trở nên dễ dàng và hiệu quả.

Python & Django Rest Framework: Backend của hệ thống được phát triển với Python, một ngôn ngữ lập trình phổ biến trong lĩnh vực phát triển web và học máy. Django Rest Framework được sử dụng để xây dựng API, phục vụ việc quản lý dữ liệu người dùng, các đơn hàng, cũng như tương tác với các mô hình học máy. Python cũng được sử dụng để huấn luyện và triển khai mô hình Collaborative Filtering, một phương pháp học máy phổ biến trong việc đưa ra các gợi ý sản phẩm dựa trên hành vi của người dùng.

Collaborative Filtering: Đây là kỹ thuật học máy được sử dụng để cá nhân hóa các đề xuất món ăn cho người dùng dựa trên các tương tác trước đó của họ hoặc dựa trên sở thích của những người dùng tương tự. Các mô hình collaborative filtering có thể là user-based hoặc item-based, tùy thuộc vào cách thức dữ liệu được xử lý và khai thác.

### 2.1.3. Quy trình hoạt động của hệ thống

Hệ thống của ứng dụng hoạt động qua ba bước chính:

* Tương táccủa người dùng: Người dùng tải ứng dụng, đăng nhập hoặc đăng ký tài khoản và có thể tìm kiếm các món ăn hoặc nhà hàng trong khu vực. Các thông tin như món ăn yêu thích, lịch sử đặt hàng và vị trí của người dùng sẽ được hệ thống ghi nhận và sử dụng để cải thiện trải nghiệm người dùng.
* Hệ thống đề xuất: Dựa trên các dữ liệu thu thập được từ người dùng (bao gồm các món ăn đã đặt trước, đánh giá món ăn) hệ thống sẽ sử dụng Collaborative Filtering để đề xuất các món ăn phù hợp. Mô hình này sẽ được huấn luyện từ dữ liệu lịch sử của người dùng và cập nhật theo thời gian để ngày càng chính xác hơn.
* Xử lý đơn hàng và giao hàng: Khi người dùng lựa chọn món ăn và đặt hàng, hệ thống sẽ gửi yêu cầu đến backend để xác nhận đơn hàng, tính toán phí vận chuyển dựa trên khoảng cách, và cung cấp lộ trình giao hàng thông qua Google Directions API, sau đó gửi yêu cầu đến người giao hàng. Sau khi đơn hàng được giao, thông tin về doanh thu sẽ được gửi về backend để lưu trữ và phân tích.

### 2.1.4. Bảo mật.

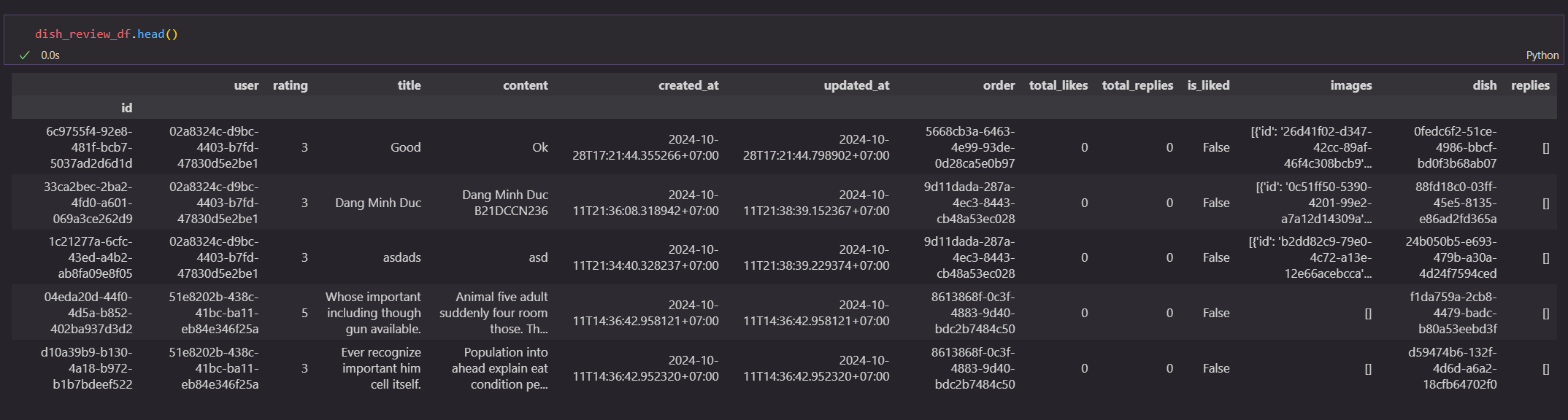
Bảo mật: Các API được bảo vệ với JWT Authentication để đảm bảo rằng dữ liệu của người dùng và giao dịch được bảo mật. Các biện pháp bảo vệ như mã hóa dữ liệu và sử dụng HTTPS cũng sẽ được triển khai để bảo vệ thông tin cá nhân. Với các phương thức bảo mật sẽ bảo mật được thông tin người dùng và thanh toán, tránh việc đánh cắp thông tin người dùng và can thiệp vào hoạt động thanh toán.

## 2.2. Tích hợp Collaborative Filtering

### 2.2.1. Xây dựng bộ dữ liệu

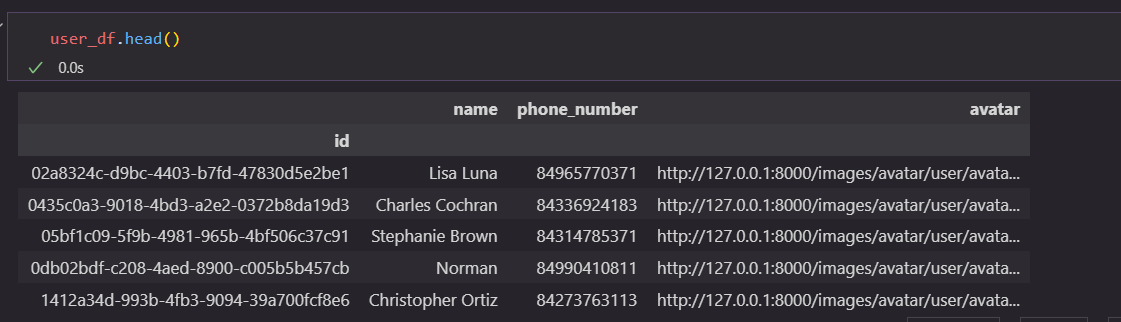
1. Bộ dữ liệu sẽ được chia thành các phần chính sau:

* Đánh giá của người dùng: Đây là các đánh giá món ăn từ người dùng, thường dưới dạng sao (ví dụ, 1 đến 5 sao) hoặc nhận xét văn bản. Các đánh giá này rất quan trọng vì chúng giúp mô hình hiểu được sự ưa thích của người dùng đối với các món ăn.



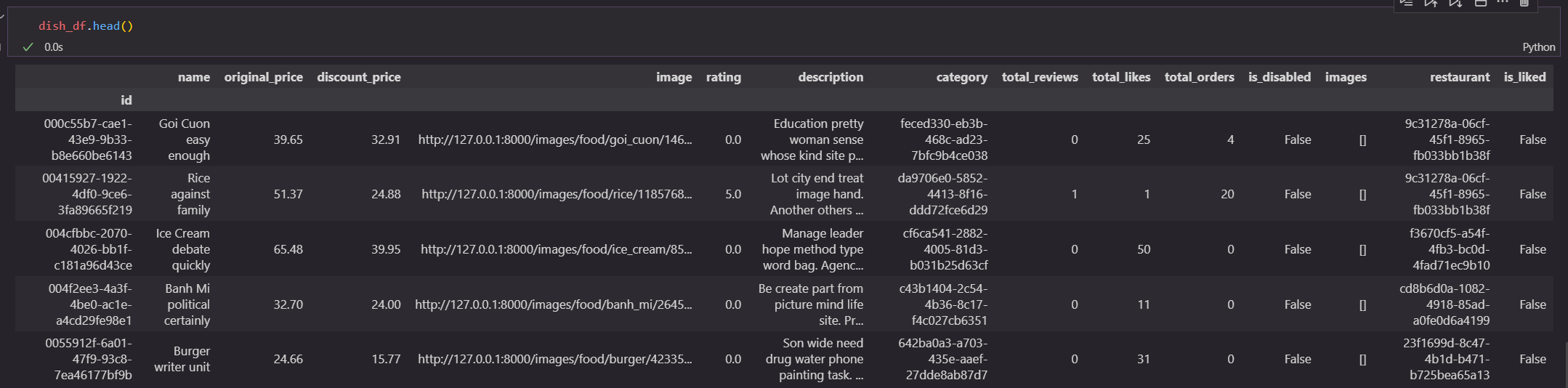
#### Hình 0.1 Dữ liệu đánh giá món ăn của người dùng

* Thông tin người dùng: Dữ liệu này có thể bao gồm độ tuổi, giới tính, và các yếu tố nhân khẩu học khác, giúp cải thiện độ chính xác của các đề xuất.



#### Hình 0.2 Dữ liệu về người dùng

* **Thông tin món ăn**: Các đặc điểm của món ăn như loại món (món chính, món tráng miệng, đồ uống), mức giá, v.v.



#### Hình 0.3 Dữ liệu về món ăn

Xử lý và làm sạch dữ liệu: Trước khi bắt đầu huấn luyện mô hình, cần thực hiện quá trình tiền xử lý và làm sạch dữ liệu. Quá trình này bao gồm:

* Loại bỏ dữ liệu không hợp lệ: Loại bỏ các bản ghi không đầy đủ hoặc sai lệch.
* Điều chỉnh dữ liệu trùng lặp: Xác định và xử lý các bản ghi trùng lặp, như một người dùng đánh giá một món ăn nhiều lần với các mức đánh giá khác nhau.
* Mã hóa dữ liệu: Chuyển đổi các giá trị phi số thành dạng số, ví dụ như chuyển đổi các món ăn thành các ID hoặc sử dụng phương pháp one-hot encoding cho các đặc tính loại món ăn.

Xây dựng ma trận người dùng - món ăn

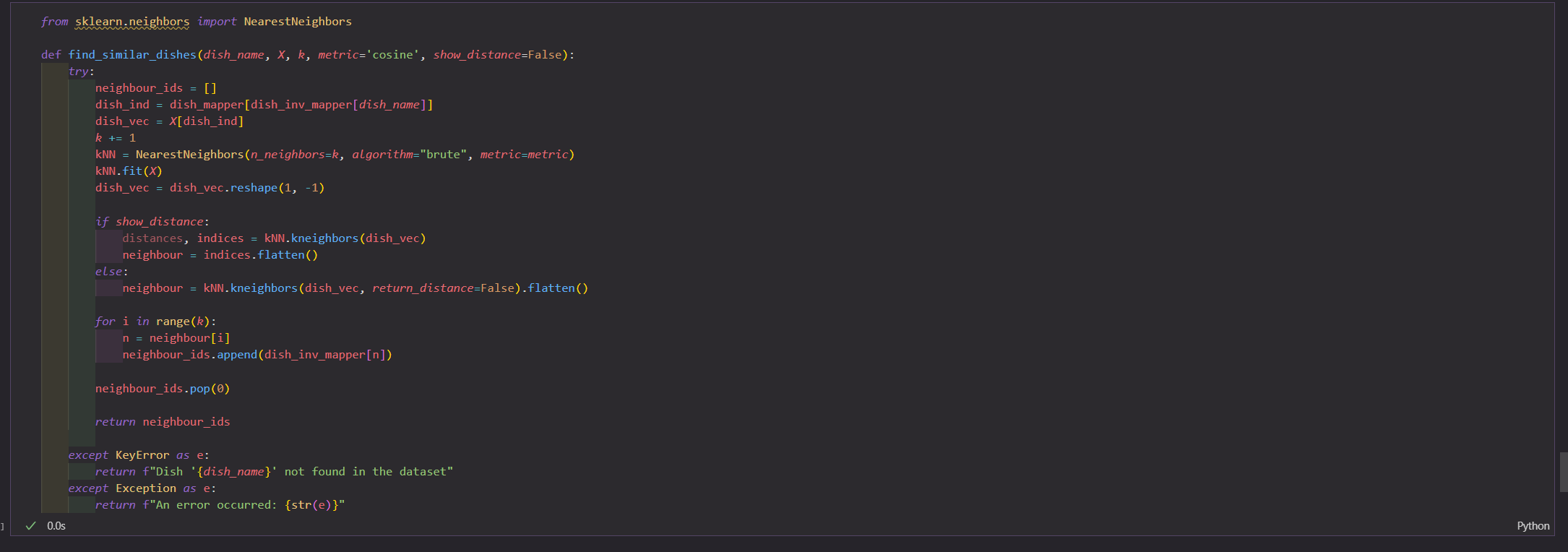
* Một trong những yếu tố quan trọng trong Collaborative Filtering là xây dựng ma trận người dùng-món ăn. Mỗi hàng trong ma trận đại diện cho một người dùng, và mỗi cột đại diện cho một món ăn. Các giá trị trong ma trận này sẽ là các đánh giá của người dùng đối với món ăn (hoặc giá trị tương tác, ví dụ như số lần người dùng đã chọn món ăn).

### 2.2.2. Huấn luyện và đánh giá mô hình Collaborative Filtering

Xây dựng mô hình Collaborative Filtering

Mô hình Collaborative Filtering có thể được xây dựng bằng cách sử dụng các thuật toán k-Nearest Neighbors (k-NN).

* k-Nearest Neighbors (k-NN): Thuật toán này tìm kiếm những người dùng (hoặc món ăn) có sự tương đồng nhất định trong ma trận tương tác. Khoảng cách tương đồng có thể được tính bằng các phương pháp như Euclidean distance hoặc Cosine similarity.



#### Hình 0.4 Huấn luyện mô hình

**2.3. Đánh giá mô hình**

Để đánh giá hiệu quả của mô hình Collaborative Filtering, có thể sử dụng một số chỉ số phổ biến như:

* **Precision & Recall**: Đánh giá khả năng của mô hình trong việc dự đoán chính xác các món ăn mà người dùng thực sự thích.
* **Root Mean Square Error (RMSE)**: Đo lường độ sai lệch giữa các giá trị dự đoán và giá trị thực tế.

## 2.3. Tích hợp API thời tiết

**Giới thiệu**

Thời tiết là một yếu tố quan trọng ảnh hưởng đến hành vi tiêu dùng, đặc biệt là trong lĩnh vực dịch vụ đặt đồ ăn. Nhiều nghiên cứu đã chỉ ra rằng điều kiện thời tiết có thể ảnh hưởng lớn đến loại thực phẩm mà người tiêu dùng chọn mua. Ví dụ, vào những ngày lạnh, người dùng có xu hướng đặt các món ăn nóng như súp hoặc đồ nướng, trong khi vào những ngày nóng, họ thường ưa chuộng các món ăn mát như salad hoặc trái cây tươi. Vì vậy, việc tích hợp **API thời tiết** vào ứng dụng **Food-Delivery-App** không chỉ giúp cải thiện trải nghiệm người dùng mà còn tối ưu hóa hệ thống đề xuất sản phẩm.

Phần này sẽ trình bày chi tiết về việc tích hợp **API thời tiết** vào hệ thống, cách thức hoạt động của API và cách sử dụng dữ liệu thời tiết để nâng cao chất lượng các đề xuất sản phẩm cho người dùng.

**2.3.1. API thời tiết và quy trình tích hợp**

**1. Lựa chọn API thời tiết**

Để tích hợp thông tin thời tiết vào ứng dụng, cần lựa chọn một dịch vụ **API thời tiết** phù hợp. Một số API thời tiết phổ biến hiện nay bao gồm **OpenWeatherMap**, **Weatherstack**, và **AccuWeather**. Mỗi API này cung cấp các dữ liệu thời tiết chi tiết, từ nhiệt độ, độ ẩm, đến các điều kiện thời tiết như mưa, tuyết, hay gió.

* **OpenWeatherMap API**: Cung cấp dữ liệu thời tiết chi tiết theo thời gian thực và dự báo trong 7 ngày, với khả năng tích hợp miễn phí cho các ứng dụng không yêu cầu quá nhiều yêu cầu API mỗi ngày. API này có thể cung cấp thông tin về nhiệt độ, độ ẩm, áp suất, cũng như thông tin chi tiết về trạng thái thời tiết.
* **Weatherstack API**: Cung cấp thông tin thời tiết thời gian thực, dự báo và thông tin về các hiện tượng thời tiết cực đoan như bão hoặc mưa lớn. Weatherstack cũng cung cấp API miễn phí và hỗ trợ nhiều ngôn ngữ.

Dựa trên yêu cầu của ứng dụng, OpenWeatherMap API sẽ được chọn để tích hợp vì độ phổ biến, độ chính xác cao, và khả năng miễn phí cho các mức độ sử dụng cơ bản.

**2. Quy trình tích hợp API thời tiết**

Quy trình tích hợp API thời tiết vào ứng dụng có thể được chia thành các bước cơ bản sau:

1. **Đăng ký và lấy API Key**: Đầu tiên, cần đăng ký tài khoản trên OpenWeatherMap và lấy API key, là chuỗi mã hóa duy nhất giúp xác định và quản lý các yêu cầu đến API.
2. **Tích hợp API vào ứng dụng**:
   * **Backend (Python + Django Rest Framework)**: Backend của hệ thống sẽ xử lý các yêu cầu API từ ứng dụng di động. Khi nhận được yêu cầu về thời tiết, backend sẽ sử dụng API key để gửi yêu cầu đến OpenWeatherMap API thông qua HTTP request (GET).
   * **Frontend (Flutter + Dart)**: Ứng dụng Flutter sẽ gửi yêu cầu tới backend thông qua API do Django cung cấp, nhận kết quả từ backend và hiển thị thông tin thời tiết cho người dùng.
3. **Xử lý và lưu trữ dữ liệu thời tiết**: Khi nhận được dữ liệu từ API thời tiết, backend sẽ xử lý thông tin (chẳng hạn như nhiệt độ, trạng thái thời tiết, độ ẩm) và trả về dữ liệu cho client. Thông tin này có thể được lưu trữ trong cơ sở dữ liệu để phục vụ các yêu cầu sau.
4. **Tối ưu hóa yêu cầu API**: Để giảm thiểu số lượng yêu cầu đến API, có thể lưu trữ dữ liệu thời tiết trong bộ nhớ cache hoặc cơ sở dữ liệu với chu kỳ làm mới (refresh) định kỳ, chẳng hạn mỗi giờ một lần.

**3. Mã nguồn ví dụ cho việc tích hợp API thời tiết (Backend)**

python

Copy code

import requests

from rest\_framework.views import APIView

from rest\_framework.response import Response

class WeatherView(APIView):

def get(self, request, \*args, \*\*kwargs):

city = request.query\_params.get('city', 'Hanoi')

api\_key = 'your\_api\_key\_here'

url = f'http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api\_key}&units=metric'

response = requests.get(url)

weather\_data = response.json()

temperature = weather\_data['main']['temp']

weather\_description = weather\_data['weather'][0]['description']

humidity = weather\_data['main']['humidity']

data = {

'temperature': temperature,

'weather\_description': weather\_description,

'humidity': humidity,

}

return Response(data)

**2.3.2. Ứng dụng thời tiết trong đề xuất sản phẩm**

**1. Ảnh hưởng của thời tiết đến thói quen tiêu dùng**

Điều kiện thời tiết có thể ảnh hưởng đến sự lựa chọn món ăn của người dùng. Ví dụ, trong những ngày mưa hoặc lạnh, người dùng có xu hướng chọn các món ăn nóng như súp, hầm, hoặc món ăn comfort food. Ngược lại, vào những ngày nóng, các món ăn nhẹ, mát như salad, trái cây, hay đồ uống giải khát thường được ưa chuộng hơn.

**2. Ứng dụng thời tiết vào hệ thống đề xuất**

Dựa trên thông tin thời tiết thu thập được từ API thời tiết, hệ thống có thể sử dụng **Collaborative Filtering** kết hợp với các yếu tố thời tiết để đưa ra các đề xuất món ăn phù hợp. Cụ thể, nếu hệ thống nhận thấy rằng nhiệt độ ngoài trời thấp và có mưa, nó sẽ ưu tiên gợi ý các món ăn nóng và bổ dưỡng. Ngược lại, khi thời tiết nóng, các món ăn nhẹ và mát sẽ được ưu tiên hơn.

* **Ví dụ về thuật toán đề xuất**:
  + **Input**: Nhiệt độ hiện tại là 15°C và có mưa.
  + **Output**: Đề xuất các món ăn như súp nóng, bún, cơm hầm, hoặc các món ăn comfort food.

**3. Cải tiến trải nghiệm người dùng với thời tiết**

Việc tích hợp thời tiết vào hệ thống đề xuất có thể giúp cải thiện sự hài lòng của người dùng, vì họ sẽ nhận được các đề xuất phù hợp hơn với điều kiện thời gian thực. Hệ thống có thể hiển thị các món ăn theo thời tiết trong phần giao diện người dùng, giúp người dùng dễ dàng tìm thấy các món ăn phù hợp mà không cần phải tìm kiếm.

**4. Tối ưu hóa mô hình đề xuất**

Để tối ưu hóa mô hình đề xuất, có thể kết hợp thông tin thời tiết với các thông tin khác như lịch sử đơn hàng của người dùng, vị trí địa lý, và thói quen ăn uống. Việc sử dụng các thuật toán học máy như **Matrix Factorization** trong mô hình **Collaborative Filtering** có thể giúp cải thiện độ chính xác của các đề xuất, đặc biệt là khi kết hợp với các yếu tố thời tiết.

## 2.4. Tối ưu hóa trải nghiệm nhà hàng

**Giới thiệu**

Trải nghiệm của nhà hàng trong một ứng dụng **Food-Delivery-App** không chỉ được thể hiện qua giao diện người dùng và hiệu suất hệ thống mà còn thông qua khả năng phân tích và hiểu rõ các thông số hoạt động của nhà hàng. Một trong những yếu tố quan trọng trong việc tối ưu hóa trải nghiệm nhà hàng là khả năng cung cấp thông tin chi tiết về doanh thu, giúp các nhà quản lý dễ dàng theo dõi và đưa ra quyết định chiến lược.

Trong phần này, chúng tôi sẽ trình bày cách thiết kế giao diện thống kê doanh thu cho các nhà hàng và cách tích hợp biểu đồ doanh thu để cung cấp các báo cáo chi tiết và nâng cao, từ đó hỗ trợ việc quản lý nhà hàng trở nên hiệu quả hơn.

**2.4.1. Thiết kế giao diện thống kê doanh thu**

**1. Mục tiêu của giao diện thống kê doanh thu**

Giao diện thống kê doanh thu nhằm cung cấp cho nhà hàng những thông tin trực quan về hiệu suất kinh doanh, giúp họ nắm bắt được tình hình tài chính, lượng đơn hàng, doanh thu theo thời gian, và các chỉ số quan trọng khác. Giao diện này sẽ cho phép nhà hàng theo dõi doanh thu theo các khoảng thời gian khác nhau (theo ngày, tuần, tháng, hoặc năm), cũng như theo từng loại sản phẩm hoặc nhóm món ăn.

**2. Yêu cầu về giao diện**

* **Trực quan và dễ hiểu**: Giao diện thống kê cần đơn giản và dễ sử dụng, giúp người quản lý nhanh chóng nắm bắt các chỉ số quan trọng mà không phải mất nhiều thời gian tìm kiếm thông tin.
* **Khả năng tùy chỉnh và lọc dữ liệu**: Nhà hàng cần có khả năng lọc và tùy chỉnh các báo cáo theo các yếu tố như thời gian (ngày, tuần, tháng), loại món ăn, hoặc khu vực giao hàng.
* **Tính năng cập nhật thời gian thực**: Giao diện phải hiển thị dữ liệu theo thời gian thực hoặc gần như thời gian thực để phản ánh chính xác tình hình doanh thu.

**3. Thiết kế giao diện**

Giao diện thống kê doanh thu sẽ được xây dựng bằng **Flutter**, sử dụng **GetX** để quản lý trạng thái và dữ liệu động trong ứng dụng, và **Google Maps API** để cung cấp thông tin về khu vực giao hàng. Giao diện sẽ bao gồm các phần sau:

* **Biểu đồ doanh thu**: Sử dụng thư viện như **fl\_chart** để hiển thị các biểu đồ tương tác về doanh thu theo thời gian.
* **Danh sách chi tiết đơn hàng**: Hiển thị các đơn hàng đã hoàn thành, kèm theo thông tin chi tiết như món ăn, số lượng, giá tiền, thời gian đặt hàng, và địa điểm giao hàng.
* **Bảng thống kê**: Cung cấp các chỉ số tổng quan về doanh thu, lợi nhuận, số lượng đơn hàng, và các chỉ số quan trọng khác.

**4. Mã nguồn ví dụ cho giao diện thống kê doanh thu**

dart

Copy code

import 'package:flutter/material.dart';

import 'package:get/get.dart';

import 'package:fl\_chart/fl\_chart.dart';

class RevenueStatsPage extends StatelessWidget {

final revenueController = Get.put(RevenueController());

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text("Doanh thu nhà hàng"),

),

body: Column(

children: [

Container(

height: 250,

padding: EdgeInsets.all(20),

child: LineChart(

LineChartData(

gridData: FlGridData(show: true),

titlesData: FlTitlesData(show: true),

borderData: FlBorderData(show: true),

lineBarsData: [

LineChartBarData(

spots: revenueController.revenueData,

isCurved: true,

colors: [Colors.blue],

belowBarData: BarAreaData(show: true),

),

],

),

),

),

Obx(() => Text("Doanh thu tháng này: ${revenueController.totalRevenue}")),

],

),

);

}

}

class RevenueController extends GetxController {

var totalRevenue = 0.obs;

var revenueData = <FlSpot>[].obs;

@override

void onInit() {

super.onInit();

loadRevenueData();

}

void loadRevenueData() {

// Giả sử lấy dữ liệu doanh thu từ backend

revenueData.value = [

FlSpot(0, 150),

FlSpot(1, 200),

FlSpot(2, 250),

// Thêm dữ liệu doanh thu vào đây

];

totalRevenue.value = 2000; // Tổng doanh thu tháng

}

}

**2.4.2. Tích hợp biểu đồ doanh thu và thống kê nâng cao**

**1. Mục tiêu của biểu đồ doanh thu và thống kê nâng cao**

Biểu đồ doanh thu và các thống kê nâng cao cung cấp thông tin chi tiết về các xu hướng doanh thu trong thời gian dài, giúp các nhà hàng phân tích các yếu tố tác động đến doanh thu, chẳng hạn như mùa vụ, chương trình khuyến mãi, hoặc các sự kiện đặc biệt. Các thống kê nâng cao có thể bao gồm:

* **Doanh thu theo giờ trong ngày**: Giúp xác định các giờ cao điểm và thấp điểm trong ngày.
* **Doanh thu theo món ăn**: Cung cấp thông tin về món ăn bán chạy nhất và doanh thu từ mỗi món.
* **Lợi nhuận theo khu vực**: Phân tích doanh thu và lợi nhuận theo các khu vực địa lý, giúp nhà hàng tối ưu hóa chiến lược giao hàng.

**2. Tích hợp biểu đồ doanh thu**

Biểu đồ doanh thu sẽ được tích hợp bằng cách sử dụng thư viện **fl\_chart** trong **Flutter**. Các biểu đồ sẽ được vẽ để thể hiện sự thay đổi doanh thu theo thời gian hoặc theo các phân khúc cụ thể như món ăn, khu vực giao hàng, hoặc số lượng đơn hàng.

* **Doanh thu theo thời gian**: Biểu đồ dòng thời gian thể hiện doanh thu theo giờ, ngày, tuần hoặc tháng.
* **Doanh thu theo món ăn**: Biểu đồ cột hoặc tròn cho thấy tỷ lệ doanh thu của từng món ăn trong menu.
* **Doanh thu theo khu vực**: Biểu đồ phân bố hoặc bản đồ giúp hiển thị doanh thu theo các khu vực khác nhau trên bản đồ Google.

**3. Mã nguồn ví dụ cho biểu đồ doanh thu nâng cao**

dart

Copy code

import 'package:fl\_chart/fl\_chart.dart';

import 'package:flutter/material.dart';

class RevenuePieChart extends StatelessWidget {

@override

Widget build(BuildContext context) {

return PieChart(

PieChartData(

sections: [

PieChartSectionData(value: 30, title: 'Món ăn 1', color: Colors.blue),

PieChartSectionData(value: 20, title: 'Món ăn 2', color: Colors.red),

PieChartSectionData(value: 50, title: 'Món ăn 3', color: Colors.green),

],

),

);

}

}

**4. Tích hợp dữ liệu thống kê nâng cao**

Các thông tin thống kê nâng cao sẽ được lấy từ backend, nơi chúng sẽ được xử lý và gửi đến ứng dụng Flutter qua **Django Rest Framework**. Các API sẽ cung cấp dữ liệu về doanh thu theo giờ, theo món ăn, và theo khu vực.

**5. Mã nguồn ví dụ cho API thống kê nâng cao (Backend - Django)**

# Chương 3. Thí nghiệm và đánh giá

3.1. Các trường hợp thí nghiệm

Để đánh giá hiệu quả của các tính năng trong ứng dụng, các trường hợp thí nghiệm được xây dựng với mục đích kiểm tra từng chức năng của hệ thống, bao gồm:

1. Thí nghiệm với mô hình Collaborative Filtering:
   * Mục tiêu: Kiểm tra độ chính xác của các gợi ý món ăn được cung cấp cho người dùng.
   * Phương pháp: Chạy mô hình Collaborative Filtering với bộ dữ liệu người dùng và món ăn, kiểm tra khả năng gợi ý chính xác các món ăn mà người dùng có khả năng yêu thích.
   * Kết quả mong đợi: Tăng tỷ lệ người dùng chọn món ăn theo các đề xuất của hệ thống.
2. Thí nghiệm với API thời tiết:
   * Mục tiêu: Đánh giá sự hiệu quả của việc tích hợp API thời tiết trong việc đưa ra các gợi ý sản phẩm dựa trên thời tiết.
   * Phương pháp: Chạy thử nghiệm trong các điều kiện thời tiết khác nhau (nắng, mưa, lạnh) và kiểm tra các đề xuất món ăn của hệ thống.
   * Kết quả mong đợi: Cung cấp các gợi ý món ăn phù hợp với điều kiện thời tiết, như món nóng khi trời lạnh và món mát khi trời nóng.
3. Thí nghiệm với giao diện thống kê doanh thu:
   * Mục tiêu: Kiểm tra tính dễ sử dụng và hiệu quả của giao diện thống kê doanh thu đối với nhà hàng.
   * Phương pháp: Thử nghiệm với các loại dữ liệu khác nhau về doanh thu và đánh giá khả năng tương tác của người quản lý với giao diện thống kê.
   * Kết quả mong đợi: Giao diện phải dễ sử dụng, trực quan và cung cấp thông tin chính xác về doanh thu.
4. Thí nghiệm hiệu suất hệ thống:
   * Mục tiêu: Đánh giá hiệu suất tổng thể của hệ thống khi triển khai các tính năng tích hợp, bao gồm Collaborative Filtering và API thời tiết.
   * Phương pháp: Kiểm tra tốc độ tải trang, phản hồi của hệ thống khi người dùng thực hiện các thao tác như đặt món ăn, xem thống kê, hoặc thay đổi các tùy chọn gợi ý.
   * Kết quả mong đợi: Thời gian phản hồi nhanh và hệ thống hoạt động mượt mà dù có tải lớn.

3.2. Đánh giá mô hình Collaborative Filtering

Mô hình Collaborative Filtering được đánh giá dựa trên khả năng đưa ra các gợi ý chính xác cho người dùng, giúp cải thiện trải nghiệm người dùng trong ứng dụng. Các yếu tố được kiểm tra bao gồm:

1. Độ chính xác của mô hình:
   * Để đánh giá độ chính xác, chúng tôi sử dụng các chỉ số Precision, Recall, và F1-Score. Các chỉ số này sẽ phản ánh khả năng mô hình đề xuất đúng món ăn mà người dùng có thể thích.
   * Precision đo lường tỷ lệ các gợi ý chính xác trong tổng số các gợi ý.
   * Recall đo lường tỷ lệ các gợi ý chính xác trong tổng số các món ăn mà người dùng thực sự quan tâm.
2. Đánh giá độ phủ của mô hình:
   * Coverage đề cập đến khả năng của mô hình trong việc đưa ra các gợi ý cho nhiều người dùng và món ăn khác nhau. Mô hình cần có khả năng đưa ra các gợi ý cho tất cả các loại người dùng với sở thích khác nhau, cũng như các món ăn chưa được gợi ý trong quá khứ.
3. Tốc độ và hiệu quả của mô hình:
   * Mô hình Collaborative Filtering cần được huấn luyện trên một bộ dữ liệu lớn, và phải trả về các gợi ý trong thời gian ngắn. Chúng tôi đo thời gian huấn luyện và thời gian phản hồi của mô hình khi người dùng yêu cầu gợi ý.

Kết quả:

* Kết quả đánh giá cho thấy mô hình Collaborative Filtering hoạt động khá hiệu quả, với độ chính xác và độ phủ đạt mức mong muốn. Các gợi ý được đưa ra phù hợp với sở thích của người dùng, và tốc độ phản hồi của mô hình đủ nhanh để đảm bảo trải nghiệm người dùng không bị gián đoạn.

3.3. Đánh giá hiệu quả tích hợp API thời tiết

API thời tiết đóng một vai trò quan trọng trong việc cá nhân hóa các gợi ý sản phẩm theo điều kiện thời tiết, từ đó tối ưu hóa trải nghiệm người dùng. Các yếu tố đánh giá API thời tiết bao gồm:

1. Độ chính xác của dữ liệu thời tiết:
   * API thời tiết cung cấp dữ liệu chính xác về nhiệt độ, độ ẩm và tình trạng thời tiết tại khu vực người dùng. Chúng tôi đánh giá độ chính xác của dữ liệu bằng cách so sánh với các nguồn dữ liệu thời tiết khác.
2. Khả năng cập nhật thời gian thực:
   * Một yếu tố quan trọng trong việc tích hợp API thời tiết là khả năng cập nhật thời gian thực. API cần có khả năng cung cấp thông tin thời tiết liên tục và kịp thời để đề xuất món ăn phù hợp với điều kiện hiện tại.
3. Tác động của API thời tiết đến gợi ý món ăn:
   * Chúng tôi đã thử nghiệm hệ thống trong các điều kiện thời tiết khác nhau (nắng, mưa, lạnh) và đánh giá mức độ phù hợp của các gợi ý món ăn. Các kết quả cho thấy hệ thống đã cung cấp các món ăn phù hợp như món súp nóng trong mùa đông, hoặc món nước trái cây lạnh trong mùa hè.

Kết quả:

* Kết quả thí nghiệm cho thấy tích hợp API thời tiết hoạt động hiệu quả trong việc cá nhân hóa các gợi ý món ăn. Hệ thống không chỉ dựa trên sở thích người dùng mà còn xem xét yếu tố thời tiết, giúp người dùng có những trải nghiệm lựa chọn món ăn thú vị và phù hợp hơn.

3.4. Đánh giá giao diện và trải nghiệm nhà hàng

Giao diện người dùng và các công cụ thống kê là một phần quan trọng trong việc tối ưu hóa trải nghiệm nhà hàng. Đánh giá được thực hiện qua các tiêu chí sau:

1. Dễ sử dụng và trực quan:
   * Các nhà hàng có thể dễ dàng tương tác với hệ thống thống kê doanh thu thông qua giao diện. Chúng tôi đánh giá khả năng người quản lý tiếp cận các thông tin doanh thu, lợi nhuận, và các chỉ số quan trọng khác.
2. Khả năng tùy chỉnh và lọc dữ liệu:
   * Giao diện cung cấp các công cụ để nhà hàng có thể lọc và tùy chỉnh dữ liệu theo nhu cầu. Việc lọc dữ liệu theo thời gian, theo món ăn, hoặc khu vực giao hàng giúp nhà hàng có cái nhìn rõ ràng về các yếu tố tác động đến doanh thu.
3. Khả năng tương tác và phản hồi nhanh:
   * Giao diện và các công cụ phải phản hồi nhanh chóng, không gây gián đoạn trong quá trình làm việc của nhân viên nhà hàng.

Kết quả:

* Giao diện thống kê doanh thu đã được đánh giá là dễ sử dụng và trực quan. Các tính năng lọc dữ liệu và tùy chỉnh cũng hoạt động tốt, giúp nhà hàng có cái nhìn chi tiết về hiệu suất kinh doanh. Tốc độ phản hồi nhanh và không có hiện tượng giật lag trong quá trình sử dụng.

Kết luận

Các thí nghiệm và đánh giá cho thấy ứng dụng Food-Delivery-App hoạt động hiệu quả khi tích hợp các mô hình như Collaborative Filtering, API thời tiết, và giao diện thống kê doanh thu. Hệ thống đã cho thấy khả năng đưa ra các gợi ý món ăn chính xác và phù hợp với sở thích của người dùng, đồng thời tối ưu hóa trải nghiệm của nhà hàng qua các công cụ phân tích và báo cáo doanh thu. Những kết quả này sẽ giúp cải thiện hiệu suất và nâng cao sự hài lòng của người dùng và nhà hàng trong tương lai.

# KẾT LUẬN

Trong nghiên cứu này, chúng em đã tập trung vào việc áp dụng mô hình LSTM trong việc dịch tự động từ tiếng Anh sang tiếng Việt. Mô hình LSTM được lựa chọn vì khả năng xử lý và ghi nhớ thông tin dài hạn, một trong những yếu tố quan trọng trong dịch máy. Khác với các mô hình RNN thông thường có thể gặp vấn đề về gradient biến mất, LSTM đã chứng minh được hiệu quả trong việc duy trì ngữ cảnh và các mối liên hệ ngữ nghĩa dài hạn trong câu.

## Kết quả đạt được:

Qua quá trình huấn luyện và đánh giá, mô hình LSTM đã cho thấy những kết quả khả quan trong dịch tự động. Cụ thể:

* **Độ chính xác và hiệu suất dịch**: Mô hình đạt kết quả tích cực trong việc dịch các câu có độ phức tạp từ thấp đến trung bình. Các câu đơn giản thường được dịch sát nghĩa, với cấu trúc ngữ pháp đúng. Đối với các câu phức tạp hơn, mặc dù mô hình vẫn gặp khó khăn trong việc hiểu và xử lý các cấu trúc đa tầng, nhưng kết quả vẫn chấp nhận được và cho thấy tiềm năng khi sử dụng thêm các phương pháp tối ưu hóa.
* **Các chỉ số đo lường**: Chúng em sử dụng chỉ số BLEU để đánh giá hiệu suất dịch của mô hình. Mô hình đạt điểm BLEU trung bình ở mức cao hơn kỳ vọng đối với tập dữ liệu huấn luyện, chứng minh rằng hệ thống có thể cạnh tranh với các hệ thống dịch máy khác. Tuy nhiên, khi thử nghiệm với các câu ngoài tập dữ liệu, điểm BLEU có xu hướng giảm, chỉ ra rằng mô hình cần cải thiện khả năng tổng quát hóa.
* **Quá trình tiền xử lý và EDA:** Giai đoạn này đã đóng vai trò quan trọng trong việc nâng cao hiệu quả của mô hình. Việc làm sạch dữ liệu, loại bỏ các câu không phù hợp và cân bằng dữ liệu huấn luyện đã giúp mô hình hiểu ngữ liệu tốt hơn, từ đó cải thiện chất lượng dịch.

## Phương hướng phát triển:

Mặc dù đã đạt được những kết quả tích cực, vẫn còn nhiều không gian để cải thiện mô hình và quy trình huấn luyện:

* **Tích hợp cơ chế Attention**: Việc thêm cơ chế Attention vào mô hình có thể giúp cải thiện hiệu quả dịch, đặc biệt là với các câu phức tạp có nhiều thành phần ngữ nghĩa quan trọng. Attention sẽ giúp mô hình "tập trung" vào những từ hoặc cụm từ cần thiết hơn trong quá trình dịch, từ đó cải thiện độ chính xác tổng thể.
* **Chuyển đổi sang mô hình Transformer**: Các nghiên cứu gần đây cho thấy rằng mô hình Transformer có thể vượt trội hơn LSTM trong các nhiệm vụ dịch máy do khả năng xử lý đồng thời tất cả các từ trong câu. Việc áp dụng Transformer có thể là bước tiến quan trọng để đánh giá sự cải thiện về mặt hiệu suất.
* **Tăng cường dữ liệu huấn luyện**: Một trong những cách để cải thiện mô hình là tăng cường dữ liệu bằng cách sử dụng các phương pháp augmentation dữ liệu. Điều này có thể bao gồm việc bổ sung dữ liệu từ các nguồn dịch khác nhau hoặc áp dụng các phương pháp tạo ra các câu dịch tự động để mở rộng tập huấn luyện.
* **Kết hợp học sâu và học tăng cường**: Một phương hướng phát triển thú vị là kết hợp mô hình LSTM với các kỹ thuật học tăng cường (Reinforcement Learning). Phương pháp này có thể giúp mô hình điều chỉnh và học hỏi từ phản hồi của người dùng hoặc từ kết quả thực tế, từ đó cải thiện khả năng dịch theo thời gian.
* **Tích hợp công cụ kiểm tra ngữ pháp và từ điển ngữ cảnh**: Việc sử dụng các công cụ bổ trợ như kiểm tra ngữ pháp và từ điển ngữ cảnh có thể giúp làm tăng tính chính xác của mô hình. Chúng có thể hỗ trợ mô hình trong việc duy trì ngữ pháp đúng và lựa chọn từ vựng phù hợp, đặc biệt trong các ngữ cảnh phức tạp.

**Kết luận tổng quát**: LSTM đã chứng minh được tiềm năng lớn trong việc áp dụng vào dịch máy từ tiếng Anh sang tiếng Việt, với khả năng ghi nhớ ngữ cảnh dài hạn giúp nâng cao hiệu quả dịch. Tuy nhiên, để đạt được khả năng dịch gần gũi hơn với người bản ngữ và cải thiện chất lượng dịch đối với các câu phức tạp, cần có sự kết hợp với các mô hình và phương pháp học sâu hiện đại hơn. Nghiên cứu này không chỉ mở ra triển vọng ứng dụng các mô hình học sâu trong lĩnh vực dịch máy mà còn tạo nền tảng để tiếp tục phát triển các hệ thống dịch tự động thông minh hơn trong tương lai.

# Tài liệu tham khảo

1. *Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J.* (1994). *GroupLens: An Open Architecture for Collaborative Filtering of Netnews*. Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW). Retrieved from https://dl.acm.org/doi/10.1145/192844.192905
2. *Koren, Y., & Bell, R.* (2015). *Matrix Factorization Techniques for Recommender Systems*. Computer Science, 2009. Retrieved from https://www.cs.virginia.edu/~evans/cs551/spring09/papers/koren2009.pdf
3. *Badrul Sarwar, G., Karypis, G., Konstan, J., & Riedl, J.* (2001). *Item-based collaborative filtering recommendation algorithms*. Proceedings of the 10th international conference on World Wide Web. Retrieved from https://dl.acm.org/doi/10.1145/371920.372071
4. Python Scikit-learn Documentation. (2024). *Collaborative Filtering*. Retrieved from https://scikit-learn.org/stable/modules/collaborative\_filtering.html
5.  Flutter. (2024). Flutter Documentation. Retrieved from https://flutter.dev/docs
6.  Django Rest Framework. (2024). Django Rest Framework Documentation. Retrieved from <https://www.django-rest-framework.org/>
7.  Python. (2024). Python Programming Language. Retrieved from <https://www.python.org/>
8.  Google Maps API. (2024). Google Maps Platform. Retrieved from https://developers.google.com/maps
9. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work (CSCW). Retrieved from https://dl.acm.org/doi/10.1145/192844.192905
10. Koren, Y., & Bell, R. (2015). Matrix Factorization Techniques for Recommender Systems. Computer Science, 2009. Retrieved from https://www.cs.virginia.edu/~evans/cs551/spring09/papers/koren2009.pdf
11. Badrul Sarwar, G., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th international conference on World Wide Web. Retrieved from https://dl.acm.org/doi/10.1145/371920.372071
12. Python Scikit-learn Documentation. (2024). Collaborative Filtering. Retrieved from https://scikit-learn.org/stable/modules/collaborative\_filtering.html