HealthAI Project Documentation

1.Introduction

Project Title: HealthAI:Inteligent Healthcare Assistant Uising IBM Granite

Team Leader: VISHAL R

Team Member: VIJAY KUMAR T

Team Member: THIRAVIDA MANIKANDAN C

2.Project Overview

- **Purpose:**

The purpose of HealthAI is to enhance healthcare delivery by integrating artificial intelligence into diagnosis, treatment, and patient care. It provides doctors with clinical decision support, assists patients with personalized wellness recommendations, and enables healthcare systems to operate more efficiently.

- **HealthAI empowers:**

Doctors – by summarizing medical records, analyzing reports, and predicting risks.

Patients – through health chatbots, symptom checkers, reminders, and lifestyle advice.

Healthcare Systems – with forecasting tools, anomaly detection, and performance monitoring.

Ultimately, HealthAI bridges medical expertise and intelligent automation to create a healthcare ecosystem that is accurate, accessible, and patient-centered.

3.Architecture

- **Frontend (Streamlit or Gradio):**

Provides an interactive dashboard with patient records, symptom checker, chat interface, and report viewers.

Navigation through sidebar menus with multiple pages (Diagnosis, Reports, Wellness, Feedback).

- **Backend (FastAPI):**

Powers APIs for diagnosis support, health record analysis, chatbot interactions, and medical forecasting.

Supports real-time queries and integrates Swagger UI for testing.

- **LLM Integration (IBM Watsonx Granite / OpenAI API):**

Used for clinical summarization, report generation, and health Q&A.

Prompt-engineered to ensure accuracy, safety, and medical relevance.

- **Vector Search (Pinecone / FAISS):**

Stores and retrieves patient reports, prescriptions, and medical guidelines.

Enables natural language queries over medical documents.

- **ML Modules (Forecasting & Anomaly Detection):**

Predicts patient health risks (e.g., readmissions, abnormal lab trends).

Flags anomalies in vitals or treatment outcomes using time-series analysis.

4.Setup Instructions

- **Prerequisites:**

  Python 3.9+

  pip & virtual environment tools

  API keys (IBM Watsonx / Pinecone / OpenAI)

  Access to EHR or sample health datasets

- **Installation Process:**

  Clone the repository

Install dependencies from requirements.txt

Create a .env file with API keys and credentials

Start backend server with FastAPI

Launch frontend dashboard with Streamlit

Upload medical records or input symptoms to interact with modules

5.Folder Structure

```
healthai/
├── app/                # FastAPI backend logic
│   ├── api/            # API routes (chat, diagnosis, reports, feedback)
│   ├── models/         # ML models for forecasting & anomaly detection
│   ├── utils/          # Helper scripts
│
├── ui/                 # Streamlit frontend components
│   ├── pages/          # Dashboard pages (chat, reports, wellness tips)
│   ├── layouts/        # Card layouts, forms, and visualizations
│
├── main_dashboard.py   # Entry point for launching dashboard
├── health_llm.py       # Handles AI model communication
├── record_embedder.py  # Converts patient records to embeddings
├── risk_forecaster.py  # Predicts patient risks
├── anomaly_checker.py  # Detects abnormal health values
├── report_generator.py # Generates AI-based medical reports
```

6.Running the Application

Start the FastAPI server → exposes backend endpoints.

Run the Streamlit dashboard → launches the frontend interface.

Navigate via sidebar → choose between Diagnosis, Reports, or Wellness pages.

Upload patient data, interact with chatbot, and view reports/forecasts.

Download health reports in PDF/CSV format.

7.API Documentation

Backend Endpoints:

POST /chat/ask → Submit patient query, get AI-based response

POST /upload-record → Upload and embed medical records in Pinecone

GET /search-records → Retrieve similar cases or guidelines

GET /get-wellness-tips → Personalized lifestyle & wellness suggestions

POST /submit-feedback → Store patient/doctor feedback

All endpoints are tested in Swagger UI.

8.Authentication

Token-based Authentication (JWT / API Keys)

OAuth2 integration for hospital/clinic systems

Role-based access (Admin, Doctor, Patient, Researcher)

Planned features: session tracking and secure health data encryption

9.User Interface

Sidebar navigation (Diagnosis, Reports, Wellness, Feedback)

Patient record upload & visualization (charts, summary cards)

Chatbot interface for health queries

Downloadable PDF reports

Real-time health monitoring dashboards

10.Testing

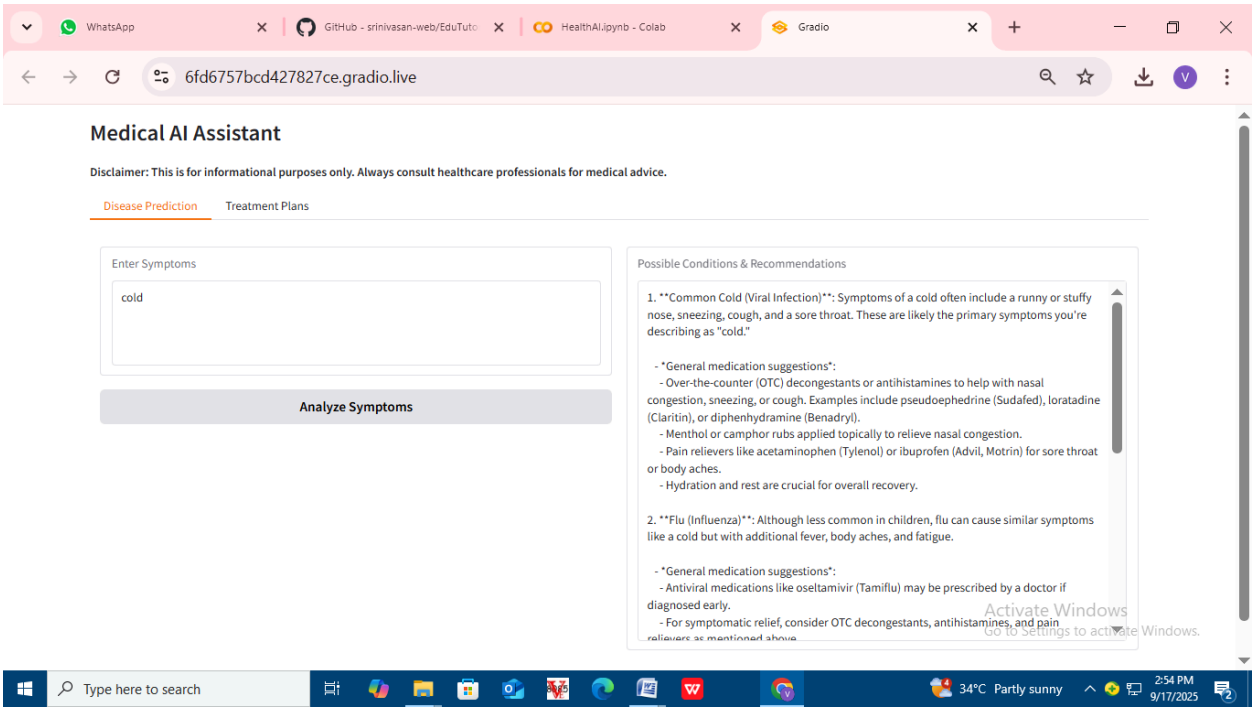
Unit Testing → LLM prompts, data parsers, anomaly detection

API Testing → Swagger UI, Postman, automated test scripts

Manual Testing → Symptom inputs, chat responses, medical report accuracy

Edge Case Handling → Large datasets, missing health fields, invalid API keys

11.screenshots



12.known Issues

- **Data Quality Dependence**: AI predictions heavily depend on the quality and completeness of patient records; missing or inconsistent data may reduce accuracy.
- **Limited Medical Context**: LLMs may occasionally generate generic advice that lacks full clinical context if prompts are not carefully engineered.
- **Latency in Large Data Processing**: Uploading and embedding large medical records (PDFs/CSVs) can lead to slower response times.
- **Integration with EHR Systems**: Compatibility challenges may arise when connecting with hospital-specific electronic health record platforms.
- **Privacy Concerns**: Without proper encryption and compliance (HIPAA/GDPR), sensitive patient data may be at risk in open deployments.

13.Future enhancement

- **Stronger Data Security** → Full HIPAA/GDPR compliance with advanced encryption and anonymization of patient data.
- **Multi-Language Support** → Expanding chatbot and report generation to support regional languages for wider accessibility.
- **Wearable Device Integration** → Real-time data ingestion from smartwatches, fitness trackers, and IoT medical devices.
- **Advanced Predictive Analytics** → Deep learning models for disease progression, readmission risks, and personalized treatment outcomes.
- **Telemedicine Integration** → Embedding video consultation modules with AI-assisted note-taking and prescription generation.
- **Continuous Learning System** → Fine-tuning AI models with new healthcare data for improved accuracy over time.
- **Mobile App Version** → Lightweight Android/iOS app for patient access outside the hospital environment.