**1. Introduction**

**Project Title: Health AI – Intelligent Healthcare Assistant**

**Team Leader:**

**• M.Thendral**

**Team Members:**

* **J.Rakshaya**
* **M.Sowmiya**
* **S.Supradha**

**---**

**2. Project Overview**

**Purpose:**

**The purpose of Health AI is to provide smart, reliable, and easy-to-understand medical assistance using IBM Granite models. It enables patients and healthcare providers to interact through natural conversations, predict diseases, receive personalized treatment plans, and access medical guidance quickly. The system combines AI-powered language understanding with an intuitive interface to support healthcare decision-making in real time.**

**Features:**

**Patient Chat**

**Key Point: Conversational healthcare interaction**

**Functionality: Provides real-time responses to patient queries in natural language.**

**Disease Prediction**

**Key Point: Preventive healthcare support**

**Functionality: Uses patient symptoms and history to suggest possible conditions.**

**Treatment Plan Suggestions**

**Key Point: Personalized medical guidance**

**Functionality: Offers AI-based recommendations for treatment steps and lifestyle improvements.**

**Healthcare Summarization**

**Key Point: Simplified medical knowledge**

**Functionality: Summarizes medical literature, patient history, and reports into concise, actionable insights.**

**Multi-Modal Support (Future Scope)**

**Key Point: Flexible data handling**

**Functionality: Support for text, PDFs, and medical data files for enhanced healthcare insights.**

**User-Friendly UI (Gradio)**

**Key Point: Easy accessibility**

**Functionality: Provides a lightweight, interactive interface for patients and healthcare staff.**

**---**

**3. Architecture**

**Frontend (Gradio):**

**Interactive web interface for patients and doctors to chat with the assistant, upload data, and receive predictions.**

**Backend (Google Colab + Python):**

**Powered by Python with support for AI libraries such as Transformers and Torch.**

**Executes AI model inference and data processing tasks.**

**LLM Integration (IBM Granite Models):**

**Uses Hugging Face’s IBM Granite (e.g., granite-3.2-2b-instruct) for natural language understanding and generation.**

**Supports disease prediction, medical Q&A, and treatment recommendations.**

**Version Control (GitHub):**

**Project code is version-controlled and hosted on GitHub for collaborative development.**

**---**

**4. Setup Instructions**

**Prerequisites:**

**Python 3.9 or later**

**Gradio framework**

**IBM Granite model access (via Hugging Face)**

**Google Colab account with T4 GPU enabled**

**GitHub account for version control**

**Installation Process:**

**1. Open Google Colab and create a new notebook.**

**2. Change runtime to T4 GPU.**

**3. Install dependencies:**

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

**4. Load the IBM Granite model from Hugging Face.**

**5. Run the project code to launch the Gradio application.**

**6. Upload the project to GitHub for version control.**

**---**

**5. Folder Structure**

**/notebooks/ – Google Colab notebooks with implementation.**

**/models/ – Model integration scripts for IBM Granite.**

**/ui/ – Gradio UI code for chat and predictions.**

**/data/ – Sample patient data and test inputs.**

**main.py – Entry script for launching the Health AI application.**

**---**

**6. Running the Application**

**1. Open the Colab notebook and execute the cells.**

**2. Launch the Gradio app – an access URL will be generated.**

**3. Interact with the chatbot for health queries.**

**4. View predictions, treatment suggestions, and summaries.**

**---**

**7. API Documentation (Planned)**

**POST /chat – Input patient queries and receive AI-generated responses.**

**POST /predict-disease – Input symptoms to get possible conditions.**

**POST /treatment-plan – Generate treatment recommendations.**

**GET /summarize – Summarize medical documents.**

**---**

**8. Authentication (Future Enhancement)**

**For now, the project runs in an open environment for testing. Future deployments can include:**

**API keys or JWT authentication**

**OAuth2 integration with healthcare systems**

**Role-based access (patient, doctor, admin)**

**---**

**9. User Interface**

**Gradio-based chatbot window**

**Input box for patient symptoms**

**Buttons for prediction and treatment generation**

**Summarized medical reports displayed in real time**

**---**

**10. Testing**

**Unit Testing: Python modules and prompt engineering.**

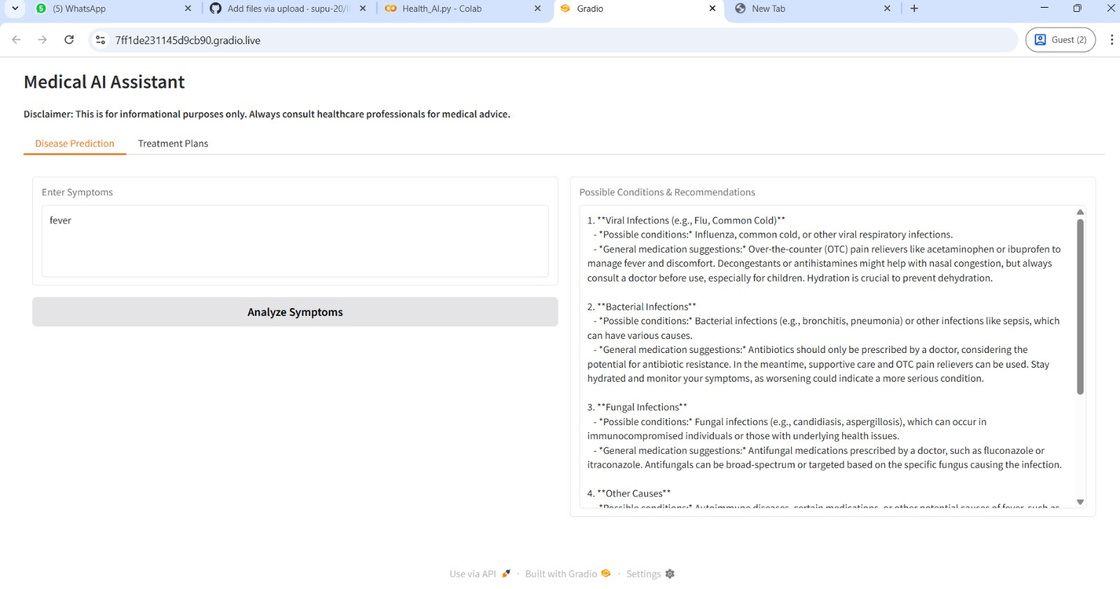
**Application Testing: Running in Colab and validating outputs.**

**Manual Testing: Chat interactions, symptom predictions, and report summaries.**

**Edge Case Testing: Invalid inputs, empty queries, long text handling.**

**---**

**11. Screenshots**



**---**

**12. Known Issues**

**Limited to general medical guidance, not a replacement for professional advice.**

**Dependent on internet connectivity (Colab + Hugging Face models).**

**Lacks integrated EHR (Electronic Health Record) support.**

**---**

**13. Future Enhancements**

**Add multimodal data support (PDF reports, lab results).**

**Expand disease prediction accuracy with more datasets.**

**Integration with hospital management systems.**

**Enhanced authentication and data privacy features.**