Global Journal of Engineering Innovations & Interdisciplinary Research

Original Article

Diet Recommendation System

L. Suresh[1], Kanturi Vikas[2], Jannapureddy Kalyani[2], Guduru Shruthilaya Kiran[2], Bitla Uday Kiran[2]

[1]AssociateProfessor, Department of CSE, Christu Jyothi Institute of Technology & Science, Jangaon, Telangana, India
[2]UG Students, Department of CSE, Christu Jyothi Institute of Technology & Science, Jangaon, Telangana, India

Correspondence

Suresh L

Assistant Professor, Department of CSE, Christu Jyothi Institute of Technology & Science, Jangaon, Telangana, India

- Received Date: 25 Jan 2026
- Accepted Date: 14 Mar 2026
- Publication Date: 20 Mar 2026

Keywords

Python, Pandas, Numpy, Fast API, Streamlit, Docker, ML model: KNN

Abstract

Irregular eating patterns and poor nutritional awareness have led to increased health complications in modern lifestyles. This paper presents a Diet Recommendation System that generates personalized food suggestions by directly matching user-defined nutritional requirements with a large-scale recipe dataset. Unlike traditional systems that depend on user history, the proposed model uses a content-based filtering approach, eliminating the cold-start limitation. The system applies the K-Nearest Neighbors (KNN) algorithm with cosine similarity to identify recipes that closely align with user-defined calorie and macronutrient targets. The application is built using FastAPI for backend processing and Streamlit for the frontend interface, with Docker ensuring consistent deployment. Experimental results demonstrate that the system provides recommendations within a ±10% accuracy range while maintaining low response latency. The approach ensures transparency, scalability, and immediate usability for new users.

Introduction

The pervasive shift towards modern, fast-paced lifestyles has introduced significant challenges to maintaining balanced and healthy eating habits. Globally, there is a mounting health crisis linked to poor diet, characterized by rising rates of obesity, diabetes, and cardiovascular disease. Traditional dietary guidance often relies on generalized recommendations that fail to account for an individual's unique biological data, lifestyle factors, or specific nutrient needs. To overcome this limitation, our project, the Diet Recommendation System, leverages the power of machine learning and modern web technologies to deliver highly personalized and actionable dietary advice.

Background

In today's modern environment, people are increasingly concerned with their health and lifestyle. However, many existing diet management tools and applications suffer from common limitations. Most basic diet apps offer generic meal plans or calorie trackers, but they often fail to provide recommendations truly balanced for an individual's specific metrics, such as height, weight, and age. Many recommendation systems are collaborative, meaning they recommend items that similar users liked. This approach fails for new users, as the system has no data on them or their "neighbors," leading to what is known as the "cold start" problem..

Problem Statement

The core problem is the lack of an intelligent, transparent, and data-driven platform capable of bypassing the "cold start" problem to deliver highly personalized, nutritionally accurate dietary recommendations tailored to an individual's specific biometric profile and health objectives. Traditional dietary guidance and existing diet management applications primarily rely on a "one-size-fits-all" approach, offering generalized meal plans or basic calorie tracking that fails to account for a user's unique biological data. Users are restricted to a highly repetitive list of generic "healthy" foods, which drastically reduces long-term diet adherence, and current apps act as "black boxes," recommending diets without explaining the nutritional justification, thereby reducing user trust.

Literature Survey

The design and development of this Diet Recommendation System were grounded in research across three key domains: Machine Learning Algorithms, Nutritional Science, and Software Architecture. To build a reliable recommendation engine, we referenced methodologies for K-Nearest Neighbors (KNN), adopting techniques from the study on "Career Recommendation System using KNN," which applied similar similarity-based logic. The strategies for curating and processing dataset inputs were informed by the paper "Automated web usage data mining and recommendation system using (KNN) classification method".

Citation: Suresh L, Kanturi V, Jannapureddy K, Guduru SK, Bitla UK. Diet Recommendation System. GJEIIR. 2026;6(3):0171.

To support the automatic generation of tailored diet plans, we drew guidelines from the "Personalised nutrition and health (Food4Me Study)" published in the BMJ, which highlights the effectiveness of personalized interventions. Furthermore, the choice to use FastAPI for high-performance service delivery was based on "Model Algorithm Research based on Python Fast API," utilizing its asynchronous capabilities for faster inference times. To standardize the application environment and ensure seamless deployment, we followed best practices found in "Utilizing Docker Containers for Reproducible Builds and Scalable Web Application Deployments".

Proposed system

The proposed system is a Diet Recommendation System built using machine learning designed to overcome the limitations of existing systems by providing personalized, transparent, and varied food recommendations. The system uses the characteristics or content of an item (e.g., its nutritional profile) to recommend similar items, utilizing the Nearest Neighbors algorithm (specifically BallTree, KDTree, and brute-force algorithms) from scikit-learn. The application features a user-friendly web interface built with Streamlit, allowing for a clean and interactive user experience with dual recommendation modes: an Automatic Diet Recommendation Page based on biometrics, and a Custom Food Recommendation Page. The application logic is served via a web API built with FastAPI, which uses the machine learning model to generate a list of recommended foods and returns them to the user.

Modules Used

- **User Interface (Frontend) Module** This module is the sole point of interaction for the end-user, developed using the Streamlit Python framework. It provides an Automatic Diet Recommendation page that collects personal biometrics to calculate daily caloric needs, and a Custom Food Recommendation page allowing users to manually input specific nutritional targets like target calories, grams of protein, and water intake.
- **Backend API Processing Module** Operating as the central nervous system of the application, the backend module is built using FastAPI to ensure rapid, asynchronous request handling. It exposes the core /recommend endpoint, which receives formatted JSON payloads, constructs a nutritional query vector, and routes this request to the machine learning engine. Utilizing startup events, this module pre-loads the serialized machine learning model and the massive recipe dataset directly into memory when the server starts.
- **Machine Learning & Recommendation Engine** Module Powered by the Scikit-Learn library, this is strictly a content-based recommendation engine that matches user requests against the actual nutritional properties of food items. It runs offline data preprocessing to scale numerical nutritional features and clean raw text, then uses the K-Nearest Neighbors (KNN) algorithm and Cosine Similarity to mathematically locate the most suitable food items.
- **Data Management Module** This module is responsible for loading and parsing the extensive Food.com Kaggle dataset, which contains detailed nutritional profiles and ingredient lists for over 500,000 distinct recipes. It heavily relies on Pandas and Numpy for high-speed data manipulation and in-memory lookups.
- **Containerization and Deployment Module** To ensure the system is reproducible, portable, and easily deployable, the entire architecture is packaged into isolated environments using Docker and Docker-Compose. It utilizes specific Dockerfile instructions to define the runtime environment and a docker-compose.yml file to map necessary host ports and establish a secure shared internal network between the UI and API.

Technologies Used

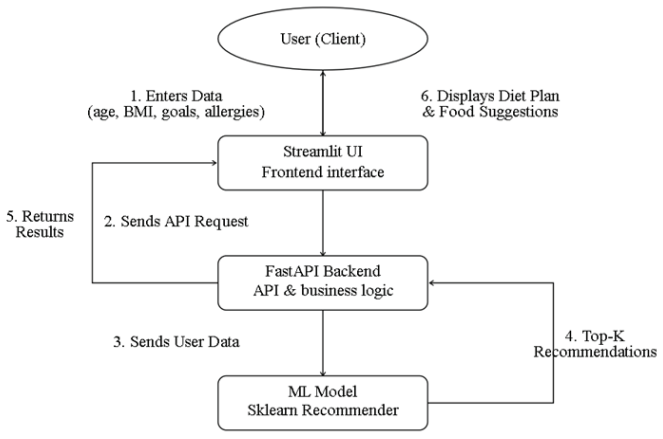
Programming Language: Python (v3.10.8)
Backend Framework: FastAPI (v0.88.0)
Frontend Framework: Streamlit (v1.16.0)
Machine Learning: Scikit-Learn (v1.1.3), Pandas (v1.5.1), Numpy (v1.21.5)
Containerization Platform: Docker and Docker-Compose
Data Parsing: BeautifulSoup4 (v4.11.1)

Advantages of proposed system

- **Avoids the "Cold Start" Problem:** Because the system is content-based, it does not require data from other users to start making recommendations, providing relevant suggestions to a brand-new user immediately.
- **Transparent and Relevant Recommendations:** The recommendations are transparent as the system suggests items because their nutritional profile matches the user's explicit request.
- **Promotes Dietary Variety:** By recommending a variety of healthy foods based on their characteristics, the system helps users discover new and nutritious options, directly combating "food boredom".
- **Ease of Development:** Content-based filtering systems are generally easier to create and maintain than complex collaborative filtering systems.
- **Modern and Fast:** The use of FastAPI for the backend ensures the application is fast and efficient, while Streamlit allows for a modern, interactive frontend.

System Architecture

The system architecture follows a clear Data Flow Diagram. The User initiates the process by providing "Personal Data & Requests" to the Streamlit User Interface. The User Interface forwards the user's needs as a request to the FastAPI Recommendation System. The Recommendation System processes this request by pulling User Profile data and querying the Food & Nutrition Database using the processed diet request vector.



The Food & Nutrition Database responds by sending Food Nutritional Data back to the Recommendation System. Finally, the Recommendation System processes the data and sends the final JSON response to the Streamlit UI, which displays the formatted Diet Plan & Recommendations back to the User.

Results

The system was evaluated using a comprehensive suite of UI, API, and Model logic tests. The Streamlit UI successfully captured user biometrics, handled boundary constraints gracefully, and dynamically rendered complex nutritional pie charts mapping macronutrients. The FastAPI backend proved highly resilient, successfully catching all improperly formatted JSON requests and returning standard 422 Unprocessable Entity errors rather than suffering internal server errors. The core Machine Learning recommendation engine effectively bypassed the "cold start" problem, directly calculating high-quality recommendations based entirely on the inputted nutritional metrics, ensuring calorie suggestions reliably stayed within a ±10% threshold.

Conclusion

The Diet Recommendation System is a powerful, data-driven tool designed to help users find recipes that align with their specific nutritional goals. With features such as content-based filtering via a NearestNeighbors model, an interactive Streamlit frontend, and a scalable FastAPI backend, the application provides a streamlined way to get personalized meal suggestions. It successfully allows users to input complex targets, such as a 2100 calorie goal or a protein range of 112-154g, and receive a highly relevant list of recipes from a massive dataset. By utilizing Docker containerization, this framework represents a highly scalable and computationally efficient solution for personalized digital health management..

Future scope

User Account and Profile Management: Implement a secure user authentication system (e.g., with a database like PostgreSQL) to allow users to create profiles, save their personal goals, and log daily food and water intake.

Full-Scale Meal Plan Generation: Develop an algorithm to generate a full 7-day meal plan complete with variety, and implement a "Pantry" feature to prioritize recipes using ingredients users already have.

Enhanced Recommendation Engine: Integrate Collaborative Filtering with the existing Content-Based model to create a more accurate hybrid model, and utilize Ingredient-Level NLP for more complex queries.

Integration with External Services: Connect with services like Google Fit or Apple Health to automatically pull user activity data to dynamically adjust calorie recommendations, and automatically generate grocery shopping lists.

References

1. Rajput, S. H. et al. "Career Recommendation System using KNN," International Journal of Creative Research Thoughts (IJCRT), 2024.
2. Ordovas, Jose M. et al. "Personalised nutrition and health (Food4Me Study)," BMJ, 2018.
3. Agrawal, Kushagra et al. "Artificial intelligence in personalized nutrition and food manufacturing," PubMed Central (PMC), 2025.
4. Anonymous. "Model Algorithm Research based on Python Fast API," BC Publication, 2023. Adeniyi, D. A. et al. "Automated web usage data mining and recommendation system using (KNN) classification method," Applied Computing and Informatics, 2014.
5. Anonymous. "Utilizing Docker Containers for Reproducible Builds and Scalable Web Application Deployments," Inpressco, 2025.