HealthAI: Intelligent Healthcare Assistant Using IBM Granite

1.Introduction

- Project title : HealthAI – Intelligent Healthcare Assistant
- Team members:

1. Hemaprakash D
2. Naveen M
3. Manoj K
4. Mohammed yaseer A

2. Project Overview

- Purpose:

  The purpose of this project is to develop an Intelligent Healthcare Assistant that provides patients with accurate, accessible, and personalized medical guidance. By leveraging AI and generative models, the assistant helps users analyze symptoms, receive preliminary insights, and explore safe treatment recommendations. It also empowers healthcare providers with decision-support tools, offering data-driven predictions, summaries, and visualizations. Ultimately, this project aims to bridge the gap between patients and technology, creating a reliable, user-friendly platform that enhances awareness, promotes preventive care, and supports better health outcomes.

- Features:

  Conversational Interface

  - Key Point: Natural language interaction
  - Functionality: Enables patients to ask health-related questions and receive AI-generated guidance in plain, easy-to-understand language.

Symptom-Based Disease Prediction

- Key Point: Probabilistic diagnosis support
- Functionality: Analyzes reported symptoms with patient data to predict possible conditions along with likelihood scores and recommended next steps.

Treatment Plan Generator

- Key Point: Personalized healthcare guidance
- Functionality: Suggests lifestyle changes, safe medication information (*informational only*), tests, and follow-up recommendations tailored to each patient.

Health Analytics Dashboard

- Key Point: Visual trend insights
- Functionality: Provides interactive charts for vitals such as heart rate, blood pressure, and glucose levels, along with AI-driven improvement tips.

Patient Feedback Loop

- Key Point: Continuous improvement
- Functionality: Collects and evaluates user feedback to refine predictions, improve accuracy, and enhance patient experience.

KPI Tracking & Forecasting

- Key Point: Preventive health planning
- Functionality: Projects health performance indicators (e.g., blood sugar trends, heart health metrics) to help patients and clinicians monitor progress.

Anomaly Detection

- Key Point: Early warning system
- Functionality: Detects unusual health patterns in vitals or reported symptoms, alerting users to potential risks.

Multimodal Input Support

- Key Point: Flexible data handling
- Functionality: Accepts symptom text, uploaded health reports (PDFs/CSVs), and structured patient data for more accurate predictions and recommendations.

Gradio Interface

- Key Point: User-friendly platform
- Functionality: Provides an intuitive dashboard with tabbed navigation for chat, predictions, treatment plans, and analytics, making it accessible to patients and healthcare providers.

3. Architecture

The architecture of the HealthAI system is divided into several key components:

I. Frontend (Gradio):
The user interface is developed with Gradio, providing a clean and intuitive platform for interaction. It organizes the system into different tabs such as Patient Chat, Disease Prediction, Treatment Plan, and Health Analytics. Each tab acts like a separate text box, handling a specific functionality while keeping the interface simple for both patients and healthcare providers.

II. Backend (Google Colab + Hugging Face):
The backend logic is executed on Google Colab, making use of Python modules for handling patient data, analyzing symptoms, and generating treatment plans. Gradio communicates directly with these backend modules. The backend acts as the central processing unit, connecting inputs from the interface with outputs from the AI models.

III. LLM Integration (Granite LLM via Hugging Face):
The core intelligence of the system comes from Granite LLM models, accessed through Hugging Face. These models are responsible for natural language understanding and generation. They process queries, analyze symptoms, summarize patient information, and generate recommendations. Prompts are carefully designed to ensure outputs are structured, safe, and clinically cautious.

IV. Data Handling & Analytics:
Health data, such as vitals and patient history, is managed in simple JSON and CSV formats. Pandas and Matplotlib are used to analyze and visualize this data, producing trends and charts. This module works like an analytical text box, transforming raw patient data into actionable insights.

V. Deployment (Hugging Face Spaces / Colab):
The entire system can be run in Google Colab for testing and research or deployed to Hugging Face Spaces for broader accessibility. Both methods ensure the system is lightweight, easy to access, and does not require heavy infrastructure.

4. Setup Instructions

Prerequisites:

- Google account (to access Google Colab)
- Internet connection
- Hugging Face account

Installation Process:

1. Open Google Colab.
2. Install the required library inside Colab using pip:
    - For Gradio: pip install gradio
    - (Optional) For Hugging Face Transformers: pip install transformers
3. Once installed, you can upload your project code into Colab cells and run it.

5. Folder Structure

The project is implemented in a single Python notebook file for simplicity, making it easy to run directly in Google Colab without setting up multiple modules.

- healthai.ipynb – The main notebook file containing all functionality, including:
    - Installation of required libraries (gradio, transformers, torch).
    - Loading of the Granite model from Hugging Face.

    - Function to generate responses from the model.
    - Disease Prediction module: Analyzes symptoms and provides possible conditions with general recommendations.
    - Treatment Plan module: Generates personalized treatment guidance based on condition, age, gender, and medical history.
    - Gradio user interface with two main tabs:
        - *Disease Prediction*
        - *Treatment Plans*
    - Built-in disclaimer to ensure responsible use.
- requirements.txt (optional) – Minimal dependencies if someone wants to run the project outside Colab:
- gradio
- transformers
- torch
- README.md (optional) – Setup and usage instructions.

6. Running the Application

To start the project:

➢ Open Google Colab and upload or open the healthai.ipynb file.

➢ Run the first cell to install the required libraries using pip (gradio, transformers, torch).

➢ Execute the notebook cells to load the Granite model and initialize the application.

➢ When the final cell is run, Gradio will launch and display a shareable public link.

➢ Open the link in a browser to access the web application.

➢ Use the provided tabs (*Disease Prediction* and *Treatment Plans*) to enter inputs and view AI-generated outputs in real-time.

Frontend (Gradio):

The frontend is developed with Gradio, offering an interactive tabbed interface. It contains input text boxes, dropdowns, and number fields for entering patient data.

Outputs are displayed directly as text boxes with predictions or treatment plans. Navigation is simple and handled via tabs, ensuring a smooth user experience.

Backend (Colab + Hugging Face):
The backend logic is embedded within the same notebook. Hugging Face's Granite LLM model processes all inputs to generate predictions and treatment suggestions. The Gradio interface directly connects user inputs to the backend functions, ensuring instant results without needing a separate server.

7. API Documentation

This project does not use external REST APIs. Instead, it provides simple functions inside the Gradio app that act like APIs for users.

- Disease Prediction
    - Input: Symptoms (text)
    - Output: Possible conditions and basic advice with disclaimer
- Treatment Plan
    - Input: Condition, Age, Gender, Medical History
    - Output: Personalized treatment suggestions (informational only)
- Generate Response
    - Input: Prompt text
    - Output: AI-generated reply from the Granite model

All these functions are connected to the Gradio interface. Users just type inputs in the app, and results appear instantly without calling separate endpoints.

8. Authentication

This version of the project runs in an open environment (Google Colab with Gradio) for demonstration purposes. No login or API key is required to access the interface.

For a more secure deployment, future enhancements can include:

- API Keys or Tokens – Protect access to the app using simple keys.

- OAuth2 – Secure login with Hugging Face or cloud provider credentials.
- Role-Based Access – Different permissions for patients, doctors, and administrators.
- User Sessions & History Tracking – Allow authenticated users to save past queries and results.

9. User Interface

The interface is designed with Gradio to be simple and easy for non-technical users. It includes:

- Tabbed Layout – Separate tabs for *Disease Prediction* and *Treatment Plans*.
- Input Forms – Text boxes, dropdowns, and number fields to collect symptoms, condition details, and patient information.
- Output Areas – Large text boxes to display AI-generated results clearly.
- Real-Time Interaction – Results are generated instantly when the user clicks a button.
- Help Texts & Disclaimers – Clear messages guide the user and emphasize that outputs are informational only.

The design prioritizes clarity, accessibility, and responsiveness, making it easy for anyone to interact with the system.

10. Testing

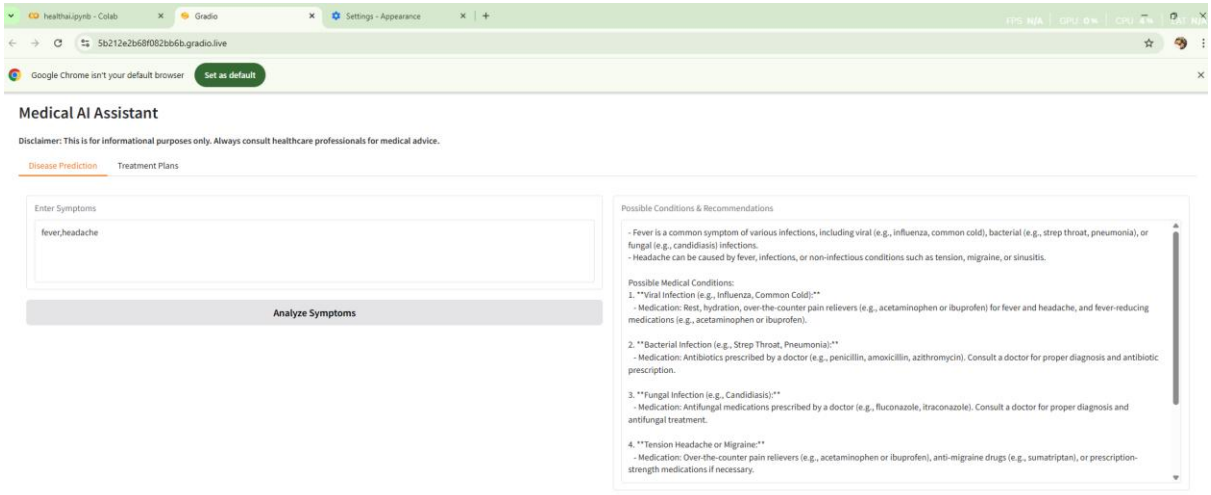
Testing of the application was carried out in simple phases:

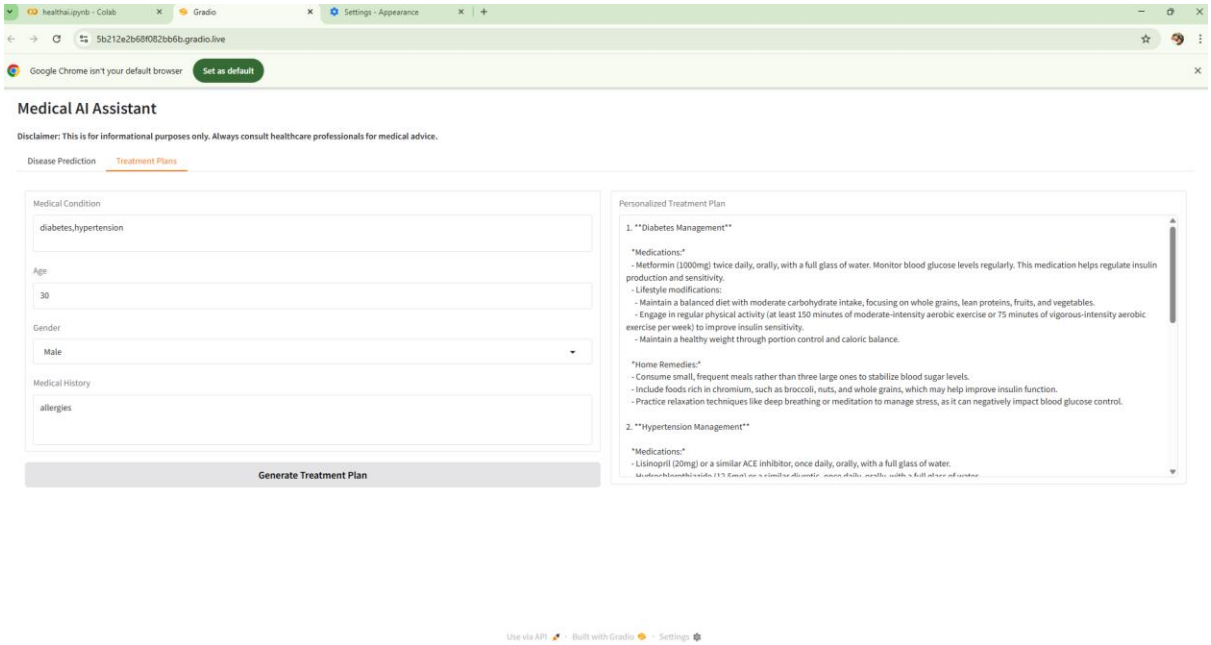
- Unit Testing – Core functions such as disease_prediction() and treatment_plan() were checked with different inputs to ensure valid outputs.
- Manual Testing – The Gradio interface was tested by entering symptoms, conditions, and patient details to confirm that responses were generated correctly.
- Edge Case Handling – Inputs such as empty text, unusual symptom combinations, or unrealistic patient data (e.g., negative age) were tested to observe system behavior.

- **Performance Check** – The app was run in Google Colab with and without GPU to verify smooth execution and response times.

All tests confirmed that the application produces outputs reliably within its demonstration scope.

11.screen shots





12. Known Issues

- Responses may sometimes be too long or less accurate.
- Results can vary for the same input.
- Slower performance without GPU in Colab.
- No login or data saving features yet.
- Outputs are only for information, not real medical advice.

13. Future Enhancements

- Add login and user accounts.
- Save past queries and user history.
- Support uploads like PDFs or CSVs for analysis.
- Add more health charts and tracking features.
- Deploy outside Colab for easier access.
- Improve model accuracy with training.
- Add multiple language support.