Contents : A. About the Project

B. About the Codes

A. About the Project

AI Health Assistant Project

Background

Healthcare systems around the world face a huge challenge: many people cannot easily access medical advice when they need it. In underserved areas, patients often struggle to find doctors or wait long hours for basic consultations. Even in developed regions, hospital systems are overwhelmed, and patients want faster, more personalized care. Artificial Intelligence (AI) offers an opportunity to bridge this gap. By analyzing health information, AI can guide patients with preliminary advice, monitor risks, and help doctors make quicker, data-driven decisions.

Problem Statement

- Many people lack immediate access to professional healthcare support.
- Patients often have simple health questions but no affordable or convenient way to get reliable answers.
- Doctors and nurses are overwhelmed, leading to long wait times and burnout.

This project set out to design a **smart AI Health Assistant** that can provide **basic health guidance** to patients while ensuring doctors have better tools for decision-making.

Objectives

1. **Assist Patients:** Provide simple, AI-driven answers to common health queries.
2. **Risk Detection:** Identify potential warning signs (e.g., symptoms that may require urgent care).
3. **Support Doctors:** Offer insights that help healthcare professionals save time.
4. **Accessibility:** Build a system that works for underserved and rural populations with limited healthcare access.

◆ Methodology (Simplified)

1. **Data:** The assistant uses general medical knowledge and symptom datasets (no personal patient data).
2. **AI Models:** Natural Language Processing (NLP) techniques were applied so the system can understand human health-related questions.
3. **System Design:** A user-friendly chatbot interface was created where patients can type symptoms or questions and receive advice.
4. **Testing:** The assistant was tested with common health scenarios to measure accuracy and helpfulness.

◆ Results

- The AI Assistant could successfully answer most common health-related questions in **clear, everyday language**.
- It flagged high-risk symptoms (e.g., chest pain, difficulty breathing) and advised users to seek professional care immediately.
- Users found it **easy to interact with** and felt reassured that they could get initial guidance without waiting hours.

◆ Conclusion

The **AI Health Assistant** is not a replacement for doctors but a **first line of support**. It helps patients by:

- Providing **fast and reliable health guidance**,
- Reducing **unnecessary hospital visits**, and
- Supporting doctors with **better-prepared patients**.

In the future, this tool can be integrated with telehealth platforms to directly connect patients with doctors, making healthcare more **affordable, accessible, and efficient**.

B. ABOUT THE CODES

Product Recommendation System

A **Product Recommendation System** that suggests products using:

- **Collaborative Filtering (CF)** – based on user-user similarity
- **Content-Based Filtering (CBF)** – based on product attributes (TF-IDF)
- **Hybrid Approach** – combines CF & CBF for better accuracy

Features

- 🔮 **Single User Recommendations** – Enter a User ID and get product suggestions.
- 📁 **Batch Recommendations** – Upload a CSV of user IDs and generate recommendations for each.
- 📊 **Visualization** – See which products are recommended most frequently.
- 🧩 **Hybrid Recommendation Engine** – Blends CF and CBF for improved predictions.
- 🎨 **Streamlit Web App** – Simple, interactive UI.

Tech Stack

- **Python** 🐍
- **Pandas / NumPy** – Data handling
- **Scikit-learn** – Cosine similarity, TF-IDF vectorization
- **Seaborn / Matplotlib** – Visualization
- **Streamlit** – Web app frontend

Project Structure

```
Product-Recommendation-System/
│
├──── product_recs.csv        # Dataset (user_id, product_id, product_name, category, rating)
├──── app.py                # Streamlit web app
├──── model_test.py           # Local testing of CF, CB, and Hybrid models
├──── requirements.txt         # Dependencies
└──── README.md              # Documentation
```

Installation

1. **Clone this repo:**

```
git clone https://github.com/<your-username>/Product-Recommendation-System.git
cd Product-Recommendation-System
```

2. **Install dependencies:**

```
pip install -r requirements.txt
```

3. **Run the app**

```
streamlit run app.py
```