Restaurant Recommendation System

Project Report

Date – 8 November, 2023

Team Id – 592691

INDEX

1. **INTRODUCTION**
   - 1.1 Project Overview
   - 1.2 Purpose
2. **LITERATURE SURVEY**
   - 2.1 Existing problem
   - 2.2 References
   - 2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   - 3.1 Empathy Map Canvas
   - 3.2 Ideation & Brainstorming
4. **REQUIREMENT ANALYSIS**
   - 4.1 Functional requirement
   - 4.2 Non-Functional requirements
5. **PROJECT DESIGN**
   - 5.1 Data Flow Diagrams & User Stories
   - 5.2 Solution Architecture
6. **PROJECT PLANNING & SCHEDULING**
   - 6.1 Technical Architecture
   - 6.2 Sprint Planning & Estimation
   - 6.3 Sprint Delivery Schedule
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   - 7.1 Feature 1
   - 7.2 Feature 2
   - 7.3 Database Schema (if Applicable)
8. **PERFORMANCE TESTING**
   - 8.1 Performance Metrics
9. **RESULTS**
   - 9.1 Output Screenshots
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX** Source Code

    GitHub & Project Demo Link

INTRODUCTION

Project Overview:

The Restaurant Recommendation System is an innovative project designed to enhance dining experiences by providing personalized restaurant suggestions to users. Leveraging advanced machine learning algorithms, this system analyses user preferences, historical dining data, and real-time reviews to generate accurate and tailored recommendations.

The system begins by asking user to select the restaurant whose similar restaurant they want. Through collaborative filtering and content-based recommendation techniques, the model identifies patterns and similarities between users, enabling it to suggest restaurants based on the preferences of similar individuals. Real-time sentiment analysis on reviews helps ensure that recommendations align with current positive trends and user sentiments.

The project incorporates a user-friendly interface, allowing users to select a restaurant from a large drop-down list containing all the restaurants present (all the restaurants present in Bangalore in this case).

Ultimately, the Restaurant Recommendation System aims to simplify the decision-making process for users, offering them a curated list of dining options that match their unique preferences and ensuring an enjoyable dining experience.

Purpose:

The Restaurant Recommendation System serves a multifaceted purpose in the realm of dining and hospitality. Firstly, it acts as a personalized guide for users, helping them navigate the overwhelming abundance of dining options by offering tailored suggestions based on their individual preferences. This not only simplifies the decision-making process but also ensures that users discover restaurants that align with their culinary tastes and preferences.

Secondly, the system contributes to user satisfaction by minimizing the risk of unsatisfactory dining experiences. By analysing user feedback and sentiments from reviews, it strives to recommend venues that consistently receive positive acclaim, thereby enhancing the likelihood of a pleasant dining outing.

Furthermore, the Restaurant Recommendation System plays a pivotal role in promoting diversity in the culinary landscape. By introducing users to a variety of restaurants and cuisines, it encourages exploration and supports local businesses. This diversity not only enriches the dining experiences of users but also contributes to the vibrancy of the local restaurant industry.

In summary, the Restaurant Recommendation System aims to simplify decision-making, enhance user satisfaction, and promote culinary diversity, collectively working to create a more enjoyable and enriching dining experience for users.

LITERATURE SURVEY

Existing problem:

One prevalent challenge in restaurant recommendation systems is the "cold start" problem, where the system struggles to provide accurate suggestions for new users or establishments lacking sufficient historical data. Sparse user input and the constant evolution of user preferences present additional hurdles, leading to less precise recommendations. Overcoming these issues necessitates the development of robust algorithms that can adapt to sparse data, effectively handle new user scenarios, and dynamically adjust to changing preferences to enhance the overall accuracy and relevance of restaurant recommendations.

References:

Key research journals in the field include the "Journal of Foodservice Business Research," "International Journal of Hospitality Management," and conferences like the "ACM Conference on Recommender Systems (RecSys)." Notable papers include "Personalized Restaurant Recommendations: Mining User-Generated Data" by Cheng-Kang Hsieh et al. and "Matrix Factorization Techniques for Recommender Systems" by Y. Koren et al. Online platforms such as arXiv.org and Google Scholar are valuable resources for accessing up-to-date research on restaurant recommendation systems.

Problem Statement Definition:

The restaurant recommendation system faces the challenge of efficiently providing accurate and personalized suggestions, particularly in scenarios involving new users or establishments with limited historical data. The "cold start" problem, stemming from insufficient information on user preferences, hampers the system's ability to deliver relevant recommendations. Sparse user input further complicates the task, leading to less precise suggestions. Additionally, the dynamic nature of user preferences and the ever-changing culinary landscape pose ongoing challenges, demanding adaptability to evolving trends and diverse user tastes. Addressing these issues requires the development of robust algorithms that can handle sparse data, effectively tackle the cold start problem, and dynamically adjust to shifting user preferences to enhance the overall performance and user satisfaction of the restaurant recommendation system.

IDEATION & PROPOSED SOLUTION

Empathy Map Canvas:

The restaurant recommendation system empathy map canvas is a tool designed to help project teams understand and empathize with the various stakeholders involved in the system's development and operation. This canvas consists of sections like: "See," "Think and Feel," "Hear," and "Say and Do."

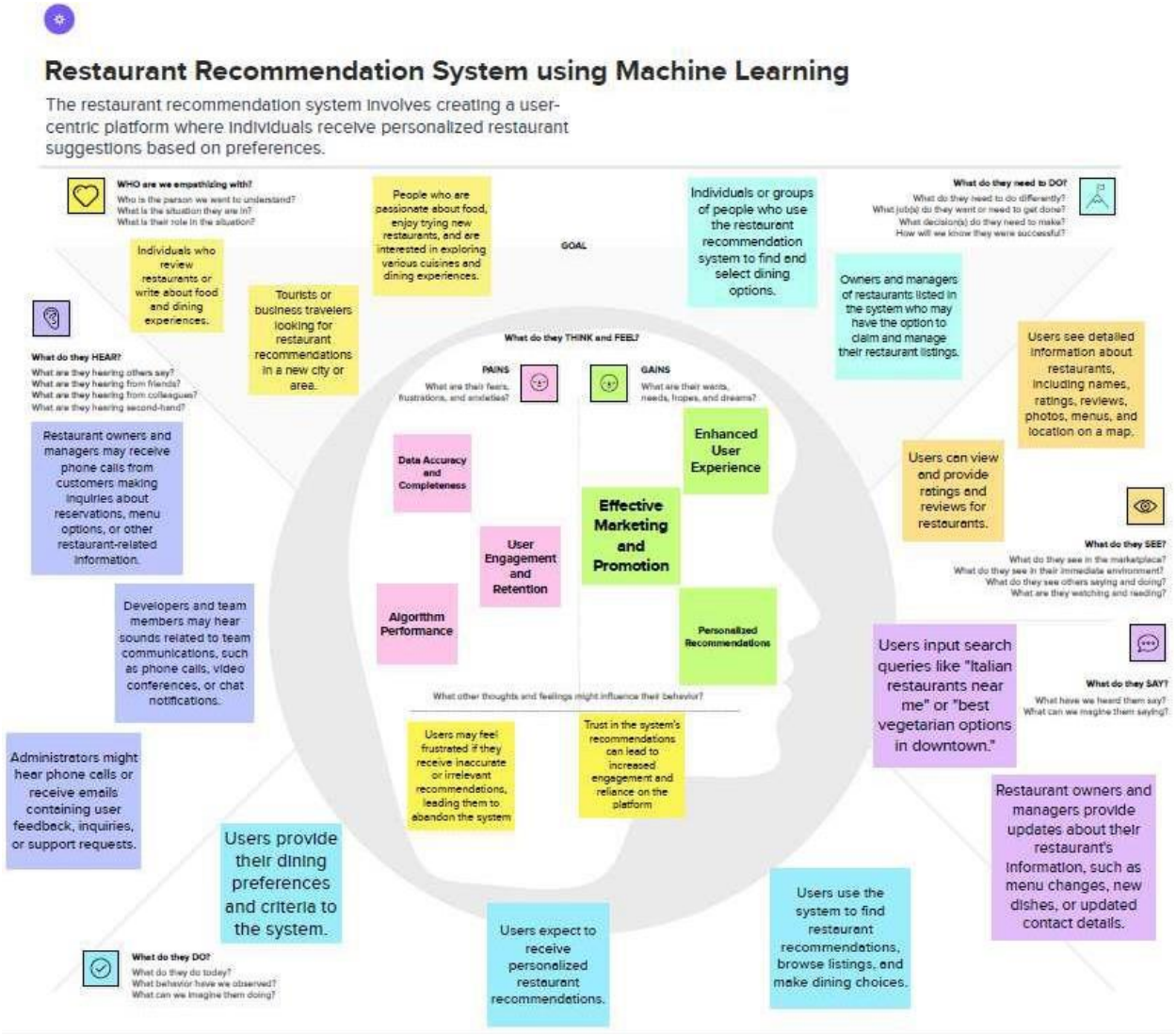
In the "See" section, stakeholders visually perceive elements such as user interfaces, restaurant listings, reviews, and data dashboards. This helps stakeholders understand what information and visuals are presented to users and how the system operates.

The "Think and Feel" section delves into stakeholders' thoughts and emotions. Users may experience excitement, trust, or frustration, while developers may feel pressure and satisfaction. Understanding these emotions guides decision-making and system improvements.

In the "Hear" section, stakeholders hear notification sounds, communicate with team members, and engage with user inquiries. Effective auditory elements can enhance the user experience and team collaboration.

The "Say and Do" section highlights user input, developers' coding and testing, administrators' content management, and marketing teams' promotional activities. Aligning communication and actions with stakeholder needs is crucial for project success.

Overall, the empathy map canvas serves as a valuable tool for project teams to gain insights into stakeholders' perspectives, ultimately leading to a more user-centred and successful restaurant recommendation system.



Reference link –
https://app.mural.co/t/restaurantrecommendationsyst8322/m/restaurantrecommendationsyst8322/1697607327806/7be6752168dac01fbdb6f46c4471e7c5ee672bd9?sender=u32fc412a972dc0ab35fb7270

Ideation & Brainstorming:

Brainstorming and idea prioritization are valuable processes for a project involving a restaurant recommendation system. It encourages team members to come up with a wide range of ideas. In the context of a restaurant recommendation system, this could include ideas for user interface features, recommendation algorithms, data sources, or user engagement strategies. More ideas provide a greater pool of potential improvements.

Brainstorming sessions often involve team members with diverse perspectives and expertise. This diversity can lead to more comprehensive and innovative ideas,

ensuring that different aspects of the project are considered. It also helps in fostering creativity and innovation. The brainstorming session typically begins with a discussion of project goals and challenges. This helps participants generate ideas that are aligned with the project's objectives, ensuring that the suggestions are relevant and valuable.

Idea prioritization involves evaluating each idea against predefined criteria, such as impact, feasibility, and alignment with project goals. This evaluation process helps identify which ideas are most likely to contribute significantly to the success of the restaurant recommendation system.

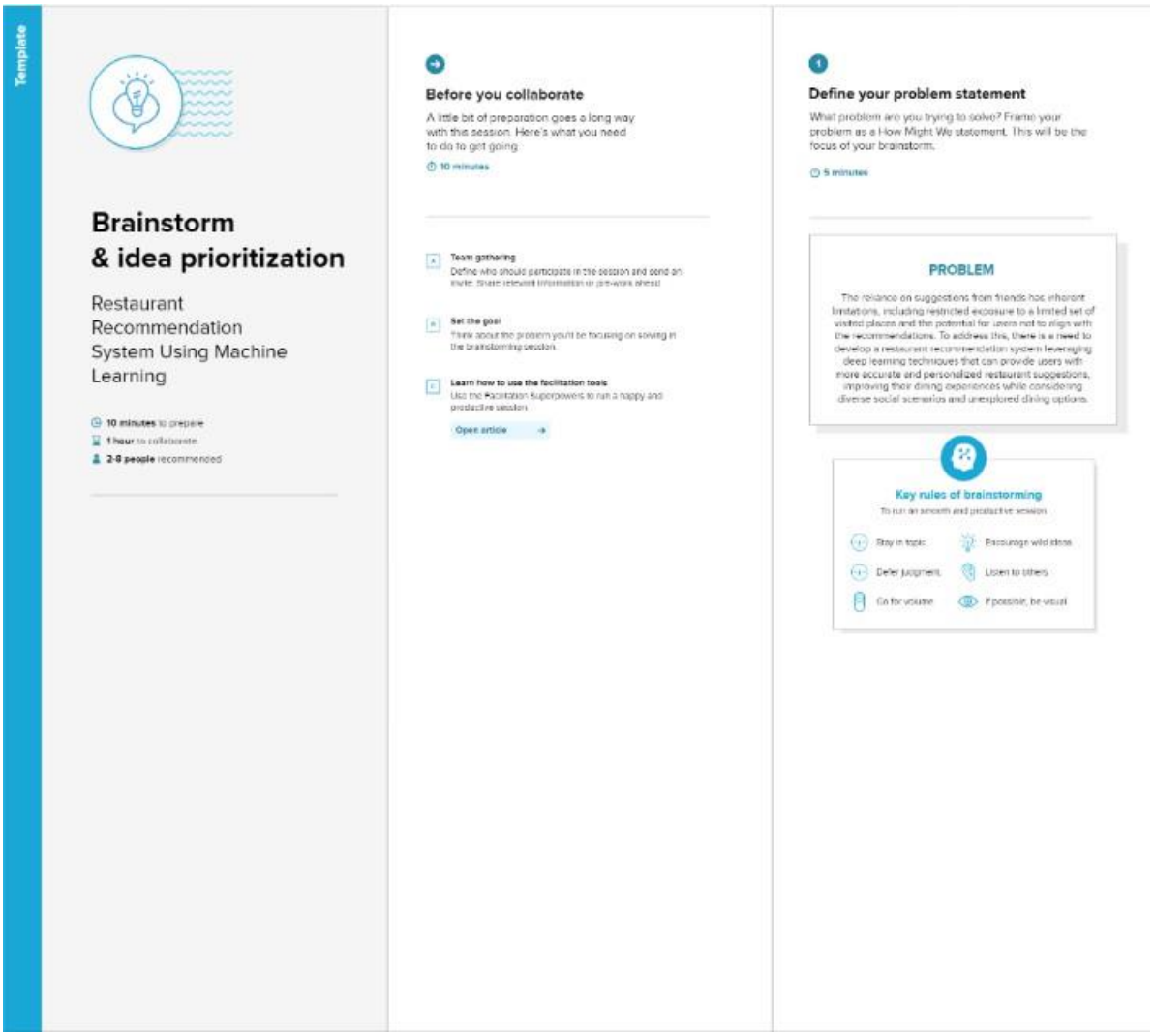
Prioritization helps allocate resources, such as time, budget, and manpower, more effectively. By focusing on ideas with the highest potential impact and feasibility, the team can make informed decisions on where to invest their efforts. It clarifies which ideas are most important and should be implemented as "Must-Have" features. This prevents the project from becoming overly complex and ensures that essential functionalities are addressed first.

Through evaluation, prioritization also helps identify potential risks and challenges associated with specific ideas. This allows the team to address these issues early in the project's development, reducing the likelihood of unexpected roadblocks.

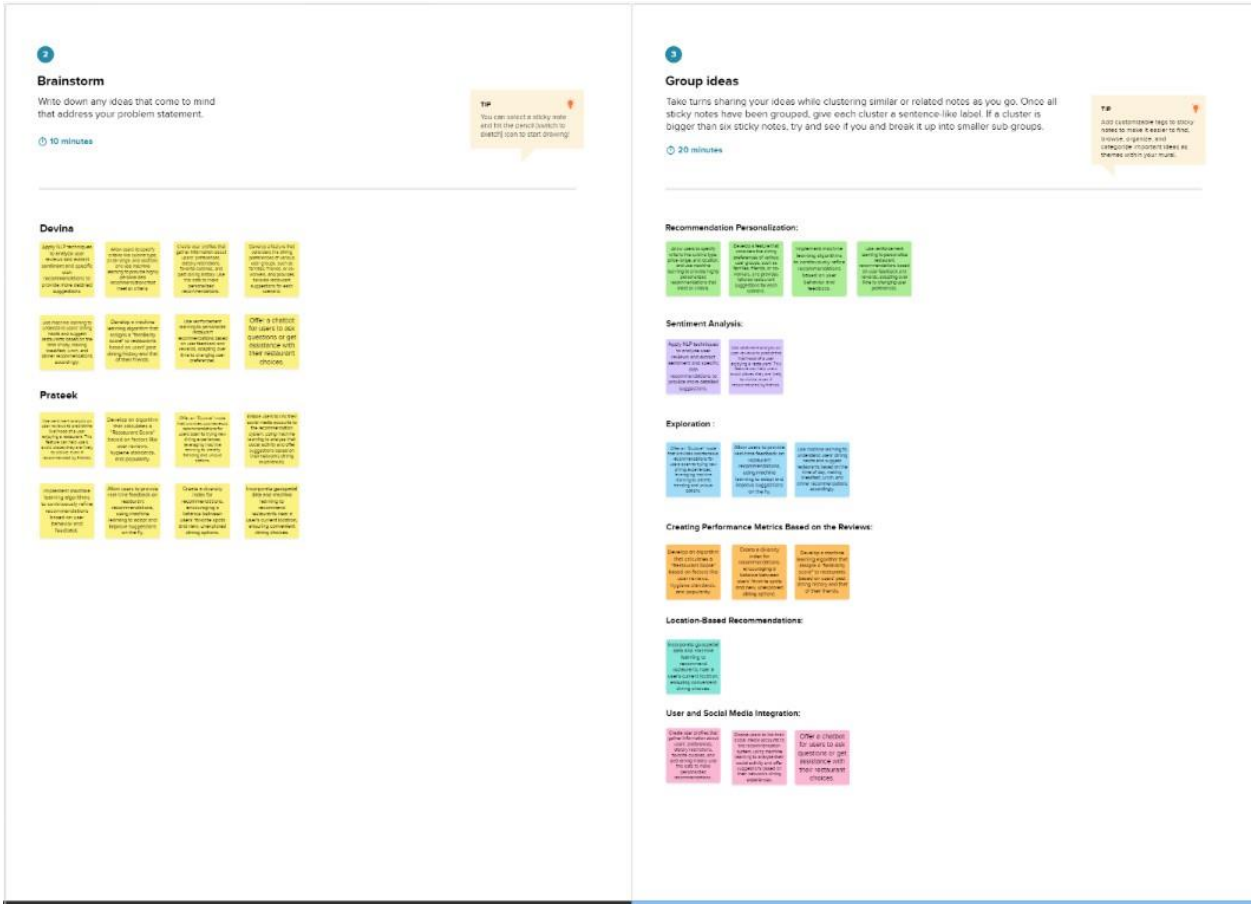
The prioritization process considers the potential impact on users, ensuring that the most valuable features are designed with the end-user in mind, ultimately leading to a more user-friendly restaurant recommendation system. By categorizing ideas as "Nice-to-Have" or "Future Consideration," prioritization allows for the possibility of incremental development.

In the context of a restaurant recommendation system, these processes are instrumental in creating a more effective, user-centric, and efficient system. They help the project team make informed decisions about which features and functionalities will provide the most value to users while staying within the project's constraints.

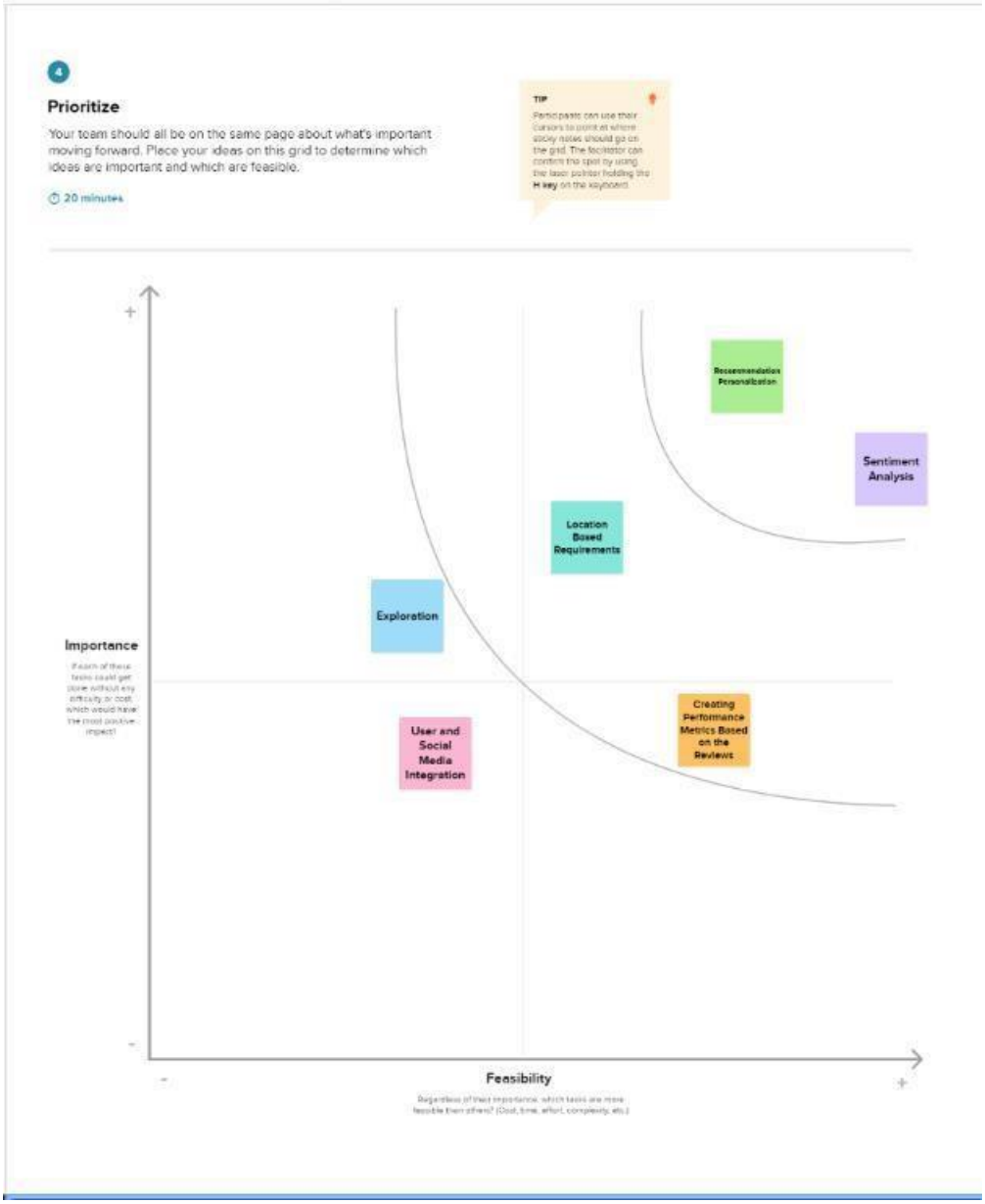
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



Reference link-
https://app.mural.co/t/devinagoel6641/m/devinagoel6641/1697635193908/e48f01960dbca529b5197b69b6b79b36a768b33c?sender=u27169e62cc239610fd242716

REQUIREMENT ANALYSIS

Functional Requirement:

The restaurant recommendation system must fulfill several key functional requirements to ensure an effective and user-friendly experience. Firstly, the system should collect and analyze user preferences, considering factors such as cuisine type, budget constraints,

and location. It must incorporate collaborative filtering and content-based recommendation techniques to identify patterns and similarities between users, delivering personalized suggestions based on historical dining data.

Real-time sentiment analysis of reviews is essential for maintaining the system's accuracy and relevance, ensuring that recommendations align with current positive trends and user sentiments. The interface should allow users to input specific preferences, providing a seamless and interactive experience.

The system should continuously learn and adapt, refining recommendations over time based on user feedback, evolving culinary trends, and changes in user preferences. To address the "cold start" problem, the system must intelligently handle new users or establishments by incorporating innovative algorithms that can generate meaningful recommendations in the absence of historical data.

Overall, the system should prioritize efficiency, accuracy, and user satisfaction, delivering a curated list of restaurant options that align with individual preferences and contribute to a positive and enjoyable dining experience.

Non-functional Requirement:

Non-functional requirements are crucial for shaping the overall performance, reliability, and user experience of a restaurant recommendation system. Firstly, performance requirements dictate the system's responsiveness, specifying that it must generate recommendations promptly, even during peak usage times, to ensure a seamless user experience. Availability requirements demand that the system be accessible and operational consistently, minimizing downtime and disruptions.

Scalability is vital to accommodate a growing user base and expanding restaurant database, ensuring the system can handle increased loads without compromising performance. Reliability requirements necessitate a robust and fault-tolerant system, capable of recovering gracefully from failures or errors to maintain continuous service.

User experience considerations involve usability requirements, ensuring an intuitive and easy-to-navigate interface for users to input preferences effortlessly. Security requirements are imperative, safeguarding user data and ensuring the confidentiality and integrity of sensitive information.

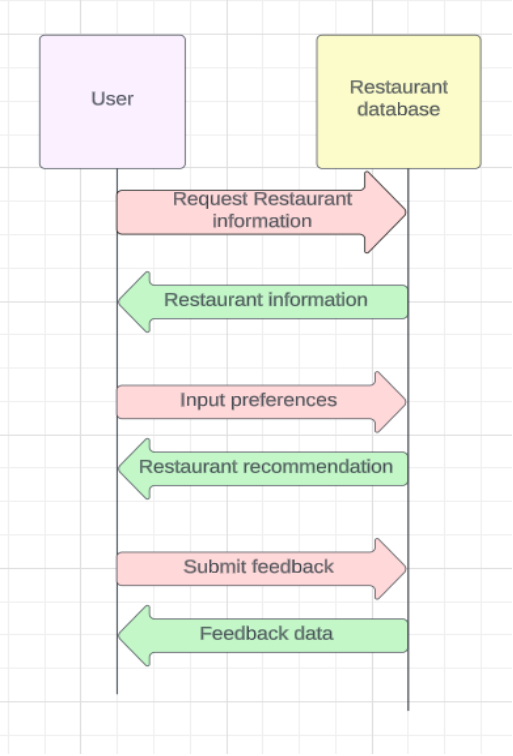
In terms of compatibility, the system should be compatible with various devices and platforms, ensuring a broad user reach. Finally, maintainability requirements dictate that the system should be designed for easy updates, enhancements, and troubleshooting to facilitate long-term sustainability.

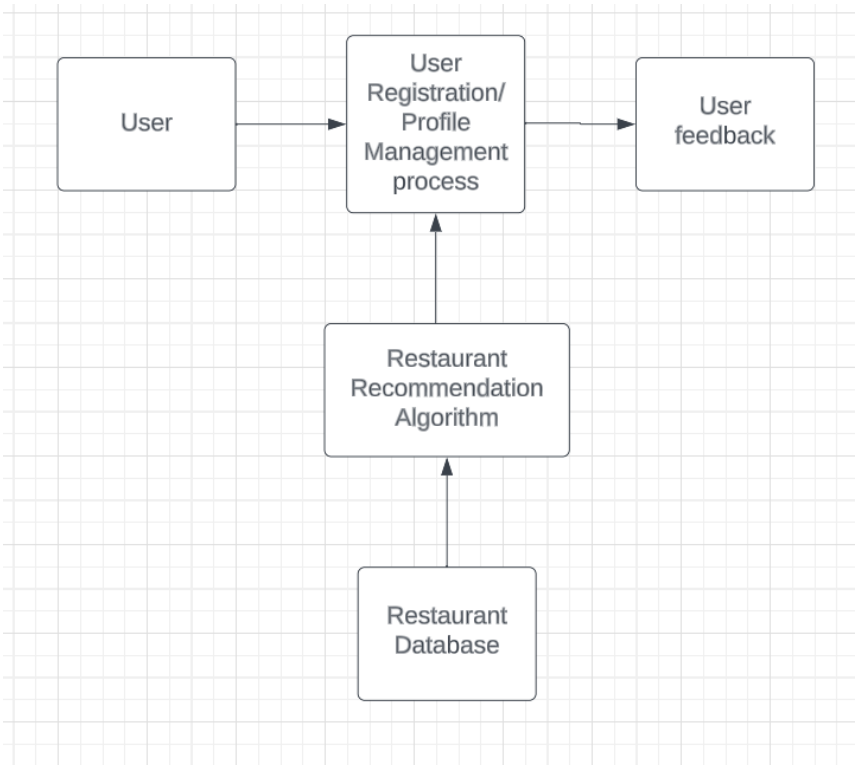
By addressing these non-functional requirements comprehensively, the restaurant recommendation system can deliver a reliable, scalable, secure, and user-friendly experience, meeting user expectations and industry standards.

PROJECT DESIGN

Data Flow diagram & User stories:

A Data Flow Diagram (DFD) for a restaurant recommendation system is a visual representation that outlines how data is processed and moves through the system. It typically includes external entities like users and databases, processes such as user profile management, recommendation algorithms, and feedback processing, and data stores for storing user profiles, restaurant information, and feedback data. Data flows depict the transfer of information between these elements, like user preferences feeding into the recommendation algorithm and the algorithm providing restaurant recommendations to users. DFDs help in understanding the system's data flow, highlighting key components, and identifying how data is transformed and utilized to offer personalized restaurant recommendations.





User stories-

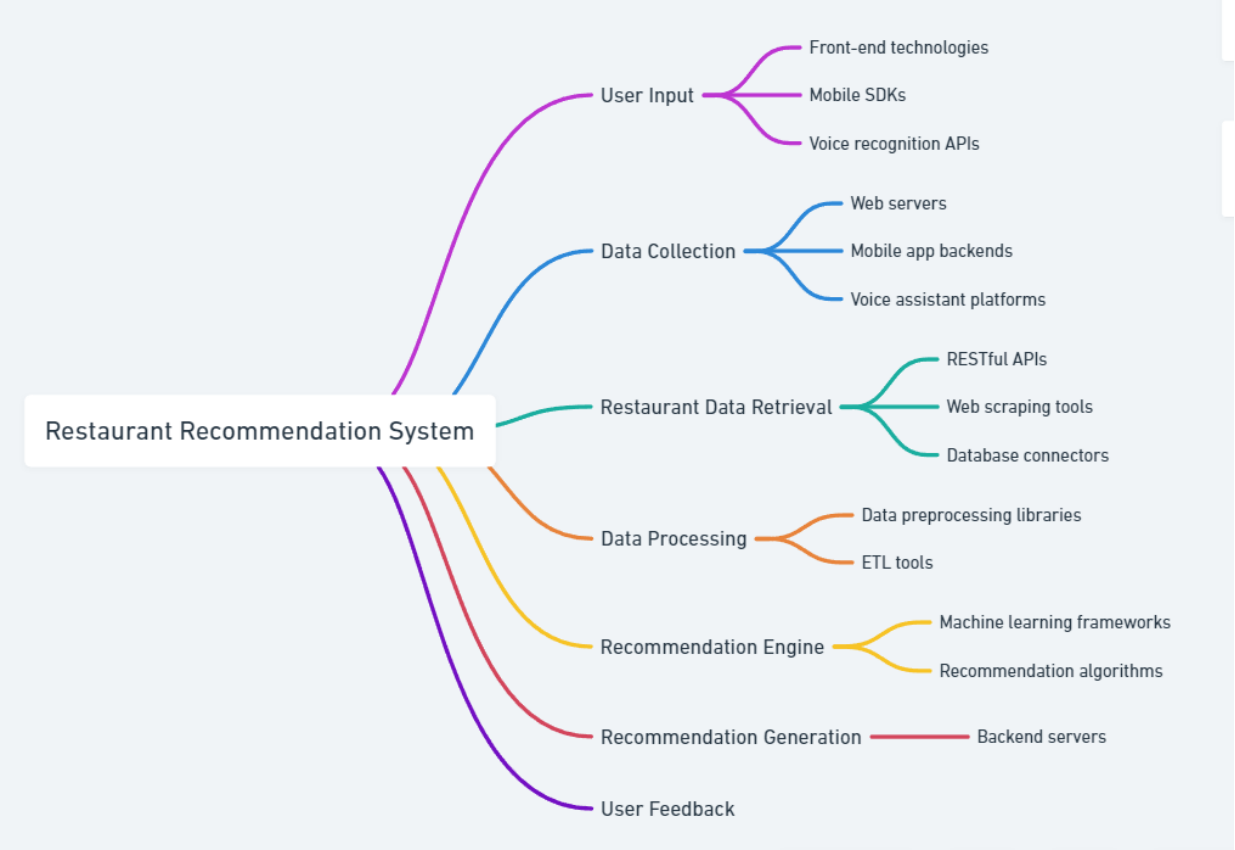
User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Web user)	Recommendations	USN-1	As a user, I want to browse and search for restaurants based on location, cuisine, price range, and user ratings, to discover new dining options.	I can browse and search based on location, cuisine, price, rating, etc.	High	Sprint-1
		USN-2	As a user, I want to specify my dining preferences, such as cuisine, dietary restrictions, and preferred location, so that the system can provide personalized restaurant recommendations.	I can specify my dining preferences.	High	Sprint-1
		USN-3	As a user, I want the system to learn from my feedback and past restaurant choices to improve the relevance of future recommendations.	I can learn from past feedbacks and choices.	Low	Sprint-2
Restaurant owner	Dashboard	USN-4	As a restaurant owner, I want to add my restaurant's information, including menus, photos, and special offers, to attract potential customers.	I can add restaurant's information.	Low	Sprint-2
		USN-5	As a restaurant owner, I want to respond to user reviews and feedback to maintain a positive online presence.	I can respond to user reviews.	Low	Sprint-2

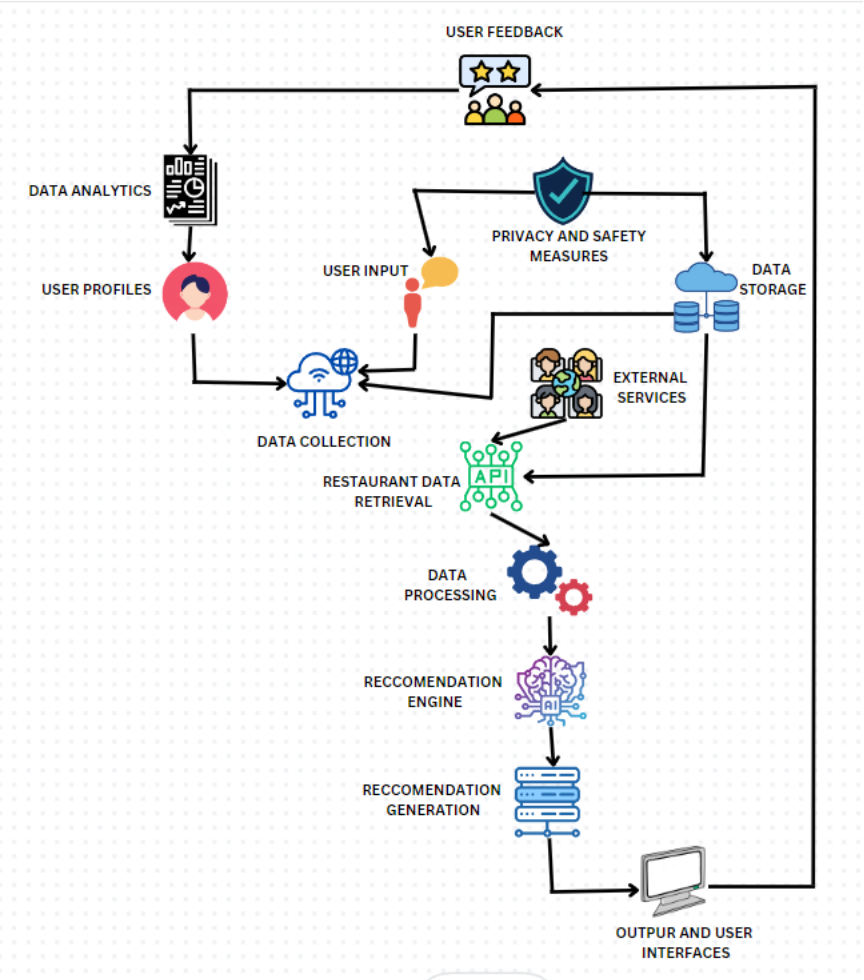
Solution Architecture:

A solution architecture for a restaurant recommendation system outlines the high-level design and structure of the system. In brief, here are the key components and concepts typically included in such an architecture:

1. **User Interfaces:** This includes web and mobile applications, as well as voice assistants, where users interact with the system to request restaurant recommendations.

2. **Recommendation Engine:** The core component that employs machine learning algorithms, such as collaborative filtering and content-based filtering, to generate personalized restaurant suggestions based on user preferences and behavior.
3. **Data Management:** Encompasses the collection, storage, and integration of restaurant data, user profiles, and external data sources. It ensures that data is clean, structured, and readily available for recommendation algorithms.
4. Security and Privacy: Addresses user data protection, privacy, and security through user authentication, data encryption, and compliance with relevant regulations.
5. **Scalability and Performance:** Ensures the system can handle increasing numbers of users and data while maintaining optimal performance. This may involve load balancing, caching, and content delivery networks.
6. **User Feedback and Ratings:** Allows users to rate and review restaurants, enabling continuous improvement of the recommendation system.
7. **Integration with External Services:** Integrates with social networks, reservation services, mapping services, and other external platforms to enhance user experience.
8. **Monitoring and Analytics:** Monitors system performance and user behavior, offering insights for system optimization and user engagement.
9. **Development and Deployment:** Implements DevOps practices for continuous integration and deployment, potentially hosting the system on cloud infrastructure for flexibility.
10. **Documentation and Knowledge Sharing:** Provides comprehensive documentation for maintenance, troubleshooting, and onboarding of team members.
This architecture serves as a roadmap for building a restaurant recommendation system that aligns with business goals, optimizes user experience, and addresses technical requirements.




USER FEEDBACK
DATA ANALYTICS
PRIVACY AND SAFETY MEASURES
USER INPUT
DATA STORAGE
USER PROFILES
EXTERNAL SERVICES
DATA COLLECTION
RESTAURANT DATA RETRIEVAL
API
DATA PROCESSING
RECCOMENDATION ENGINE
AI
RECCOMENDATION GENERATION
OUTPUR AND USER INTERFACES

PROJECT PLANNING & SCHEDULING

Technical Architecture:

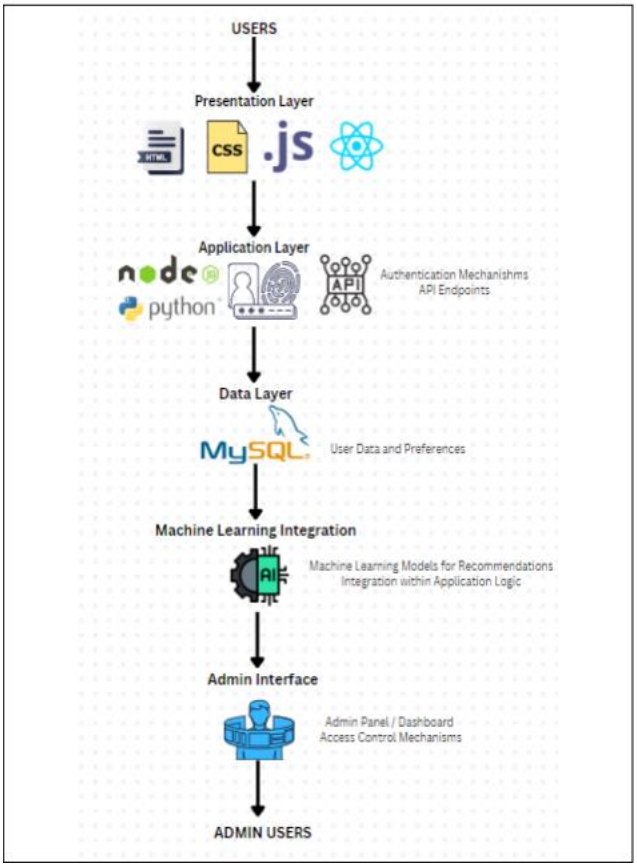


Table-1 : Components & Technologies:

S. No	Component	Description	Technology
1.	Data Collection	Gathering restaurant data from various sources	Web Scraping (BeautifulSoup, Scrapy)
2.	Data Preprocessing	Cleaning and preparing data for modelling	Pandas, Numpy
3.	Data Storage	Storing restaurant information	SQL or NoSQL Databases (MySQL, MongoDB)
4.	Feature Engineering	Extracting and engineering relevant features	Pandas, Scikit-learn
5.	Modelling	Building recommendation models	Scikit-learn, TensorFlow, PyTorch
6.	API Development	Creating backend APIs for model interaction	Flask.
7.	Serialization	Handling data interchange between frontend and backend	JSON
8.	Frontend Development	Designing user interfaces	HTML, CSS, Javascript
9.	Frontend Frameworks	Building interactive UIs	React, Vue.js, Angular
10.	Mapping and Geolocation	Incorporating mapping and location-based services	Google Maps API
11.	Cloud Services and Deployment	Hosting and deploying the application	AWS, Google Cloud Platform, Azure, Docker, Kubernetes.
12.	Recommendation Algorithms	Implementing recommendation strategies	Collaborative Filtering, Content-based Filtering
13.	User Feedback Mechanisms	Collecting and utilizing user ratings and reviews	Ratings, Reviews, Implicit Feedback
14.	Version Control	Collaborative development and version control	Git

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Utilized open-source frameworks for component development	React, Python, Java, Node.js, MySQL, Google Cloud Storage, Flask, Skicit-learn
2.	Security Implementations	Implemented security measures and access controls	SHA-256, TLS/SSL, Encryption, IAM (Identity and Access Management), OWASP (Open Web Application Security Project)
3.	Scalable Architecture	Three-tier architecture provides scalability by allowing independent scaling of presentation, application, and data layers. Its simplicity, modular scalability, and flexibility to scale individual components make it a suitable choice.	HTML, CSS, JavaScript, React, Python, Java, Node.js, Flask, MySQL, Web Servers, ORMs (Object-Relational Mapping), API Protocols
4.	Availability	Ensuring application availability through load balancers	Load Balancers, Distributed Servers
5.	Performance	Caching Mechanisms, Content Delivery Networks (CDNs), Optimized Request Handling	HTTP/3, Optimized Routing Algorithms, HTTP Cache-Control Headers, Web Accelerators etc.

Reference:
https://www.canva.com/design/DAF0hXhXTp0/lMAEthodcXNRRTi-zttV0Q/view?utm_content=DAF0hXhXTp0&utm_campaign=designshare&utm_medium=link&utm_source=editor

Sprint Planning & Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint - 1	Data Collection and Preprocessing	US1	Scrape restaurant data from various sources	5	High	Prateek
		US2	Clean and preprocess scraped data	3	High	Prateek
Sprint - 1	Database Setup and Management	US3	Design database schema	5	High	Devina
		US4	Implement database setup	3	High	Devina
Sprint - 1	Feature Engineering	US5	Extract relevant features	5	Medium	Prateek
Sprint - 2	Model Development	US6	Research machine learning models	5	High	Prateek
		US7	Begin training initial models	3	High	Prateek
Sprint - 2	API Development	US8	Design API endpoints	5	High	Devina

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint - 3	Model Development	US9	Evaluate model performance	5	High	Prateek
Sprint - 3	Frontend Development	US10	Design user interface	5	High	Devina
		US11	Implement frontend	5	High	Devina
Sprint - 4	Frontend Development	US12	Complete frontend development	8	High	Devina
Sprint - 4	Testing and Quality Assurance	US13	Conduct unit and integration tests	5	High	Prateek
Sprint - 5	Model Development	US14	Refine machine learning models	8	High	Prateek
	API Development	US15	Complete API development	5	High	Devina
Sprint - 6	Testing and Bug Fixes	US16	Comprehensive testing	8	High	Devina
Sprint - 6	Deployment	US17	Deploy system on cloud	5	High	Prateek

Sprint Delivery Schedule:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint - 1	21	6 Days	14 Oct 2023	19 Oct 2023	20	19 Oct 2023
Sprint - 2	13	6 Days	20 Oct 2023	25 Oct 2023	13	25 Oct 2023
Sprint - 3	20	6 Days	26 Oct 2023	31 Oct 2023	20	31 Oct 2023
Sprint - 4	13	6 Days	1 Nov 2023	6 Nov 2023	13	6 Nov 2023
Sprint - 5	13	6 Days	7 Nov 2023	12 Nov 2023	13	12 Nov 2023
Sprint - 6	13	3 Days	13 Nov 2023	15 Nov 2023	13	15 Nov 2023

CODING & SOLUTIONING

Feature 1:

PRE-REQUISITES:

To complete the project successfully, you need to install following software & packages:

Activity 1: Install Anaconda IDE / Anaconda Navigator.

- In order develop a solution to this problem statement, we need an environment to write and test the code.
- We use Anaconda IDE (Integrated Developing Environment).
- Refer to the below link to download & install Anaconda Navigator.

Link: https://www.youtube.com/watch?v=5mDYijMfSzs

1. Activity 2: To build Machine learning models you must require the following packages

- Numpy:

It is an open-source numerical Python library. It contains a multidimensional array and matrix data structures and can be used to perform mathematical operations

- Numpy:

It is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy

- Matplotlib and Seaborn

Matplotlib is mainly deployed for basic plotting. Visualization using Matplotlib generally consists of bars, pies, lines, scatter plots and so on. Seaborn: Seaborn, on the other hand, provides a variety of visualization patterns. It uses fewer syntax and has easily interesting default themes.

- Flask:

Web framework used for building Web applications

If you are using anaconda navigator, follow below steps to download required packages:

- Open anaconda prompt.
- Type “pip install pandas” and click enter.
- Type “pip install matplotlib” and click enter.
- Type “pip install seaborn” and click enter.
- Type “pip install plotly” and click enter.
- Type “pip install numpy” and click enter.
- Type “pip install scikit-image” and click enter.
- Type “pip install scikit-learn” and click enter.
- Type “pip install Flask” and click enter.

Link: Introduction to Scikit-Learn (sklearn) in Python • datagy

PRIOR KNOWLEDGE

One should have knowledge on the following Concepts:

Link: Supervised and Unsupervised Learning

Watch the below video to know about the types of machine learning

Link: Regression, Classification and Clustering

Link: ML - Content Based Recommender System - GeeksforGeeks

Link: NLTK :: Natural Language Toolkit

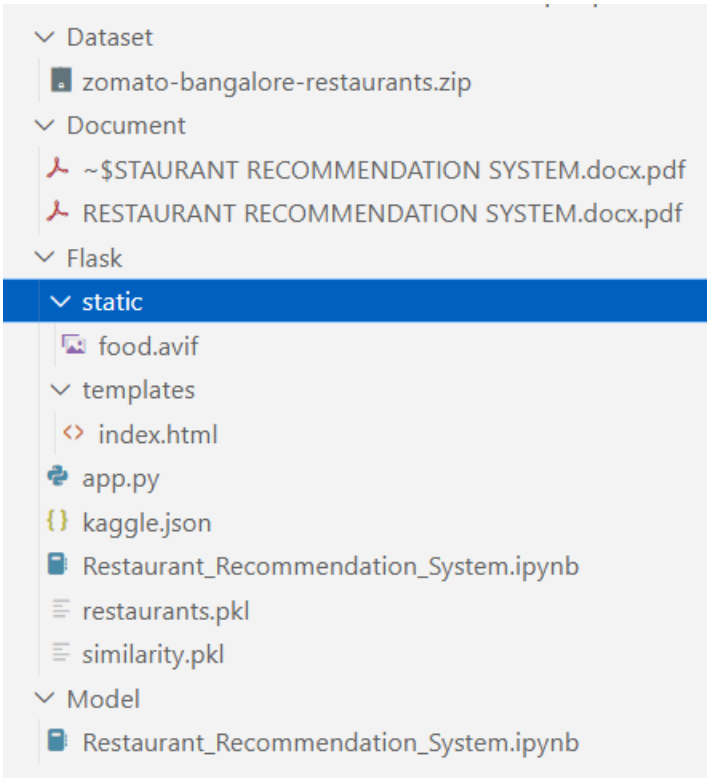
Link: Flask:

Link: Recommendation System

It is recommended to watch above video’s to understand the concepts before you start your project.

PROJECT WORK FLOW

- User interacts with the UI (User Interface) to enter the input features.
- Entered features/input is analysed by the model which is integrated
- Once model analyses the entered inputs, the prediction is showcased on the UI.



TASKS

1. Data Collection.

- Collect the dataset or Create the dataset

2. Data Pre- processing.

Import the Libraries.
Importing the dataset. Exploratory Data Analysis Data Visualization.
3. Content Based Filtering

Merging datasets
Creating the recommender system
Predicting the results
4. Application Building Create an HTML file

Build a Python Code

Milestone 1: Data Collection

Now, the milestone-2 is all about creation or collection of dataset.
we will use the **Zomato** Bangalore for our analysis to draw conclusions using the content filtering method.
Here is the dataset link: **dataset**

Milestone 2: Data Pre-processing

In this milestone, you need to complete all the below activities to build the model.

Activity 1: Import Libraries

Import the below essential libraries for data pre-processing and creating recommendation system. Pandas and NumPy are used for data pre-processing and cleaning. Seaborn, Plotly and Matplotlib helped in creating visual graphics and bar plots for the dataset. Also, since there would be cleaning of text data (reviews) as well, therefore for that we will use nltk and sklearn library.

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
import plotly.offline as py
import plotly.graph_objs as go
import seaborn as sns

import nltk
from nltk.corpus import stopwords
from sklearn.metrics.pairwise import linear_kernel
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

Activity 2: Read the Dataset:

Our dataset format might be in .csv, excel files, .txt, json, etc. We can read the dataset with the help of pandas.

In pandas we have a function called read_csv () to read the dataset. As a parameter we have to give the directory of csv file.

```
[142]
import numpy as np
import pandas as pd
```
Python

```
[143]
df = pd.read_csv('/content/zomato.csv')
df.head()
```
Python

	url	address	name	online_order	book_table	rate	votes	phone	location	rest_type	dish_liked
0	https://www.zomato.com/bangalore/jalsa-banasha...	942, 21st Main Road, 2nd Stage, Banashankari, ...	Jalsa	Yes	Yes	4.1/5	775	080 42297555\r\n+91 9743772233	Banashankari	Casual Dining	Pasta, Lunch Buffet, Masala Papad, Paneer Laja...
1	https://www.zomato.com/bangalore/spice-elephan...	2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...	Spice Elephant	Yes	No	4.1/5	787	080 41714161	Banashankari	Casual Dining	Momos, Lunch Buffet, Chocolate Nirvana, Thai G...
2	https://www.zomato.com/SanchurroBangalore?cont...	1112, Next to KIMS Medical College, 17th Cross...	San Churro Cafe	Yes	No	3.8/5	918	+91 9663487993	Banashankari	Cafe, Casual Dining	Churros, Cannelloni, Minestrone Soup, Hot Choc...
3	https://www.zomato.com/bangalore/addhuri-udupi...	1st Floor, Annakuteera, 3rd Stage, Banashankar...	Addhuri Udupi Bhojana	No	No	3.7/5	88	+91 9620009302	Banashankari	Quick Bites	Masala Dosa

Activity 3: Analyse the Dataset:

Data preprocessing

```
[144] df.shape
```

```
(51717, 17)
```

The dataset contains 51717 records with 17 features.

Checking the columns in the dataset.

Columns description

1. **URL** contains the url of the restaurant on the Zomato website

2. **address** contains the address of the restaurant in Bengaluru

3. **name** contains the name of the restaurant

4. **online_order** whether online ordering is available in the restaurant or not

5. **book_table** table book option available or not

6. **rate** contains the overall rating of the restaurant out of 5

7. **votes** contain the total number of rating for the restaurant as of the above-mentioned date

8. **phone** contains the phone number of the restaurant

9. **location** contains the neighbourhood in which the restaurant is located

10. **rest_type** restaurant type

11. **dish_liked** dishes people liked in the restaurant

12. **cuisines** food styles, separated by comma

13. **approx_cost(for two people)** contains the approximate cost for a meal for two people

14. **reviews_list** list of tuples containing reviews for the restaurant, each tuple

15. **menu_item** contains a list of menus available in the restaurant

16. **listed_in(type)** type of meal

17. **listed_in(city)** contains the neighbourhood in which the restaurant is listed.

Understanding Overview of features

- How the information is stored in a DataFrame or Python object affects what we can do with it and the outputs of calculations as well. There are two main types of data those are numeric and text data types.
- Numeric data types include integers and floats.
- Text data type is known as Strings in Python, or Objects in Pandas.

Strings can contain numbers and / or characters.

- For example, a string might be a word, a sentence, or several sentences.
- Will see how our dataset is, by using **info ()** method.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 43499 entries, 0 to 51716
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   address       43499 non-null  object
 1   name          43499 non-null  object
 2   online_order  43499 non-null  object
 3   book_table    43499 non-null  object
 4   rate          43499 non-null  object
 5   votes         43499 non-null  int64 
 6   location      43499 non-null  object
 7   rest_type     43499 non-null  object
 8   cuisines      43499 non-null  object
 9   cost          43499 non-null  object
 10  reviews_list  43499 non-null  object
 11  menu_item     43499 non-null  object
 12  type          43499 non-null  object
 13  city          43499 non-null  object
dtypes: int64(1), object(13)
memory usage: 5.0+ MB
```

• As you can see in our dataset, except 'votes', all other features are categorical data, but it is not necessary that all the continuous data which we are seeing has to be continuous in nature. There may be a case that some categorical data is in the form of numbers but when we perform info () operation we will get numerical output. So, we need to take care of those type of data also.

Checking for null values in the dataset

```
df.isnull().sum()
```

```
url                                0
address                            0
name                               0
online_order                       0
book_table                         0
rate                            7775
votes                              0
phone                           1208
location                          21
rest_type                        227
dish_liked                     28078
cuisines                          45
approx_cost(for two people)      346
reviews_list                       0
menu_item                          0
listed_in(type)                    0
listed_in(city)                    0
dtype: int64
```

Data cleaning as our dataset contains null values and some special characters

```
df = df.drop(columns = ['url','phone','dish_liked'])
df.head(2)
```

```
df = df.rename(columns = {'approx_cost(for two people)':'cost','listed_in(type)':'type','listed_in(city)':'city'})
df.head(5)
```

```
# Converting the rate column to float type
df = df.loc[df['rate'] != 'NEW']
df = df.loc[df['rate'] != '-'].reset_index(drop = True)
slash = lambda x: x.replace('/5','') if type(x) == str else x
df['rate'] = df['rate'].apply(slash).str.strip().astype('float')
```

```
# Converting the cost column to float type
df['cost'] = df['cost'].astype(str)
df['cost'] = df['cost'].apply(lambda x: x.replace(',','.'))
df['cost'] = df['cost'].astype(float)
```

Checking for null values after cleaning & data Processing

```
zomato_df.isnull().sum()
```

```
address         0
name            0
online_order    0
book_table      0
rate            0
votes           0
location        0
rest_type       0
cuisines        0
cost            0
reviews_list    0
menu_item       0
type            0
city            0
dtype: int64
```

Checking mean rating with restaurant name and rating for each restaurant using below line codes

Mean rating for each restaurants

```
## Computing Mean Rating
restaurants = list(zomato_df['name'].unique())
zomato_df['Mean Rating'] = 0
for i in range(len(restaurants)):
    zomato_df['Mean Rating'][zomato_df['name'] == restaurants[i]] = zomato_df['rate'][zomato_df['name'] == restaurants[i]].mean()
#Scaling the mean rating values
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range = (1,5))
zomato_df[['Mean Rating']] = scaler.fit_transform(zomato_df[['Mean Rating']]).round(2)
```

Checking the mean rating with restaurant name and rating

```
zomato_df[['name','rate','Mean Rating']].head()
```

	name	rate	Mean Rating
0	Jalsa	4.1	3.99
1	Spice Elephant	4.1	3.97
2	San Churro Cafe	3.8	3.58
3	Addhuri Udupi Bhojana	3.7	3.45
4	Grand Village	3.8	3.58

We will be using the 'Review' and 'Cuisines' feature in order to create a recommender system. So we need to prepare and clean the text in those columns.

Operations performed: Lower Casing, Removal of Punctuations, Removal of Stop words, Removal of URLs, Spelling correction

Milestone 3: Data Visualization

Data visualization is where a given data set is presented in a graphical format. It helps the detection of patterns, trends and correlations that might go undetected in text-based data. Understanding your data and the relationship present within it is just as important as any algorithm used to train your machine learning model. In fact, even the most sophisticated machine learning models will perform poorly on data that wasn't visualized and understood properly.

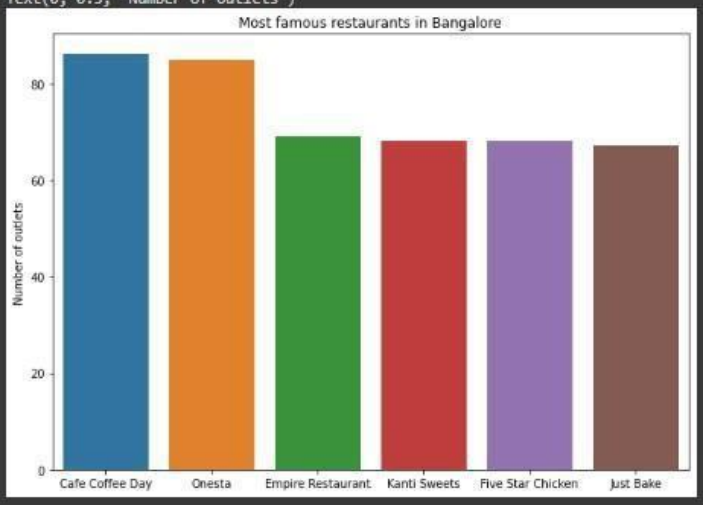
To visualize the dataset, we need libraries called Matplotlib, Seaborn. The Matplotlib library is a Python 2D plotting library which allows you to generate plots, scatter plots, histograms, bar charts etc.

Let's visualize our data using Matplotlib and seaborn library.

At first, we will be plotting a bar plot using matplotlib for showing the top 6 restaurants in Bangalore by value counts.

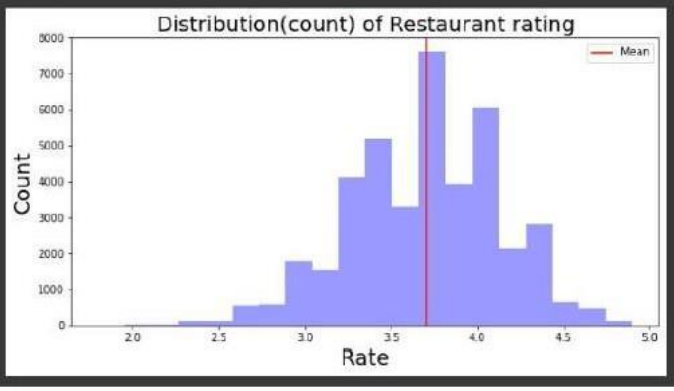
```
#Most Famous 6 restaurants in Banglore

plt.figure(figsize=(10,7))

chains=zomato_df['name'].value_counts()[:6]

sns.barplot(x=chains.index,y=chains,palette='tab10')

plt.title("Most famous restaurants in Bangalore")

plt.ylabel("Number of outlets")
```

```
Text(0, 0.5, 'Number of outlets')
```



Checking the distribution of restaurant rating, for that we are using distplot from seaborn library.

```
#Distribution of Restaurant Rating

fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(10, 5))
sns.distplot(zomato_df.rate,kde=False,color = 'b',ax =ax,bins=20);
ax.axvline(zomato_df.rate.mean(), 0, 1, color='r', label='Mean')
ax.legend();
ax.set_ylabel('Count',size=20)
ax.set_xlabel('Rate',size=20)
ax.set_title('Distribution(count) of Restaurant rating',size=20);
```



Here we can see the Top favourite cuisine among people of Bangalore is 'North Indian', 'Indian Chinese' and 'Fast food'.

Milestone 4: CONTENT-BASE RECOMMENDER SYSTEM

Creating the required dataset for recommendation system

```
# Group by 'name' column and calculate the mean of 'rate' column
grouped_df = df.groupby('name', as_index=False)['rate'].mean().round(2)

# Combine all reviews and cuisines for each unique name
combined_df = df.groupby('name', as_index=False).agg({'reviews_list': 'sum', 'cuisines': 'sum'})

# Merge the two dataframes on 'name' column
new_df = pd.merge(grouped_df, combined_df, on='name')

# Print the resulting dataframe
new_df.tail()
```

	name	rate	reviews_list	cuisines
6597	i-Bar - The Park Bangalore	3.80	rated 40 ratedn a fab place to hangout with f...	North Indian Chinese MediterraneanNorth Indian...
6598	iFruit Live Ice Creams	3.40	rated 40 ratedn the sharjah milkshake used to...	Ice CreamIce CreamIce CreamIce Cream
6599	iSpice Resto Cafe	3.70	rated 50 ratedn food 55 expertise in breakfa...	Cafe North Indian Chinese Fast FoodCafe North ...
6600	nu.tree	4.31	rated 40 ratedn i ordered their veg meal at o...	North Indian Healthy Food BeveragesNorth India...
6601	re:cess - Hilton Bangalore Embassy GolfLinks	4.10	rated 50 ratedn a big thanks for the last nig...	South Indian North Indian Continental European...

Recommendation system

```
# Importing required modules
import nltk
from nltk.corpus import stopwords
from sklearn.metrics.pairwise import linear_kernel
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
cv = CountVectorizer(max_features=5000,stop_words='english')
```

```
vector = cv.fit_transform(new_df['tags']).toarray()
```

```
vector.shape
```

```
(6602, 5000)
```

```
from sklearn.metrics.pairwise import cosine_similarity
similarity = cosine_similarity(vector)
similarity
```

```
def recommend(restaurant):
    index = new_df[new_df['name'] == restaurant].index[0]
    distances = sorted(enumerate(similarity[index]), reverse=True, key=lambda x: x[1])
    for i in distances[1:11]:
        print(new_df.iloc[i[0]]['name'])
```

Calculating Cosine Similarity

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

The formula for Cosine Similarity

And in the last line of code, we are calculating the cosine similarity of each item with every other item in the dataset. So we just pass the matrix as an argument.

Querying recommendation for 4 Restaurants:

```
recommend('Jalsa')
```

```
The Black Pearl
Fenny's Lounge And Kitchen
Crawl Street
Vapour Brewpub and Diner
Funjabi
The Hidden Home
Hangover
Sotally Tober
Vapour Pub & Brewery
Urban Tamaasha
```

```
recommend('San Churro Cafe')
```

```
Cafe Mondo
Sidewalk Cafe - Nahar's Heritage Hotel
Caffe Pascucci
The French Loaf
Three Dots & A Dash
Cafe Pink Pajamas
Mr. Beans - Home Cafe
Cafe Happytizing
House Of Commons
Dyu Art Cafe
```

```
recommend('Grand Village')
```

```
Curry with a 'K' - St. Mark's Hotel
24th Main
1947
Atithi
Flavours - Octave Hotel & Spa
Elmas Restaurant
Kakal Kai Ruchi
Dal Tadkaa
Citrus Cafe - Lemon Tree Hotel
Pepper's - The Palladium
```

```
recommend('#FeelTheROLL')
```

```
Happy Hours
Cafe Aladdin
Dessi Cuppa
Ilyazsab The House Of Chicken
ANTIGRAVITY
7th Heaven
Ackley Kitchen
Jay Bhavani
Shizusan Shophouse & Bar
Bloomsbury's Global Kitchen & Bakehouse
```

```
recommend('1000 B.C')
```

```
Subway
Apna Hotel
North Wale Restaurant
Bombay Sandwich Company
This Cafe Has No Name Cafe
Silver Spoon Restaurant
Raenss Cafe
Hungry Paunch
Food Point
Chakh Le India
```

Milestone 5: Application Building

Activity 1: Create an HTML File

We use HTML to create the front end part of the web page.
Here, we created 2 html pages- index.html, web.html. index.html displays home page. web.html accepts the values from the input and displays the prediction. For more information regarding HTML refer the link below

https://www.w3schools.com/bootstrap/bootstrap_forms_inputs.asp

- We also use JavaScript-main.js and CSS-main.css to enhance our functionality and view of HTML pages.
  - **Link :https://www.w3schools.com/css/**
  - **https://www.w3schools.com/js/DEFAULT.asp**

Activity 2: Build python code

- Let us build flask file 'app1.py' which is a web framework written in python for server-side scripting. Let's see step by step procedure for building the backend application.
- **App starts running when "__name__" constructor is called in main.**
- render template is used to return html file.
- **"GET" method is used to take input from the user.**
- **"POST" method is used to display the output to the user.**

Importing libraries

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
import plotly.offline as py
import plotly.graph_objs as go
import seaborn as sns
import warnings
warnings.filterwarnings('always')
warnings.filterwarnings('ignore')
import nltk
from nltk.corpus import stopwords
from sklearn.metrics.pairwise import linear_kernel
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
import flask
from flask import Flask,render_template, request
import pickle
```

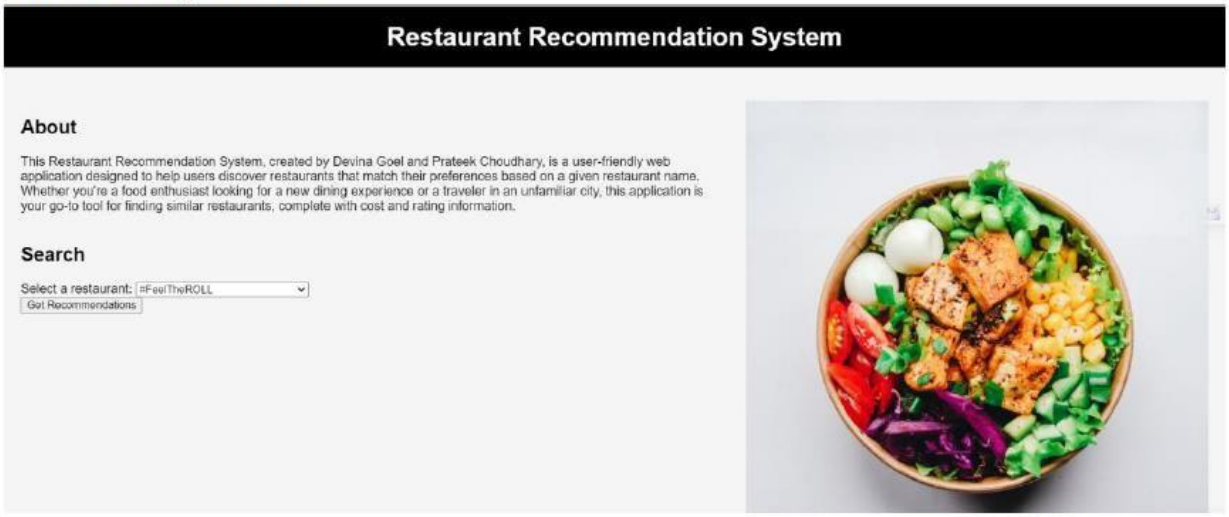
Libraries required for the app to run are to be imported.

Creating our flask app and loading the newly created dataset

Now after all the libraries are import we will be creating our flask app with the updated dataset

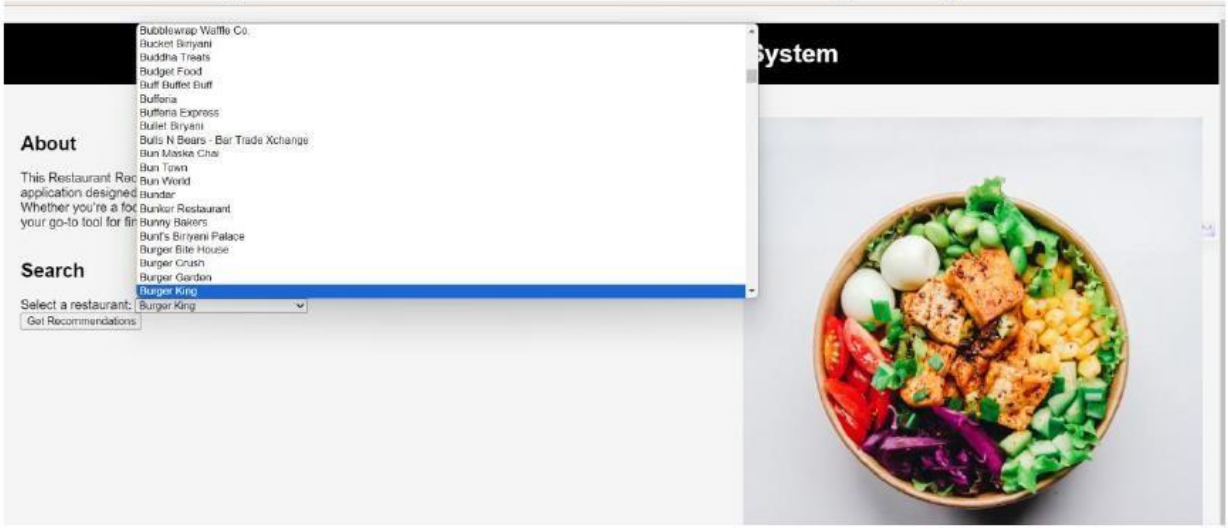
```
import pickle
import pandas as pd
from flask import Flask, render_template, request

app = Flask(__name__)

# Load the pickled data
with open('restaurants.pkl', 'rb') as file:
    restaurant_data = pickle.load(file)

with open('similarity.pkl', 'rb') as file:
    similarity_matrix = pickle.load(file)

# Define a function to get restaurant recommendations
def get_recommendations(restaurant_name, top_n=10):
    # Find the index of the given restaurant
    restaurant_index = restaurant_data[restaurant_data['name'] == restaurant_name].index[0]

    # Get cosine similarity scores for the given restaurant
    similarity_scores = list(enumerate(similarity_matrix[restaurant_index]))

    # Sort the restaurants by similarity score
    sorted_restaurants = sorted(similarity_scores, key=lambda x: x[1], reverse=True)

    # Get the top N similar restaurants (excluding itself)
    top_recommendations = [restaurant_data.iloc[x[0]] for x in sorted_restaurants[1:top_n+1]]

    return top_recommendations

@app.route('/')
def index():
    return render_template('index.html', restaurant_data=restaurant_data)

@app.route('/recommend', methods=['POST'])
def recommend():
    restaurant_name = request.form['restaurant_name']
    recommendations = get_recommendations(restaurant_name)
    return render_template('index.html', restaurant_name=restaurant_name, recommendations=recommendations, restaurant_data=restaurant_data)

if __name__ == '__main__':
    app.run(debug=True)
```

Showcasing The UI

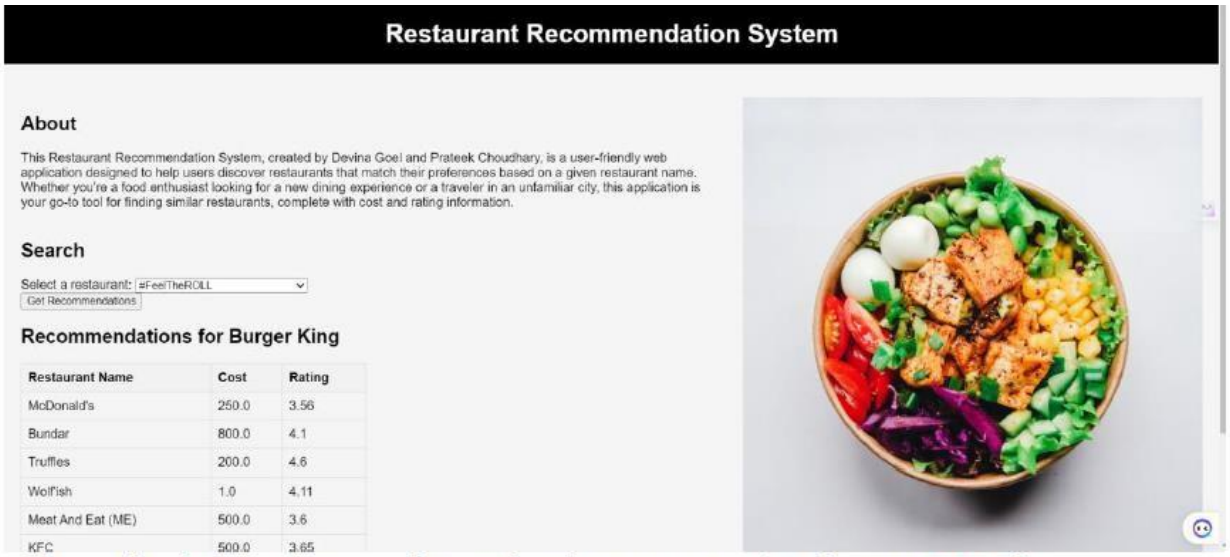


This is the home main page that describes the project and summarizes it.

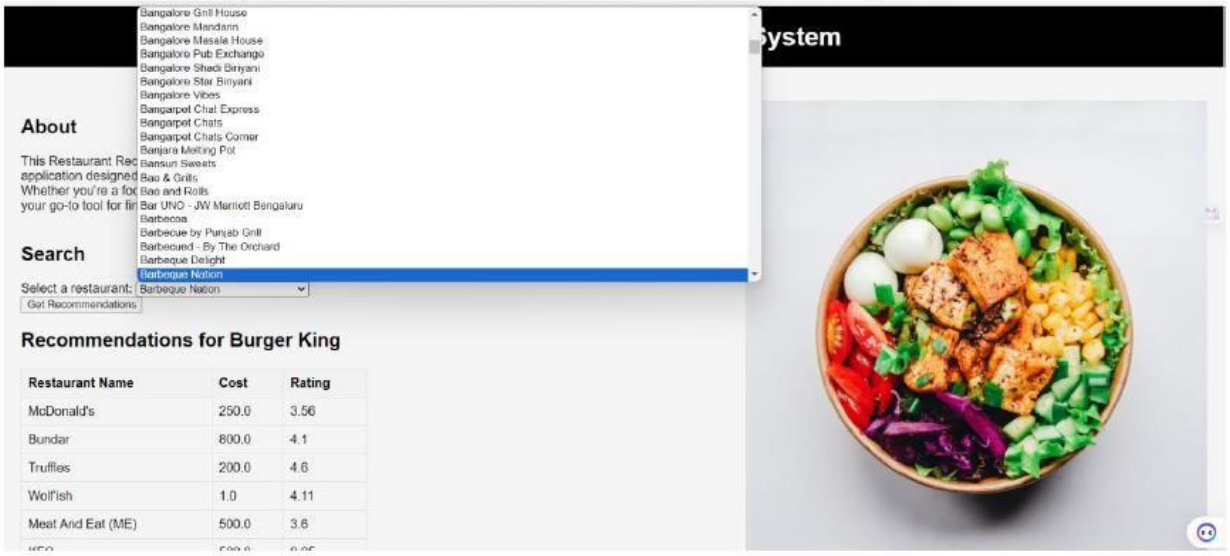
I. Checking recommendation for the restaurant: 'Burger King'



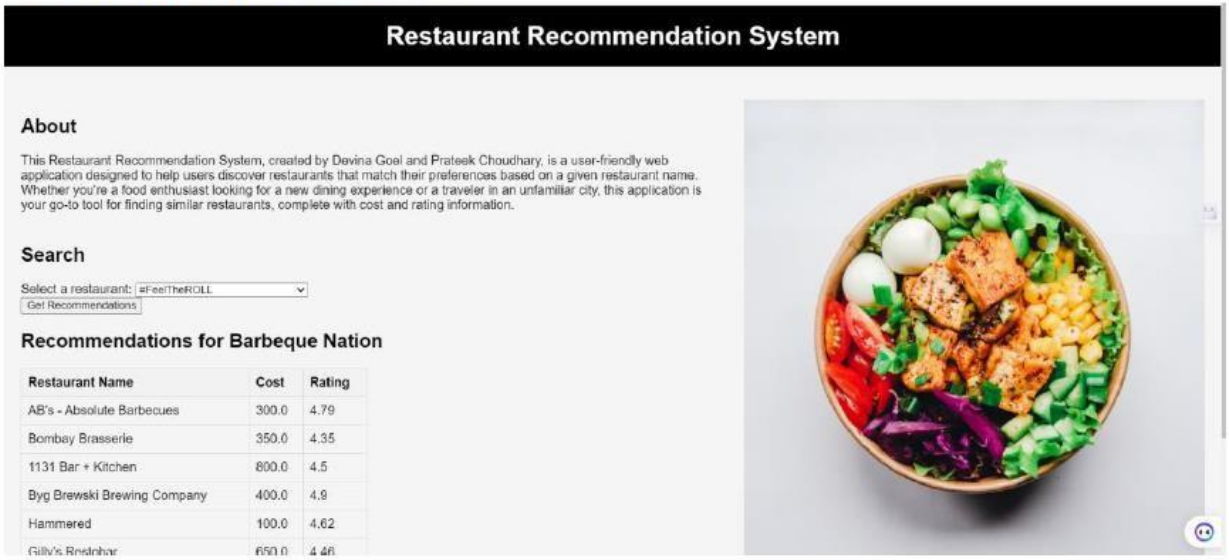
This is the prediction page where we will provide a restaurant name for which we will get the top recommended restaurants, which based on cuisines, mean rating (out of 5), cost in thousands.



II. Checking recommendation for the restaurant 'Barbeque Nation'



And here is the recommendation



Finally, the prediction for the given restaurant inputs is shown.

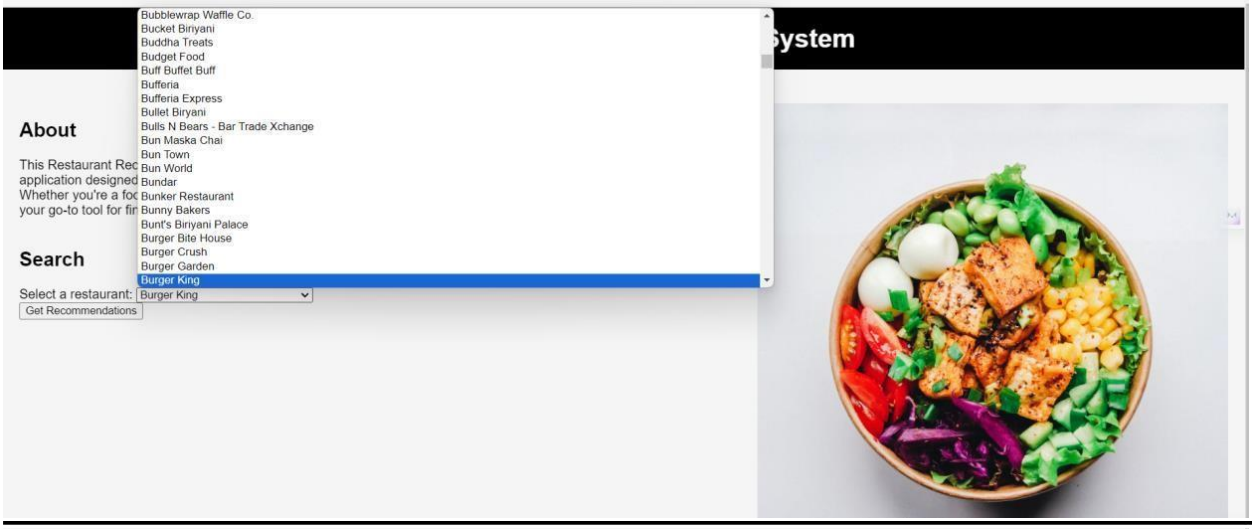
PERFORMANCE TESTING

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Parameter example 1	I have used Natural Language Processing as my model, so there's no as such the parameters. I can show some of the examples of the recommendations. This is for Burger King	`recommend('Burger King')` McDonald's Bundar Truffles Wolf'ish Meat And Eat (ME) KFC Inhouse Burger The Bundle Co. Bean Flickers Mugs N Burgers
2.	Parameter example 2	Example 2 is of Barbeque Nation.	`recommend('Barbeque Nation')` AB's - Absolute Barbecues Bombay Brasserie 1131 Bar + Kitchen Byg Brewski Brewing Company Hammered Gilly's Restobar Biergarten Flechazo Stories Kopper Kadai

RESULTS



Restaurant Recommendation System

About

This Restaurant Recommendation System, created by Devina Goel and Prateek Choudhary, is a user-friendly web application designed to help users discover restaurants that match their preferences based on a given restaurant name. Whether you're a food enthusiast looking for a new dining experience or a traveler in an unfamiliar city, this application is your go-to tool for finding similar restaurants, complete with cost and rating information.

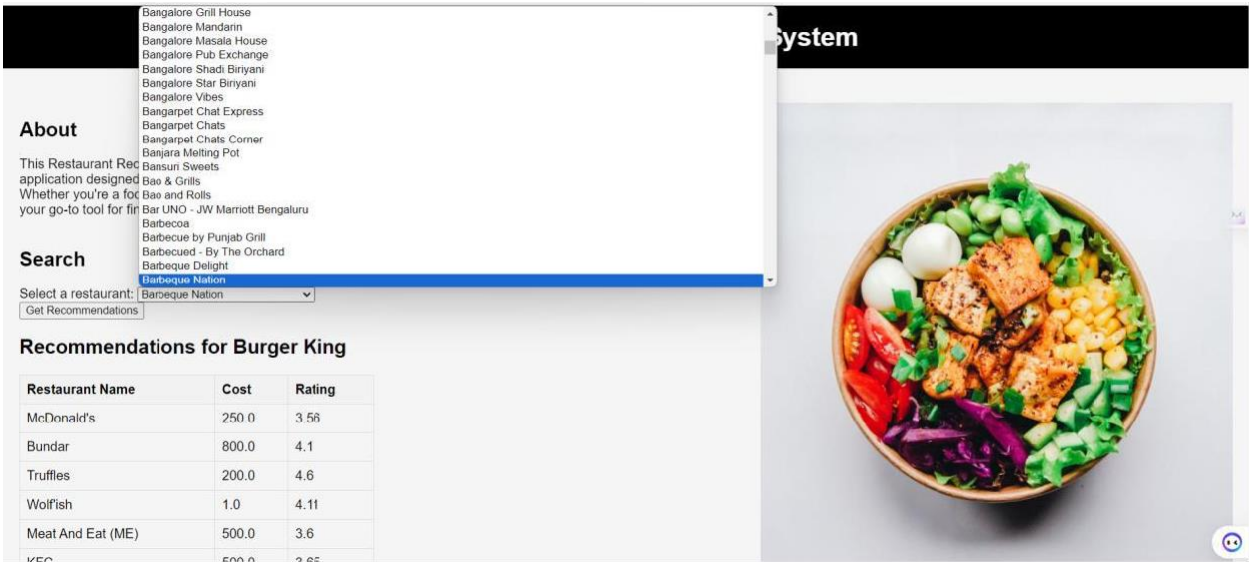
Search

Select a restaurant: #FeelTheROLL

Get Recommendations

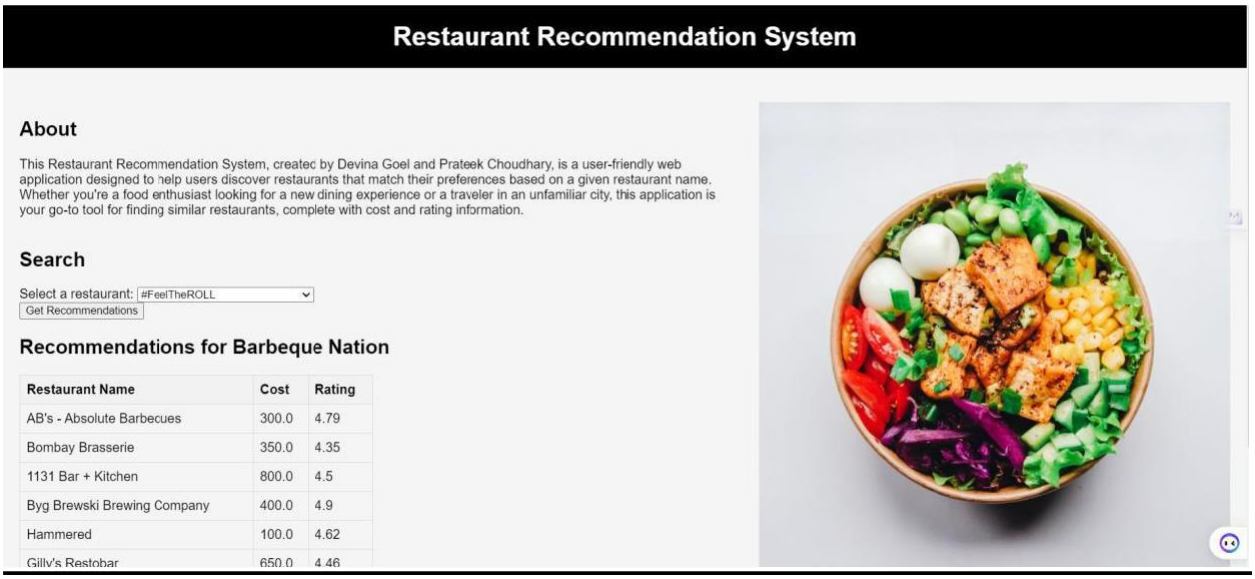
Recommendations for Burger King

Restaurant Name	Cost	Rating
McDonald's	250.0	3.56
Bundar	800.0	4.1
Truffles	200.0	4.6
Wolf'ish	1.0	4.11
Meat And Eat (ME)	500.0	3.6
KFC	500.0	3.65



Recommendations for Burger King

Restaurant Name	Cost	Rating
McDonald's	250.0	3.56
Bundar	800.0	4.1
Truffles	200.0	4.6
Wolf'ish	1.0	4.11
Meat And Eat (ME)	500.0	3.6
KFC	500.0	3.65



ADVANTAGES & DISADVANTAGES

Advantages:

A restaurant recommendation system offers several advantages:

Personalization: It tailors suggestions based on individual preferences, considering factors such as cuisine type, dietary restrictions, and budget, providing users with a more personalized dining experience.

Time Efficiency: Users save time in deciding where to dine as the system streamlines the restaurant selection process, presenting curated options that align with their tastes.

Discovery of New Places: The system introduces users to new and diverse culinary experiences, encouraging them to explore different restaurants and cuisines they might not have considered.

Increased User Satisfaction: By offering relevant and appealing recommendations, the system enhances user satisfaction, contributing to positive dining experiences and potentially building customer loyalty.

Adaptability: A well-designed system can adapt to changing user preferences, evolving culinary trends, and dynamically adjust recommendations over time, ensuring ongoing relevance.

Data-Driven Insights: The system generates valuable data on user preferences and trends, providing insights that can be used by restaurants for menu optimization, marketing strategies, and enhancing overall customer satisfaction.

Reduced Decision Fatigue: The system alleviates decision-making stress by presenting users with a curated list of options, minimizing the cognitive load associated with choosing a restaurant from a vast array of choices.

Improved Business Opportunities: Restaurants recommended by the system may experience increased visibility and customer traffic, contributing to their business success.

Overall, a restaurant recommendation system enhances user convenience, satisfaction, and the overall dining experience, benefiting both consumers and the restaurant industry.

Disadvantages:

While restaurant recommendation systems offer numerous advantages, there are also some potential disadvantages:

Over-Reliance on Algorithms: Users might become overly dependent on the system, limiting their exploration of new restaurants or cuisines outside the system's recommendations.

Bias and Lack of Diversity: Recommendation systems can perpetuate biases present in the data, potentially leading to a lack of diversity in the types of restaurants recommended and reinforcing existing user preferences.

Privacy Concerns: The collection and analysis of user data for personalized recommendations raise privacy concerns. Users may be hesitant to share personal information, impacting the system's ability to provide accurate suggestions.

Cold Start Problem: For new users or restaurants without sufficient data, the system may struggle to provide accurate recommendations, leading to a less effective user experience.

Inability to Capture Context: Some systems may not effectively capture contextual factors such as special occasions, mood, or specific dietary requirements, limiting the accuracy of recommendations in certain situations.

Limited Understanding of User Preferences: The system may not fully grasp the nuances of user preferences, potentially leading to inaccurate recommendations or failing to consider factors that are important to users.

Influence of Fake Reviews: If the recommendation system relies heavily on user reviews, the presence of fake or biased reviews can mislead the system and result in inaccurate recommendations.

Difficulty Handling Evolving Tastes: Adapting to sudden changes in user preferences or shifts in culinary trends can be challenging for recommendation systems, potentially leading to recommendations that become outdated.

Understanding and mitigating these disadvantages are essential for developing recommendation systems that balance personalization with ethical considerations, privacy concerns, and the need for diverse and accurate suggestions.

CONCLUSION

In conclusion, the restaurant recommendation system stands as a transformative tool that significantly enriches the dining experience for users. By leveraging advanced algorithms, it successfully addresses the challenges of information overload and decision fatigue, providing tailored suggestions that align with individual preferences. The system's ability to adapt and evolve over time, learning from user feedback and staying attuned to dynamic culinary trends, ensures ongoing relevance and user satisfaction.

While the advantages, such as personalization and time efficiency, are prominent, it is crucial to acknowledge potential drawbacks, including biases and privacy concerns. Striking a balance between algorithmic efficiency and ethical considerations is imperative for the sustained success and user trust in these systems.

Looking ahead, the continuous refinement of recommendation algorithms, addressing the cold start problem, and improving contextual understanding will further enhance the system's efficacy. Additionally, efforts to mitigate biases, prioritize user privacy, and promote diversity in recommendations will contribute to the development of more responsible and user-centric restaurant recommendation systems. Ultimately, as technology advances and user expectations evolve, the restaurant recommendation system remains a dynamic and indispensable tool in shaping enjoyable, diverse, and personalized dining experiences for individuals worldwide.

FUTURE SCOPE

The future scope for restaurant recommendation systems is promising, with potential advancements in several key areas.

Integration of Emerging Technologies: Incorporating emerging technologies like augmented reality (AR) and virtual reality (VR) could revolutionize the way users interact with restaurant recommendations. Users might virtually explore restaurant interiors, menus, and ambiance, enhancing their decision-making process.

Enhanced Personalization: Future systems will likely employ more sophisticated machine learning algorithms, leveraging deep learning and natural language processing to better understand and predict user preferences. This will result in even more accurate and personalized recommendations.

Cross-Domain Recommendations: Recommendation systems could expand beyond restaurants, providing users with holistic suggestions for dining, entertainment, and other leisure activities, creating a comprehensive lifestyle recommendation experience.

Real-Time Contextual Recommendations: Systems that can dynamically adapt recommendations based on real-time contextual factors such as weather, events, or user mood will offer a more responsive and tailored experience.

Blockchain for Data Security: Integration of blockchain technology may address privacy concerns by ensuring secure and transparent handling of user data, fostering greater trust among users.

Collaborative Filtering in Social Networks: Utilizing social network data for collaborative filtering could lead to recommendations based on the preferences and experiences of a user's social circle, enhancing the social aspect of dining choices.

Incorporating Sustainability Metrics: Future systems may consider users' interest in sustainable dining options, providing recommendations that align with environmental and ethical values.

As technology continues to advance, the future of restaurant recommendation systems holds the promise of more intelligent, personalized, and ethical solutions, catering to the evolving preferences and expectations of users in the dynamic culinary landscape.

APPENDIX

GitHub & Project Link:

https://github.com/smartinternz02/SI-GuidedProject-600981-1697476596