Project Title

HealthAI – Intelligent Healthcare Assistant

Project Documentation

1. Introduction

• Project title : HealthAI – Intelligent Healthcare Assistant (IBM Project)
• Team member : J.SAMVEL
• Team member : R.MANIGANDAN
• Team member : R.CHANDRAN
• Team member : K.PERARASU

2. project overview

• Purpose :
The purpose of HealthAI – Intelligent Healthcare Assistant is to empower hospitals, doctors, and patients with AI-driven insights, real-time monitoring, and decision support. By leveraging IBM Watsonx LLMs and real-time health data, the assistant helps optimize patient care, summarize medical reports, forecast health risks, and provide personalized wellness guidance. For healthcare professionals, it acts as a decision-making partner—offering clear insights, predictive analytics, and summarizations of complex medical documents to support strategic planning. Ultimately, this assistant bridges technology, healthcare delivery, and patient engagement to foster smarter, more inclusive, and resilient healthcare systems.

• Features:
Conversational Interface
Key Point: Natural language interaction
Functionality: Allows patients and doctors to ask health-related questions, get updates, and receive guidance in plain language

Medical Report Summarization
Key Point: Simplified medical understanding
Functionality: Converts lengthy medical documents into concise, actionable summaries

Health KPI Forecasting
Key Point: Predictive analytics
Functionality: Estimates future patient health indicators using historical and real-time data

Wellness Tip Generator
Key Point: Personalized health advice

Functionality: Recommends daily actions to improve health and reduce risks based on patient data

Patient Feedback Loop
Key Point: Engagement & service improvement
Functionality: Collects and analyzes patient input to inform treatment and service improvements

Anomaly Detection
Key Point: Early warning system
Functionality: Identifies unusual patterns in patient vitals or data to flag potential risks

Multimodal Input Support
Key Point: Flexible data handling
Functionality: Accepts text, PDFs, and CSVs for medical document analysis and forecasting

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

3. Architecture

Frontend (Stream lit):
The frontend is built with Streamlit, offering an interactive web UI with multiple pages including dashboards, medical file uploads, chat interface, feedback forms, and report viewers. Navigation is handled through a sidebar using the streamlit-option-menu library. Each page is modularized for scalability.

Backend (Fast API):
Fast API serves as the backend REST framework that powers API endpoints for medical document processing, chat interactions, wellness tip generation, report creation, and vector embedding. It is optimized for asynchronous performance and easy Swagger integration.

LLM Integration (IBM Watsonx Granite):
Granite LLM models from IBM Watsonx are used for natural language understanding and generation. Prompts are carefully designed to generate summaries, health tips, and medical explanations.

Vector Search (Pinecone):
Uploaded medical documents are embedded using Sentence Transformers and stored in Pinecone. Semantic search is implemented using cosine similarity to allow users to search documents using natural language queries.

ML Modules (Forecasting and Anomaly Detection):
Lightweight ML models are used for forecasting and anomaly detection using Scikit-learn.

Time-series health data is parsed, modeled, and visualized using pandas and matplotlib.

4. Setup Instructions

Prerequisites:
- **o Python 3.9 or later**
- **o pip and virtual environment tools**
- **o API keys for IBM Watsonx and Pinecone**
- **o Internet access to access cloud services**

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

5. Folder Structure

app/ – Contains all Fast API backend logic including routers, models, and integration modules.
app/api/ – Subdirectory for modular API routes like chat, feedback, report, and document vectorization.
ui/ – Contains frontend components for Stream lit pages, card layouts, and form UIs.
health_dashboard.py – Entry script for launching the main Stream lit dashboard.
granite_llm.py – Handles all communication with IBM Watsonx Granite model including summarization and chat.
document_embedder.py – Converts medical documents to embeddings and stores in Pinecone.
health_forecaster.py – Forecasts patient health trends using regression.
anomaly_checker.py – Flags unusual values in uploaded KPI data.
report_generator.py – Constructs AI-generated healthcare reports.

6. Running the Application

To start the project:
- **➢ Launch the FastAPI server to expose backend endpoints.**
- **➢ Run the Streamlit dashboard to access the web interface.**
- **➢ Navigate through pages via the sidebar.**
- **➢ Upload medical documents or CSVs, interact with the chat assistant, and view outputs like reports, summaries, and predictions.**
- **➢ All interactions are real-time and use backend APIs to dynamically update the frontend.**

Frontend (Stream lit):
The frontend is built with Stream lit, offering an interactive web UI with multiple pages including dashboards, file uploads, chat interface, feedback forms, and report viewers. Navigation is handled through a sidebar using the stream lit-option-menu library. Each page is modularized for scalability.

Backend (Fast API):
Fast API serves as the backend REST framework that powers API endpoints for document processing, chat interactions, wellness tip generation, report creation, and vector embedding. It is optimized for asynchronous performance and easy Swagger integration.

7. API Documentation

Backend APIs available include:
POST /chat/ask – Accepts a user query and responds with an AI-generated message
POST /upload-doc – Uploads and embeds medical documents in Pinecone
GET /search-docs – Returns semantically similar documents to the input query
GET /get-wellness-tips – Provides personalized health tips for selected areas like nutrition, exercise, or chronic condition management
POST /submit-feedback – Stores patient feedback for later review or analytics
Each endpoint is tested and documented in Swagger UI for quick inspection and trial during development.

8. Authentication

each endpoint is tested and documented in Swagger UI for quick inspection and trial during development.
This version of the project runs in an open environment for demonstration.
However, secure deployments can integrate:
• Token-based authentication (JWT or API keys)
• OAuth2 with IBM Cloud credentials
• Role-based access (admin, doctor, patient)
• Planned enhancements include user sessions and history tracking.

9. User Interface

The interface is minimalist and functional, focusing on accessibility for non-technical users. It includes:
Sidebar with navigation
KPI health visualizations with summary cards
Tabbed layouts for chat, wellness tips, and forecasting
Real-time form handling
PDF report download capability
The design prioritizes clarity, speed, and user guidance with help texts and intuitive flows.

10. Testing

Testing was done in multiple phases:

Unit Testing: For prompt engineering functions and utility scripts

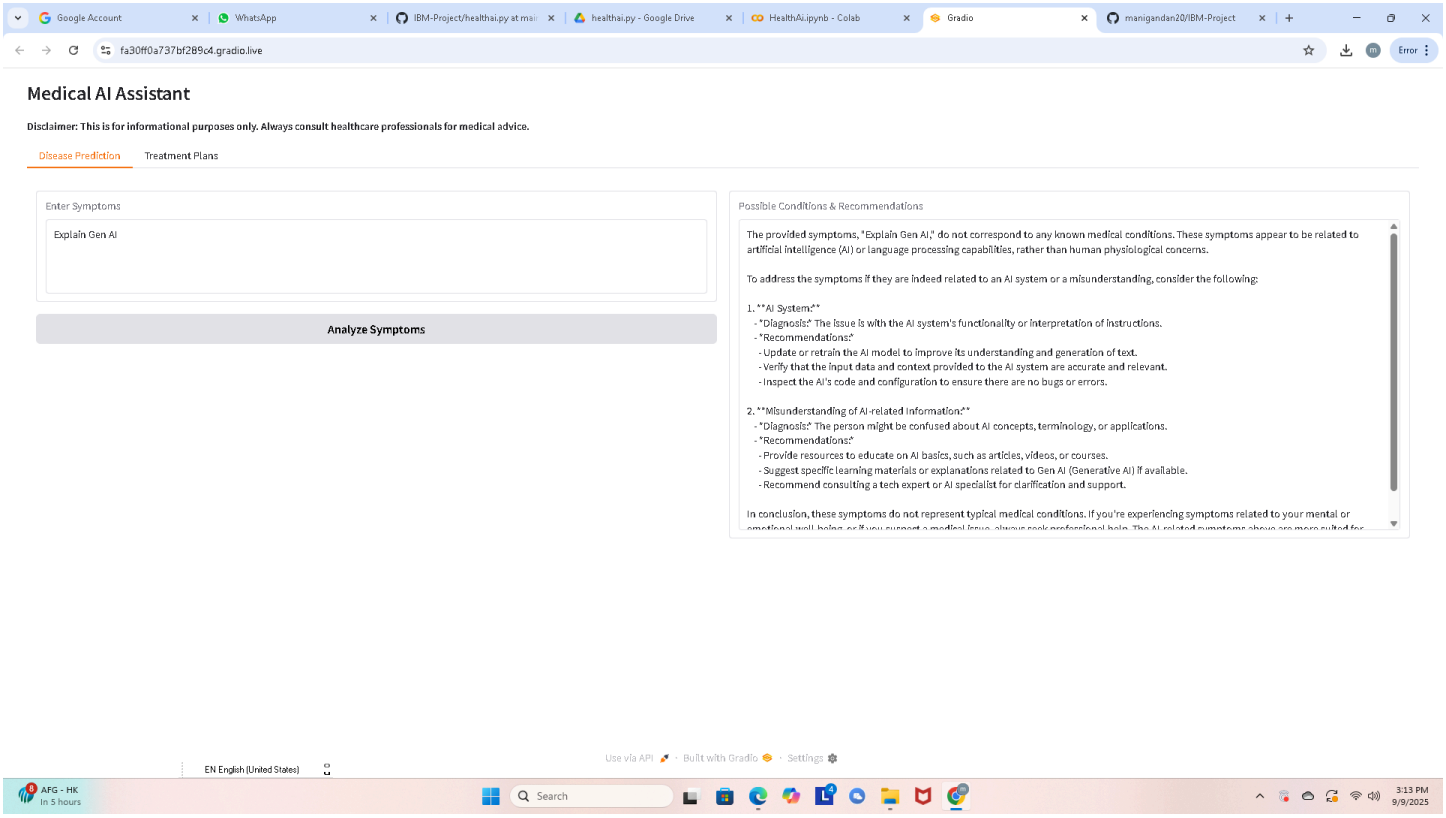
API Testing: Via Swagger UI, Postman, and test scripts

Manual Testing: For file uploads, chat responses, and output consistency

Edge Case Handling: Malformed inputs, large files, invalid API keys

Each function was validated to ensure reliability in both offline and API connected modes.

11. screen shots



12. Known Issues

- **Limited to English medical documents in this demo.**
- **Requires stable internet for IBM Watsonx & Pinecone services.**
- **Forecasting models may need larger, labelled clinical datasets for clinical-grade accuracy.**

13. Future enhancement

- **Multilingual support for patients and caregivers.**
- **Integration with IoT health devices and wearables for continuous monitoring.**
- **Voice-based interaction and accessibility enhancements.**