**COE-MAJOR-PROJECT**

**AI -Powered Fitness Assistant and Adaptive Meal Planner**

**COE-AIDS**

A logo of a college of engineering

AI-generated content may be incorrect.

**Project Title:** AI-Powered Fitness Assistant and Adaptive Meal Planner  
**submitted by:**

K Hemanth Kumar-22B81A6717

N. Raj Kumar-22B81A6736  
**Institution:** CVR College of Engineering  
**Date:**6th August 2025

**Table of Contents**

1. **Introduction**

1.1. Project Overview

1.2. Purpose and Scope

1.3. Target Audience

1. **Key Features**
2. **System Architecture and Technology**

3.1. System Architecture

3.2. Technology Stack

1. **User Guide**
2. **Local Setup and Installation**

5.1. Prerequisites

5.2. Installation Steps

1. **Conclusion and Future Work**

6.1. Conclusion

6.2. Future Work

**1.Introduction:**

**1.1. Project Overview**

The AI-Powered Fitness & Adaptive Meal Assistant is a smart, two-in-one health companion designed to deliver personalised fitness advice and meal plans. It uses advanced AI technologies like large language models (LLMs), semantic search, and retrieval-augmented generation (RAG) to give users guidance that's not only intelligent but also grounded in real data.

The system includes:

* A Fitness Assistant Bot that answers workout-related questions based on your organisation’s fitness materials.
* An Adaptive Meal Planner that analyses users’ medical health reports and creates tailored diet plans to match their health needs.

**1.2. Purpose and Scope**

This project aims to close the gap between generic advice and truly personalised health support. It brings AI into the fitness and nutrition space in a meaningful way, allowing users to receive recommendations that are actually relevant to their own health conditions or training goals.

**The scope of the project includes:**

* Uploading medical reports or fitness PDFs
* Ask questions or get personalised advice through a chatbot
* Receive health-conscious meal and workout plans
* Interact through an easy-to-use web interface

**1.3. Target Audience**

The web application is primarily intended for:

* **Patients managing chronic conditions** seeking adaptive diet plans grounded in medical evidence .
* **Personal trainers and fitness coaches** looking to automate personalised workout recommendations aligned with client specific data.
* **Nutritionists and dieticians** who wish to streamline cline consultations using AI-generated meal strategies based on health reports.
* **Tech-forward wellness startups** aiming to integrate smart assistants into their platforms using LLM’s, RAG Pipelines, and semantic search systems.

**2. Key Features:**

**Smart AI-Powered Search & Generation**

Combines the power of semantic search with generative AI, so users get responses that are not just accurate but also meaningful and tailored to their needs.

**Fitness and Medical PDF Analysis**

The system reads and understands content from uploaded documents—like training manuals or medical reports—and converts them into data the AI can use.

**User-Friendly Web Apps**

The Fitness Assistant runs through a simple Flask web app that looks and feels like a chatbot.

The Meal Planner uses Streamlit for a clean, interactive experience that guides users step-by-step.

**Personalised Fitness Plans**

Based on your uploaded fitness material, the assistant can suggest custom workouts, rep counts, training styles, and more—adapted to your specific goals.

**Adaptive Meal Planning**

After reading your health reports, the assistant recommends meals that are safe, healthy, and personalised—especially for people dealing with conditions like diabetes, obesity, high blood pressure, or nutrient deficiencies.

**3. System Architecture and Technology:**

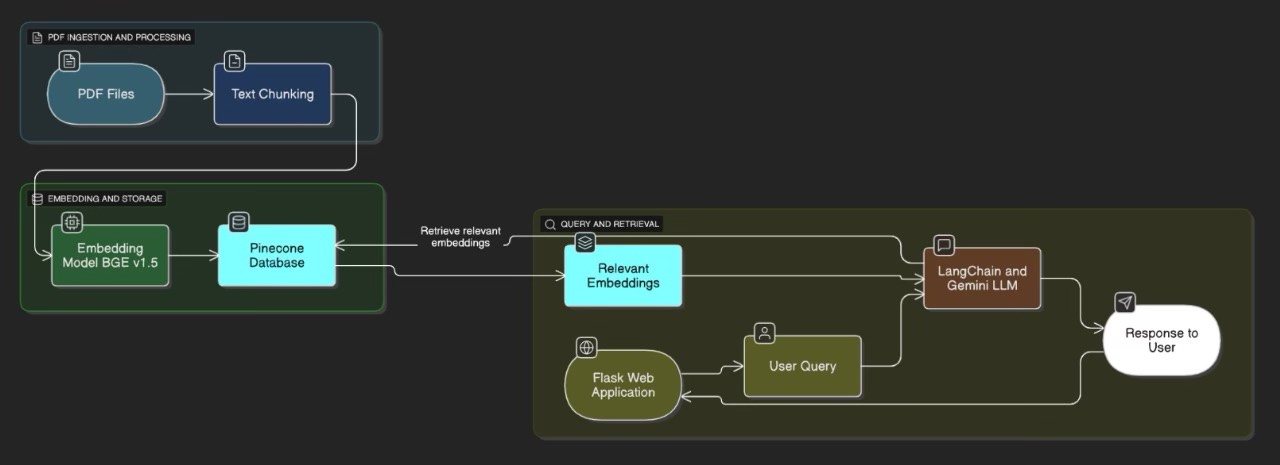
**3.1. System Architecture**

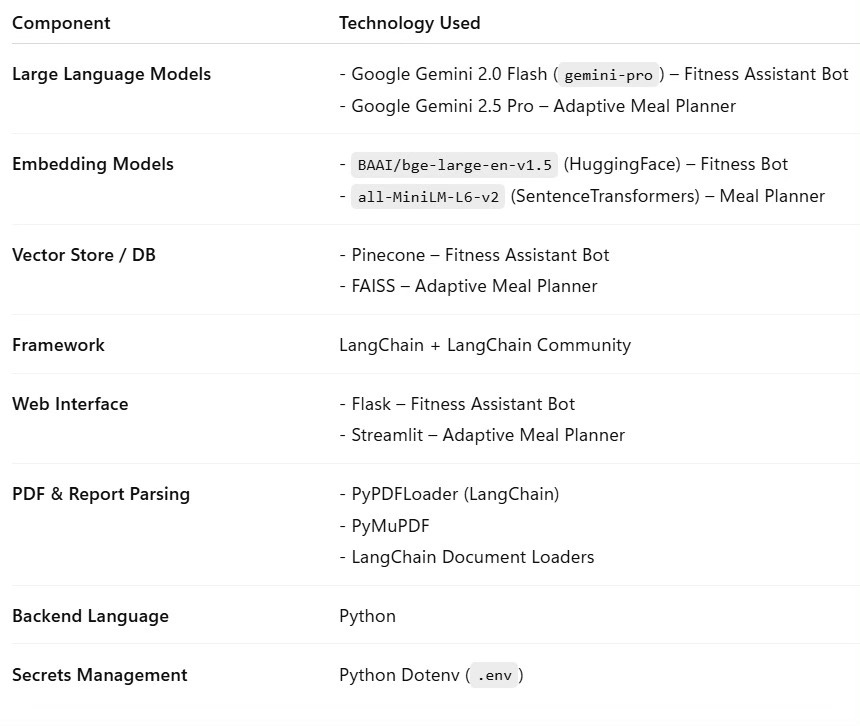
The application is built on a client-server model and utilises a Retrieval-Augmented Generation (RAG) pipeline to provide answers.

**The workflow is as follows:**

1. **Frontend Interaction**: The user interacts with the web interface built with HTML, CSS, and JavaScript. They use the interface to upload PDFs and submit questions.
2. **Backend Request Handling**: A Flask server on the backend receives these requests.
3. **Document Processing (RAG Pipeline)**:
   * **Loading**: When a PDF is uploaded, the Flask server uses PyPDFLoader from the LangChain library to load its content.
   * **Chunking**: The extracted text is split into smaller, manageable chunks using RecursiveCharacterTextSplitter.
   * **Embedding**: Each chunk of text is converted into a numerical vector representation (an embedding) using Google's gemini-embedding-001 model.
   * **Indexing**: These vectors are stored in a FAISS (Facebook AI Similarity Search) vector store, creating an efficient, searchable index of the document content. This index is stored in memory for the user's session.
4. **Query and Retrieval**: When a user asks a question, it is also converted into an embedding. The FAISS index finds the text chunks with similar embeddings
5. **Answer Generation**: The retrieved text chunks and the user's original question are passed to the Google Gemini LLM. The model is instructed to formulate a final answer based only on the provided context.

**3.2.System Architecture Diagram:**



**3.2. Technology Stack**

**4. User Guide:**

**Fitness Assistant:**

* Users visit the chatbot via a browser interface.
* They can type queries like “What’s a good workout plan for fat loss?” or “Explain HIIT vs LISS”
* The bot will fetch and summarise relevant info from the uploaded fitness documents using Gemini-powered responses.

**Meal Planner :**

* Users upload their medical health reports and ask questions like “Suggest a meal plan for type-2 diabetes” or “How should I eat if I have high blood pressure?”.
* The assistant reads and understands the report, retrieves relevant content, and generates a tailored meal plan in real time.

**5. Local Setup and Installation:**

**5.1. Prerequisites**

* Python 3.9 or higher
* Git version control
* A Google AI API Key

**5.2. Installation Steps**

1. **Clone the Repository**:

git clone <https://github.com/hemanth18624/adaptive-meal-planning>

git clone https://github.com/rajkumar9030/Fitness\_Assistant\_RAG

1. **Create and Activate a Virtual Environment**:
2. **Install Dependencies**:
3. **Set Up Environment Variables**: Create a file named .env in the project root and add your Google API key:
4. **Run the Application**:

**6. Conclusion and Future Work:**

**6.1. Conclusion**

This project represents a powerful integration of AI-driven technologies to solve real-world problems in health, nutrition, and fitness. By leveraging LangChain-based Retrieval-Augmented Generation (RAG) pipelines, state-of-the-art embedding models like BGE v1.5 and MiniLM, and Google’s Gemini LLM, we’ve built two intelligent systems. A Fitness Assistant Bot that provides contextual, document-aware fitness guidance from domain-specific PDFs. An Adaptive Meal Planner Assistant that reads and interprets medical reports to generate personalised meal plans. Both systems demonstrate the seamless orchestration of document processing, semantic search, vector databases (Pinecone & FAISS), and conversational AI, delivering deeply personalised and domain-grounded user experiences. This project not only reflects advanced skills in LLM integration but also emphasises responsible AI usage in the healthcare space.

**6.2. Future Work**

The project has a strong foundation with several avenues for future enhancement:

**Multilingual Support:** Enable cross-language understanding to support a wider user base.

**Meal Plan Export & Integration:** Allow users to export their adaptive plans as PDFs or sync with calendar apps.

**User Profile Learning:** Build persistent user history to adapt recommendations over time.

**Real-time Nutrition APIs:** Integrate food databases for up-to-date caloric and macro information.

**Mobile-Optimised Frontend:** Develop a companion mobile app for on-the-go access to health insights.

**Data Privacy Enhancements:** Further strengthen HIPAA-compliant practices for handling sensitive health data.