HEALTH AI: INTELLIGENT HEALTH CARE ASSISTANT

1. Introduction

- Team Member: Vidhya Lakshmi K
- Team Member: Nivasini Mazumder
- Team Member: YuvaShree G
- Team Member: Blessy Pooja N
- Team Member: Swetha R

2. Project Overview

Purpose:

The purpose of HEALTH AI: Intelligent Healthcare Assistant is to provide a smart, accessible, and user-friendly tool that assists individuals in understanding their health conditions through symptom analysis and general treatment guidance. With the increasing demand for quick and reliable health information, many people rely on online resources that may not always be accurate, structured, or personalized. This project aims to bridge that gap by using Artificial Intelligence to deliver context-aware, reliable, and structured healthcare information.

Conversational Interface

Key Point: AI-powered health symptom analyzer

Functionality: Users can enter their symptoms, and the system predicts possible medical conditions with general guidance.

Resource Forecasting

Functionality: Predicts patient demand, Estimates medicine and supply need

Policy Summarization

Key Point: Summaries complex healthcare policies and regulations into simple, accessible insights for patients, doctors, and administrators

Functionality: Provides automated summarization of healthcare policies, guidelines, and regulations into concise, easy-to-understand information for effective decision-making

Eco-Tip Generator

Key Point: Generates personalized eco-friendly health and lifestyle tips to promote sustainable healthcare practices.

Functionality: Provides users with automated, practical eco-friendly health and lifestyle suggestions to encourage sustainability in daily healthcare practices

Health Feedback Loop

Key Point: Creates a continuous cycle of monitoring, analysis, and feedback to improve user health outcomes and system accuracy.

Functionality: Enables continuous tracking of user health data, provides timely feedback, and refines recommendations for improved healthcare outcomes.

KPI Forecasting

Key Point: Predicts key healthcare performance indicators to support proactive planning and better decision-making.

Functionality: Analyzes historical health data to forecast key performance indicators, enabling proactive healthcare management and resource optimization.

Anomaly Detection

Key Point: Identifies unusual patterns or deviations in health data to detect potential risks at an early stage.

Functionality: Continuously monitors user health data to detect irregularities and provide early alerts for timely medical intervention.

Multimodal Input Support

Key Point: Allows users to interact through text, voice, images, and sensors for a seamless healthcare experience.

Functionality: Enables the system to process and integrate inputs from text, voice, images, and sensors, ensuring accurate and user-friendly healthcare assistance.

Stream Lit or Gradio UI

Key Point: Provides an interactive and user-friendly interface for seamless access to healthcare insights and AI functionalities.

Functionality: Delivers an intuitive, web-based interface using Stream lit/ Gradio to visualize data, interact with AI models, and provide real-time healthcare assistance.

3. Architecture

1. AI Assistant Layer (IBM Granite LLM Integration)

The AI Assistant Layer forms the core intelligence of the HEALTH AI system, enabling natural and context-aware interactions between users and the platform.

- **Key Functions**:
  - Natural Language Understanding – Interprets patient queries, medical terms, and contextual intent
  - Intelligent Response Generation – Provides accurate, human-like, and personalized healthcare responses.
  - Multimodal Interaction Support – Processes text, voice, and image-based inputs for seamless communication.
  - Continuous Learning & Adaptation – Improves accuracy through user feedback and health data trends.
- **Tools**:
  - IBM Granite LLM – Core large language model for intelligent healthcare query processing.
  - Watsonx.ai – IBM's AI platform for model deployment, training, and fine-tuning.
  - Python – Programming backbone for integration, APIs, and workflow automation.
  - Stream lit / Gradio – UI frameworks for interactive user interfaces.

2. Data Integration Layer

The Data Integration Layer acts as the backbone for consolidating and managing diverse healthcare data sources in the HEALTH AI system. It ensures seamless collection, cleaning, and harmonization of structured and unstructured data from hospitals, wearable devices, patient records, labs, and external healthcare databases.

Data Sources:

- Pharmacy & Prescription Databases – Medication history, drug interactions, and dosage information.
- Research Publications & Clinical Guidelines – Medical journals, treatment protocols, and healthcare best practices.
- Patient-Generated Data – Surveys, health questionnaires, and self-reported symptoms.

Purpose:

To consolidate, standardize, and manage diverse healthcare data from multiple sources, ensuring accurate, secure, and real-time information is available for AI-driven analysis and decision-making.

4. Setup Instructions

Prerequisites:

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

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 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 documents to embeddings and stores in Pinecone.

kpi_file_forecaster.py – Forecasts future energy/water trends using regression.anomaly_file_checker.py – Flags unusual values in uploaded KPI data.

6. Running the Application

To start the project:

- ➢ Launch the Fast API server to expose backend endpoints.
- ➢ Run the Stream lit dashboard to access the web interface.
- ➢ Navigate through pages via the sidebar.
- ➢ Upload 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, eco 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 documents in Pinecone GET /search-docs – Returns semantically similar policies to the input query

GET /get-eco-tips – Provides sustainability tips for selected topics like energy, water, or waste

POST /submit-feedback – Stores citizen 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 (symptoms, fever, mediation)
- 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 visualizations with summary cards
- Tabbed layouts for chat, eco 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

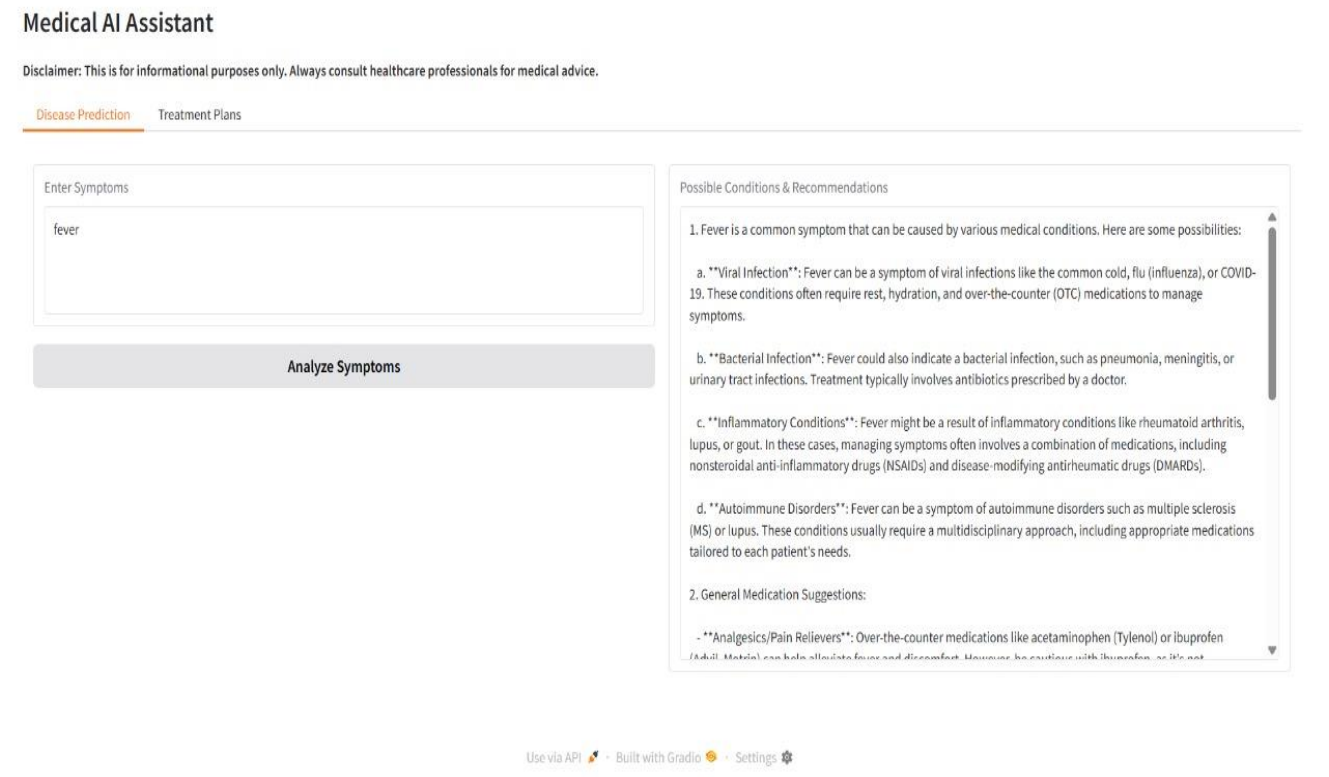
Testing was done in multiple phases:

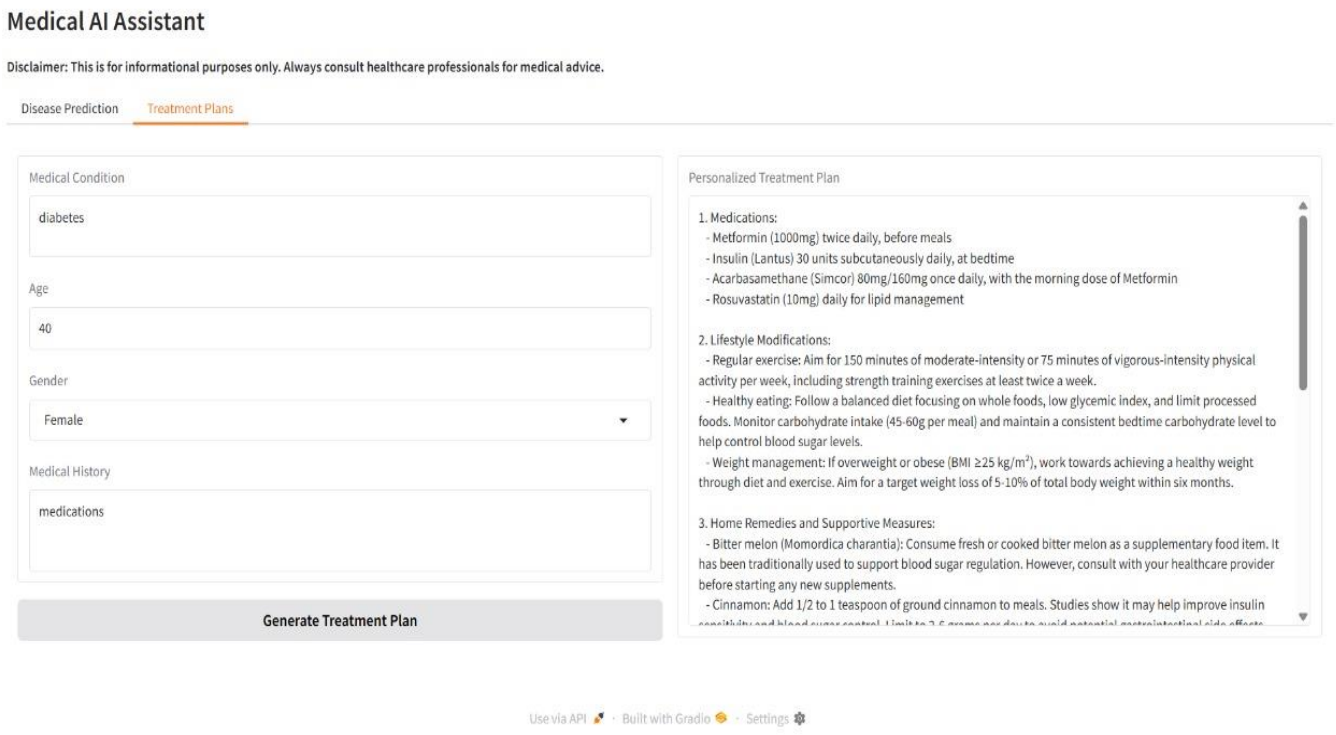
Unit Testing: For prompt engineering functions and utility scripts API Testing: Via Swagger UI, Postman, and test scripts

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 shorts





12. Known Issues

- **Accuracy Limitations:**
- The system relies on predefined datasets and rules, which may not cover all rare or complex medical conditions.
- Predictions may be generalized and not personalized enough for patients with multiple health conditions.

- **Limited Scope of Diseases:**
- Only common medical conditions and general symptoms are included. Rare diseases, comorbidities, and age-specific conditions are not fully supported.

- **Dependency on User Input**:
- The system heavily depends on correct symptom/medical history input. Incorrect or incomplete inputs can lead to misleading suggestions.

13. Future enhancement

Expansion of Disease Database:

Include rare diseases, pediatric conditions, geriatric health issues, and mental health disorders for broader medical coverage.

AI-Powered Symptom Analysis:

Use advanced machine learning models (like NLP + Deep Learning) to analyze multiple symptoms at once and predict possible conditions more accurately.

Personalized Treatment Plans:

Incorporate patient-specific factors such as allergies, genetic background, and comorbidities for tailored recommendation.

Continuous Learning System:

Allow the AI to learn from new patient cases, updated medical guidelines, and doctor feedback to improve accuracy over time.