HEALTH AI: INTELLIGENT HEALTHCARE ASSISTANT

Project Documentation

1.Introduction

- HealthAI:Intelligent healthcare assistant
- Team Member : Swathi K
- Team Member : Vidhya Sri H
- Team Member : Vinisha D
- Team Member : Yuvasri S

2.Project overview

- Purpose

The purpose of a Health AI: Intelligent Healthcare Assistant is to empower healthcare systems and individuals to thrive in a more proactive, personalized, and data-driven medical environment. By harnessing AI and real-time health data, the assistant helps optimize critical resources such as clinical workflows, diagnostics, and patient monitoring, while also promoting healthier behaviors through tailored guidance and support. For medical professionals, it acts as a strategic partner—offering predictive insights, summarizing complex medical literature, and streamlining decision-making to enhance patient outcomes. Ultimately, this assistant bridges technology, care delivery, and patient engagement to foster healthcare ecosystems that are more efficient, equitable, and resilient.

- Feature

Conversational Interface

Key Point: Natural language interaction

Functionality: Enables patients, caregivers, and healthcare professionals to ask health-related questions, receive medical updates, and get personalized guidance—all in plain, accessible language.

Medical Summarization

Key Point: Simplified clinical understanding

Functionality: Transforms complex medical research, treatment protocols, and health policies into concise, actionable summaries for both professionals and patients.

Health Prediction

Key Point: Predictive analytics

Functionality: Uses historical health records and real-time biometric data to anticipate potential health risks, disease progression, and resource needs—supporting early intervention and better care planning.

Patient Feedback Loop

Key Point: Community-driven care improvement

Functionality: Collects and analyzes patient input on healthcare experiences, treatment satisfaction, and service accessibility to inform better clinical practices and system design.

Anomaly Detection

Key Point: Early warning system

Functionality: Detects unusual patterns in patient vitals, lab results, or wearable device data to flag potential health risks—such as sudden spikes in blood pressure, irregular heart rhythms, or medication non-adherence.

Multimodal Input Support

Key Point: Flexible data handling

Functionality: Accepts diverse formats like clinical notes (text), lab reports (PDFs), and patient records (CSVs) for seamless analysis, summarization, and predictive modeling.

Symptom Checker

Key Point: First-line triage

Functionality: Lets users input symptoms in natural language and receive possible causes, urgency level, and next steps—based on medical databases.

Health Education

Key Point: Empowered self-care

Functionality: Provides bite-sized, easy-to-understand articles and videos on topics like nutrition, chronic disease management, and preventive care.

Streamlit or Gradio UI

Key Point: User-friendly interface

Functionality: Offers an intuitive dashboard for both patients and healthcare providers to interact with the assistant—viewing health trends, receiving alerts, and accessing personalized recommendations in real time.

3.Architecture

Frontend:

The frontend of HealthAI is developed using Streamlit, offering a clean and interactive web interface tailored for healthcare users. It includes multiple modules such as health dashboards, symptom checkers, lab report uploads, chat-based medical guidance, feedback forms, and appointment tracking. Navigation is streamlined through a sidebar using the streamlit-option-menu library, and each page is modularized to support scalability and future feature expansion.

Backend:

FastAPI serves as the backend framework, powering RESTful endpoints for document analysis, conversational health support, wellness tip generation, report creation, and feedback processing. Its asynchronous capabilities ensure responsive performance, and Swagger integration simplifies API documentation and testing for developers and healthcare partners.

LLM Integration (IBM Granite):

Natural language understanding and generation are handled by IBM Granite LLMs. These models are used to interpret patient queries, summarize complex medical literature, generate personalized health insights, and support empathetic, context-aware conversations. Prompt engineering ensures outputs are medically relevant and user-friendly

Vector Search:

HealthAI uses Pinecone to store and search embedded medical documents such as discharge summaries, prescriptions, and diagnostic reports. Sentence Transformers convert these documents into vector representations, enabling semantic search so users can retrieve relevant information using natural language queries—ideal for both patients and clinicians seeking quick access to critical data.

4. Setup Instructions

Prerequisites:

- Python 3.9 or later
- pip and virtual environment tools (venv, virtualenv, or similar)
- API keys for relevant healthcare or AI services (if applicable, e.g., OpenAI, IBM Watson, etc.)
- Internet access for accessing cloud-based features or APIs

Installation Process

- Clone the repository
- Install dependencies from requirements.txt
- Create a .env file and configure credentials
- Run the backend server using Fast API
- Launch the frontend via Stream lit
- Upload data and interact with the modules

5.Folder Structure

app/ – Contains all FastAPI backend logic, including routers, models, services, and utility functions.

app/api/ – Modular API routes for features like symptom checking, diagnosis, chat, and feedback.

ui/ – Streamlit-based frontend with interactive pages and reusable UI components.

health_dashboard.py – Entry point to launch the Streamlit dashboard interface.

health_llm.py – Manages interactions with the language model for medical Q&A and symptom analysis.

risk_predictor.py – Uses patient data and ML models to predict potential health risks.

report_generator.py – Generates personalized health reports and summaries using AI.

6.Running The Application

To start the project:

➢ Launch the FastAPI server to expose backend endpoints for health data processing, chat, predictions, and report generation.

➢ Run the Streamlit dashboard to access the healthcare assistant web interface.

➢ Navigate through the interface using the sidebar menu to access different features.

➢ Upload patient records or CSV files, interact with the AI health assistant, and explore outputs like health summaries, risk predictions, and generated reports.

➢ All interactions occur in real-time, leveraging backend APIs to dynamically update the frontend.

Frontend(Streamlit):
The frontend is built with Streamlit, offering an interactive web UI with multiple pages including health dashboards, data upload interfaces, AI chat, feedback forms, and report viewers. Navigation is handled through a sidebar using the streamlit-option-menu library. Each page is designed as a separate module to ensure scalability and clean separation of functionality.

Backend(FastAPI):
FastAPI serves as the backend REST framework that powers API endpoints for health data processing, chat interactions, risk prediction, anomaly detection, and report generation. It is optimized for asynchronous performance and includes built-in Swagger integration for API documentation and testing.

7. API Documentation

Backend APIs available include:

POST /chat/ask – Accepts a user query related to symptoms or health concerns and responds with an AI-generated medical advice or suggestions.

POST /upload-doc – Uploads and embeds patient records, medical documents, or CSV files into the vector database for semantic search.

GET /search-docs – Returns semantically relevant medical documents or patient notes based on the input query.

POST /submit-feedback – Stores user feedback or health-related reports for later analysis or improvement of the system.

GET /get-risk-predictions – Provides risk assessment results based on uploaded patient data or inputs.

8. Authentication

This version of the project is intended for demonstration purposes and runs in an open environment. For secure deployments, you can integrate:

- Token-based authentication
- authentication compatible with healthcare cloud platforms
- Role-based access control (e.g., patients, doctors, admins)
- Future enhancements planned include user session management, audit logs, and history tracking of health queries and reports.

9. User Interface

The interface is designed to be user-friendly and accessible, targeting both technical and non-technical users. It includes:

- Sidebar navigation for easy access to different modules like chat, data upload, and report viewing
- Interactive dashboards with health metrics and risk summary cards
- Tabbed layouts for symptom checking, AI chat, patient data uploads, and report generation
- Real-time form handling and instant feedback from the AI assistant
- Ability to download personalized PDF health reports and summaries

The design emphasizes simplicity, clarity, and responsiveness, with helpful tooltips and guided workflows to assist users throughout their interaction with the assistant.

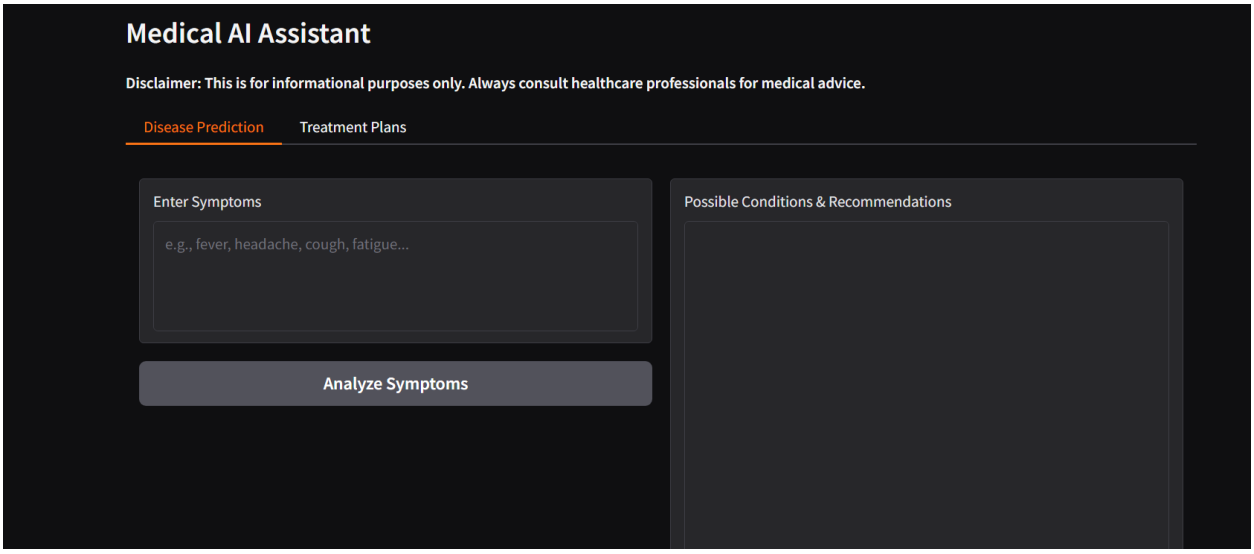
10. Testing

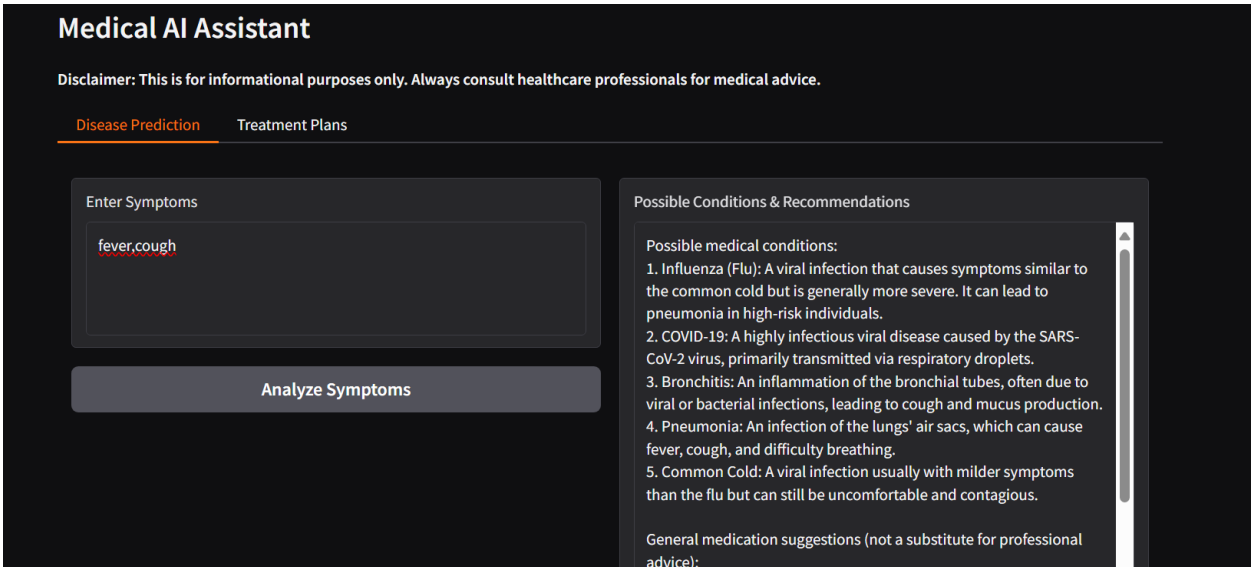
Testing was done in several phases:

- Unit tests for core functions and utilities.
- API testing using Swagger UI and Postman.
- Manual testing of file uploads, chat responses, and outputs.
- Handling edge cases like invalid inputs and large files.

All functions were verified for reliability both offline and online.

11.Screenshorts





12.Known Issues

Inaccurate or Wrong Medical Advice

The AI might give incorrect health information that could be harmful if trusted without a real doctor.

Privacy and Data Safety Risks

Sensitive patient data could be exposed or misused if the AI system is not secure.

Not Always Reliable
The AI can give different answers to the same question, depending on how it is asked, which can confuse users.

13.Future Enhancement

Integration with Real Doctors
Allow collaboration between AI and human doctors to review and validate AI responses for better safety.

Access to Real-Time Medical Data
Connect the AI to live databases (e.g., WHO, hospitals) to give more accurate and up-to-date health information.

Stronger Privacy and Security
Implement advanced encryption and follow strict privacy laws (like HIPAA) to protect patient data.

Multi-Language and Accessibility Support
Add support for different languages and accessibility features (voice commands, large text, etc.) to help more users.