Health AI Assistant

1. Introduction

Project title: Health AI Assistant
Team Leader: A. Abinaya
Team member: S. Alagumeenal
Team member: P. Anusiya
Team member: T. Atchaya
Team member: P. Bhuvaneswari

2. Project Overview

Purpose:
The purpose of the Health AI Assistant is to provide intelligent, personalized, and accessible healthcare support. By leveraging AI and real-time medical data, the assistant helps patients manage their health, track vital signs, and receive tailored health recommendations. For doctors and healthcare providers, it acts as a clinical decision support system—offering medical insights, patient data summaries, and anomaly detection to improve diagnosis and treatment efficiency. Ultimately, the assistant bridges technology, healthcare professionals, and patients to create safer, more connected, and efficient healthcare systems.

Features:
Conversational Interface
Key Point: Natural language interaction
Functionality: Allows patients and doctors to ask health-related questions, track symptoms, and receive AI-driven recommendations in plain language
Medical Report Summarization
Key Point: Simplified clinical understanding
Functionality: Converts lengthy lab reports and medical histories into concise, patient-friendly summaries
Health Forecasting
Key Point: Predictive analytics
Functionality: Predicts possible health risks (e.g., diabetes, heart conditions) using historical and wearable data

Personalized Health Tips
Key Point: Preventive care guidance
Functionality: Recommends diet, exercise, and lifestyle modifications based on patient profile and medical history

Patient Feedback Loop
Key Point: Continuous improvement
Functionality: Collects patient feedback on treatments and recommendations to refine healthcare services

Anomaly Detection
Key Point: Early warning system
Functionality: Identifies unusual patterns in vitals or lab results to flag potential health risks early

Multimodal Input Support
Key Point: Flexible health data handling
Functionality: Accepts text, PDFs (reports), and device data (wearables, IoT health sensors)

Streamlit or Gradio UI
Key Point: User-friendly interface
Functionality: Provides an intuitive dashboard for both patients and doctors to interact with the assistant

3. Architecture

Frontend (Streamlit):
Interactive web UI with dashboards for patient health tracking, file uploads (lab reports), chat interface, and doctor–patient collaboration features.

Backend (FastAPI):
Serves as the backend REST framework powering API endpoints for medical data processing, chat interactions, health-tip generation, and anomaly detection.

LLM Integration (Medical AI Models):
Fine-tuned LLMs for healthcare (HIPAA-compliant) used for summarization, recommendations, and patient-friendly explanations.

Vector Search (Pinecone or FAISS):
Medical records and reports are embedded for semantic search, enabling natural language queries on patient history.

ML Modules (Forecasting & Anomaly Detection):
Predictive models trained on healthcare datasets for risk prediction and anomaly detection in vitals/lab values.

4. Setup Instructions

Prerequisites:
Python 3.9+
pip and virtual environment tools
API keys for LLM (Watsonx, OpenAI, or Hugging Face) and vector DB
Secure cloud storage (HIPAA/GDPR compliance)
Installation Process:
Clone repository
Install dependencies (requirements.txt)
Configure .env with credentials
Run backend with FastAPI
Launch frontend with Streamlit
Upload reports / connect wearable devices .

5. Folder Structure

app/ – FastAPI backend logic
API routes for chat, health reports, feedback, and embeddings
Streamlit frontend with patient/doctor dashboards
health_forecaster.py – Predicts potential risks from vitals and history
report_summarizer.py – Generates simplified summaries of medical documents
anomaly_checker.py – Detects abnormal values in uploaded health data
feedback_handler.py – Stores patient feedback
dashboard.py – Main Streamlit entry script

6. Running the Application

Start FastAPI server
Run Streamlit dashboard
Navigate via sidebar (Chat, Reports, Health Tips, Forecasting)
Upload reports or connect wearable devices
Interact with AI for real-time health insights

7. API Documentation

POST /chat/ask – Patient/doctor queries → AI response
POST /upload-report – Uploads lab reports/records for summarization
GET /search-records – Retrieves patient history insights
GET /get-health-tips – Provides personalized health recommendations
POST /submit-feedback – Stores patient feedback

8. Authentication

JWT tokens or OAuth2 for secure logins
Role-based access: patient, doctor, admin
HIPAA/GDPR compliance for medical data handling

9. User Interface

Sidebar navigation
Patient health dashboards (vitals, history, trends)
Doctor dashboards (summarized patient reports, alerts)
Real-time AI chat for Q&A
Report download (PDF)

10. Testing

Unit Testing: For NLP & ML modules
API Testing: Swagger UI & Postman
Manual Testing: Upload reports, connect wearables, validate outputs
Edge Cases: Incomplete records, abnormal values.