HealthAI: Intelligent Healthcare assistant

Project Documentation

1.Introduction

- Project title : Intelligent Healthcare assistant
  Team ID : NM2025TMID00545
- Team Leader : SYED ASHRAF S N
- Team member : SHARATH S
- Team member : SAI GOWRI SHANKAR R
- Team member : NESA KUMAR S

2.project overview

- Purpose :
  The purpose of HealthAI is to provide an intelligent healthcare assistant that helps users get quick, reliable, and easy-to-understand health-related information. By leveraging IBM Granite models from Hugging Face, HealthAI can predict possible diseases based on user symptoms, suggest treatment plans, and provide health guidance in natural language.
- Features:
  Conversational Interface
  Key Point: Natural language interaction.
  Functionality: Users can chat with the assistant to describe symptoms and get advice.
  Disease Prediction
  Key Point: ML-driven prediction
  Functionality: Predicts possible diseases based on symptoms using ML models.
  Treatment Recommendation
  Key Point: AI-generated treatment plans
  Functionality: Suggests next steps (self-care or doctor visit) based on prediction.
  Explainable Output (Optional Future Enhancement)
  Key Point: Transparency

Functionality: Shows reasoning for predictions using feature importance or explainability tools.

User-Friendly UI (Gradio)

Key Point: Easy interaction

Functionality: Provides a clean web interface where users can input symptoms and get real-time results.

3. Architecture

Frontend (Gradio):

Gradio is used to build an interactive web interface. It allows users to enter symptoms and receive predictions with treatment recommendations in real time.

Backend (Google Colab + Python):

The backend is implemented in Python and runs on Google Colab with GPU support. It uses Hugging Face transformers and torch libraries to load the IBM Granite model and process user inputs.

LLM Integration (IBM Granite on Hugging Face):

Granite models are used for natural language understanding and generation. The model processes user symptoms and generates appropriate medical responses.

ML Module (Disease Prediction):

Uses a lightweight ML model to predict diseases based on symptom input. Future scope includes integrating statistical models and explainability.

4. Setup Instructions

Prerequisites:

Python 3.8+

Google Colab access (preferably with T4 GPU runtime)

Hugging Face account (API token if needed)

Required libraries: transformers, torch, gradio

Installation Process:

1. Open Google Colab and create a new notebook.

2. Change runtime to GPU (T4).

3. Run the command:

```
!pip install transformers torch gradio -q
```

4. Copy the project code from the provided Google Drive link.

5. Run all cells and launch the Gradio interface.

5. Folder Structure

```
Healthai/
|
 ├── healthai_notebook.ipynb    # Main Google Colab notebook
 ├── requirements.txt           # List of required libraries
 ├── README.md                  # Project overview and usage instructions
 ├── assets/                    # Screenshots and sample outputs
└── healthai.py                 # Script version of the app (optional)
```

---

6. Running the Application

1. Open the Colab notebook.

2. Install dependencies and run all cells.
3. Launch Gradio UI → Enter symptoms → View predictions and recommendations.
4. Copy the shareable link (demo link) to submit.

7. **API / Model Documentation**

Hugging Face Model: ibm-granite/granite-3.2-2b-instruct
Input: Text (symptoms)
Output: Text (predicted disease name + suggested treatment)

8. Authentication

For demonstration purposes, the project runs without authentication. Future versions can include:

User login & medical history tracking

Secure data storage using databases

Role-based access for doctors vs. patients

9. User Interface

Minimal Gradio UI

Input box for symptoms

Output box with predicted disease & treatment recommendation

10. Testing

Unit Testing: Tested model response for common symptom inputs.

Manual Testing: Verified chatbot functionality and accuracy with sample cases.

Edge Cases: Tested with invalid or empty input to ensure graceful handling.

11. **Known Issues**

Model may not always provide accurate predictions for very rare diseases.

Limited dataset → cannot cover all possible conditions.

12. **Future Enhancements**

- Add statistical disease probability calculation (chi-square, logistic regression).
- Integrate SHAP/LIME for explainable output.
- Build a Tableau dashboard to visualize most common predicted diseases.
- Deploy on Hugging Face Spaces or Streamlit Cloud for permanent hosting.