📘 **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 Members:**
  - **Boppana Nagasai(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.
  - **Boppudi Hanumatharao& DevisettyPramod kumar (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.
  - **Devarapalli Karthik& Dokku syam kumar (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.

---

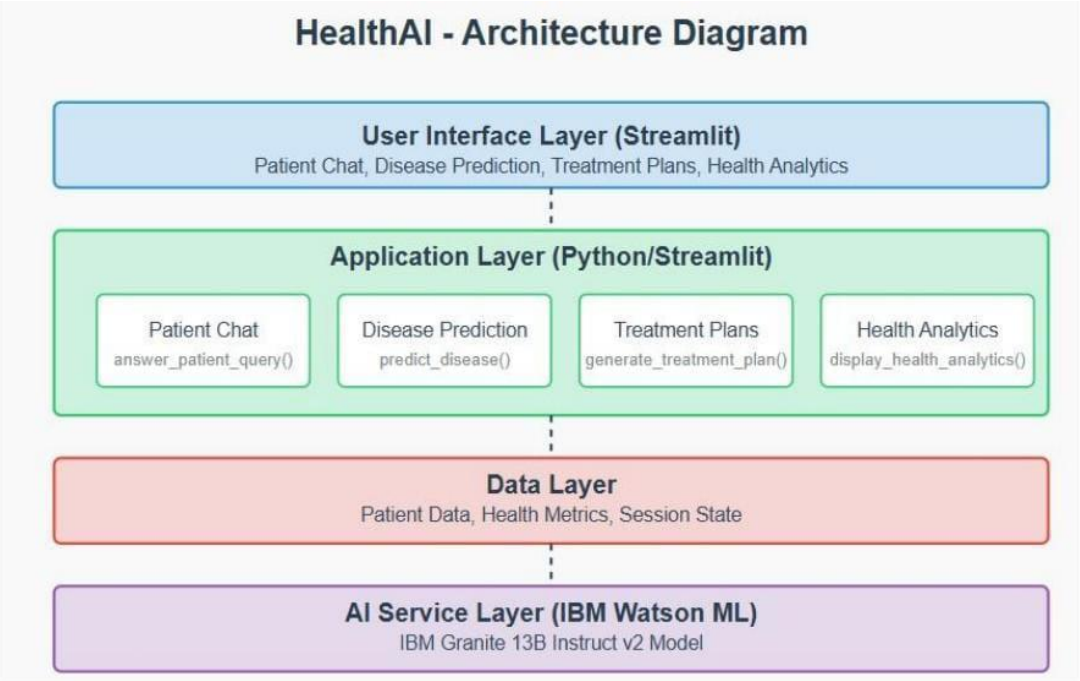
2. Project Overview

- **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
  - 🧠 Disease Prediction from user symptoms
  - 💊 Treatment Plan Suggestions
  - 📊 Health Analytics Dashboard
  - ☁️ Centralized shared model for performance optimization

---

3. 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-3.3-2b-instruct
  - 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

https://github.com/nagasai-boppana/healthai

cd Health-ai

pip install -r requirements.txt

Environment Variables

Create a .env file in the root folder:

HUGGINGFACEHUB_API_TOKEN=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

✅ .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
```

---

6. 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"

---

7. 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..."
}
```

---

8. 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

---

9. 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

---

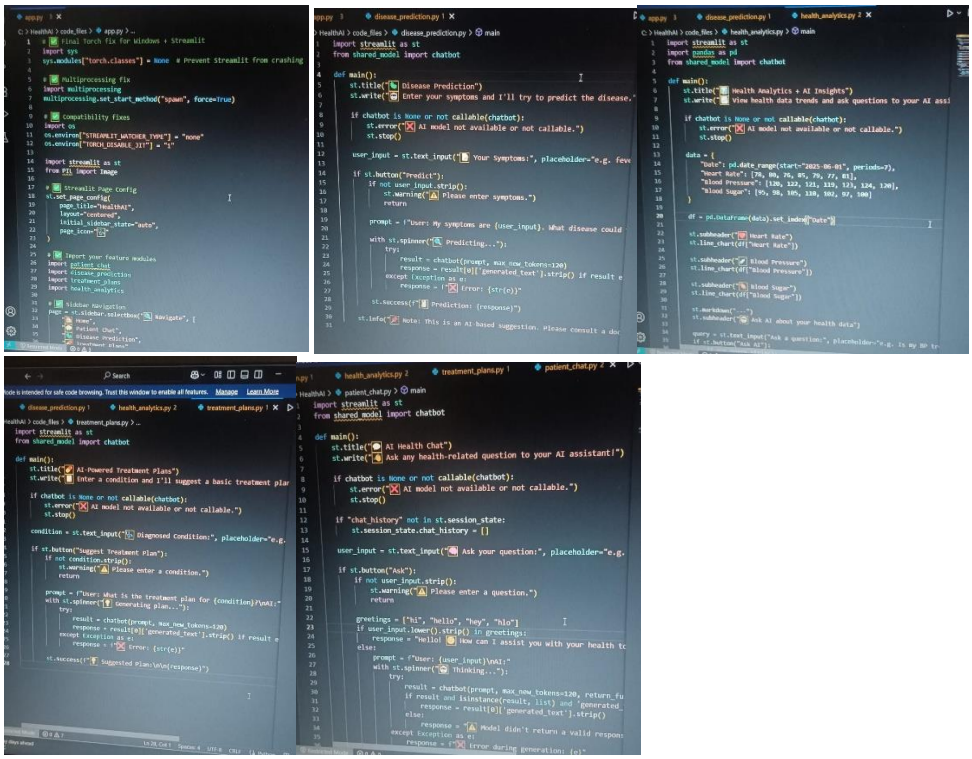
10. Testing

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

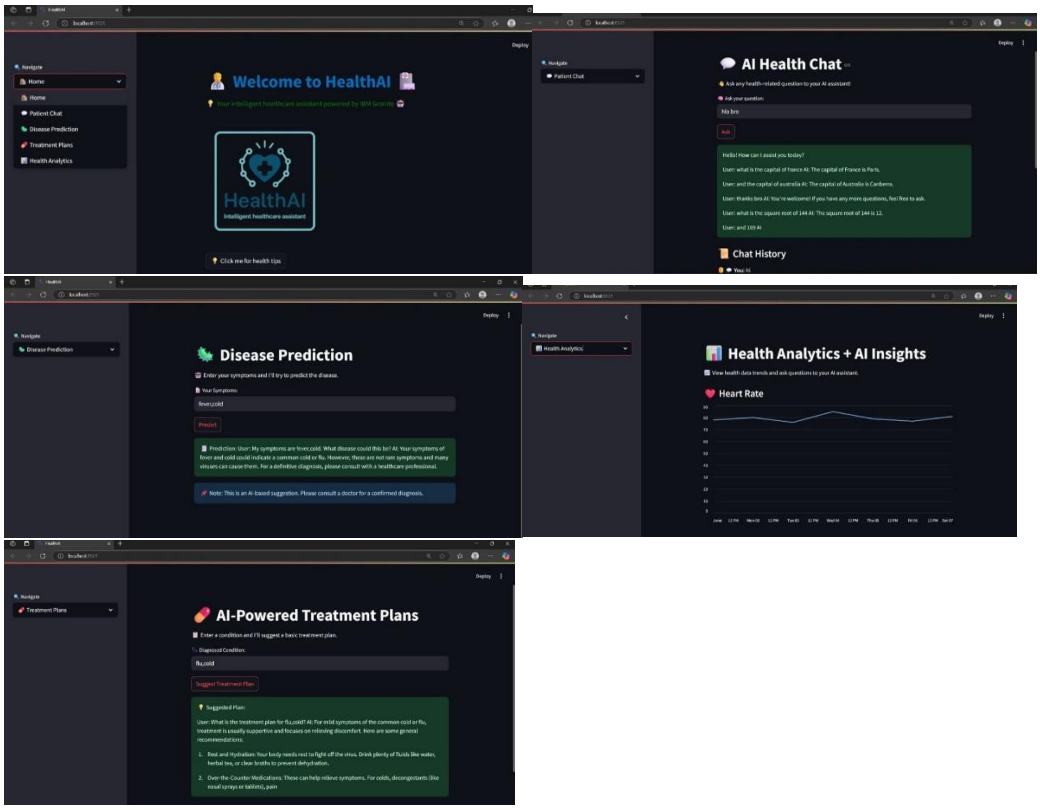
---

11. Screenshots or Demo

- 📺 [Demo Video on YouTube]
- **INPUTS (CODES) :**



- **OUTPUT :**

- 

---

12. 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)

---

13. 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
