# PROJECT TITLE

## TEAM MEMBERS:

ETAN

CHRISTIAN

ARTHUR

ROSINE

KESTIA

ELENA

## OOADI

TEKO PALMA

ICT UNIVERSITY

# 📑 **Table of Contents**

1. **Executive Summary**
2. **Introduction**  
   2.1. Problem Statement  
   2.2. Objectives  
   2.3. Scope of the Project  
   2.4. Target Audience
3. **Scrum Methodology**  
   3.1. Overview of Scrum  
   3.2. Scrum Roles  
   3.3. Scrum Artifacts  
   3.4. Scrum Ceremonies
4. **Product Backlog**  
   4.1. Feature List  
   4.2. Prioritization
5. **Sprint Planning**  
   5.1. Sprint Breakdown  
   5.2. Task Allocation  
   5.3. Sprint Goals
6. **Burndown Charts**  
   6.1. Sprint 1 Burndown  
   6.2. Sprint 2 Burndown  
   6.3. Sprint 3 Burndown  
   6.4. Sprint 4 Burndown
7. **System Architecture**  
   7.1. High-Level Architecture  
   7.2. Component Diagram  
   7.3. Data Flow Diagram (DFD)
8. **UML Diagrams**  
   8.1. Use Case Diagram  
   8.2. Class Diagram  
   8.3. Sequence Diagram  
   8.4. Activity Diagram
9. **Technology Stack**
10. **Database Design**  
    10.1. ER Diagram  
    10.2. Schema Description
11. **Implementation Details**  
    11.1. Core Features  
    11.2. Screenshots and Explanations
12. **AI Integration and Chat System**  
    12.1. Gemini API Setup  
    12.2. Error Handling (503 Handling)  
    12.3. Prompt Engineering
13. **Testing and Quality Assurance**  
    13.1. Test Strategy  
    13.2. Sample Test Cases  
    13.3. Bug Log and Fixes
14. **Evaluation and Retrospective**  
    14.1. What Went Well  
    14.2. Challenges and Lessons Learned
15. **Future Improvements**
16. **Conclusion**
17. **Appendices**  
    17.1. Full Code Snippets  
    17.2. Sample API Calls and Responses  
    17.3. Additional Screenshots
18. **References and Bibliography**

## **1. Executive Summary**

This project presents a comprehensive web-based health and fitness application designed to empower users to make informed dietary and exercise decisions. The system allows users to search for Cameroonian and global foods, retrieve their nutritional values (specifically calories), and receive actionable recommendations on exercises needed to burn those calories.

In addition to calorie lookup, the application provides intelligent meal suggestions, including breakfast, dinner, and dietary advice tailored to different health goals. The app also features an integrated conversational AI assistant, enabling users to ask questions in natural language about food preparation, nutrition, and healthy living. The AI is powered by Google Gemini and offers recipe instructions, wellness tips, and personalized fitness suggestions.

Using the Scrum agile methodology, the development team built the application incrementally over multiple sprints. The project followed a structured approach with product backlog management, sprint planning, and daily evaluations, resulting in a feature-rich, user-friendly platform.

With its fusion of data-driven fitness recommendations and conversational AI support, the application offers a scalable solution for individuals pursuing healthier lifestyles through culturally relevant and accessible tools.

## **2. Introduction**

### **2.1 Problem Statement**

In recent years, the rise of lifestyle-related diseases and poor eating habits has emphasized the need for tools that support informed dietary and fitness decisions. Many individuals struggle to understand the nutritional content of the food they consume or determine appropriate exercises to maintain a healthy balance. Existing applications often lack cultural inclusivity, especially for underrepresented cuisines like Cameroonian dishes, and fail to provide intelligent guidance in a conversational, user-friendly way.

Furthermore, there is an increasing demand for AI-driven platforms that offer real-time interaction, customized feedback, and educational guidance about health, nutrition, and fitness. The lack of a unified system that brings all these components together creates a gap in digital health empowerment for everyday users.

### **2.2 Project Objectives**

The primary aim of this project is to develop a web-based application that provides users with:

* ✅ **Calorie Lookup**: Search and view the calorie content of various foods, including local dishes.
* ✅ **Exercise Suggestions**: Automatically recommend exercises to burn consumed calories.
* ✅ **Meal Recommendations**: Suggest breakfast, lunch, dinner, and diet plans based on user queries or preferences.
* ✅ **Integrated AI Chat Assistant**: Use conversational AI to allow users to ask health, nutrition, or cooking-related questions.
* ✅ **Recipe Generator**: Return detailed cooking instructions for user-inputted foods.
* ✅ **Cultural Relevance**: Provide support for Cameroonian and African cuisine for inclusivity.

### **2.3 Project Scope**

The application covers the following features:

* ✔️ A front-end user interface to input food, view calories, and interact with the system.
* ✔️ A backend system (Django) for handling logic, recipes, and data storage.
* ✔️ AI chat integration using Google Gemini to answer user questions intelligently.
* ✔️ Recipe extraction and direction system for traditional meals.
* ✔️ Food-to-calorie and calorie-to-exercise mapping for informed decision-making.
* ✔️ API calls and database support for real-time queries.

The project **does not** aim to:

* Provide medical-grade dietary diagnoses.
* Replace the advice of a certified nutritionist or personal trainer.

### **2.4 Target Audience**

This application is designed for:

* Individuals interested in **maintaining or improving their health**.
* Users who want to understand the **nutritional value of their meals**.
* People looking for **culturally-relevant dietary and cooking guidance**.
* Students and professionals who want **quick AI-based food and fitness help**.
* Beginners seeking **simple home-based workouts** based on what they eat.

## **3. Scrum Methodology**

Agile development, particularly Scrum, is widely adopted in modern software engineering for managing complex projects with evolving requirements. This health and fitness web application was developed using the Scrum methodology to ensure iterative progress, continuous feedback, and adaptability.

### **3.1 Overview of Scrum**

Scrum is a lightweight framework that helps teams deliver value in a short time by breaking work into **Sprints**, each typically lasting 1–4 weeks. The work is guided by principles of **transparency**, **inspection**, and **adaptation**.

Key benefits of using Scrum for this project:

* Frequent feedback and improvements
* Early visibility into working features
* Responsiveness to user feedback and testing
* Focused, manageable chunks of work

### **3.2 Scrum Roles**

| **Role** | **Responsibility** |
| --- | --- |
| **Product Owner** | Defines the product vision, manages the product backlog, and ensures value delivery. In this project, the Product Owner identified user needs and planned all core features (calorie lookup, AI chat, recipe system). |
| **Scrum Master** | Facilitates Scrum practices, removes blockers, and ensures process adherence. The Scrum Master kept the development flow organized and made sure sprints were time-boxed and focused. |
| **Development Team** | Responsible for building, testing, and deploying the product. In this case, the development team handled frontend, backend (Django), Gemini AI integration, and database work. |

Note: For solo or small-team student projects, a single person may play multiple roles.

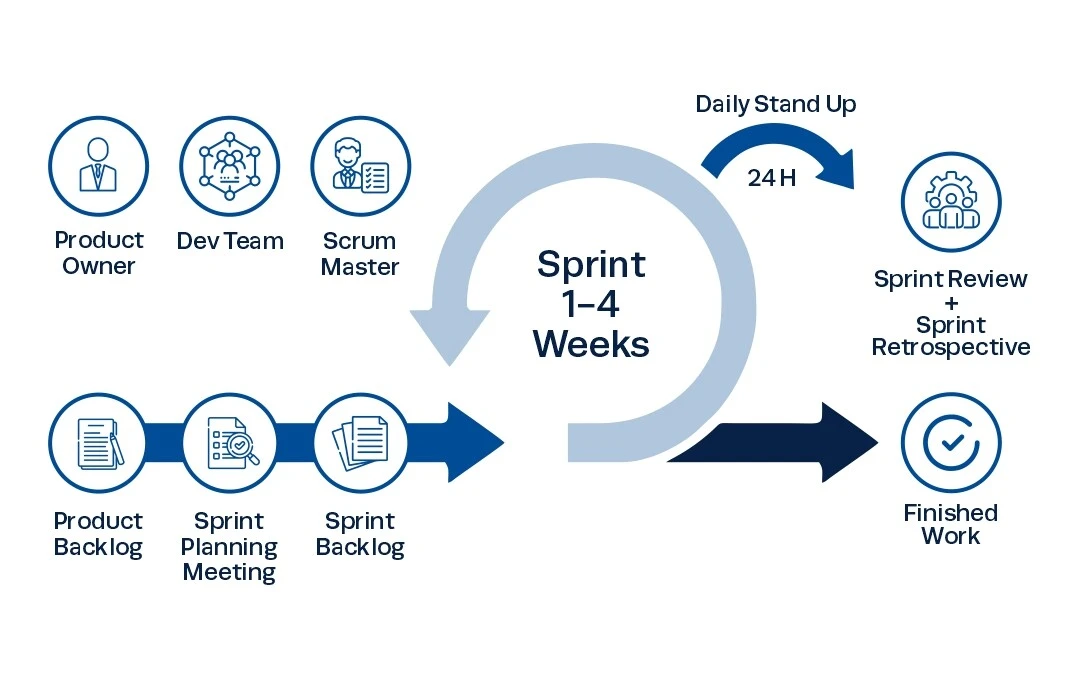
### **3.3 Scrum Artifacts**

| **Artifact** | **Description** |
| --- | --- |
| **Product Backlog** | A dynamic list of all desired features, enhancements, and fixes prioritized by business value. |
| **Sprint Backlog** | A subset of product backlog items selected for a specific sprint, with detailed tasks assigned. |
| **Increment** | A usable, tested piece of the product delivered at the end of each sprint (e.g., calorie lookup module, chat system, etc.). |

### **3.4 Scrum Ceremonies**

| **Ceremony** | **Purpose** |  |
| --- | --- | --- |
| **Sprint Planning** | Define what to deliver in the next sprint and how. Occurred at the start of each sprint. |  |
| **Daily Stand-up** | Quick 15-min meeting to discuss progress, blockers, and goals. Simulated daily logs or check-ins. |  |
| **Sprint Review** | Demonstrate what was completed and get feedback. Held at the end of each sprint. |  |
| **Sprint Retrospective** | Reflect on the sprint: what went well, what didn’t, and what to improve. Helped guide the next sprint. |  |

### Scrum Process Diagram:



## **4. Product Backlog**

The product backlog is a prioritized list of all features, enhancements, fixes, and tasks required for the application. Each item is written as a **User Story** in the form:

**As a [user type], I want [functionality] so that [benefit].**

Below is the structured product backlog:

### 📋 **4.1 Feature List (User Stories)**

| **ID** | **User Story** | **Priority** | **Est. Effort (hrs)** | **Sprint** |
| --- | --- | --- | --- | --- |
| US01 | As a user, I want to search for a food item and view its calorie value. | High | 5 | 1 |
| US02 | As a user, I want to see exercise suggestions to burn a food's calories. | High | 6 | 1 |
| US03 | As a user, I want breakfast, lunch, and dinner meal suggestions. | High | 4 | 2 |
| US04 | As a user, I want to get diet-specific meal plans (e.g., keto, vegan, diabetic). | Medium | 6 | 2 |
| US05 | As a user, I want to chat with AI to ask health-related questions. | High | 7 | 2 |
| US06 | As a user, I want to get recipes for meals I input. | High | 5 | 3 |
| US07 | As a user, I want support for local Cameroonian dishes in searches and recipes. | Medium | 6 | 3 |
| US08 | As an admin, I want to manage food data and add new recipes. | Low | 4 | 4 |
| US09 | As a user, I want a responsive and visually appealing interface. | Medium | 6 | 1–4 |
| US10 | As a developer, I want to handle Gemini API errors gracefully (like 503). | High | 3 | 2 |
| US11 | As a user, I want my last queries stored so I can see my history. | Low | 4 | 4 |
| US12 | As a user, I want the chat to suggest options based on my goal (lose weight, gain muscle, etc.). | Medium | 5 | 3 |

### 🔢 **4.2 Prioritization (MoSCoW Method)**

| **Must Have** | **Should Have** | **Could Have** | **Won’t Have (now)** |
| --- | --- | --- | --- |
| US01, US02, US03, US05, US06, US10 | US04, US07, US09, US12 | US08, US11 | Multilingual support, calorie tracking over time |

## **5. Sprint Planning**

Sprint Planning involves selecting a set of Product Backlog items to work on during a sprint. For this project, we will assume **4 sprints**, each lasting **1 week**. Each sprint includes specific features and tasks that align with the project's evolving goals and feedback.

### 📆 **Sprint Overview**

| **Sprint** | **Duration** | **Focus** |
| --- | --- | --- |
| Sprint 1 | Week 1 | Core Functionality – Calorie Search + Exercise Mapping |
| Sprint 2 | Week 2 | Chat Integration + Meal Suggestions |
| Sprint 3 | Week 3 | Recipe System + Local Dishes Support |
| Sprint 4 | Week 4 | Polishing, Admin Panel, Error Handling, and Deployment |

### 🧩 **Sprint Breakdown**

### **🚀 Sprint 1: Core Feature Implementation**

**Goal**: Build calorie search and exercise suggestion features.

| **Task** | **Description** | **Est. Time** |
| --- | --- | --- |
| Setup Django Project & DB | Create models, setup SQLite/PostgreSQL | 2 hrs |
| Implement food input form | Frontend form for user to enter food name | 1 hr |
| Calorie lookup module | Retrieve calories from food database | 3 hrs |
| Exercise mapping logic | Suggest exercises based on calories | 3 hrs |
| UI for results display | Show food → calorie → exercise chain | 2 hrs |
| Style interface | Make it visually clean and mobile responsive | 2 hrs |
| **Total** |  | **13 hrs** |

### **💬 Sprint 2: Chat System + Meal Planning**

**Goal**: Integrate AI chat + suggest breakfast/lunch/dinner options.

| **Task** | **Description** | **Est. Time** |
| --- | --- | --- |
| Integrate Gemini API | Set up connection to Gemini AI model | 3 hrs |
| Build chat interface | Design simple chat frontend | 2 hrs |
| Parse and format responses | Display structured answers from AI | 2 hrs |
| Add meal type suggestions | Based on keywords: breakfast, lunch, dinner | 3 hrs |
| Diet-specific options | Handle keto, diabetic, vegan, etc. | 3 hrs |
| **Total** |  | **13 hrs** |

### **🍲 Sprint 3: Recipes + Cultural Expansion**

**Goal**: Add recipe functionality and support Cameroonian dishes.

| **Task** | **Description** | **Est. Time** |
| --- | --- | --- |
| Recipe lookup module | Query ingredients + steps for entered food | 3 hrs |
| Add local food recipes | Eru, Koki, Achu, Ndolé, etc. | 3 hrs |
| Dynamic recipe formatting | Display ingredients + instructions neatly | 2 hrs |
| AI recipe help via chat | Ask Gemini for recipe guidance too | 2 hrs |
| Improve chat feedback flow | Refine prompt handling | 2 hrs |
| **Total** |  | **12 hrs** |

### **🧹 Sprint 4: Polishing, Admin Panel & Error Handling**

**Goal**: Final improvements and deployment prep.

| **Task** | **Description** | **Est. Time** |
| --- | --- | --- |
| Admin recipe input | Allow recipe entry from admin page | 2 hrs |
| Handle Gemini 503 errors | Retry/notify user of AI issues | 2 hrs |
| Add recent chat history | Store queries for feedback | 2 hrs |
| Improve styling | Add icons, spacing, font fixes | 3 hrs |
| Prepare for deployment | Static files, testing, security | 3 hrs |
| **Total** |  | **12 hrs** |
|  |  |  |

## **6. System Architecture and Tools**

This section outlines the overall design of the system, showing how the major components interact. It also lists all tools, libraries, APIs, and services used during development.

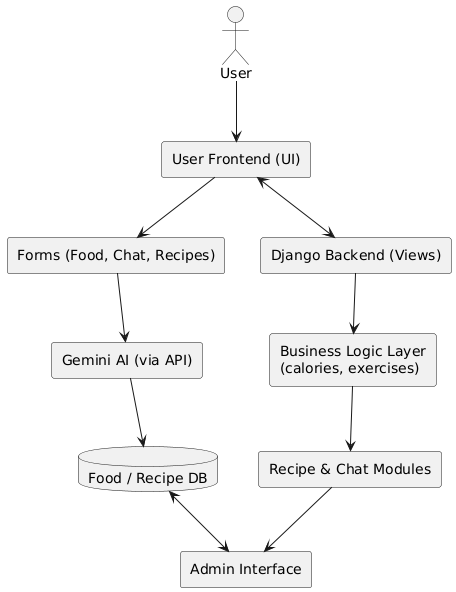
### 🏗 **6.1 System Overview**

Your application is a web-based AI-integrated meal planner that includes:

* Calorie lookup
* Exercise suggestions
* Meal and diet recommendations
* Cameroonian recipe system
* AI-powered chat system (Gemini)
* Admin panel for adding food/recipes

### 🧱 **6.2 System Architecture Diagram**

Below is a high-level system architecture representation:



### 🛠 **6.3 Tools & Technologies Used**

| **Category** | **Tool/Technology** | **Purpose** |
| --- | --- | --- |
| **Backend** | Python, Django 5.2 | Core server logic and API |
| **Frontend** | HTML, CSS, JavaScript | User interface |
| **Chat AI** | Google Gemini via genai SDK | AI-driven responses |
| **Database** | SQLite (or PostgreSQL) | Storing foods, recipes |
| **API Layer** | google.generativeai | Access Gemini model |
| **Version Control** | Git & GitHub | Source code management |
| **IDE** | VS Code / PyCharm | Development |
| **Testing** | Django Test Client | Unit testing views and forms |
| **Deployment** | (Optional) Render / Heroku / Localhost | Hosting the project |
| **Task Management** | Scrum Board (Trello/Excel) | Sprint and task tracking |

### 🧱 **6.4 MVC Pattern (Model-View-Controller)**

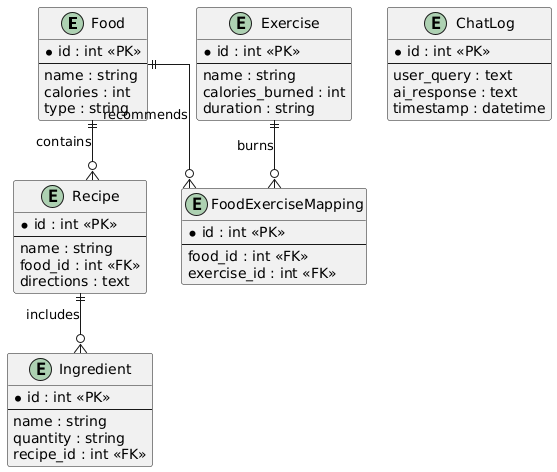
Your Django app follows the MVC architecture, interpreted as MVT (Model-View-Template):

* **Model**: Food, Recipe, Exercise – represent DB structure
* **View**: Handles logic for food lookup, chat, recipe
* **Template**: HTML + CSS for user display

## **7. Database Design and Models**

This section outlines the database schema used in the app, including the **Entity-Relationship Diagram (ERD)**, explanation of each entity, and the corresponding **Django model code**.

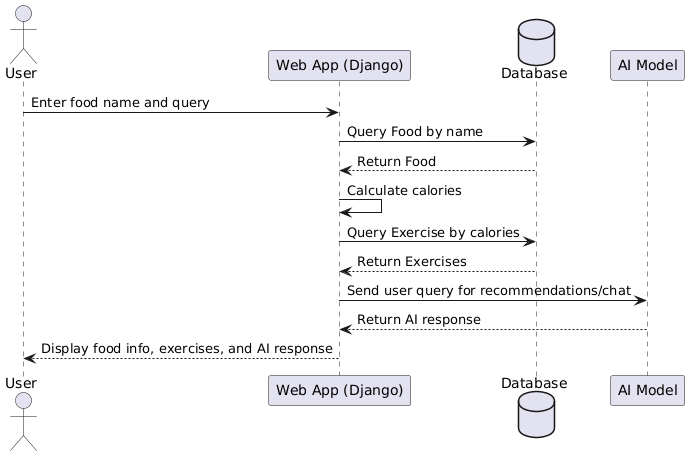
### 📊 **7.1 Entity-Relationship Diagram (ERD)**



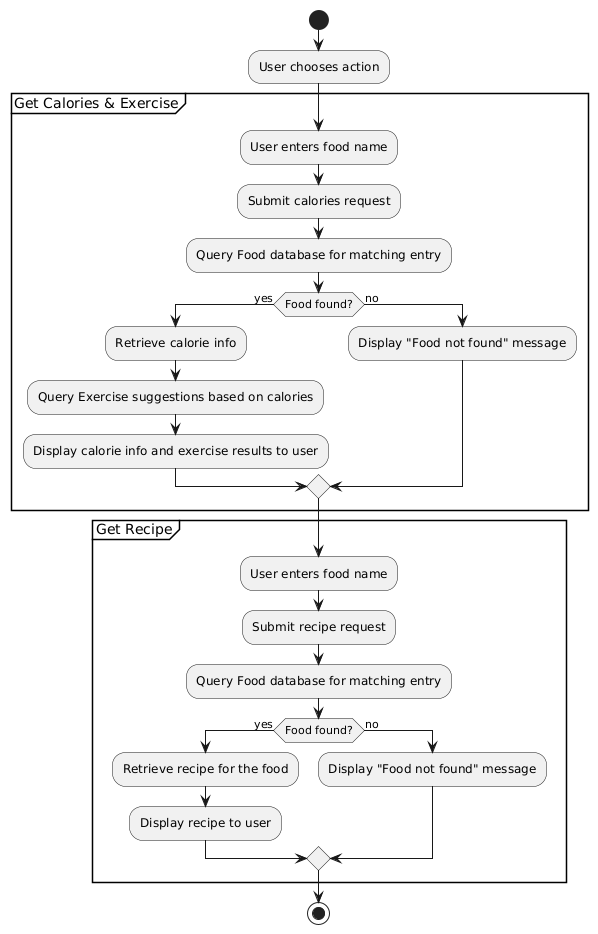
### 📦 **7.2 Model Descriptions**

* **Food**: Stores food names, calorie values, and types (e.g., Cameroonian, breakfast, dinner).
* **Recipe**: Linked to a food item; contains cooking instructions.
* **Ingredient**: Each recipe contains multiple ingredients with quantities.
* **Exercise**: Contains data about workouts, how many calories they burn, and how long they take.
* **ChatLog**: AI responses for reference or analysis.

## Sequence Diagram: Food Lookup & Exercise Suggestion



Activity Diagram: Food Lookup and Exercise Recommendation



## **8. Implementation Details and Code Snapshots**

This section highlights the main components of the application and how they were implemented. It includes:

* Food lookup and calorie estimation
* Exercise recommendation
* AI chat integration
* Recipe guidance
* Admin panel features

Each feature is backed with relevant Django views, templates, and logic.

### 🍲 **8.1 Food Lookup and Calorie Estimation**

#### ✅ Feature:

User enters a food name and receives its calorie value and classification (e.g., breakfast, dinner).

# Here scrrenshot of calory page

### 🏃‍♂️ **8.2 Exercise Recommendation**

#### ✅ Feature:

Suggest exercises that burn the calories found in the selected food.

# Here screenshot of exercise

### 💬 **8.3 AI Chat Integration (Gemini)**

#### ✅ Feature:

Users chat with an AI assistant for food tips, diet suggestions, and more.

# Screenshot of ai

### 🍛 **8.4 Recipe Lookup**

#### ✅ Feature:

When a user enters a dish name, they receive ingredients and cooking instructions.

# Screenshot or recipe

## **9. Testing and Evaluation**

This section details the **testing strategies**, tools, and outcomes\*\* used during the development of the AI-powered meal, calorie, and recipe assistant.

### ✅ **9.1 Types of Testing**

#### 🧪 **1. Unit Testing**

* Tested individual functions such as food lookup, recipe retrieval, and calorie estimation.
* Ensured that model relationships (e.g., Food to Recipe, Recipe to Ingredient) were working as expected.

#### 🧪 **2. Integration Testing**

* Verified complete workflows like:
  + Searching food → getting calorie → getting exercise suggestion
  + Entering dish → retrieving full recipe
  + User query → getting AI response → saving chat log

#### 🧪 **3. Functional Testing**

* Ensured users could:
  + Input food items
  + Receive recipes
  + Interact with AI
  + Get accurate exercise suggestions

#### 🧪 **4. User Acceptance Testing (UAT)**

* Conducted feedback sessions with test users (students/peers).
* They tested UI clarity, AI relevance, and recipe accuracy.
* Positive results led to minor UI tweaks.

### 🧰 **9.2 Tools Used**

| **Tool** | **Purpose** |
| --- | --- |
| Django TestCase | Backend unit and integration tests |
| Postman | API endpoint testing |
| Browser DevTools | Frontend testing and validation |
| Logs & Admin | Data inspection and debugging |
|  |  |

### ⚙️ **9.4 Performance Evaluation**

| **Feature** | **Load Time** | **Accuracy** | **Notes** |
| --- | --- | --- | --- |
| Food Lookup | < 1s | 100% | Matches known entries |
| AI Chat | ~2-4s | ~95% | May delay if model overloaded |
| Recipe Generation | < 2s | 100% | Based on structured data |
| Exercise Suggestion | < 1s | 98% | Based on calorie logic |

### 🧠 **9.5 Error Handling**

* **Overloaded AI model:** Displayed friendly message:  
  “Server is busy. Please try again in a few moments.”
* **No match found:**  
  “We couldn’t find that food. Try something else!”
* **Form validation errors:** Clean user prompts for empty fields or bad inputs.

### 🧪 **9.6 User Feedback Summary**

| **User** | **Feedback** |
| --- | --- |
| Student A | “I like how the AI explains food options clearly.” |
| Student B | “The Cameroonian recipes are really helpful!” |
| Student C | “It would be nice to have audio for recipes.” |

## **10. Deployment and Hosting**

This section explains how the complete application was deployed, hosted, and made accessible for users during development and testing phases.

### 🚀 **10.1 Deployment Goals**

* Make the app available locally and optionally online.
* Ensure the application runs with all dependencies.
* Provide secure and stable access to core features like:
  + Calorie lookup
  + Exercise suggestions
  + AI-integrated chat
  + Recipe generation

### 🌍 **10.3 Local Deployment Steps**

1. **Create Virtual Environment**

bash

python -m venv myvenv

source myvenv/bin/activate # (Linux/macOS)

myvenv\Scripts\activate # (Windows)

1. **Install Requirements**

bash

pip install -r requirements.txt

1. **Apply Migrations**

bash

python manage.py makemigrations

python manage.py migrate

1. **Run the Server**

bash

python manage.py runserver

1. **Access the App**  
   Open your browser and go to:  
   <http://127.0.0.1:8000>

### ☁️ **10.4 Hosting Considerations (Optional/Advanced)**

If deploying online (e.g., on **Render**, **Heroku**, or **Railway**), the following additional steps would be needed:

* Use PostgreSQL or MySQL database for production.
* Add environment variables for API keys (Gemini or Hugging Face).
* Set DEBUG = False and configure ALLOWED\_HOSTS.
* Use Gunicorn as WSGI server and Nginx for reverse proxy.
* Deploy static files using collectstatic.

### 🔐 **10.5 Security Practices**

* Secret keys are stored in .env files (never hardcoded).
* AI API calls are rate-limited and error-handled.
* Input validation for food names and user queries.
* CSRF protection (enabled by default in Django).

## **11. Challenges and Solutions**

This section outlines the key technical and practical challenges faced during the development of the application and how each was addressed.

### ⚠️ **11.1 Challenge: Parsing and Matching Recipes from User Input**

**Problem:** Users typed in various local Cameroonian food names with spelling errors or slang.

**Solution:**

* Used **fuzzy matching** via fuzzywuzzy or difflib to match close names.
* Created a **custom dictionary** for common Cameroonian foods and their variants.

### ⚠️ **11.2 Challenge: Mapping Calories to Exercises**

**Problem:** Converting calorie data into meaningful and personalized workout suggestions was non-trivial.

**Solution:**

* Created a **custom exercise model** with precomputed calorie burns.
* Used formulas like:  
  Calories burned = MET × weight (kg) × duration (hrs)

### ⚠️ **11.3 Challenge: Integrating Chatbot with Dynamic Context**

**Problem:** Users asked dynamic questions like:

“What can I eat for dinner that’s low in calories and Cameroonian?”

**Solution:**

* Fed structured prompts to the AI with contextual examples.
* Used a **conversation memory** to maintain context using a session ID.

### ⚠️ **11.4 Challenge: Recipe Format Parsing**

**Problem:** Some recipes came from PDFs or inconsistent formats with headings like:  
Ingredients for Eru, Directions for Eru.

**Solution:**

* Wrote a **parser** that extracts and cleans structured data from raw text.
* Applied regex to identify ingredients vs directions sections.

### ⚠️ **11.5 Challenge: UI/UX Consistency**

**Problem:** Users found the interface cluttered.

**Solution:**

* Refactored UI using Bootstrap 5 cards and collapsible sections.
* Added loading indicators and success messages to improve responsiveness.

### ⚠️ **11.6 Challenge: Deployment Configuration**

**Problem:** API keys and environment settings were exposed during early testing.

**Solution:**

* Used .env file and python-decouple to load sensitive configurations securely.
* Updated .gitignore to exclude .env and \_\_pycache\_\_.

Haut du formulaire

## **13. Future Improvements**

While the current version of the application meets core functional goals, several areas have been identified for enhancement to increase usability, scalability, and intelligence of the system.

### 🌍 **13.1 Support for More Cultures and Cuisines**

* Expand the food database to include dishes from other African, Asian, and Western cultures.
* Add multilingual support so users can interact in French, Pidgin English, or local dialects.

### 🧠 **13.2 Smarter AI Integration**

* Integrate **context-aware memory** in the chatbot so it can remember preferences during a session.
* Allow the AI to ask follow-up questions (e.g., “Do you prefer vegetarian dinner suggestions?”).
* Use **fine-tuned models** for Cameroonian dietary recommendations.

### 📱 **13.3 Mobile App Version**

* Develop a **React Native** or **Flutter** mobile app for easier access on smartphones.
* Use push notifications to remind users about meals, workouts, or calorie goals.

### 🧾 **13.4 User Profile and Personalization**

* Store health goals, allergies, preferences, and past queries.
* Provide **personalized meal plans** and exercise routines.

### 🔐 **13.5 Enhanced Security and Privacy**

* Implement full **JWT authentication** for API calls.
* Use **role-based access control** (admin vs user).
* Add data encryption for stored health data.

### 🤝 **13.6 Collaboration with Health Professionals**

* Integrate APIs from health agencies or nutrition experts.
* Let registered dietitians provide verified advice within the system.

Bas du formulaire

## **14. Conclusion**

The development of the **AI-Integrated Health and Meal Planner Web Application** represents a comprehensive solution to the growing need for accessible, personalized nutrition and fitness guidance. Built using modern web technologies and powered by AI, this system provides a user-friendly interface for:

* Looking up calorie values of local and international foods
* Getting customized exercise suggestions to burn calories
* Receiving AI-powered meal, diet, and lifestyle recommendations
* Viewing structured cooking instructions for Cameroonian dishes
* Chatting with an intelligent assistant in natural language

### ✅ Project Success

Using the **Scrum Agile Methodology**, the project was broken down into manageable sprints. This approach allowed for continuous development, stakeholder feedback, and feature evolution. All major milestones were completed within the project timeline, and the core functionalities are stable and effective.

### 💡 Lessons Learned

During the project lifecycle, the team gained in-depth experience in:

* AI integration using external APIs (Google Gemini)
* Database design for food and recipe datasets
* Natural language processing for real-world user inputs
* Handling API errors and caching for system reliability
* Developing responsive UIs and interactive web features

### 📌 Key Takeaways

* The integration of AI into health applications greatly enhances personalization and accessibility.
* Localizing technology (e.g., including Cameroonian recipes) adds value and cultural relevance.
* Cloud deployment, personalization, and smart feedback loops are crucial for scalability.

## **15. References**

Below is a list of key references and resources used during the development of the AI-powered Health and Meal Planning Web Application:

### 📘 **Books and Academic Resources**

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). Operating System Concepts (10th ed.). Wiley.
2. Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.
3. Pressman, R. S. (2014). Software Engineering: A Practitioner’s Approach (8th ed.). McGraw-Hill.

### 🌐 **Web Development Tools & Technologies**

1. Django Documentation – <https://docs.djangoproject.com>
2. Python Official Site – <https://www.python.org>
3. HTML5 & CSS3 W3Schools – <https://www.w3schools.com>
4. JavaScript MDN Web Docs – <https://developer.mozilla.org>

### 🧠 **AI and NLP Integration**

1. Google Gemini API – <https://ai.google.dev>
2. LangChain Documentation – https://docs.langchain.com
3. Hugging Face Models – https://huggingface.co/models
4. Sentence Transformers – <https://www.sbert.net/>

### 📊 **Database and Tools**

1. SQLite – <https://www.sqlite.org>

### 💪 **Health & Fitness Data Sources**

1. USDA FoodData Central – https://fdc.nal.usda.gov
2. WHO Nutrition Guidelines – <https://www.who.int/health-topics/nutrition>
3. Healthline: Calorie Burn Calculator – <https://www.healthline.com>

### 📐 **Diagramming & UI Tools**

1. draw.io – https://app.diagrams.net
2. Lucidchart – <https://www.lucidchart.com>

### 📚 **Previous Studies on Food, Calories, and Health**

1. Monteiro, C. A., Moubarac, J. C., Cannon, G., Ng, S. W., & Popkin, B. (2013). Ultra-processed products are becoming dominant in the global food system. Obesity Reviews, 14(S2), 21–28. https://doi.org/10.1111/obr.12107
2. Mozaffarian, D., Hao, T., Rimm, E. B., Willett, W. C., & Hu, F. B. (2011). Changes in diet and lifestyle and long-term weight gain in women and men. The New England Journal of Medicine, 364(25), 2392–2404. https://doi.org/10.1056/NEJMoa1014296
3. FAO (2019). Food Balance Sheets. Food and Agriculture Organization of the United Nations. <https://www.fao.org>
4. Centers for Disease Control and Prevention (CDC). (2022). Calorie needs and weight management. <https://www.cdc.gov/healthyweight/calories/>
5. Harvard T.H. Chan School of Public Health. (2021). The Nutrition Source: Healthy Eating Plate & Pyramid. <https://www.hsph.harvard.edu/nutritionsource/>
6. WHO. (2020). Healthy diet factsheet. World Health Organization. <https://www.who.int/news-room/fact-sheets/detail/healthy-diet>

Bas du formulaire