**Generative AI Project using IBM Cloud – HEALTHAI**

**Project Documentation Format**

1. **Introduction** 
   * **Project Title:** **HEALTHAI: Intelligent Healthcare Assistant using IBM Granite (Generative AI with IBM Cloud)**
   * **Team id: NM2025TMID08932**
   *  **Team Members:**
     + **thiru (Team Leader – Development & Integration):**

Led the complete development of the HEALTHAI application, including IBM Granite integration, Streamlit-based UI design, module creation, and model API handling.

O vengadeshwaran**(Model Interaction & Testing):**

Contributed by assisting in prompt design, testing the AI model outputs across modules like Disease Prediction and Health Chat, and refining interactions with IBM Granite.

* + - **vasu s&adhithiyan.s (UI Structuring & Feature Enhancement):**

Supported in designing user flow, organizing the Streamlit interface across all modules, and suggesting improvements in user interaction and feature behavior.

1. **Project Overview** 
   * **Purpose:**

To build a Generative AI-based healthcare assistant using IBM Granite, capable of answering health queries, predicting diseases, suggesting treatments, and displaying analytics.

* + **Features:**
    - AI Health Chat using IBM Granite o Disease Prediction from user symptoms o Treatment Plan Suggestions o Health Analytics Dashboard o Centralized shared model for performance optimization

1. **Architecture** 
   * **Frontend:**

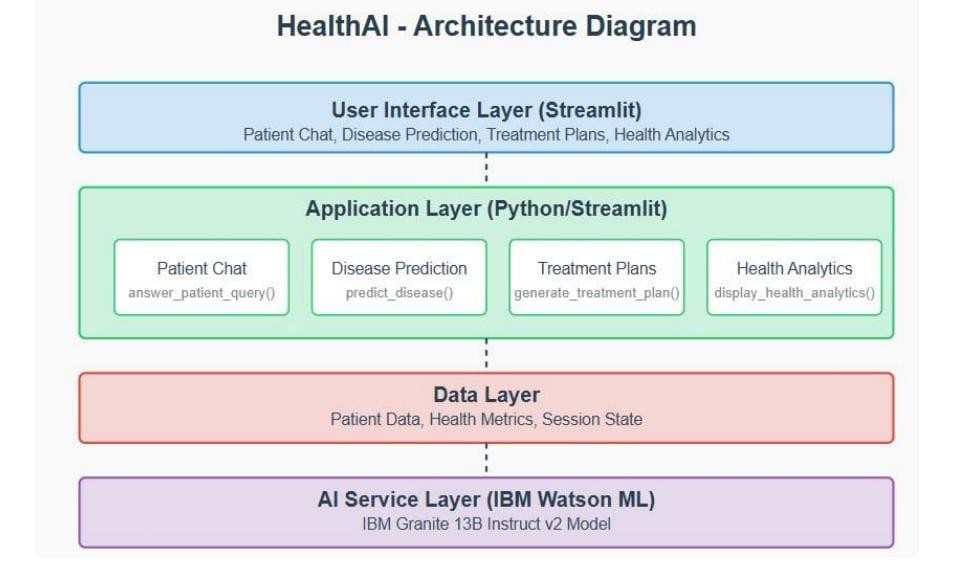
Built using **Streamlit** for a clean and responsive web interface. Each feature is modularized for easy navigation via sidebar.

* + **Backend & Model:**
    - No traditional backend. All logic handled in Streamlit using Python.
    - Uses **IBM Granite 3.3B Instruct model** from Hugging Face: ibm-granite/granite-3.32b-instruct o Supports both API and **local model loading** (granite/ folder).
  + **Shared Model Loader:**

The shared\_model.py file centrally loads and shares the AI model across modules to prevent memory crashes and redundancy.

**4**

**. Setup Instructions**



**Prerequisites**

* + Python 3.10+
  + pip
  + Hugging Face account and token
  + Installed model files if using local (granite/ folder)

**Installation**

git clone <https://github.com/harshithahoney16/Healthai.git>

cd Healthai pip install -r requirements.txt

**Environment Variables**

Create a .env file in the root folder:

HUGGINGFACEHUB\_API\_TOKEN=hf\_EPKOkQWaTrYYRwbVgrfzpiTWNrSADVyjnd

✅ .env file must be excluded in .gitignore.

**5. Folder Structure**

Health-ai/

├── app.py # Main entry point

├── shared\_model.py # Shared AI model instance

├── patient\_chat.py # AI Health Chat module

├── disease\_prediction.py # Disease Prediction logic

├── treatment\_plans.py # Treatment Plan suggestions

├── health\_analytics.py # Analytics module

├── requirements.txt # Python dependencies

├── .env # API token (not pushed to GitHub)

├── granite/ # [Optional] Local model folder

└── assets/ # Logos and screenshots

1. **Running the Application**

**For Hugging Face API:** streamlit run app.py

**For Local Model:**

Ensure granite/ folder contains the downloaded model and tokenizer files. In shared\_model.py, update: model\_path = "./granite"

1. **API Documentation**

**Endpoint:** https://api-inference.huggingface.co/models/ibm-granite/granite-3.3-2b-instruct

**Method:** POST

**Headers:**

{

"Authorization": "Bearer <HUGGINGFACEHUB\_API\_TOKEN>",

"Content-Type": "application/json"

}

**Example Request:**

{

"inputs": "What are the symptoms of diabetes?"

}

**Example Response:**

{

"generated\_text": "Common symptoms of diabetes include frequent urination..."

}

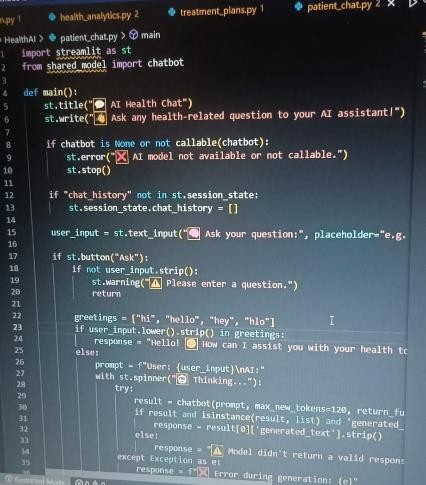
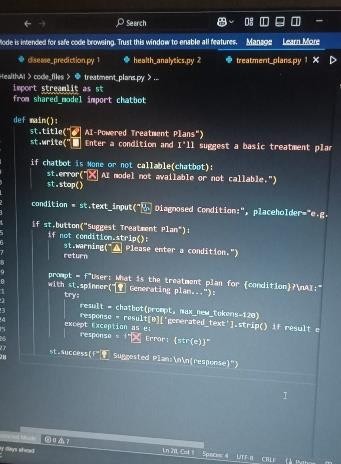
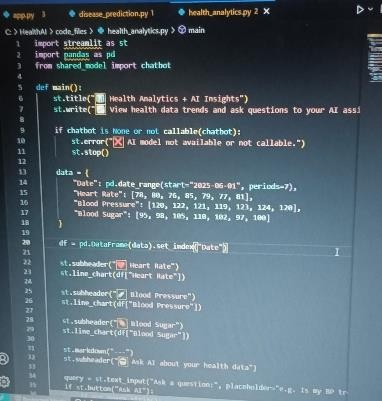
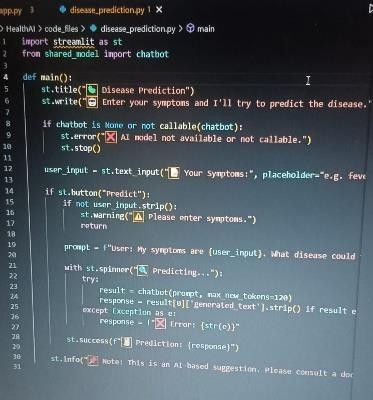
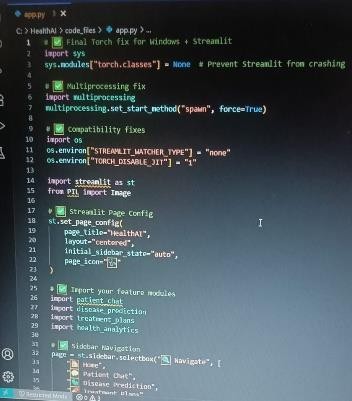
1. **Authentication** 
   * Hugging Face token is securely stored in .env
   * .env is excluded via .gitignore
   * App is currently public and stateless (no user login)
   * Streamlit or Firebase Auth can be added in future

1. **User Interface** 
   * Built entirely with **Streamlit**
   * Sidebar for navigation
   * Text/chat inputs for interaction
   * Visual graphs and health tips in Analytics
   * Centralized theme and branding

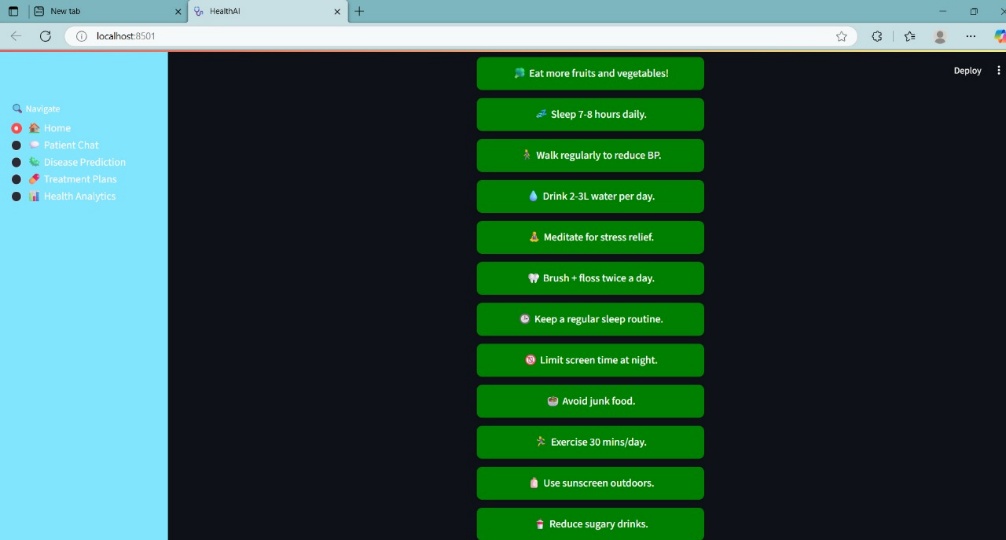
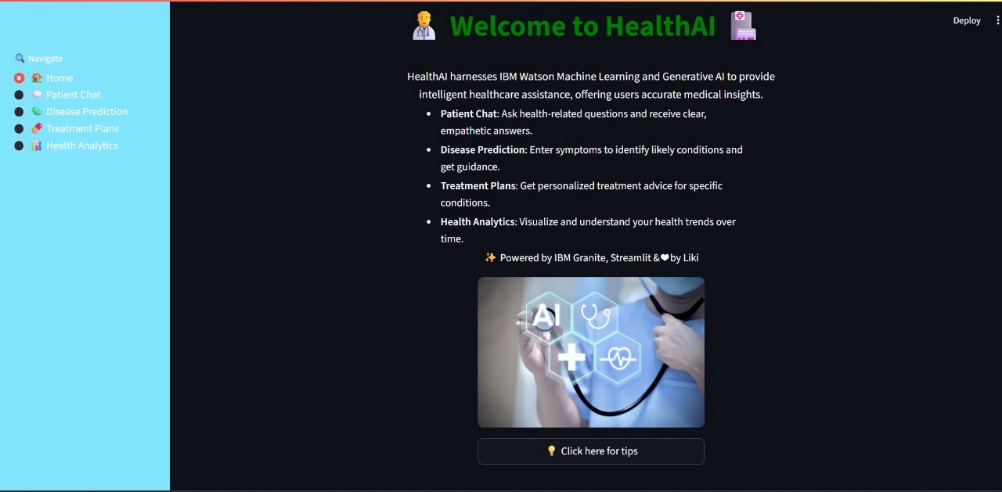
1. **Testing** 
   * ✅ Manual testing across all modules
   * ✅ Model tested with varied prompts and edge cases
   * ✅ Handled errors for invalid inputs and model timeouts

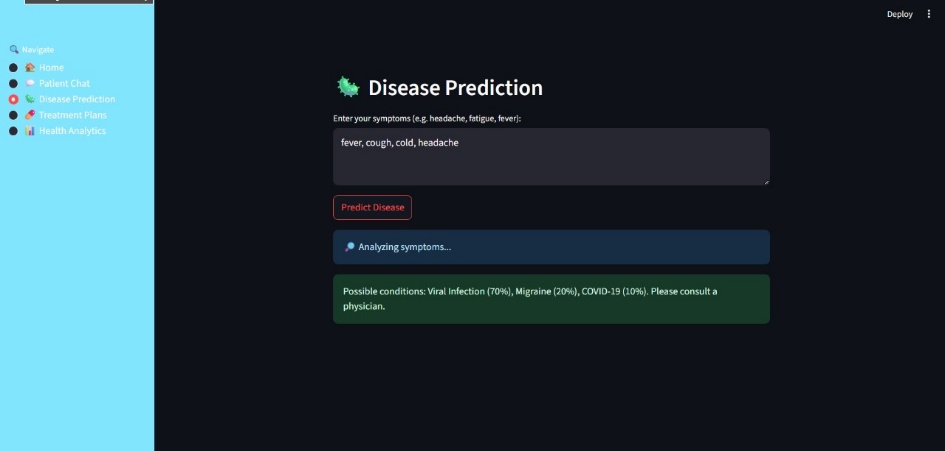
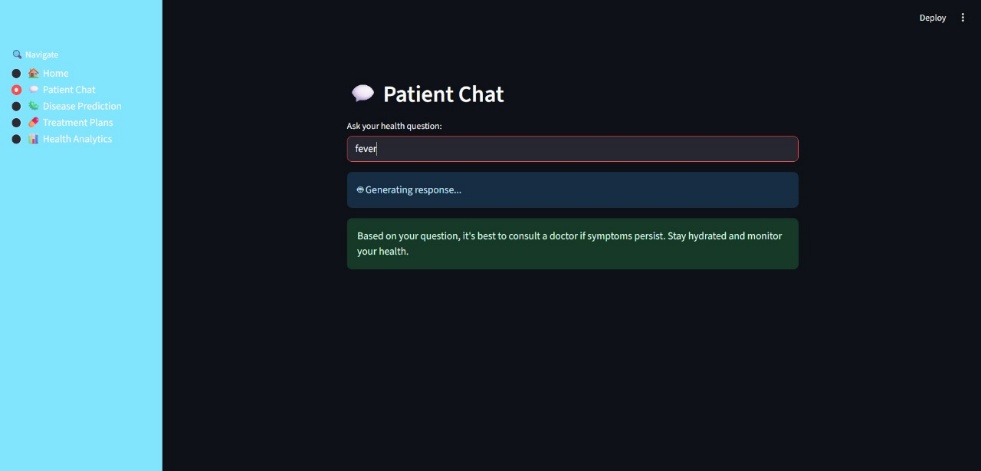
1. **Screenshots or Demo** 
   * [Demo Video on YouTube](https://youtu.be/your_demo_link" /h)  **INPUTS ( CODES ) :**

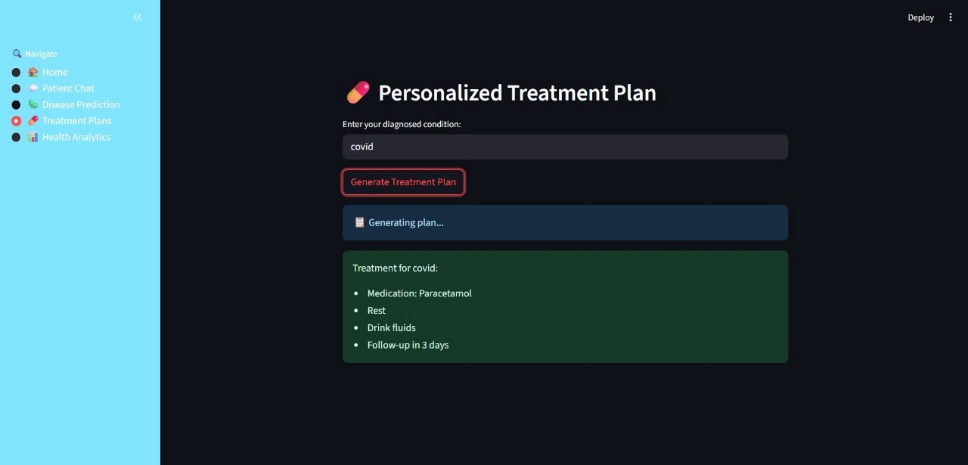


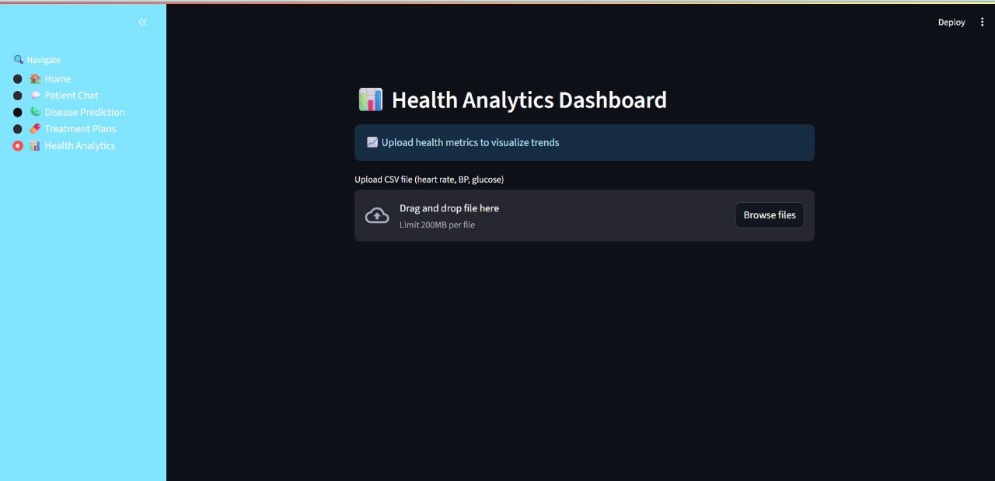


* + **OUTPUT : https://www.youtube.com/watch?v=sHO20zfQnuY**









1. **Known Issues** 
   * Generic model outputs due to lack of medical domain fine-tuning
   * Internet dependency when using Hugging Face API
   * No data persistence (currently stateless app)

1. **Future Enhancements** 
   * ✅ Add user authentication and patient record storage
   * ✅ Deploy on IBM Cloud / Hugging Face Spaces
   * ✅ Multilingual prompt support
   * ✅ Mobile version of the app
   * ✅ Integrate with real-time health APIs or EHRs