# Restaurant Recommendation System

## Problem Description :

**Dataset Link :**

**Problem Description:** The Restaurant Recommendation System is a system that recommends the best restaurant to users based on their preferences, location, and other relevant factors. The system can help users to save time in searching for restaurants and provide personalized recommendations.

**Objectives:** The main objectives of the Restaurant Recommendation System are:

- To provide personalized restaurant recommendations to users based on their preferences.
- To help users save time in searching for restaurants by recommending the best restaurants to them.
- To increase user engagement and satisfaction by providing a personalized experience.

**Deliverables:** The deliverables of the Restaurant Recommendation System are:

- A recommendation system that provides personalized restaurant recommendations to users.
- A web application that allows users to input their preferences and receive restaurant recommendations.
- A database that stores information about restaurants and user preferences.

**Dataset:** The dataset for the Restaurant Recommendation System can be obtained from various sources, such as Yelp, TripAdvisor, and Zomato. The dataset should contain information about the restaurants, such as their name, location, cuisine, ratings, reviews, and other relevant factors. It should also contain information about the users, such as their location, preferences, and ratings.

**Background Information:** The Restaurant Recommendation System is a type of recommendation system that uses collaborative filtering, content-based filtering, and

hybrid filtering techniques to provide personalized recommendations to users. Collaborative filtering uses the behavior of other users to make recommendations, while content-based filtering uses the features of items to make recommendations. Hybrid filtering combines both techniques to provide more accurate recommendations.

# Possible Framework and Steps:

The framework for the Restaurant Recommendation System can be divided into the following steps:

1. **Data Collection:** Collect data from various sources, such as Yelp, TripAdvisor, and Zomato.
2. **Data Preprocessing:** Clean and preprocess the data, such as removing duplicates, handling missing values, and encoding categorical variables.
3. **Feature Extraction:** Extract relevant features from the data, such as the restaurant's cuisine, location, and ratings.
4. **Recommendation Algorithm:** Choose a recommendation algorithm, such as collaborative filtering, content-based filtering, or hybrid filtering.
5. **Model Training:** Train the recommendation model using the extracted features and the chosen algorithm.
6. **Model Evaluation:** Evaluate the performance of the model using metrics such as precision, recall, and F1-score.
7. **Deployment:** Deploy the model as a web application that allows users to input their preferences and receive restaurant recommendations.

Overall, the Restaurant Recommendation System can help users to save time in searching for restaurants and provide personalized recommendations that increase user satisfaction.

# Code Explanation :

Here is the simple explanation for the code which is provided in the code.py file.

This code creates a restaurant recommendation system that uses collaborative filtering to recommend restaurants to users based on their past restaurant ratings and the ratings of similar users.

The first step is to import the required libraries such as pandas and numpy for data manipulation, and cosine_similarity and pairwise_distances for calculating similarities between users and restaurants.

Then, we load the data into a pandas dataframe, where each row corresponds to a user and each column corresponds to a restaurant. Each cell represents the rating given by the user to the restaurant, or NaN if the user hasn't rated the restaurant yet.

Next, we fill in the missing values in the dataframe by calculating the mean rating given by each user and filling in the NaN values with that mean.

Then, we calculate the similarity between users using cosine similarity, which measures the similarity between two vectors by taking the cosine of the angle between them. We also calculate the similarity between restaurants using pairwise distance, which measures the distance between two vectors in a high-dimensional space.

Next, we define a function to predict the rating of a user for a restaurant by taking a weighted average of the ratings given by similar users. The weight is determined by the similarity between the user and the similar user.

Finally, we define a function to recommend restaurants to a user by predicting their ratings for all restaurants they haven't rated yet and returning the top N restaurants with the highest predicted ratings.

To use the recommendation system, one can simply call the recommend_restaurants function with a user ID and the number of restaurants to recommend. The function will return a list of recommended restaurants for that user.

# Future Work :

1. **Improvement in Recommendation Algorithm:** Currently, the recommendation system is using a simple collaborative filtering algorithm. One can try to implement more advanced algorithms such as Matrix Factorization, Deep Learning, and Hybrid algorithms.
2. **Integration with Online Ordering System:** In order to enhance user experience, one can integrate the recommendation system with an online ordering system so that users can directly order their recommended dishes.
3. **Integration with Social Media:** Social media integration can be added to the recommendation system so that users can share their favorite dishes with their friends and family.
4. **User Feedback System:** A user feedback system can be implemented to collect feedback on the recommended dishes. This feedback can be used to further improve the recommendation system.
5. **Integration with Reservation System:** Integration with a reservation system can be implemented so that users can make reservations for the recommended restaurant directly from the recommendation system.

**Step-by-Step Guide:**

1. **Data Collection:** Collect data on restaurants and their menus. This data can be obtained from various sources such as restaurant websites, online ordering systems, and review websites.
2. **Data Preprocessing:** Clean and preprocess the collected data. This involves removing duplicates, handling missing data, and converting the data into a suitable format for analysis.
3. **Recommendation Algorithm:** Choose an appropriate recommendation algorithm. Collaborative filtering is a popular algorithm for restaurant recommendation systems. Implement the chosen algorithm in Python.
4. **Integration with UI:** Develop a user interface for the recommendation system. The UI can be developed using popular web frameworks such as Flask or Django.
5. **Deployment:** Deploy the recommendation system on a web server. The system can be deployed on cloud platforms such as AWS or Heroku.
6. **User Feedback System:** Implement a user feedback system to collect feedback on the recommended dishes.
7. **Integration with Reservation System:** Integrate the recommendation system with a reservation system so that users can make reservations for the recommended restaurant directly from the recommendation system.

8. **Testing and Optimization:** Test the recommendation system and optimize it for better performance. Use A/B testing to evaluate the effectiveness of different recommendation algorithms.

# Exercise :

**Try to answers the following questions by yourself to check your understanding for this project. If stuck, detailed answers for the questions are also provided.**

1. **What is the purpose of cosine similarity in the restaurant recommendation system?**
   Cosine similarity is used to measure the similarity between the user's preferences and the restaurants in the dataset. It helps to identify the restaurants that are most similar to the user's preferred restaurant attributes.

2. **How does the system handle cases where the user has not provided any restaurant preferences?**
   If the user has not provided any restaurant preferences, the system will return the top-rated restaurants from the dataset. This is because the system has no information about the user's preferences to use for recommendations.

3. **What is the purpose of the train_test_split function?**
   The train_test_split function is used to split the dataset into training and testing sets. This allows the model to be trained on a portion of the dataset and evaluated on the remaining portion, which helps to assess the model's performance.

4. **How does the system handle cases where a user provides new preferences after receiving recommendations?**
   If the user provides new preferences after receiving recommendations, the system will re-calculate the cosine similarity scores and return updated recommendations based on the new preferences.

5. **How can the system be improved to provide more accurate recommendations?**
   The system can be improved by incorporating additional features or attributes of restaurants, such as price range, cuisine type, or location. It can also be improved by incorporating user feedback or ratings into the recommendation process to better personalize the recommendations. Additionally, using more advanced machine learning algorithms or techniques could potentially improve the accuracy of the recommendations.