**Introduction**

Restaurant Recommendation System. This project uses natural language processing to assist users in discovering the perfect dining experience according to the time, location and cuisine . By seamlessly integrating into conversations, our chat bot analyzes user preferences and suggests personalized restaurant recommendations, creating a delightful and tailored dining journey. by using our project uses can find best restaurants details according to the details provided by the user with in couple of minutes so that user can save lot of time to search all details of the restaurant and cuisines.

**Objective of the project**

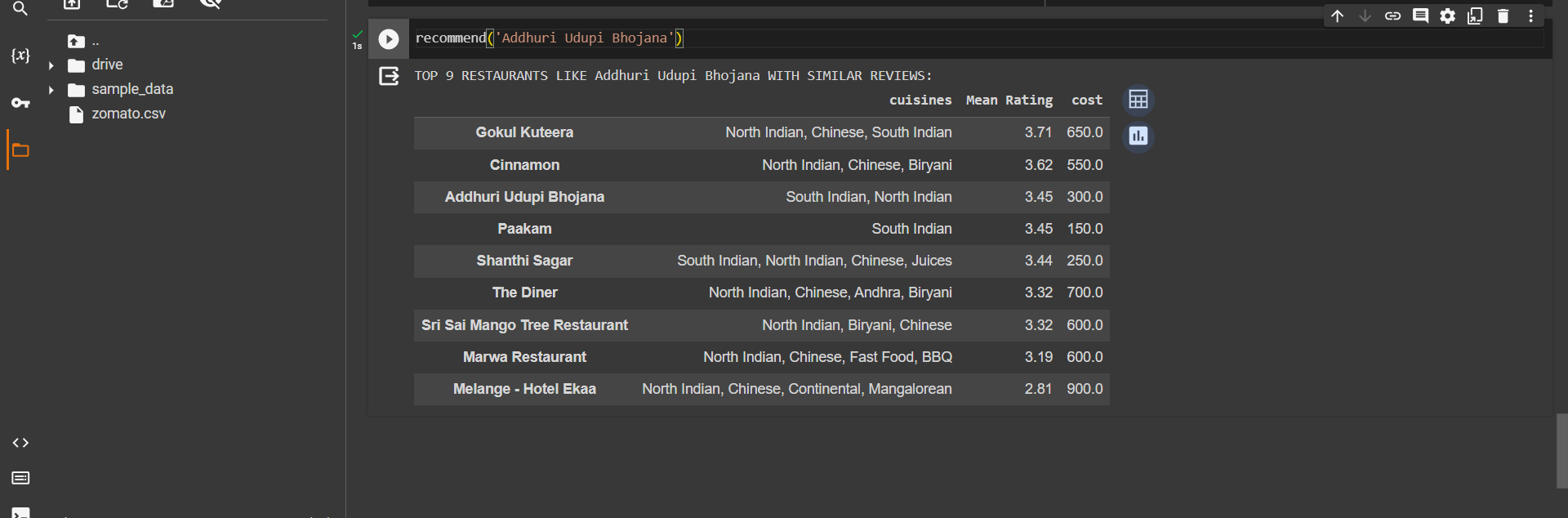
The main aim to develop a restaurant recommendation system by using the natural language processing by chatbot to provide recommendations with real time information. The primary objective of the Chat Bot Restaurant Recommendation System is to enhance user dining experiences by providing personalized and relevant restaurant suggestions. Through natural language processing, the system aims to understand user preferences, dietary restrictions, and contextual cues in order to deliver tailored recommendations. This project seeks to streamline the restaurant discovery process, making it efficient, enjoyable, and reflective of individual tastes, ultimately fostering customer satisfaction and engagement.

**The project design**

1. User Interface (UI): The system has a user-friendly chat interface where users can interact with the chat bot. This interface is designed to be intuitive and accessible, allowing users to input their preferences seamlessly.
2. Natural Language Processing (NLP): NLP algorithms are employed to understand and interpret user input. This includes extracting information about cuisine preferences, dietary restrictions, location, and any specific requirements the user may have.
3. Recommendation Engine: A robust recommendation engine is at the core of the system. It processes the user's input, analyzes the available restaurant data, and generates personalized recommendations. This engine may use techniques such as collaborative filtering or content-based filtering to enhance accuracy.
4. Database: A comprehensive database of restaurants is essential. This database includes details such as restaurant names, cuisines offered, locations, ratings, timing and reviews. Regular updates ensure that the information remains current.
5. Integration with External APIs: To enrich the recommendation system, integration with external APIs (Application Programming Interfaces) such as maps, reviews, and possibly social media can provide real-time data and additional insights.
6. User Profile Management: This will offer users to create profiles where their preferences and past interactions are stored. This contributes to the system's ability to continuously refine recommendations over time.
7. Feedback Mechanism: By taking the feedback from the user after every recommendation it will helps to update data in our database Implementing a feedback loop allows users to provide input on suggested recommendations, enabling the system to learn and improve its accuracy over time.
8. Security Measures: The sensitivity of personal preferences and potentially location-based information, the system has prioritize data security and privacy. This involves secure data storage and handling.
9. User Assistance and Error Handling: The chat bot is equipped to handle user queries gracefully. Clear responses, error messages, and assistance in case of misunderstandings contribute to a positive user experience.

By integrating these components, the Chat Bot Restaurant Recommendation System is designed to offer an intelligent, user-centric, and adaptive solution for discovering dining options to individual preferences.

**Results**

****

the recommend function takes the name of the restaurant as an input and checks for the cuisines and gives the recommendations based on those cuisines. It takes the top 30 restaurants and narrows down the list to 10. In the above example we can see that there are top 10 recommendations based on the restaurant name provided.

This function **get\_top\_words** takes a column of text data (column), the number of top words to retrieve (top\_nu\_of\_words), and the number of words in each n-gram (nu\_of\_word). It uses the CountVectorizer from scikit-learn to convert the text data into a bag-of-words representation. Then, it calculates the frequency of each word and returns a list of the top words and their frequencies.

zomato = zomato.drop(['address', 'rest\_type', 'type', 'menu\_item', 'votes'], axis=1)

This line drops specified columns ('address', 'rest\_type', 'type', 'menu\_item', 'votes') from the DataFrame zomato. It removes unnecessary columns, potentially to focus on a subset of the data for analysis.

df\_percent = zomato.sample(frac=0.5)

This line randomly samples 50% (0.5) of the rows from the DataFrame zomato. The frac parameter specifies the fraction of rows to return. This is a way to create a smaller subset of the original data for analysis, especially useful if the dataset is large.