Project Design Phase-II
Technology Stack (Architecture & Stack)

Team ID	591739
Project Name	Project – Restaurant Recommendation System
Maximum Marks	4 Marks
Team Members	- Arsh Sharma 21BCE2483 - Shaik Mohid Babu 21BCE9569 - Charith Yerram 21BBS0165 - Veerala Rupa 21BCE9563

Technical Architecture:

1. User Interface (UI):

    - Collects user preferences (food, location, budget).
    - Sends user input to the backend.
    - Also collects ratings provided by the user.

2. Backend Application Logic:

    - User Input Processing:
        - Receives and processes user input from the UI.
    - Data Retrieval:
        - Accesses the restaurant dataset stored in a data storage component.
    - AI Recommendation Engine:
        - Utilizes machine learning models (if applicable) to generate restaurant suggestions based on user preferences.
    - Filtering and Sorting:
        - Applies filters and sorting algorithms based on user preferences.
    - Third-Party APIs:
        - May interact with external services or APIs (e.g., for additional data, reviews, maps).
    - Result Compilation:
        - Gathers the final list of recommended restaurants.
    - Response to UI:
        - Sends the list of recommended restaurants to the UI for display.

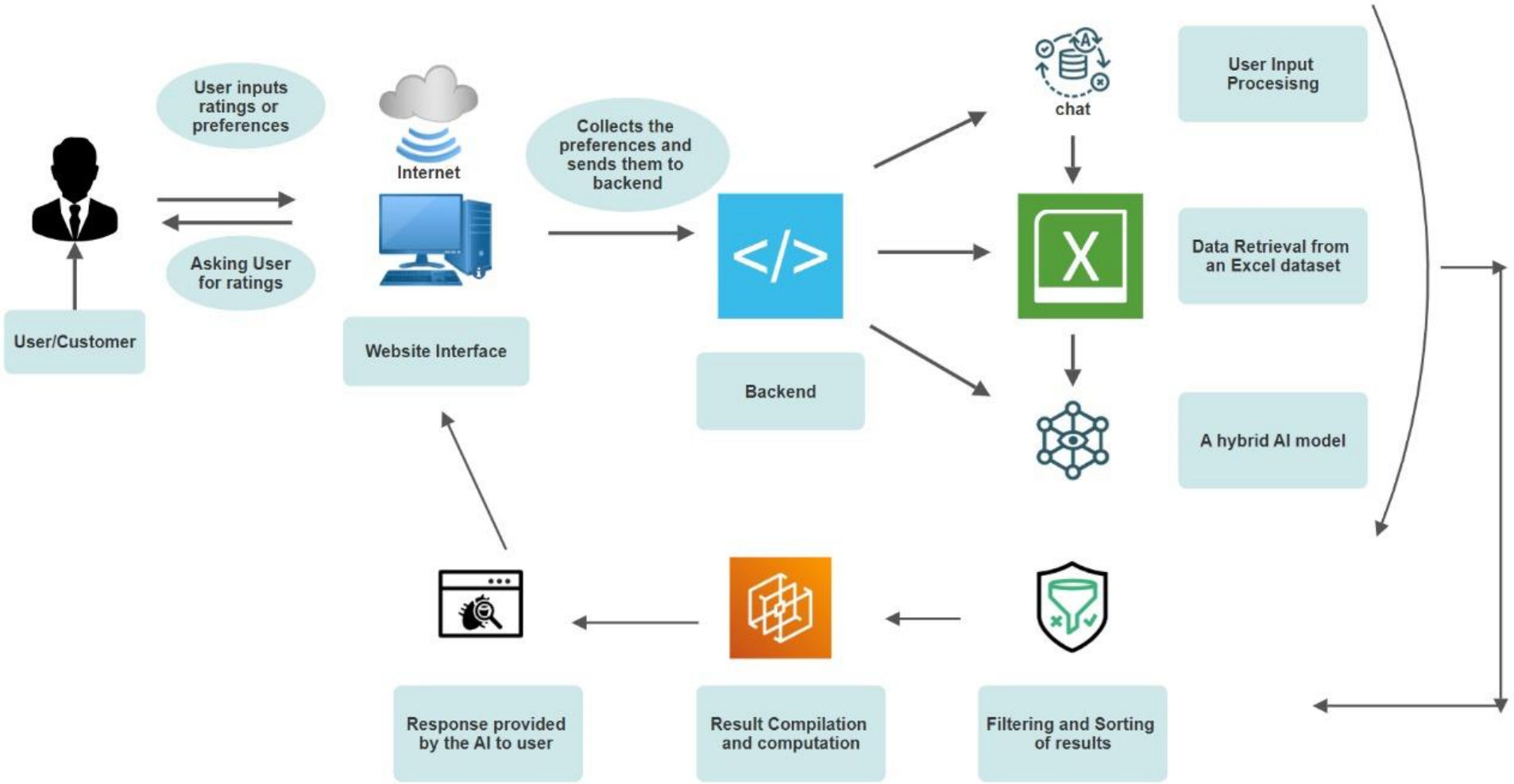
User inputs ratings or preferences
Asking User for ratings
User/Customer
Internet
Website Interface
Collects the preferences and sends them to backend
Backend
chat
User Input Procesisng
Data Retrieval from an Excel dataset
A hybrid AI model
Filtering and Sorting of results
Result Compilation and computation
Response provided by the AI to user

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The User filters to narrow down their search based on criteria like cuisine, price range, location, ratings	HTML, CSS
2.	Application Logic-1	Gather data on restaurants, including information such as location, cuisine type, price range, ratings, reviews, and user preferences	CSS / Python
3.	Application Logic-2	Clean and preprocess the data, collaborative filtering, content-based filtering for patterns for the data.	Python, HTML
4.	Application Logic-3	Implement features that enable personalized recommendations based on the user's history, preferences, and behavior	Python, CSS
5.	Database	Data like restaurant name, cuisine type, location, ratings, and reviews. Configure the parameters of the recommendation algorithm, including learning rates, regularization terms.	MySQL, NoSQL, etc.
6.	Cloud Database	Database service on the cloud is initialized when data is utilized in the cloud SQL, the data is fully structured and organized based on data sets	IBM DB2, IBM Cloudant etc.
7.	File Storage	Images and multimedia content considering the size and files quantity. Restaurant cuisines and popular dishes is stored in the pdf format.	IBM Block Storage or Other StorageService or Local Filesystem
8.	External API-1	API integrating with Google Maps helps users locate restaurants easily. Integrating with weather APIs can provide users with additional context for their dining choices	IBM Weather API, etc
9.	External API-2	Utilizing Authentication API'S like social media logins. Connecting to restaurant reservation APIs enables users to make reservations directly from the application.	Social Media API'S, etc.
10.	Machine Learning Model	Machine learning model is required for providing personalized and accurate recommendations to users.	Object Recognition Model, etc.
11.	Infrastructure (Server / Cloud)	Local system: Database, Backend server, Model deployment Cloud server: Cloud CDN, Storage, Scalability Monitoring	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Framework	Apache Mahout, TensorFlow Recommenders (TFRS), Flask	Python, Java, Flask, MySQL,
2.	Security Implementations	Data Encryption, Authorization, Securing API endpoints, Vulnerability	OpenSSL, OAuth, Django, OWASP ZAP
3.	Scalable Architecture	Microservices, Load balancing, Horizontal scaling, Caching, Auto Scaling	Kubernetes, NGINX, AWS, Redis, Azure Auto scaling
4.	Availability	Redundancy, multiple servers for preventing overloading, Maintenance	Load balancing, alerting systems, Automation tools
5.	Performance	Sharding mechanisms, CDN, Optimized database, Asynchronous processing	Amazon Aurora, Apache Kafka, Database indexing, Cloudflare

References:

https://c4model.com/

https://developer.ibm.com/patterns/online-order-processing-system-during-pandemic/

https://www.ibm.com/cloud/architecture

https://aws.amazon.com/architecture

https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d