Health AI – Personalized Healthcare Assistance with IBM Granite

Project Documentation

1. Introduction

Title: Health AI – Personalized Healthcare Assistance with IBMGranite

Team Members

1. Safani Fathima S
2. Sandhiya R
3. Poojasri B
4. Subasri J

2. Project Overview

Health AI is an innovative project that leverages IBM Granite's powerful language model to deliver intelligent healthcare support. The platform focuses on assisting patients and healthcare enthusiasts by providing quick disease predictions and general treatment recommendations. This initiative aims to bridge the gap between medical knowledge and accessibility by empowering users with instant, AI-powered guidance while emphasizing that professional consultation is always essential.

Purpose

The primary goal of Health AI is to make basic healthcare information more accessible. By using IBM Granite LLM, the system analyzes user-provided symptoms and generates possible conditions along with general care guidelines. Additionally, the system generates treatment plans by considering user demographics such as age, gender, and medical history. The project is built with simplicity and user safety in mind—it does not replace medical advice but instead functions as a supportive tool for awareness, early self-checks, and education.

Features :

- **Disease Prediction:**
  Processes symptoms entered by the user and predicts potential diseases. For example, a combination of ‘fever, cough, and headache’ may lead to conditions like flu or cold. It not only lists likely diseases but also provides preventive tips and general advice.
- **Treatment Plan Generation:**
  Once a condition is identified, the system suggests a general treatment plan. The recommendations consider age, gender, and medical history to ensure the suggestions are tailored.

- **User-Friendly Gradio Interface:**
  Designed with Gradio, providing tab-based navigation for disease prediction and treatment planning.
- **Deployment Flexibility:**
  Supports deployment in local environments, Colab, or Hugging Face Spaces.

3. Architecture

- Frontend (Gradio): Provides a web-based UI for inputting symptoms and viewing results.
- Model Layer (IBM Granite): Granite 3.2 interprets natural language and generates predictions or treatment plans.
- Backend Logic (Python): Handles prompt engineering, processing, and response generation.
- Deployment: Supports execution on local machines, Colab, or Hugging Face Spaces.

4. Setup Instructions

1. Clone or download the repository.
2. Install dependencies using: pip install -r requirements.txt
3. Run the application with: python app.py
4. Access the Gradio link (http://127.0.0.1:7860).

5. Folder Structure

- app.py: Main application script.
- requirements.txt: Lists required dependencies.
- README.md: Documentation for usage.
- /static: Optional resources such as CSS and images.
- /tests: Testing scripts.

6. Running the Application

- Locally: Run python app.py and access via browser.
- Google Colab: Run notebook code to generate Gradio link.
- Hugging Face Spaces: Upload project files, system auto-builds.

7. API Documentation

- Disease Prediction API: disease_prediction('fever, cough') → Returns possible conditions with recommendations.
- Treatment Plan API: treatment_plan('diabetes', 35, 'Male', 'No major history') → Provides structured treatment guidelines.

8. Authentication

Currently no authentication is required. For production, token-based authentication, private Hugging Face Spaces, or role-based access can be added for security.

9. User Interface

- Disease Prediction Tab – Input symptoms for instant predictions.
- Treatment Plan Tab – Input condition, age, gender, history for personalized output.
- Clear outputs displayed in text boxes.

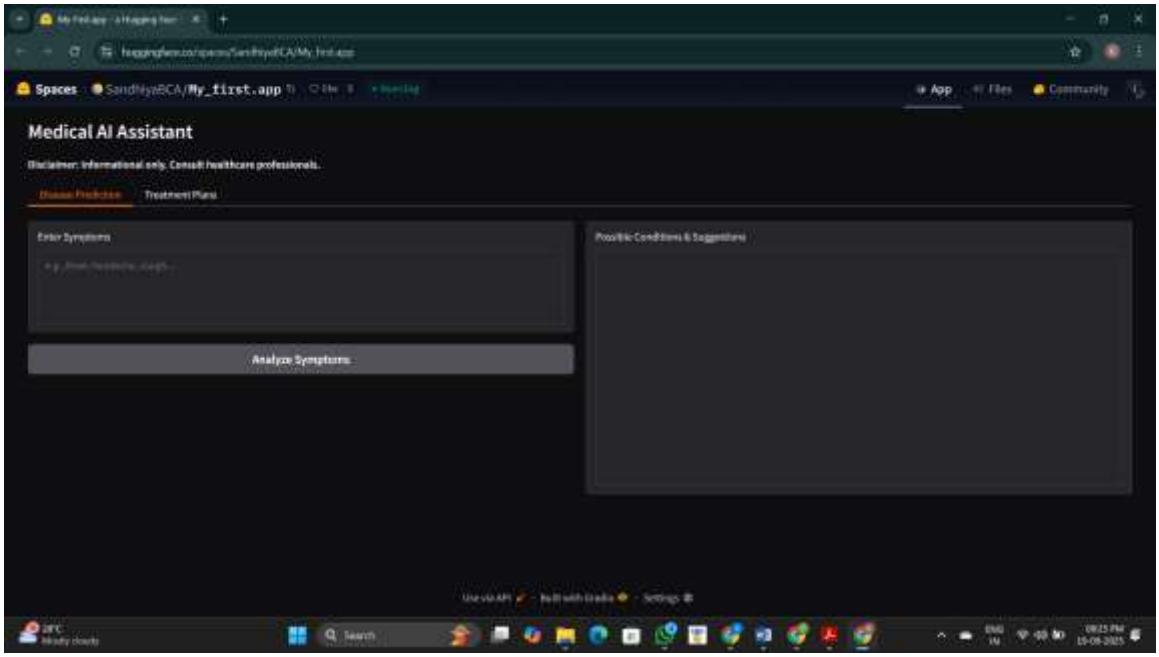
10. Testing

- Unit Testing – Validate core functions.
- Integration Testing – Ensure correct response generation.
- UI Testing – Verify Gradio inputs/outputs.
- Edge Case Testing – Handle rare symptoms or incomplete details.

11. Known Issues

- LLM may produce generic or hallucinated responses.
- No clinical validation – cannot replace doctors.
- Slower on CPU; GPU recommended.
- Confidence scores not yet available.

12. Screenshots



13. Future Enhancements

- Add voice input/output support.
- Store user history securely with compliance.
- Provide multi-language support.
- Integrate real medical APIs (ICD-10, PubMed).
- Add confidence scores for predictions.
- Develop a mobile-friendly version.