Diet Recommendation

Session: 2022-2026

Submitted by:

Saba Shahdin 2022-CS-112
Afeera Fatima 2022-CS-151

Submitted to:

Dr. Samyan Wahla

Course:

CS-371 Artificial Intelligence

Department of Computer Science

University Of Engineering And Technology,

Lahore, Pakistan

Abstract

This report details the design, development, and implementation of a personalized Diet Recommendation System powered by machine learning algorithms. The system analyzes individual health metrics such as body mass, activity levels, age, height, weight, and dietary constraints, including food allergies, to generate tailored meal plans that promote balanced nutrition and foster healthier eating habits.

Key features include a user-friendly interface for inputting health data, tracking daily meals, and adjusting diet plans to meet evolving health goals and meal preferences. Machine learning models enable dynamic, data-driven recommendations that adapt and improve over time, ensuring a high degree of personalization.

The report elaborates on the system architecture, algorithmic approach, and user experience design. Additionally, it evaluates the system's performance based on accuracy, effectiveness, and user satisfaction. Testing and validation metrics are presented to substantiate its ability to meet diverse dietary needs.

Finally, the report explores future improvements, such as integrating advanced algorithms, expanding dietary options, and incorporating real-time health tracking, with the ultimate goal of delivering a more comprehensive and adaptive diet planning solution.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

In recent years, there has been a growing awareness of the importance of maintaining a healthy lifestyle, with a significant focus on diet and nutrition. With the rise of sedentary lifestyles, poor eating habits, and the increasing prevalence of lifestyle, diseases and specific mood swings people are more conscious of the need to adopt healthy eating practices. However, finding personalized and effective diet plans can be a challenge due to the vast amount of dietary information available and the individual variability in nutritional needs.

A diet recommendation system addresses this issue by providing tailored meal plans based on a user's specific health data, preferences, and goals. Such systems leverage advances in technology, particularly machine learning, to analyze individual health factors and suggest diets that optimize nutritional intake. This can help users make informed decisions about their food choices, manage their health better, and achieve their wellness goals.

In today's fast-paced world, where individuals often struggle to plan their meals or stick to healthy eating habits, a diet recommendation system offers a practical solution. By automating the process of meal planning and considering various factors such as age, weight, activity level, and user food allergies, this system can play a pivotal role in promoting long-term health and well-being. The importance of such a system is particularly evident in the context of increasing reliance on technology for health management and the growing demand for personalized health solutions.

1.2 Problem Statement

In today's health-conscious society, many individuals struggle to manage their diet due to food allergies (e.g., milk, meat, gluten) and their physique that require specific dietary adjustments, such as maintaining a healthy BMI. Existing diet recommendation systems often overlook these factors, providing generic meal plans that may not account for allergens or the user's BMI class. This project aims to address this gap by developing a personalized diet recommendation system that considers both food allergies and health metrics, such as BMI and activity levels, ensuring safe and tailored meal plans for users.

1.3 Objectives

The primary objectives of this project are:

- To develop a machine learning-based diet recommendation application that offers personalized meal plans based on user preferences, such as weight, age, height, and food allergies (e.g., milk, meat, gluten).
- To create a user-friendly mobile application that provides dietary suggestions tailored to users' nutritional needs while ensuring allergen-free meal options.

- To implement a machine learning model that accurately predicts and suggests dietary plans based on factors such as age, gender, weight, activity level, and specific food allergies.
- To allow users to select meals from provided food options and enable the regeneration of alternative options if the suggested items do not meet their preferences or dietary needs.
- To include functionality for tracking the calories and macronutrient content (e.g., carbohydrates, proteins, fats) of selected food items, ensuring a comprehensive analysis of the nutritional intake.
- To provide a final calculation of whether the selected meal plan exceeds, meets, or falls short of the user's recommended caloric and macronutrient requirements, guiding users towards balanced nutrition.
- To evaluate the system's performance in terms of prediction accuracy, user satisfaction, and its ability to promote healthier eating habits while ensuring allergen safety.

1.4 Scope of the Report

This report covers the design, implementation, and evaluation of a diet recommendation system that takes food allergies into account, including allergens such as milk, meat, gluten, and others. The following key areas will be discussed:

- **System Design and Architecture:** The overall design of the diet recommendation app, including the architecture, user flow, and technology stack used.
- **Machine Learning Approach:** The selection, training, and evaluation of the machine learning models used to generate personalized diet recommendations considering user preferences while avoiding allergens.
- **App Development:** The process of developing both the front-end and back-end of the application, including UI/UX design, database design, and API integration.
- **Implementation:** The specific features implemented in the app, including allergen detection and meal recommendations, challenges faced during development, and solutions to overcome them.
- **Results and Evaluation:** The performance of the app in terms of user engagement, satisfaction, and its ability to meet personalized dietary needs while avoiding allergens.

The report concludes with recommendations for future enhancements and potential areas for further research in the field of allergen-aware diet recommendation systems.

2 Literature Review

Personalized diet recommendation systems have gained significant attention in recent years due to the increasing need for tailored health solutions. Traditional diet plans, while effective to some extent, often fail to account for individual differences such as food preferences, allergies, and health conditions. This has led to the development of more advanced systems that leverage machine learning algorithms to provide customized meal plans. These systems aim to optimize nutrient intake, reduce the risk of adverse health effects, and improve overall well-being by considering a variety of personal factors.

One of the key areas in personalized diet recommendation systems is the inclusion of food allergies. Many individuals suffer from allergies to common foods like milk, meat, gluten, making it essential for diet planning systems to factor in these restrictions. Without this consideration, users may inadvertently consume foods that trigger allergic reactions, leading to serious health risks. The integration of food allergies into diet recommendation systems ensures a safer and more effective meal plan for users.

Another important aspect is the incorporation of health metrics such as BMI (Body Mass Index), which plays a crucial role in determining an individual's nutritional needs. By analyzing BMI alongside other health data, personalized systems can better understand a user's health status and provide diet recommendations that align with their goals, whether it's weight loss, weight maintenance, or improving overall health.

Despite the progress made in developing these systems, several challenges remain. The complexity of accurately predicting a user's dietary needs, the vast array of food items and allergens, and the need for real-time dietary updates are some of the hurdles that need to be addressed. Additionally, existing systems may not always be accessible to a wide range of users, particularly those with unique dietary restrictions or limited access to technology.

This project addresses these gaps by creating a diet recommendation system that not only considers food allergies but also utilizes machine learning algorithms to personalize diet plans based on an individual's health metrics, preferences, and lifestyle. By integrating these features, the system aims to provide a more holistic and user-friendly approach to nutrition management, ensuring that users receive accurate, safe, and effective diet recommendations tailored to their specific needs.

3 Methodology

The methodology section outlines the detailed steps taken in developing the Personalized Diet Recommendation App. This section breaks down the entire process into four major steps: Data Collection and Preprocessing, Model Training and Validation, App Development, and Feature Integration.

3.1 Workflow Diagram

The following diagram illustrates the overall workflow of the development process. The following sections describe each step in detail.

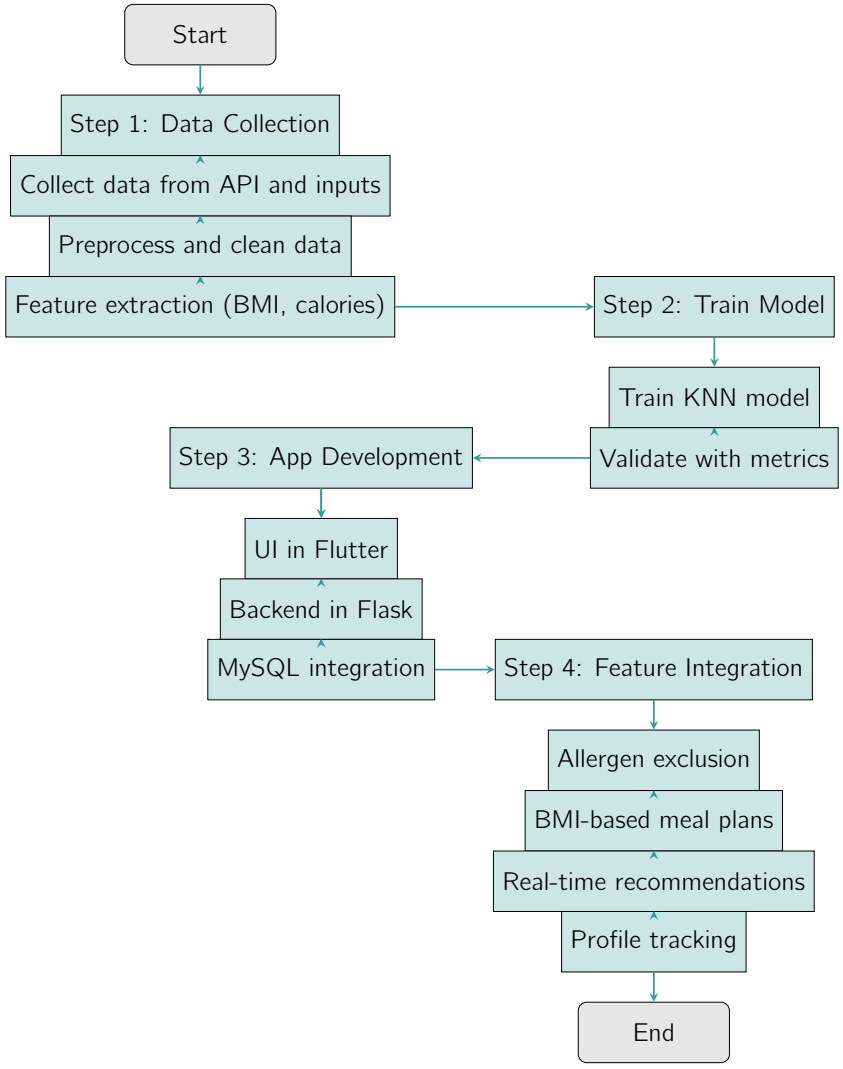

Figure 1: Development Process for the AI-Based Diet Planner

3.2 Step 1: Data Collection and Preprocessing

The initial step in building the diet recommendation system was to collect relevant data and prepare it for use in machine learning models. This step is essential to ensure that the app can provide accurate, personalized, and allergy-aware recommendations.

3.2.1 Data Collection:

- Nutritional information and food categories were gathered from the **Spoonacular API** (for allergens, macronutrients, and calories).
- A user dataset was taken from kaggle to understand the preferences, allergies, and health metrics (e.g., BMI) needed for personalization. This dataset also includes BMI classification to tailor recommendations based on the user's health status.

3.2.2 Data Preprocessing:

- **Cleaning:** We removed missing, incomplete, or irrelevant data entries to ensure that the dataset was accurate and usable.
- **Normalization:** Nutritional values (calories, fat, protein, etc.) were normalized to bring all features to a common scale, using Min-Max scaling or Z-score normalization.
- **Encoding:** Categorical features (such as lifestyle preferences) were converted into numerical values using one-hot encoding for compatibility with machine learning algorithms.
- **Feature Extraction:**
  - **BMI Calculation:** The Body Mass Index (BMI) for each user was calculated using the following formula:

    $$\text{BMI} = \frac{\text{Weight (kg)}}{\text{Height (m)}^2}$$

    The calculated BMI was used to classify users into categories such as underweight, normal weight, overweight, or obese, providing insights into their nutritional needs.
  - **Calories Calculation:** The total daily calorie requirement for each user was calculated using the *Mifflin-St Jeor Equation*:

    $$\text{BMR} = 10 \times \text{Weight (kg)} + 6.25 \times \text{Height (cm)} - 5 \times \text{Age (years)} + 5 \quad \text{(for men)}$$

    $$\text{BMR} = 10 \times \text{Weight (kg)} + 6.25 \times \text{Height (cm)} - 5 \times \text{Age (years)} - 161 \quad \text{(for women)}$$

    The BMR was then adjusted based on the user's activity level to calculate their Total Daily Energy Expenditure (TDEE), which is essential for determining the right calorie intake in the diet plan.
- **Feature Scaling:**

  - Nutritional features like calorie intake, BMI, and health metrics were scaled to ensure that all features have the same range or distribution. This step improves the performance of machine learning models, as unscaled data could lead to biased predictions.
  - We used Min-Max scaling or Z-score normalization on continuous variables (e.g., calorie intake) to standardize the values and ensure compatibility with the machine learning models.

3.2.3 Feature Selection:

- We focused on features like food preferences, allergies, BMI, and user goals (e.g., weight loss, weight gain) to tailor diet recommendations.
- Feature engineering was done to extract important attributes, such as dietary restrictions, from the raw data.

3.3 Step 2: Model Training and Validation

Once the data was preprocessed, the next step involved training and validating the machine learning model that would generate personalized diet plans.

- **Model Selection:**
  - We evaluated different algorithms such as **K-Nearest Neighbors (KNN)** and **Random Forest** for their suitability in recommendation systems.
  - After testing various models, KNN was selected for its ability to classify and recommend based on users' dietary preferences and health metrics.
- **Model Training:**
  - The dataset was split into training (80%) and testing (20%) subsets. The training dataset was used to teach the model how to predict diet recommendations based on user inputs.
  - Hyperparameters for the KNN model, such as the number of neighbors (k), were tuned using cross-validation to optimize performance.
- **Model Validation:**
  - The model was evaluated using the test dataset, and performance metrics such as **accuracy** and **precision** were measured.
  - The model's predictions were compared to actual outcomes to assess its ability to recommend relevant diets accurately.

3.4 Step 3: App Development Process

After the machine learning model was trained, the next step was to build the mobile app that would interface with the model to provide real-time diet recommendations.

- **Frontend Development:**
  - We chose **Flutter** for cross-platform app development, allowing the app to run on both iOS and Android devices.
  - The app's user interface (UI) was designed with an emphasis on simplicity and ease of use. The main features include user profile creation, input of dietary preferences, allergies, and health metrics (e.g., BMI).
  - The UI components were built using **Dart** programming language in Flutter, making the app responsive and user-friendly.
- **Backend Development:**
  - The backend was developed using **Flask**, a lightweight Python framework. It serves as an intermediary between the user's input and the machine learning model.
  - API endpoints were created to fetch user data, send it to the model, and return personalized diet recommendations in real-time.
  - We also integrated the backend with a **MySQL database** to store user data, preferences, and food-related information.
- **Database Integration:**
  - A relational database schema was designed to store essential user information, including dietary preferences, allergies, and health metrics.
  - SQL queries were implemented to retrieve and update data, such as storing users' dietary goals or suggesting diet plans based on individual profiles.

3.5 Step 4: Feature Integration

To make the app highly personalized, several advanced features were integrated into the system. These features work together to provide a more tailored and accurate diet plan for each user.

- **Allergen Detection:**
  - A key feature in the app is the allergen detection system. Users can input their food allergies, and the system cross-references them with the available food database to filter out allergens.
  - When generating recommendations, foods that trigger allergies are automatically excluded from the diet plans.

- **BMI Calculation:**
  - Users are prompted to input their height and weight, and the app calculates their **Body Mass Index (BMI)** and then **BMI Class**().
  - The BMI value is used to suggest appropriate meal plans based on the user's health goals, such as weight loss, maintenance, or muscle building.
- **Real-Time Recommendations:**
  - The app's backend communicates with the machine learning model to generate real-time diet recommendations whenever a user updates their profile or dietary preferences.
  - The recommendation algorithm considers user inputs such as health metrics, allergies, and food preferences.
- **User Profile:**
  - Each user has a unique profile that stores their preferences, dietary restrictions, allergies, and health metrics.
  - The user profile helps to customize diet recommendations by allowing them to select meal or regenerate it and track progress over time.

4 Machine Learning Model

4.1 Data Collection

- The data was collected form the open source as well as through AP. For the prediction of **BMI Category** i.e. Normal Weight , Underweight , Overweight , Obese Class 1 , Obese Class 2 , Obese Class 3 . This data has the input features of Age , weight , height and BMI.
- For the recommendation of food we use the **Spoonacular** API to fetch the data of Asian Cuisine. Our app focus on Asian cuisines only for now. It gets the food Item their calories , micronutrient and the ingredients to give user centric results

4.2 Model Selection

- The machine learning model of **K Nearest Neighbor** is used to train our model which predicts the BMI class according to user input. First the model is trained on 800 users and then it is test on unknown new user and this will predict the BMI class. The model has an accuracy of 0.92 which give the insights that the model si working fine.
- To recommend food on the basis of suer preference we use the AI based based recommendation technique of **Content Based Filtering** which works on recommending the

food based on the calculated cosine similarity score index which helps to get the results more closer to the user preferences.

- We use simple Natural Language technique to get the insights form the food allergies input which is given in comma separated form. The model analyze the user input it also check for the type mistakes using fuzzy library and also the phrase matcher to extract the food with the ingredient so that they don't recommend to the user.

4.3 Prediction of the Body Mass

The primary objective of this project was to predict a model to classify categories of BMI (body mass index) based on characteristics such as age, height, weight and BMI. The data set was processed and modeled using K-Nearest Neighbors (KNN), while handling class imbalance using SMOTE (Synthetic Minority Oversampling Technique). Evaluation metrics and visualizations were also included to measure the performance of the model.

4.3.1 Dataset

We get the data set from the open source of different users age , weight , height and bmi as input features and based on this our model predicts the bmi class in which the user lies. The output feature is divided into 6 classes.

- Normal Weight
- Underweight
- Overweight
- Obese Class 1
- Obese Class 2
- Obese Class 3

4.3.2 Tools and Frameworks

we used the python machine learning models to train our system and to predict BMI. The data set classifies the user into six classes.K Nearest neighbors helps to train the model in order to predict the BMI class which give the accuracy of 0.92.

4.3.3 Implementation

The implementation starts by preprocessing the data in order to remove the missing values which affect the model prediction. Then we perform **Standardization** technique in order to make all the features on one scale. If one feature has highest range of values it become more prominent in the prediction. When we deal with the categorical data we encode it to the numerical data so we used **One Hot Encoding**.

4.3.4 Class Imbalance

Class imbalance occurs when the distribution of target classes in a dataset one or more classes have far fewer instances than others. This imbalance often leads to machine learning models being biased towards the majority class, as they tend to predict the more frequent classes more accurately while ignoring the minority classes.The target variable, BMI Class, represents categories such as Underweight, Normal, Overweight, and Obese.These classes are likely to be unevenly distributed because, in real-world populations Most individuals tend to fall in the Normal or Overweight categories.Fewer people may be classified as Underweight or Obese, depending on regional health statistics and sample demographics.

4.3.5 Reason for Class Imbalance

- The BMI class distribution in the dataset likely reflects real-world health data, where the majority of individuals are in the "Normal" or "Overweight" categories. The "Underweight" and "Obese" categories often represent smaller proportions of the population.
- Sample Collection Bias: If the dataset was collected from a population with specific demographic characteristics (e.g., age group, region, or health status), the distribution may inherently favor some classes over others.
- Health Trends:In many regions, health trends like increased rates of overweight or obesity due to sedentary lifestyles and unhealthy diets can result in an over representation of these categories compared to "Underweight."
- Dataset Size:Smaller datasets are more prone to class imbalance because they may not capture the full diversity of the population.

4.3.6 Impact of Class Imbalance on the Model

- Bias Toward Majority Classes: Machine learning models, including KNN, tend to favor majority classes. This is because the model optimizes for overall accuracy, often at the expense of minority classes.
- Poor Prediction for Minority Classes: Without addressing the imbalance, predictions for "Underweight" and "Obese" classes may be inaccurate or non-existent.

4.3.7 Evaluation Metrics

- Accuracy
  - Formula for Accuracy The formula for Accuracy for each class is:

$$\text{Accuracy}_{\text{Class}} = \frac{\text{True Positives (TP)}}{\text{Total Instances for Class}}$$

    Where:

    * True Positives (TP): The number of correct predictions for the class.
    * Total Instances for Class: The total number of true instances (support) of the class in the dataset.

  – Class-Wise Accuracy Calculations
  The Accuracy for each class is given as follows:

Class	True Positives (TP)	Total Instances (Support)	Accuracy
Normal Weight	100	101	0.9901
Obese Class 1	1	6	0.1667
Obese Class 2	13	13	1.0000
Obese Class 3	16	19	0.8421
Overweight	52	57	0.9123
Underweight	25	27	0.9259

Table 1: Class-wise Accuracy Calculations

  – Interpretation The accuracy for each class measures the fraction of correctly classified instances for that specific class. Here's the breakdown of the calculated values:
    * **Normal Weight**: 0.9901 (99.01% of instances correctly classified as "Normal Weight").
    * **Obese Class 1**: 0.1667 (16.67% of instances correctly classified as "Obese Class 1").
    * **Obese Class 2**: 1.0000 (100% of instances correctly classified as "Obese Class 2").
    * **Obese Class 3**: 0.8421 (84.21% of instances correctly classified as "Obese Class 3").
    * **Overweight**: 0.9123 (91.23% of instances correctly classified as "Overweight").
    * **Underweight**: 0.9259 (92.59% of instances correctly classified as "Underweight").

- Precision Calculation
  – Formula for Precision The formula for Precision for each class is:

$$\text{Precision}_{\text{Class}} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$$

  Where:
    * True Positives (TP): The number of correct predictions for the class.

    * False Positives (FP): Instances predicted as the class but actually belonging to other classes.

- Class-Wise Precision Calculations For your dataset, the Precision for each class can be calculated as follows:

Class	True Positives (TP)	False Positives (FP)	Precision
Normal Weight	100	8	0.93
Obese Class 1	1	0	1.00
Obese Class 2	13	7	0.65
Obese Class 3	16	0	1.00
Overweight	52	2	0.96
Underweight	25	0	1.00

Table 2: Class-wise Precision Calculations

- Interpretation Precision measures the proportion of correct predictions out of all predictions made for a specific class. Here's the breakdown of the calculated values:

    * **Normal Weight**: 0.93 (93% of instances predicted as "Normal Weight" were correct).
    * **Obese Class 1**: 1.00 (100% of instances predicted as "Obese Class 1" were correct).
    * **Obese Class 2**: 0.65 (65% of instances predicted as "Obese Class 2" were correct).
    * **Obese Class 3**: 1.00 (100% of instances predicted as "Obese Class 3" were correct).
    * **Overweight**: 0.96 (96% of instances predicted as "Overweight" were correct).
    * **Underweight**: 1.00 (100% of instances predicted as "Underweight" were correct).

- Recall Calculation
  - Formula for Recall The formula for Recall for each class is:

$$\text{Recall}_{\text{Class}} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}}$$

    Where:
    - True Positives (TP): The number of correct predictions for the class.
    - False Negatives (FN): Instances of the class incorrectly classified as other classes.
  - Class-Wise Recall Calculations
    The Recall for each class is given as follows:

Class	True Positives (TP)	False Negatives (FN)	Recall
Normal Weight	100	1	0.99
Obese Class 1	1	5	0.17
Obese Class 2	13	0	1.00
Obese Class 3	16	3	0.84
Overweight	52	5	0.91
Underweight	25	2	0.93

Table 3: Class-wise Recall Calculations

  - Interpretation Recall measures the ability of the classifier to correctly identify all instances of a specific class. Here's the breakdown of the calculated values:
    - **Normal Weight**: 0.99 (99% of all "Normal Weight" instances were correctly identified).
    - **Obese Class 1**: 0.17 (17% of all "Obese Class 1" instances were correctly identified).
    - **Obese Class 2**: 1.00 (100% of all "Obese Class 2" instances were correctly identified).
    - **Obese Class 3**: 0.84 (84% of all "Obese Class 3" instances were correctly identified).
    - **Overweight**: 0.91 (91% of all "Overweight" instances were correctly identified).
    - **Underweight**: 0.93 (93% of all "Underweight" instances were correctly identified).
- F1 Score Calculation

– Formula for F1 Score The formula for the F1 Score for each class is:

$$\text{F1 Score}_{\text{Class}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

The F1 Score is the harmonic mean of precision and recall, providing a balance between the two metrics.

– Class-Wise F1 Score Calculations
The F1 Score for each class is given as follows:

Class	**Precision**	**Recall**	**F1 Score**
Normal Weight	0.93	0.99	00.96
Obese Class 1	1.00	0.17	0.29
Obese Class 2	0.65	1.00	0.79
Obese Class 3	1.00	0.84	0.91
Overweight	0.96	0.91	0.94
Underweight	1.00	0.93	0.96

Table 4: Class-wise F1 Score Calculations

– Complete Report Summary

Class	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Support**
Normal Weight	0.9901	0.93	0.99	0.96	101
Obese Class 1	0.1667	1.00	0.17	0.29	6
Obese Class 2	1.0000	0.65	1.00	0.79	13
Obese Class 3	0.8421	1.00	0.84	0.91	19
Overweight	0.9123	0.96	0.91	0.94	57
Underweight	0.9259	1.00	0.93	0.96	27

Table 5: Model Evaluation: Accuracy, Precision, Recall, F1 Score, and Support

– Interpretation The F1 Score provides a balance between precision and recall. Here's the breakdown of the F1 Score for each class:

* **Normal Weight**: 0.96 (The F1 score is high, indicating a good balance between precision and recall).
* **Obese Class 1**: 0.29 (A lower F1 score, reflecting a poor balance due to low recall despite perfect precision).
* **Obese Class 2**: 0.79 (A relatively good F1 score, with a moderate precision and perfect recall).
* **Obese Class 3**: 0.91 (A high F1 score, indicating a good balance of precision and recall).
* **Overweight**: 0.94 (A very high F1 score, indicating strong performance across both metrics).

  * **Underweight**: 0.96 (A high F1 score, indicating a good balance between precision and recall).

- Confusion Matrix

Actual \Predicted	Normal Weight	Obese Class 1	Obese Class 2	Obese Class 3	Overweight	Underweight
Normal Weight	100	0	0	0	1	0
Obese Class 1	1	4	0	0	1	0
Obese Class 2	0	0	13	0	0	0
Obese Class 3	0	0	0	16	0	3
Overweight	0	0	0	0	52	5
Underweight	0	0	0	0	0	25

Table 6: Confusion Matrix for K-Nearest Neighbors Classification with imbalance

4.3.8 Handling Class Imbalance: SMOTE

To mitigate the impact of class imbalance in the BMI dataset, SMOTE (Synthetic Minority Oversampling Technique) was applied: SMOTE generates synthetic data points for minority classes by interpolating between existing samples. This balances the distribution of target classes in the training dataset, ensuring the model learns equally from all categories.

- Classification Report

Class	Precision	Recall	F1-Score	Support	Accuracy
Normal Weight	0.98	0.94	0.96	101	0.94
Obese Class 1	1.00	0.50	0.67	6	0.50
Obese Class 2	0.72	1.00	0.84	13	1.00
Obese Class 3	1.00	0.84	0.91	19	0.84
Overweight	0.93	0.98	0.96	57	0.98
Underweight	0.90	0.96	0.93	27	0.96

Table 7: Classification Report after removing imbalance

- Confusion Matrix for K-Nearest Neighbors

Actual \Predicted	Normal Weight	Obese Class 1	Obese Class 2	Obese Class 3	Overweight	Underweight
Normal Weight	100	0	0	0	1	0
Obese Class 1	1	4	0	0	1	0
Obese Class 2	0	0	13	0	0	0
Obese Class 3	0	0	0	16	0	3
Overweight	0	0	0	0	52	5
Underweight	0	0	0	0	0	25

Table 8: Confusion Matrix for K-Nearest Neighbors Classification after removing imbalance

4.4 Diet Recommendation using Content Based Filtering

Content-based food recommendation system is implemented to recommend food items based on user preferences for nutrients (Calories, Fats, Proteins, Carbohydrates). The system processes a dataset of food items, each with nutritional information, and provides personalized food recommendations to users for different meal categories like Breakfast, Lunch, Dinner, Snacks, and Side Dishes based on user calories need and food allergies.

4.4.1 Dataset Loading

The dataset used for this recommendation system is generated using **spoonacular** API and most preferably gathered of Asian cuisine, which contains nutritional information for various food items. The dataset is loaded and processed by checking for null values and setting the `Food Items` column as the index. If any null values are found, they are handled accordingly.

4.4.2 Normalization

Data normalization is performed to standardize the values for each nutrient: Calories, Fats, Proteins, and Carbohydrates. This normalization ensures that each nutrient contributes equally to the cosine similarity calculation, preventing any single nutrient from dominating the recommendation process.

4.4.3 Meal Categorization

The dataset is divided into meal categories: Breakfast, Lunch, Dinner and Snack.

4.4.4 Recommendation Algorithm

- The user provides their preferred nutritional values (Calories, Fats, Proteins, Carbohydrates) and the food allergies for the recommendation system to use.
- For each meal type, the system calculates the **cosine similarity** between the user's nutritional preferences and the available food items.

4.4.5 Recommendation Output

The system outputs the top recommended food items for each meal type based on the **highest cosine similarity scores**. These recommended items are accompanied by their ingredients, providing users with information about the food they are considering.

4.4.6 Selection Process

For each meal category, the user is shown the 3 recommended dishes based on their preferences. The user selects one dish from these recommendations, ensuring that the selected dish aligns with their nutritional needs.

4.4.7 Nutrient Tracking

Once the user has selected dishes for each meal category, the system calculates the total nutritional values (Calories, Fats, Proteins, Carbohydrates) based on the selected dishes. The total nutrient intake is then compared with the user's original preferences to evaluate the alignment.

4.4.8 Nutrient Comparison

The system compares the total nutrients of the selected dishes with the user's preferences and provides feedback. It checks if the total calories exceed the user's preferred limit, and warns the user if any other nutrients (Fats, Proteins, Carbohydrates) exceed the preferred limits. The system provides warnings for any nutrient that exceeds the user's defined threshold, ensuring that the selected food items align with their dietary goals.

4.5 Allergens Detection

The Allergy Detection helps user in selecting food items based on their nutritional preferences while avoiding ingredients they are allergic to. The system is built using Python, leveraging libraries such as spaCy, fuzzywuzzy, and csv. This utilizes fuzzy string matching to detect and correct possible misspellings or variations in allergen names, ensuring accurate allergy filtering.

4.5.1 Text Processing and Normalization

The user inputs their allergies as a comma-separated list of terms. The system normalizes these terms by stripping extra spaces and converting them to lowercase.

4.5.2 Fuzzy Matching

The system uses fuzzy string matching to handle variations in how allergies might be spelled by users. Although this is not strictly NLP in the traditional sense (e.g., it doesn't involve syntax, semantics, or grammar), it does rely on string comparison algorithms.

4.5.3 Handling Synonyms and Misspellings

The system can recognize potential misspellings and synonyms in the allergy terms provided by the user. For example, "mik" and "milk" might be matched using fuzzy logic, helping the system understand terms that are phonetically similar or commonly abbreviated.

4.5.4 Text Matching (Allergy Detection)

The system attempts to match user-provided terms to a list of predefined allergens using fuzzy matching techniques. This is a form of text-based information retrieval, which is often considered a part of NLP. Although the system doesn't employ complex language models, it still processes and interprets the meaning of the user's input based on similarity to a list of known allergens.

4.5.5 NLP Elements Involved

- **Tokenization**: Splitting the allergy input into individual terms (tokens) for processing.
- **Normalization**: Converting text to a standard form (e.g., lowercase) and removing extraneous spaces.
- **Fuzzy Matching**: Finding approximate matches of text based on similarity metrics, which is a technique often used in NLP for handling variations in text.

4.5.6 Limitations

While the system works to some extent, it is not a fully developed solution and still has limitations. The fuzzy matching technique can sometimes produce false positives or negatives, especially if the input terms are highly ambiguous or too far off from the predefined allergens. Further improvements in handling complex synonyms, spelling variations, and context would be necessary to create a more robust allergen detection system.

5 System Design and Architecture

5.1 System Overview

The system is the first step towards the development of the solution that assist nutritionist and common man to generate the diet plan according to their preferences. The system gets user input and generate in accordance to user It starts by calculating the caloric need and then recommending food , it allow user to change their preferences at any time

5.2 System Architecture

5.2.1 Front-End

The front-end is the user-facing part of the application, built with Flutter. It is responsible for collecting user input, displaying results, and managing navigation.

- **Technology Stack:** Flutter (cross-platform for Android, iOS, Web, Desktop).
- **Responsibilities:**
  - Collect user input (e.g., height, weight, age , physical activity , gender).
  - Display visual feedback (e.g., BMI, Calories Needed along with breakdown of proteins , fats and carbohydrates food recommendations).
  - Communicate with the back-end via REST APIs.

5.2.2 Back-End

The back-end is responsible for business logic, data validation, and integration with databases and machine learning models.

- **Technology Stack:** Node.js, Python (Flask).
- **Responsibilities:**
  - API Gateway for communication with the front-end.
  - Data validation and processing (e.g., calculating BMI , calories and macro nutrients).
  - Integration with machine learning models for predictive analytics.

5.2.3 Database

The database stores user data, preferences, and potentially training data for machine learning.

- **Technology Stack:** MySQL.
- **Responsibilities:**

  - Store user information (e.g., name, age, height, weight).
  - Save application state (e.g., user progress, preferences).

5.2.4 Machine Learning Models

The machine learning model adds predictive or analytical capabilities to the system.

- **Technology Stack:** Python (Scikit-learn).
- **Responsibilities:**
  - Predict health metrics (e.g., BMI category) using K Nearest Neighbors.
  - Recommend food using content based filtering according to user preferences.

5.2.5 Data Flow Example

- The user enters personal details, height, and weight , age , gender , physical activity and food allergies through the Flutter app.
- The front-end sends the collected data to the back-end via a REST API.
- The back-end validates the data, calculates basic metrics (e.g., BMI) and store it in the datasbe.
- The machine learning model processes the data stored in database fetched via API and returns a prediction (e.g., BMI category , meal plan recommendation).
- User data and results are saved in the database for future use.
- The back-end sends the processed results back to the front-end, which displays them to the user.

5.3 Database Design

For the app the database is designed to store the suer information so that user don't have to input its preference every time . he just login and get the meal for the day in one click and can update only one the thing he needs to. It also have the facility to store user meals to keep track of its daily diet and calories needed.

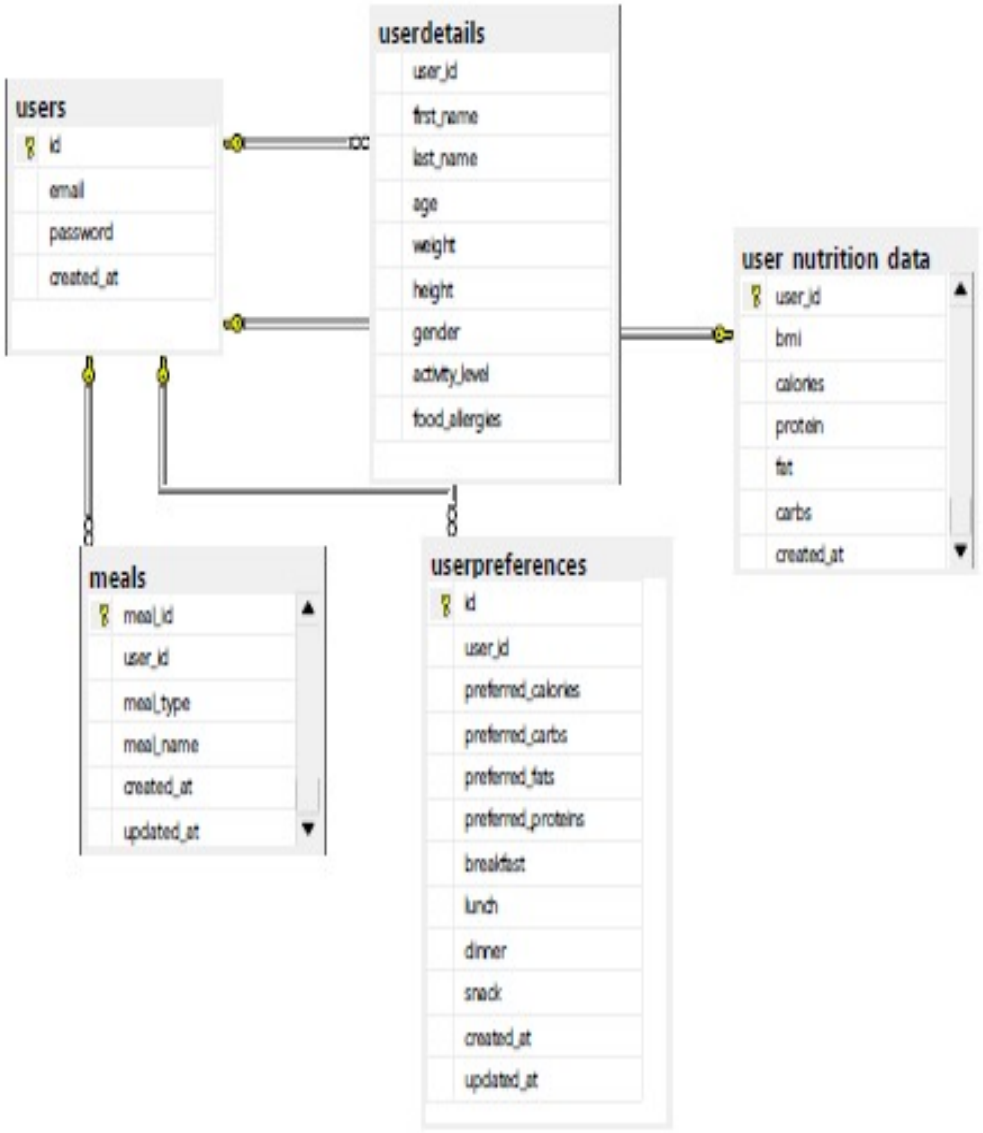

Figure 2: Database Design for the Ai based Diet Recommendation system

5.4 User Interface Design

Home Page

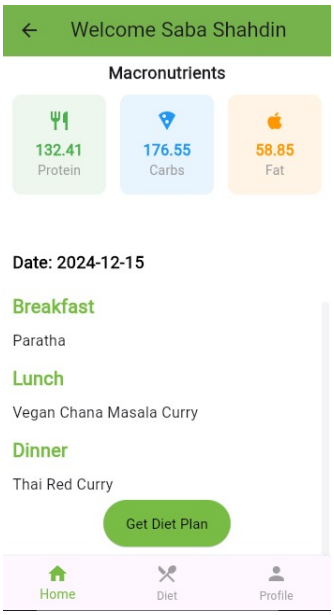

Figure 3: Home page displaying an overview of features and quick access options.

Body Mass Classification

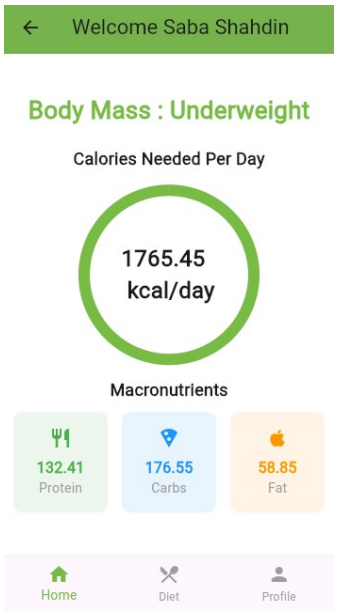

Figure 4: Page displaying calculated body mass index (BMI) classification.

Diet Recommendation Page

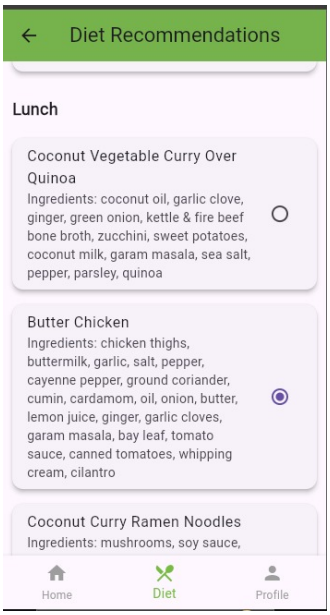

Figure 5: Personalized diet recommendations based on user data and preferences.

User's Profile Overview

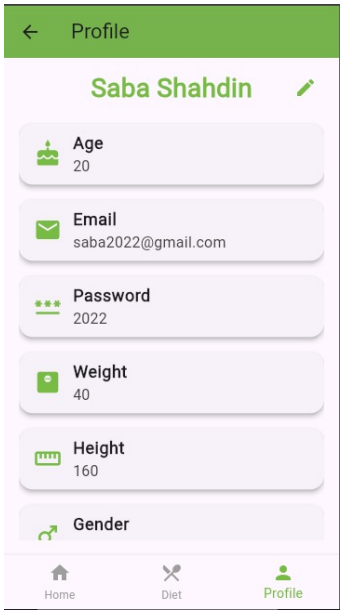

Figure 6: Overview of the user's profile, displaying personal details.

5.5 User Flow

The following diagram illustrates the step-by-step interaction of the user with the diet recommendation system, beginning from the registration process to receiving personalized diet recommendations.

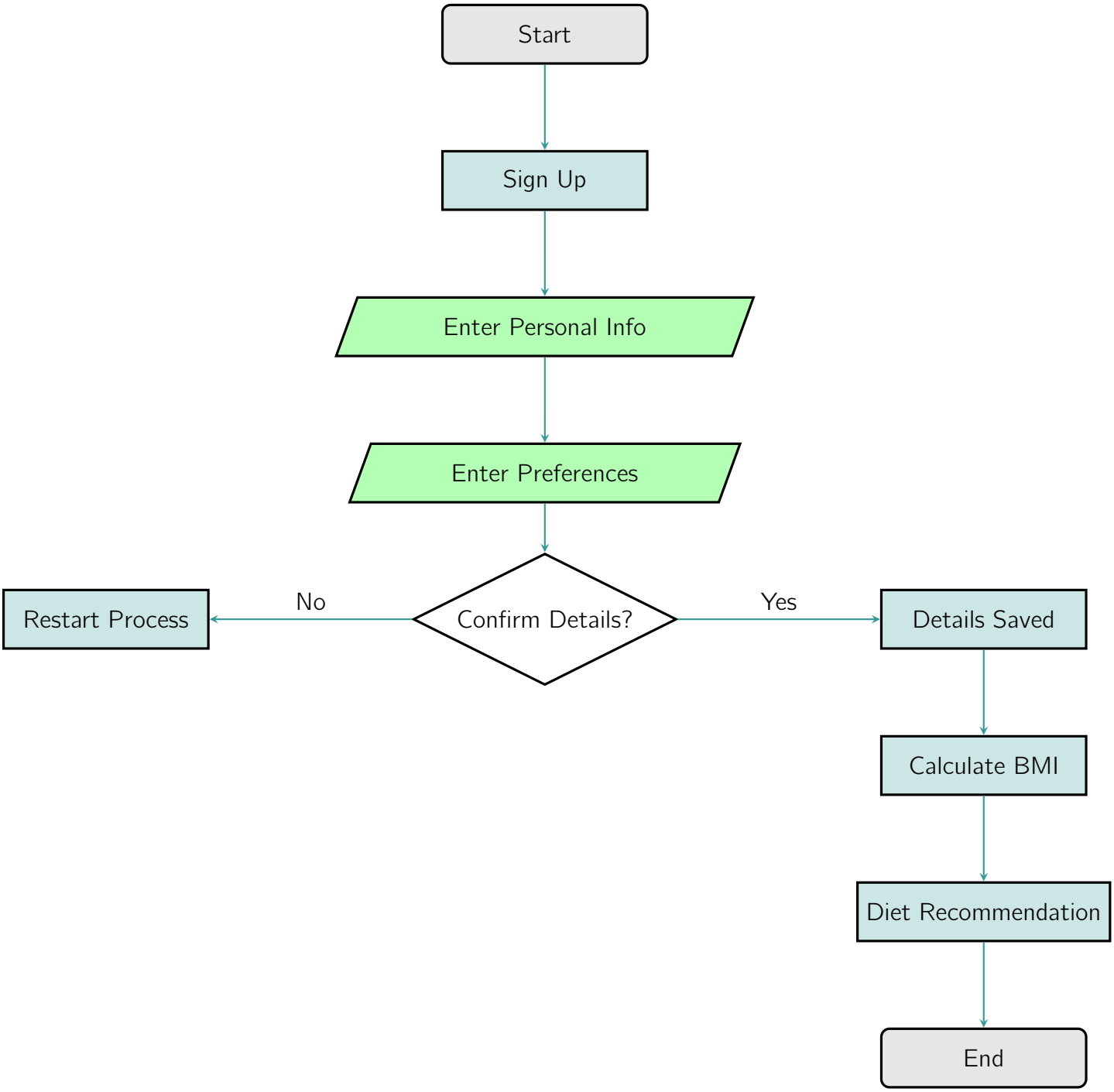

Figure 7: User Interaction with Diet Recommendation App

6 Implementation

Below is the illustratiom of the journey from user sign-up and health metric updates to BMI prediction, Macro Nutrients Calculation,allergen detection, and personalized food recommendations through machine learning and content-based filtering techniques

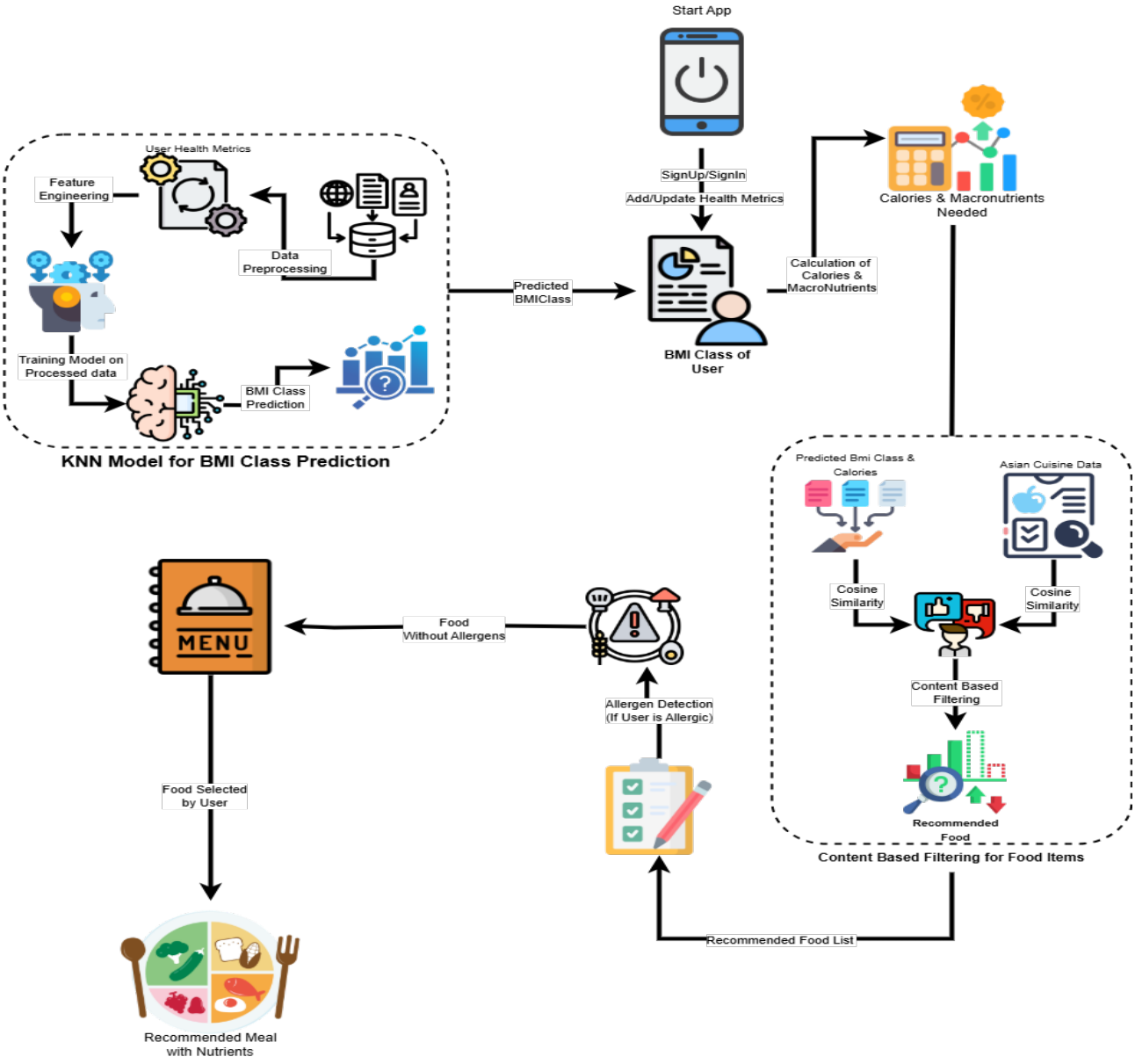

Figure 8: Methodology for Personalized Diet Recommendation System

6.1 User Data

The system collects essential user details to provide personalized recommendations, including:

- **Personal Information**: Age, weight, height, and gender.
- **Physical Activity Level**: The system considers the user's physical activity level, categorized as active, sedentary, or light.

Using the collected data, the system performs the following steps:

- **Body Mass Index (BMI) Calculation**: The BMI is calculated using the standard formula based on the user's weight and height.
- **Calorie and Macronutrient Calculation**: The system calculates the daily calorie needs and macronutrients (carbohydrates, protein, fats) based on the user's input and activity level.
- **Machine Learning Model for Body Mass Prediction**: A machine learning model is employed to predict the user's body mass and potential health risks.

6.2 Diet Recommendation

The system provides diet suggestions for the day, covering all four meal types: breakfast, lunch, dinner, and a snack.

- For each meal, the system suggests three options. The user can choose one option per meal type.
- **Allergen-Free Recommendations**: The system takes into account any allergens specified by the user and removes ingredients that the user is allergic to.
- After the user selects their meal options for the day, the system calculates the total nutrient values for the day, including calories, protein, fats, and carbohydrates.

6.3 Meal Customization

Users have the flexibility to update their meal choices for the day based on preferences or new dietary requirements.

- The system allows users to set and update preferences for future meal suggestions, ensuring that the recommendations align with their dietary goals and restrictions.

7 Challenges Faced

The major challenge in developing this system is the collection f data of the food in the Asia and their nutrient levels . Te other thing is the combination of artificial intelligence e along with recommendation approach to give more defined results. We face challenge in reading

and decrypting the allergies written in human language comma separated on which we still working but until no it is working to the extent which aligns to our sytem.

8 Summary

In this project, a personalized diet recommendation system was developed to help users manage their nutritional needs based on their physical characteristics, activity levels, and dietary preferences. The system collects essential user data such as age, weight, height, gender, and activity level, and calculates BMI, daily calorie requirements, and macronutrient distribution. Using these inputs, the system suggests meal options for the day, taking into account user preferences and allergen restrictions. A machine learning model was integrated to predict the user's body mass and the system allows users to customize their meal plans for flexibility and long-term use. The system's functionality was evaluated through rigorous testing, ensuring its accuracy in calorie and nutrient calculation, as well as user satisfaction in meal recommendations.

8.1 Key Findings

Throughout the development process, key findings included the importance of personalization for user engagement, as allowing users to customize their diet plans led to higher satisfaction. Accurate nutrient calculation built trust by providing reliable calorie and macro nutrient information. The allergen-free recommendations made the system more accessible to users with food sensitivities. The machine learning model provided valuable health insights, though its accuracy can improve with more data. Lastly, the system's flexibility in updating meal plans and preferences ensured long-term usage and adaptability to users' evolving needs.

8.2 Limitations

While the diet recommendation system offers personalized suggestions and integrates advanced features, there are certain limitations to address:

- **Limited Data Diversity:** The dataset may not fully represent global cuisines, limiting the cultural applicability of recommendations.
- **Dependency on User Inputs:** Accurate recommendations rely heavily on the correctness and completeness of user-provided data, such as health metrics and dietary preferences.
- **Basic Allergy Detection:** The allergen detection mechanism may produce false positives or negatives due to its reliance on fuzzy matching techniques.
- **Static Recommendations:** The current system does not dynamically adjust meal plans in real-time based on changes in activity or health metrics.
- **Wearable Integration:** The system lacks integration with wearable devices for real-time data collection and health monitoring.

- **Model Performance:** The KNN model's accuracy is dependent on the quality of the training data and may not scale well with larger datasets.
- **Limited Personalization:** Stress levels, mental states, and other nuanced health factors are not currently incorporated into the recommendation process.
- **Privacy Concerns:** Although basic data security measures are implemented, further enhancements are needed to ensure compliance with advanced privacy standards like GDPR and HIPAA.
- **Offline Functionality:** The system requires internet access for data processing and API integration, making it inaccessible offline.
- **Scalability:** The current architecture may face performance issues with an increase in user base or dataset size.

8.3 Future Work

Future improvements aim to enhance personalization, accuracy, and user experience:

- **Wearable Integration:** Real-time health monitoring, dynamic calorie adjustment, and hydration tracking.
- **Advanced Models:** Incorporate deep learning and reinforcement learning for adaptive recommendations.
- **Personalization:** Leverage long-term trends and stress levels for tailored dietary plans.
- **Holistic Health:** Combine diet, exercise, and sleep insights for comprehensive health management using wearable devices.
- **Gamification:** Introduce rewards for meeting dietary and activity goals.
- **Expanded Database:** Add diverse regional foods and sports nutrition options.
- **Healthcare Collaboration:** Validate with professionals and share reports securely.
- **Privacy and Security:** Enhance encryption and user privacy.
- **Multi-Language Support:** Cater to global users with multilingual recommendations.
- **Grocery Integration:** Enable direct ingredient ordering and meal prep suggestions.
- **Long-Term Analysis:** Use wearable data for tracking progress and improving health outcomes.

9 Conclusion

The personalized diet recommendation system successfully addresses the need for tailored nutritional guidance by integrating user-specific data, machine learning techniques, and dietary customization features. Through the calculation of BMI, daily calorie requirements,

and macronutrient distribution, the system provides users with meal recommendations that align with their health goals, preferences, and allergen restrictions. Its user-centric approach ensures accessibility and adaptability, enhancing the overall experience for diverse user groups.

While the project demonstrates significant achievements in accurate nutrient calculation and meal customization, it also highlights areas requiring further development, such as integrating real-time data from wearable devices, improving model performance, and expanding the food database. The inclusion of machine learning introduces a valuable predictive component, offering insights into users' health trajectories.

Overall, this system lays a strong foundation for future enhancements in personalized nutrition, combining innovative technology with practical applications to promote healthier lifestyles. By addressing the identified limitations and incorporating advanced features, the system has the potential to become a comprehensive tool for long-term health and wellness management.

10 Link of Repository

More tracking of the project can be found on the given repository below:

https://github.com/afferafatima/Diet-Recommendation

11 References

- Scikit-learn: Machine Learning in Python - https://scikit-learn.org/stable/
- Spoonacular API - https://spoonacular.com/food-api
- Pandas: A Python Data Analysis Library - https://pandas.pydata.org/
- Flask: A Micro Web Framework - https://flask.palletsprojects.com/
- spacy: Phrase Matcher -https://spacy.io/api/phrasematcher/
- Subbaraj, Karthika. "Multi-Choice Diet Recommendation Application for Indian Scenario Based on Insights from Ensemble Learning Techniques." Engineering Proceedings, 2024. - https://www.mdpi.com/2673-4591/62/1/13
- Kaggle: Data Science and Machine Learning Platform - https://www.kaggle.com/
- Geeks For Geeks: Content Based Filtering - https://www.geeksforgeeks.org/ml-content-based-recommender-system/