HealthAI Assistant

Project Documentation

1. Introduction

Project Title: HealthAI: intelligent Healthcare Assistant Using IBM Granite

Team Members:
. Team Leader : MATHUMITHA P

- Team Member 1:PANDIAN C
- Team Member 2: RASIKA R
- Team Member 3 : NIVETHA D
- Team Member 4 :ASARUDEEN T

2. Project Overview

Purpose:

The purpose of the Medical AI Assistant is to assist users by predicting possible medical conditions based on symptoms and generating personalized treatment suggestions. By leveraging IBM Granite LLM models and Gradio, the system provides natural language insights while emphasizing the importance of consulting healthcare professionals for accurate diagnosis and treatment.

Features:

- **Disease Prediction:** Predicts possible medical conditions based on user-input symptoms.
- **Personalized Treatment Plans:** Generates treatment suggestions considering age, gender, and medical history.
- **Conversational Interface:** Enables natural language interaction with an intuitive Gradio interface.
- **General Medication Guidelines:** Provides general medication and home remedies while highlighting consultation necessity.
- **Multi-tab Dashboard:** Organized UI with separate tabs for disease prediction and treatment planning.

3. Architecture

Frontend (Gradio):
Built using Gradio Blocks, the interface provides an interactive multi-tabbed dashboard where users can input symptoms, generate predictions, and access treatment plans.

Backend (Transformers & PyTorch):

The backend integrates Hugging Face Transformers with PyTorch for model inference. It loads the IBM Granite LLM model for response generation.

LLM Integration (IBM Granite):

IBM Granite's 3.2 2B Instruct model is used for natural language understanding and medical response generation.

4. Setup Instructions

1 Install Python 3.9 or later.

2 Clone the project repository.

3 Install dependencies from requirements.txt.

4 Ensure you have access to IBM Granite LLM via Hugging Face.

5 Run the Gradio application script.

6 Open the provided local or public URL to access the dashboard.

5. Folder Structure

• app/ – Contains backend model integration and response generation logic. •

ui/ – Contains Gradio UI components and dashboard configuration. •

model_loader.py – Handles Granite model loading and inference. •

requirements.txt – Lists Python dependencies.

• app.py – Entry point for running the Gradio application.

6. Running the Application

1 Launch the Gradio app by running: python app.py

2 Access the app via the local URL provided in the terminal.

3 Navigate between tabs to predict diseases or generate treatment plans.

7. User Interface

The UI is designed for simplicity and usability. It features two main tabs: one for disease prediction and another for generating treatment plans. Inputs are handled through textboxes, dropdowns, and number fields. Outputs are displayed in large multi-line textboxes for clarity.

8. Testing

• Unit Testing: For disease prediction and treatment plan generation functions. •

Integration Testing: Ensuring smooth LLM integration with Hugging Face and Gradio. •

Manual Testing: Testing edge cases like invalid inputs and large symptom lists.

9. Known Issues

- The model does not provide medically verified diagnoses. - Internet connection required for accessing Hugging Face models. - No authentication implemented; app runs in open environment.

10. Future Enhancements

• Integrate real-time medical knowledge bases for verified information. •

Add user authentication and secure data handling.

• Enable integration with wearable health monitoring devices.

• Implement advanced analytics and visualization for symptom trends.