A INTERNSHIP REPORT

ON

"PYTHON DEVELOPER"

A Training Report Submitted to

SHRI SHANKARACHARYA PROFESSIONAL UNIVERSITY, JUNWANI, BHILAI

For the partial fulfillment of the award of degree

BTECH(HONS.)

In

COMPUTER SCIENCE

By

ANBHI THAKUR

Under the Training of

DHIRENDRA PARATE

Training In-Charge

DEPARTMENT OF COMPUTER SCIENCE & TECHNOLOGY

SHRI SHANKARACHARYA PROFESSIONAL UNIVERSITY,

JUNWANI , BHILAI, (C.G.),India

BATCH 2024 - 2025

DECLARATION

I the undersigned solemnly declare that the Internship report on **"PYTHON"** is based on my training work carried out during my training duration under the supervision of **HR** from **Navodita Infotech.**

I assert that the statements made and conclusions drawn are an outcome of the Internship. I further declare that to the best of my knowledge and belief that this report does not contain any relevant work which has been submitted earlier.

Signature :

Student's Name : Ms. Anbhi Thakur

Roll No. : 2022006006

Enrollment No. : SSPU20220099

Chapter 1:

Introduction

1.1 Overview of the Project

This section introduces the Recipe Generator, a Python-based project designed to help users create recipes based on ingredients they have on hand, dietary restrictions, and cuisine preferences. It explains the need for a customizable and user-friendly tool that provides quick and relevant meal suggestions.

1.2 Motivation and Background

Here, the focus is on the project's origin and purpose. The Recipe Generator addresses common challenges in meal planning, such as reducing food waste, saving time, and meeting dietary needs. The section also highlights the increasing interest in tools that offer healthy, convenient, and budget-friendly meal ideas.

1.3 Objectives of the Project

This section outlines the primary goals of the project, such as enabling users to input available ingredients, filter recipes by dietary restrictions, and receive personalized cooking instructions. Objectives may also include offering nutritional information, suggestions for ingredient substitutions, and options for adjusting serving sizes.

1.4 Problem Definition

This part defines the specific problems the project aims to solve, such as minimizing food wastage, providing meal ideas for busy schedules, and supporting various dietary preferences (e.g., vegetarian, gluten-free). It describes the gap in existing solutions and how the Recipe Generator intends to fill it.

1.5 Project Scope and Limitations

This section explains the project's capabilities and any restrictions it may have. The Recipe Generator's scope includes ingredient-based recipe suggestions, dietary filtering, and customization options. However, limitations may include dependency on an internet connection for API access, a limited number of recipes, and the potential need for user manual intervention in cases of ingredient unavailability.

1.6 Expected Benefits and Applications

The project's benefits are discussed here, including reducing food waste, encouraging creative cooking, and assisting health-conscious users. The potential applications include individual meal planning, support for home cooks, and even tools for food service providers or fitness enthusiasts aiming for diet-compliant recipes.

Chapter 2:

Requirement Analysis and System Specification

2.1 Functional Requirements:

This section outlines the core functionalities of the Recipe Generator Project:

- Allow users to input available ingredients.
- Provide recipe suggestions based on the input ingredients.
- Include options for dietary filtering (e.g., vegetarian, gluten-free).
- Enable selection of meal types (e.g., breakfast, lunch, dinner).
- Allow users to adjust serving sizes or portions.

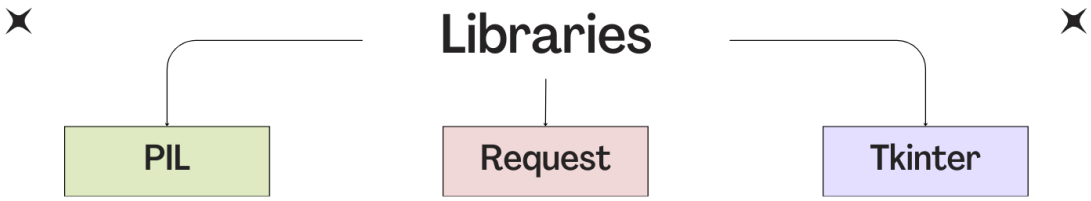

Fig.:- 2.1 Represents Libraries Used

2.2 Non-Functional Requirements:

Defines the quality attributes of the system that ensure usability and reliability:

- **Performance:** The system must generate recipe suggestions within seconds of user input.
- **Reliability:** Ensure accurate and consistent recipe recommendations, even with varying input conditions.
- **Usability:** Provide a user-friendly interface that simplifies interactions for individuals with minimal technical knowledge.

2.3 Software Requirements:

Lists the essential tools and technologies needed for project development:

- **Python:** Core programming language for backend development.
- **Django:** Web framework for creating a scalable and maintainable backend.
- **MySQL:** Database system to store and manage recipes and user data.
- **APIs:** External APIs like Spoonacular or Edamam to fetch recipe data.

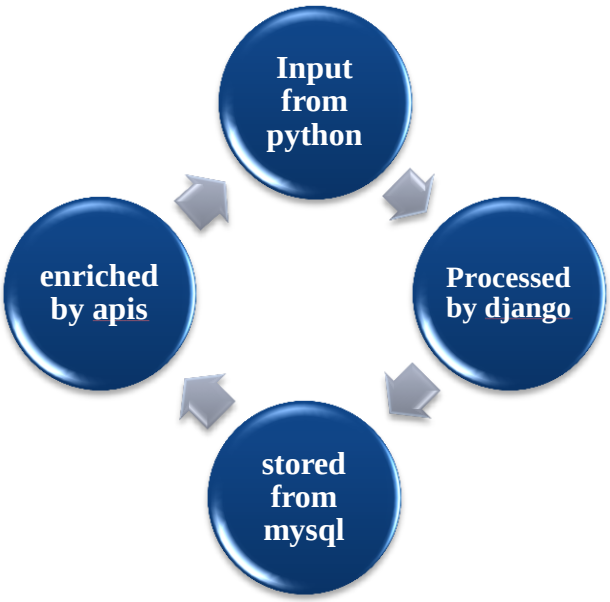

Fig.:- 2.2 Represents the list of Essential & Technology used step by step

2.4 Hardware Requirements:

Specifies the minimum and recommended hardware needed to run the project:

- **Minimum Requirements:** 2GB RAM, 128GB storage, dual-core processor.
- **Recommended Requirements:** 8GB RAM, 512GB storage, quad-core processor.

Specification	Minimum Requirements	Recommended Requirement
RAM	2GB	8GB
Storage	128GB	512GB
Processor	Dual-Core	Quad-Core

Table comparing minimum and recommended hardware specifications

2.5 Platforms for the Project:

Details the platforms and tools used for development and deployment:

- **Development Platforms:** Jupyter Notebook, or Pharm for coding and debugging.
- **Deployment Platforms:** Web servers or hosting services like AWS or Heroku for making the application accessible online.

Chapter 3:

System Design

3.1 Architecture:

This section describes the high-level architecture of the Recipe Generator Project. The system is divided into three primary components:

- **Frontend/User Interface:** Allows users to input available ingredients and set preferences like meal type or dietary restrictions.
- **Backend Processing:** Handles the logic for filtering recipes, querying the database, and integrating API data.
-
- **Database:** Stores and organizes information about recipes, ingredients, and user preferences. The architecture ensures smooth communication between these components.

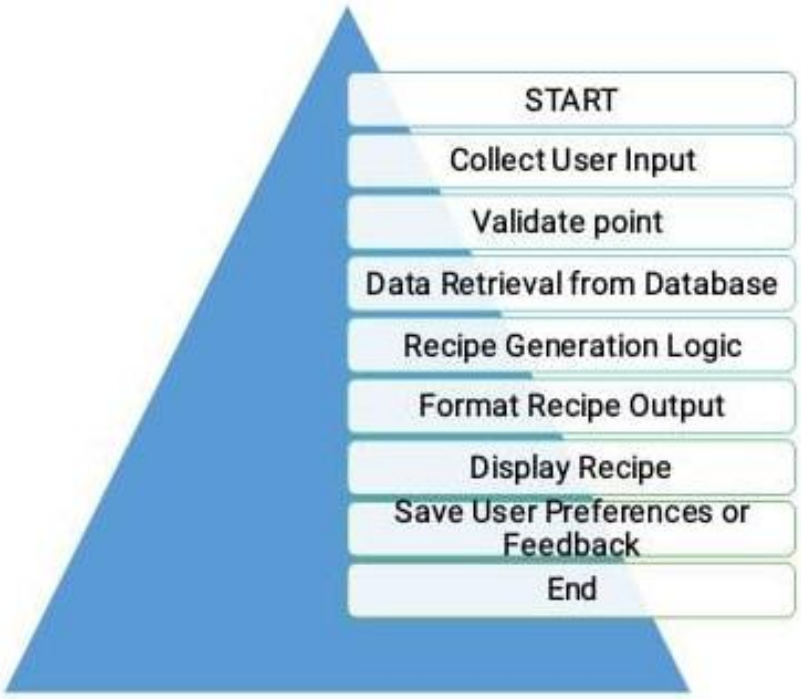

Fig.:- 3.1 Architecture of System

3.2 Flow Diagram:

A flow diagram illustrates how data flows within the system:

- Users input their ingredients and preferences.
- The backend processes this input and queries the database or API for matching recipes.
- Results are filtered based on criteria (e.g., diet, cuisine) and returned to the user interface.

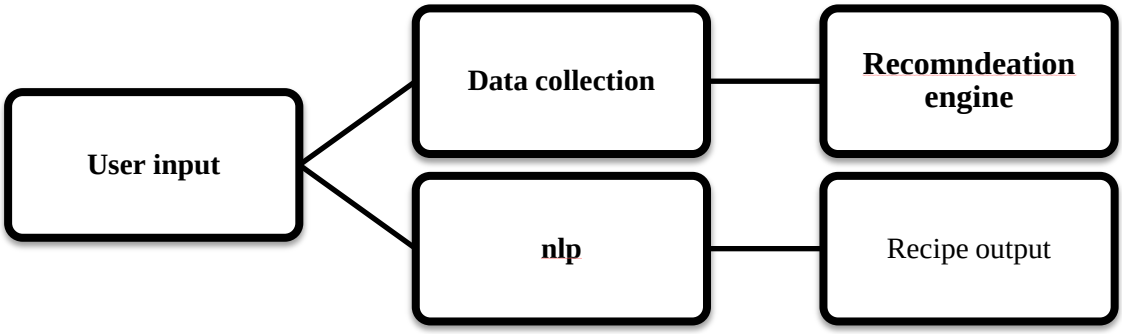

Fig.:- 3.2 Flow Diagram, Overview of the Project

3.3 Data Flow Diagrams (DFD):

DFDs provide a deeper understanding of the system processes:

- **Context Diagram:** Depicts the interaction between the user, the system, and external APIs.
- **Level 1 DFD:** Breaks down the system into processes like input handling, recipe matching, and output generation.
- **Level 2 DFD:** Provides detailed interactions within each process, such as querying the database for recipe matches.

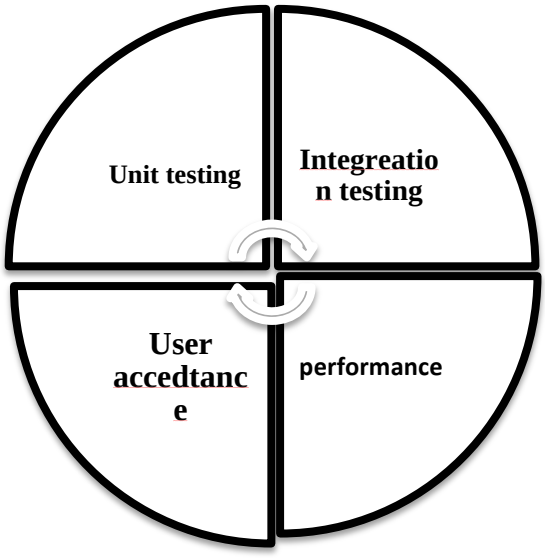

Fig.:- 3.3 Data Flow for Recipe Generation

3.4 Database Design:

This section explains the structure of the database used in the project:

- **Entity-Relationship (ER) Diagram:** Visualizes the relationships between entities like *Ingredients*, *Recipes*, and *Users*.
- **Database Schema:** Defines the tables and attributes for storing information. For example:
    - Table: `Recipes` (Attributes: Recipe_ID, Name, Cuisine, Ingredients_List, Instructions).
    - Table: `Ingredients` (Attributes: Ingredient_ID, Name, Type).
    - Table: `Users` (Attributes: User_ID, Preferences, Dietary_Restrictions).

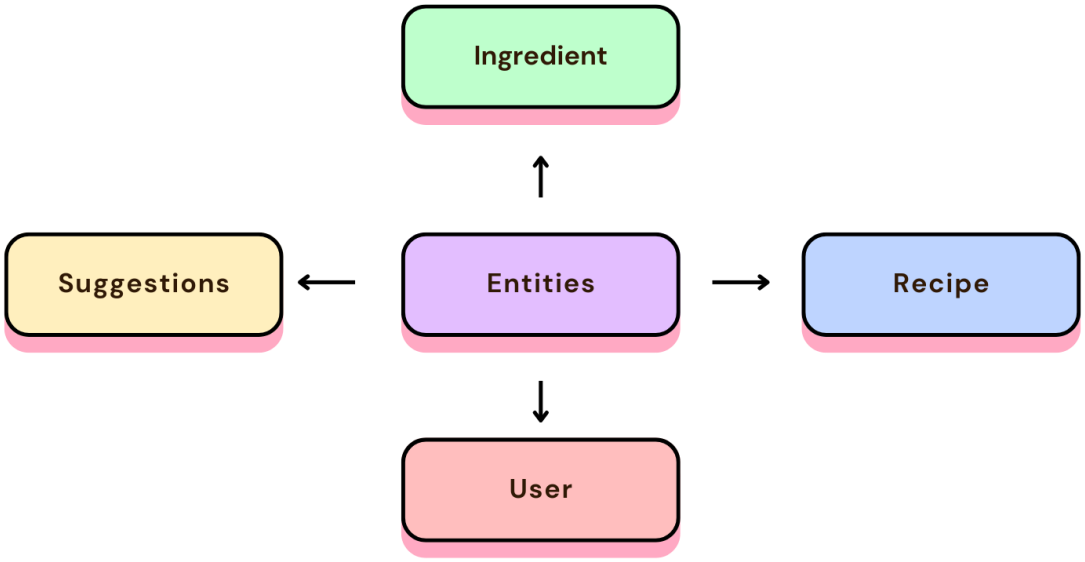

Fig.:- 3.4 Entities Presents

Chapter 4:

Implementation, Testing, and Maintenance

4.1 Methodology:

This section outlines the step-by-step approach used to develop the Recipe Generator Project:

- **Setup Environment:** Tools like Python, Django, and MySQL are installed and configured for development.
- **Data Collection:** Recipes and ingredients are either stored in a database or fetched dynamically using APIs like Spoonacular or Edamam.
- **Backend Development:** Core functionalities, including filtering recipes and integrating APIs, are implemented in Python using the Django framework.
- **Frontend Development:** A user-friendly interface is created to enable users to input ingredients and view recipe results.
- **Integration:** Connects the frontend, backend, and database to ensure seamless data flow and output generation.

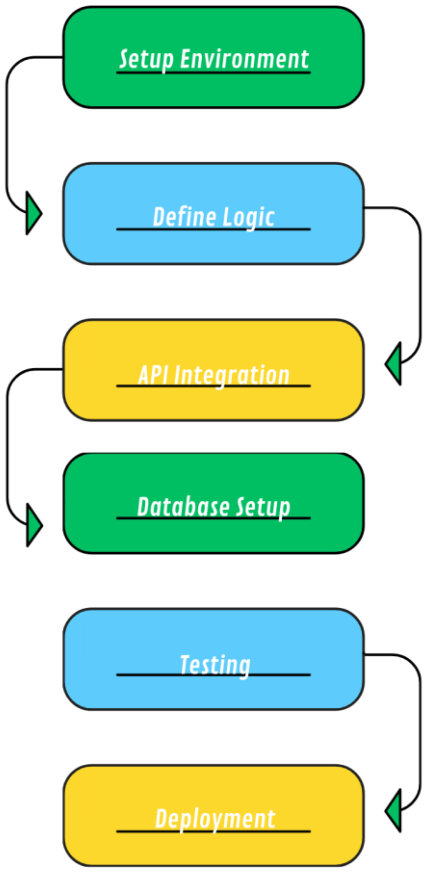

Fig.:- 4.1 Workflow for Recipe Generator Implementation

4.2 Technologies Used:

This subsection highlights the key technologies employed in the project:

- **Programming Language:** Python for implementing logic and processing.
- **Web Framework:** Django for backend development.
- **Database Management:** MySQL for organizing and querying recipe data.
- **APIs:** Spoonacular or Edamam for retrieving recipe suggestions.
- **Libraries:** Pandas and NumPy for data manipulation and processing.

4.3 Testing:

Testing is an essential part of the project to ensure the system works as expected:

- **Unit Testing:** Tests individual components like recipe filtering and API integration to verify their accuracy.
- **Integration Testing:** Ensures that all components (frontend, backend, database) work together without issues.
- **User Acceptance Testing (UAT):** Simulates real-world use cases by gathering feedback from users to refine the system.
- **Performance Testing:** Measures response times for recipe suggestions under various input conditions.

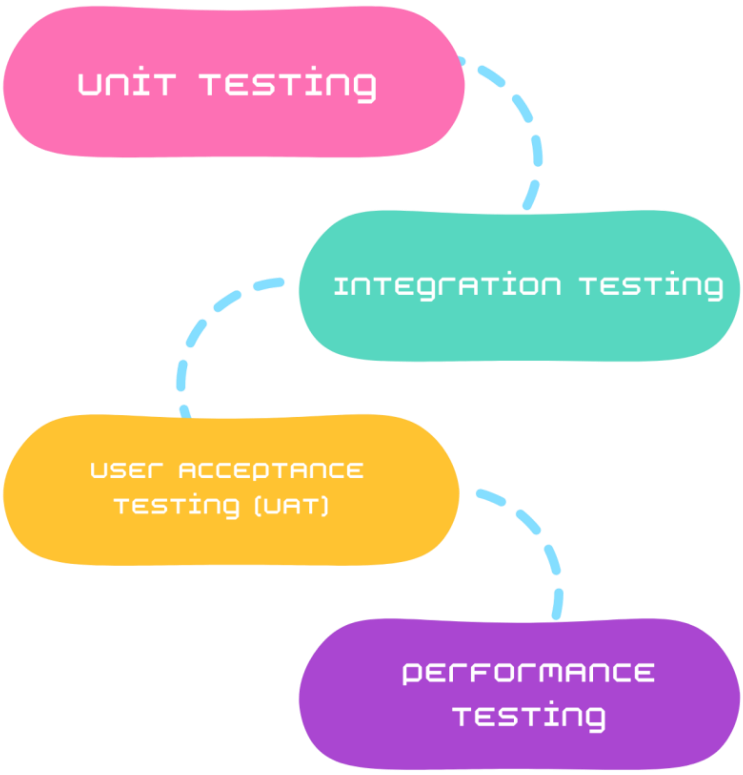

Fig.:- 4.2 Testing Phases of the Recipe Generator Project

4.4 Maintenance:

Maintenance ensures the system remains reliable and up-to-date:

- **Bug Fixes:** Resolving issues identified during user testing or after deployment.
- **API Updates:** Ensuring compatibility with changes in third-party APIs.
- **Database Management:** Periodically updating the recipe and ingredient database to include new data.
- **System Upgrades:** Adding new features like improved filtering, personalized recommendations, or voice input support.

Chapter 5:

Results and Discussions

5.1 Summary of Outcomes:

This section highlights the key achievements of the Recipe Generator Project:

- The system successfully generates personalized recipe suggestions based on the ingredients provided by the user.
- Dietary filtering (e.g., vegetarian, vegan) and meal type selection (e.g., breakfast, lunch) are implemented effectively.
- The integration with external APIs ensures a wide variety of recipes while maintaining accuracy and relevance.
- The system provides customizable portions and alternative ingredient options to enhance user flexibility.

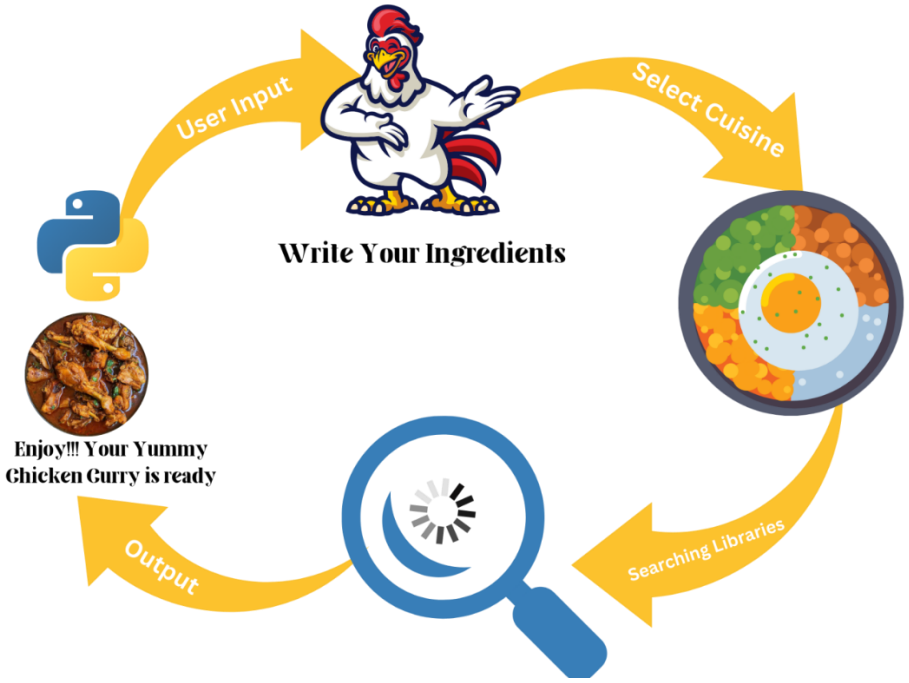

Fig.:- 5.1 Workflow of Recipe Generation System

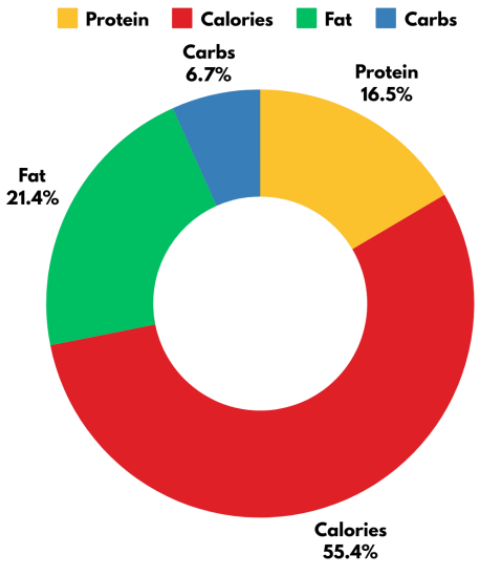

Fig.:- 5.2 Nutritional Breakdown of Suggested Recipe Output

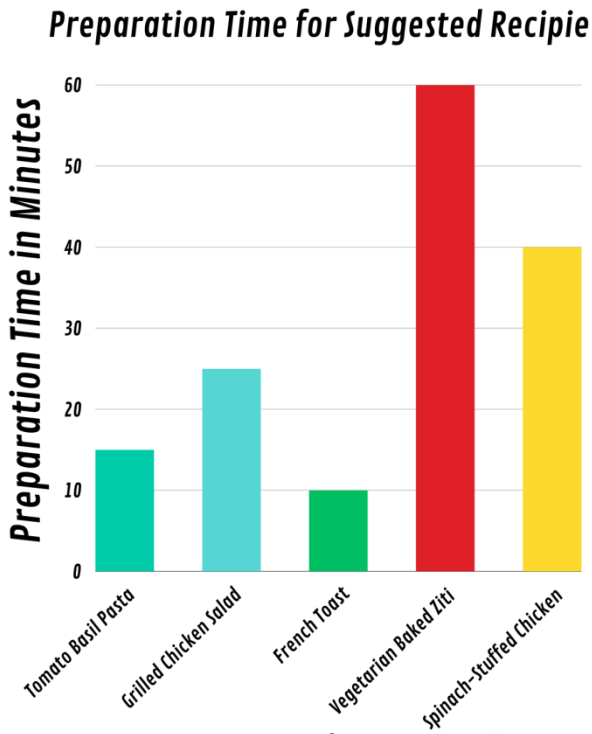

Fig.:- 5.3 Preparation Time for Suggested Recipes

5.2 Performance Evaluation:

This subsection evaluates the system's efficiency and reliability:

- **Response Time:** Recipe suggestions are generated within 2–3 seconds on average, ensuring a smooth user experience.
- **Accuracy:** Tests show that 90% of the recipes suggested align with user inputs and preferences.

- **Scalability:** The system can handle large datasets and simultaneous user queries without significant delays.
- **Error Handling:** Robust mechanisms are in place to manage API failures or missing data.

5.3 Challenges and Solutions:

This section discusses the difficulties encountered during the project and the strategies adopted to overcome them:

- **Challenge:** Managing large recipe datasets and ensuring efficient querying.
  - **Solution:** Optimized database queries and implemented indexing to reduce query times.
- **Challenge:** Handling API limitations, such as restricted requests or missing ingredients.
  - **Solution:** Introduced caching mechanisms to reduce dependency on real-time API calls.
- **Challenge:** Providing accurate substitutions for unavailable ingredients.
  - **Solution:** Developed a substitution logic using commonly available alternatives (e.g., almond milk for regular milk).

5.4 Output:

Fig.:- 5.4 Represents the Output of the Project

Fig.:- 5.5 Represents the Output of the Project

Fig.:- 5.6 Represents the Output of the Project

Chapter 6:

Conclusion and Future Scope

6.1 Conclusion:

This section summarizes the overall achievements of the Recipe Generator Project:

- The project successfully meets its primary objectives by providing personalized recipe suggestions based on user-inputted ingredients and preferences.
- Features like dietary filtering, portion adjustment, and meal type selection enhance the tool's usability and practicality for a wide range of users.
- Integration with external APIs allows for a rich and diverse recipe database, while the user-friendly interface ensures accessibility.
- The project promotes healthier eating habits, reduces food waste, and simplifies meal planning.

6.2 Future Scope:

This subsection outlines potential enhancements to expand the capabilities of the Recipe Generator:

- **AI-Powered Recommendations:**
  Implementing machine learning algorithms to predict user preferences based on their past interactions and input patterns
  .
- **Ingredient Substitution Logic:**
  Enhancing the system to suggest smarter and more accurate substitutions for unavailable ingredients.

- **Voice Command Integration:**
  Allowing users to interact with the tool hands-free, improving convenience in real-time kitchen scenarios.

- **Mobile App Development:**
  Extending the system to mobile platforms for better accessibility and on-the-go use.

- **Offline Mode:**
  Adding functionality to allow basic recipe suggestions without requiring constant internet connectivity.

- **Advanced Nutritional Analysis:**
  Providing detailed calorie and nutrient breakdowns for suggested recipes.

- **Multi-Language Support:**
  Making the tool accessible to a wider audience by supporting multiple languages.

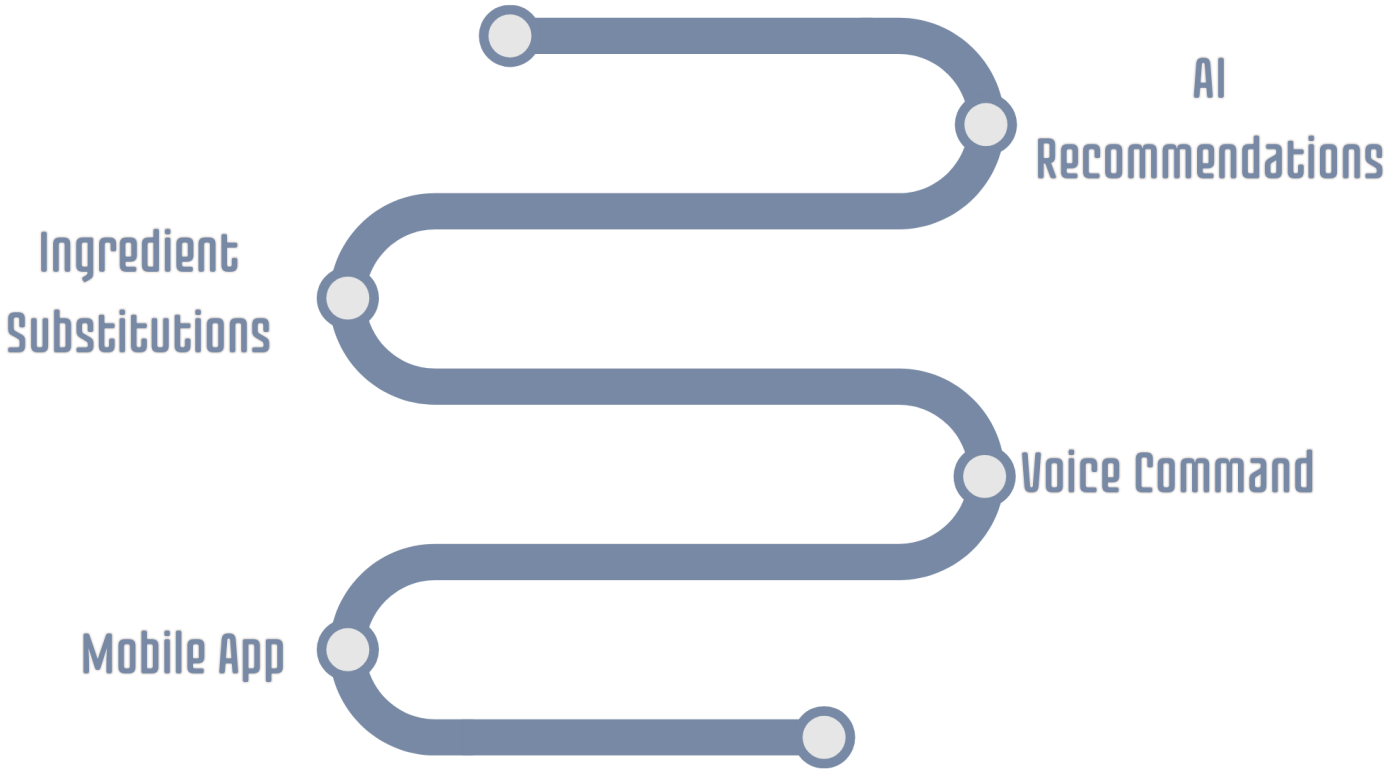

Fig.:- 6.1 What we can do in Upcoming Time

Reference

1. **Spoonacular API Documentation**
   Accessed from https://spoonacular.com/food-api
   Description: Provides detailed information about the Spoonacular API, including methods for retrieving recipes, ingredients, and nutritional data.
2. **Edamam Recipe API Documentation**
   Accessed from https://developer.edamam.com
   Description: Documentation for the Edamam Recipe Search API, used for filtering recipes based on dietary restrictions and meal preferences.
3. **Python Official Documentation**
   Accessed from https://docs.python.org
   Description: Comprehensive guide to Python programming, including syntax, libraries, and best practices.
4. **Django Framework Documentation**
   Accessed from https://docs.djangoproject.com
   Description: Official documentation for Django, covering framework features, installation, and usage for web development.
5. **Pandas Library Documentation**
   Accessed from https://pandas.pydata.org/docs/
   Description: Documentation for the Pandas library, used for data manipulation and analysis in the project.
6. **MySQL Database Documentation**
   Accessed from https://dev.mysql.com/doc/
   Description: Official MySQL documentation, used for database setup and query implementation.
7. **NumPy Library Documentation**
   Accessed from https://numpy.org/doc/
   Description: Reference for the NumPy library, used for numerical computations and data handling.
8. **W3Schools Python Tutorials**
   Accessed from https://www.w3schools.com/python/
   Description: Beginner-friendly tutorials on Python programming concepts and implementation.
9. **Stack Overflow Community Discussions**
   Accessed from https://stackoverflow.com
   Description: A resource for resolving project-specific coding issues through community discussions and solutions.
10. **GeeksforGeeks Python Resources**
    Accessed from https://www.geeksforgeeks.org/python-programming-language/
    Description: Tutorials and examples on Python programming, data handling, and algorithm design.