HealthAI Project Documentation

1. Introduction

• **Project Title:** Health AI: Intelligent Healthcare Assistant Using IBM Granite

• **Team Members:**

- Amsavalli V
- Abinaya V
- Giritha A

2. Project Overview

• Purpose:

The purpose of HealthAI is to harness IBM Watson Machine Learning and Generative AI to provide intelligent healthcare assistance. The system delivers accurate medical insights, predicts diseases, recommends treatment plans, and visualizes patient health analytics. By leveraging IBM Granite-13B Instruct v2, the platform improves healthcare accessibility, empowers users to make informed decisions, and enhances the patient experience.

• Features:

Patient Chat
Key Point: Conversational healthcare guidance
Functionality: Provides natural language interaction for health-related questions with AI-generated responses.

Disease Prediction
Key Point: Symptom-based diagnosis

Functionality: Analyzes user symptoms and health data to suggest possible conditions with likelihoods and next steps.

Treatment Plans

Key Point: Personalized medical advice

Functionality: Generates tailored treatment recommendations including medications, lifestyle changes, and follow-up testing.

Health Analytics

Key Point: Data-driven insights

Functionality: Visualizes patient health metrics (vital signs, trends) and provides AI-generated insights.

Secure API Management

Key Point: Data safety

Functionality: Ensures responsible handling of healthcare data with API key protection.

3. Architecture

Frontend (Streamlit): Provides an interactive interface for chat, prediction, treatment, and analytics with intuitive dashboards and visualizations.

Backend (FastAPI): Manages requests, communicates with IBM Granite, and handles core healthcare functionalities.

LLM Integration (IBM Watsonx Granite): IBM Granite-13B Instruct v2 model processes natural language queries and generates medical insights.

Data Visualization (Plotly, Pandas): Displays patient metrics and trends in interactive graphs.

ML Modules: Support disease prediction and health analytics using patient-reported data.

Error! Hyperlink reference not valid.4. Setup Instructions

Prerequisites:

- Python 3.9+
- pip & virtual environment
- IBM Watsonx API key
- Streamlit, Plotly, Pandas installed

Installation Process:
• Clone the repository
• Install dependencies from requirements.txt
• Configure credentials in .env file
• Run backend server (FastAPI)
• Launch frontend via Streamlit
• Upload health data and interact with modules

5. Folder Structure

app/ – FastAPI backend logic including chat, prediction, treatment, and analytics modules

ui/ – Streamlit frontend components for dashboards and health visualization

app.py – Entry script to run the main Streamlit interface

granite_llm.py – Handles IBM Granite model interactions

prediction_engine.py – Implements disease prediction logic

treatment_planner.py – Generates treatment recommendations

health_dashboard.py – Visualizes health data and insights

6. Running the Application

- Launch FastAPI server
- Run Streamlit dashboard
- Navigate via sidebar
- Input symptoms, request treatment plans, or view analytics Receive
- AI-generated responses in real-time

7. API Documentation

POST /chat/ask – Submit health-related queries

POST /disease/predict – Submit symptoms for disease prediction

POST /treatment/generate – Generate personalized treatment plan

GET /analytics/view – Retrieve health metrics and visualizations

POST /upload-data – Upload patient health data

8. Authentication

- Token-based authentication (JWT / API Keys)
- OAuth2 with IBM Cloud
- Role-based access (patient, doctor, researcher)
- Secure API credential handling via .env file

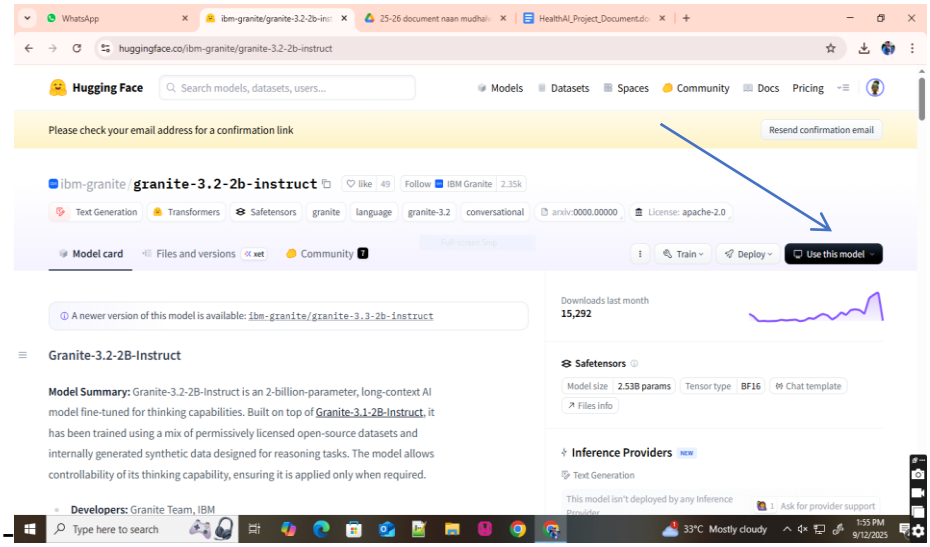
9. User Interface

- Sidebar navigation
- Chat interface for Patient Chat
- Symptom input and prediction display
- Treatment recommendation output
- Interactive health dashboard with visualizations

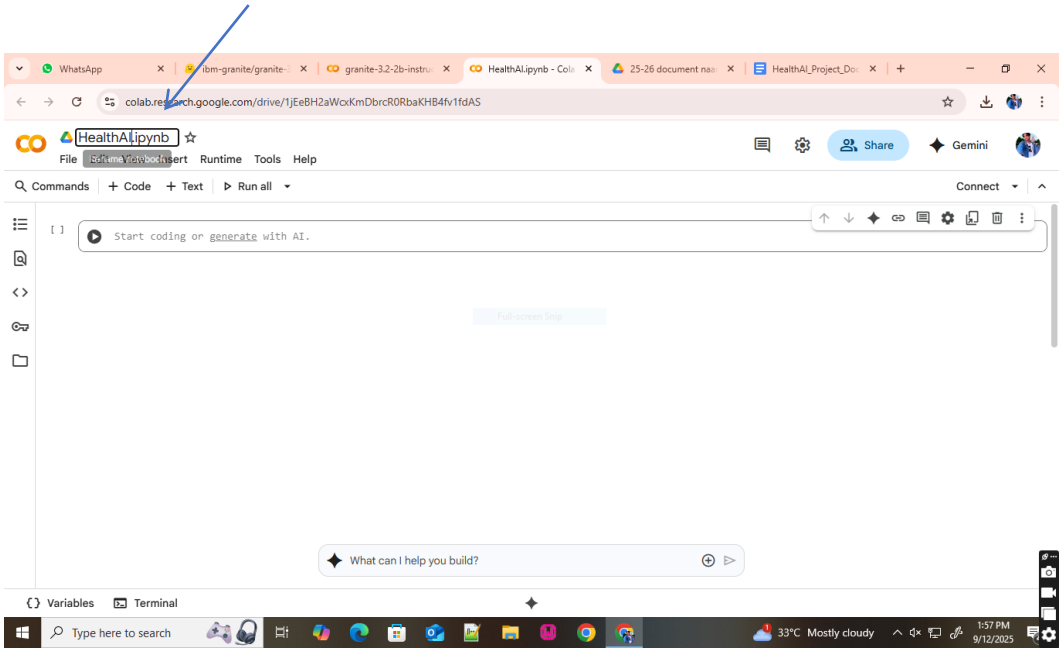
10. Testing

- Unit Testing: For AI prompting and data utilities
- API Testing: Swagger UI and Postman
- Manual Testing: For chat, prediction, and visualization consistency
- Edge Case Handling: Invalid inputs, missing symptoms, large datasets

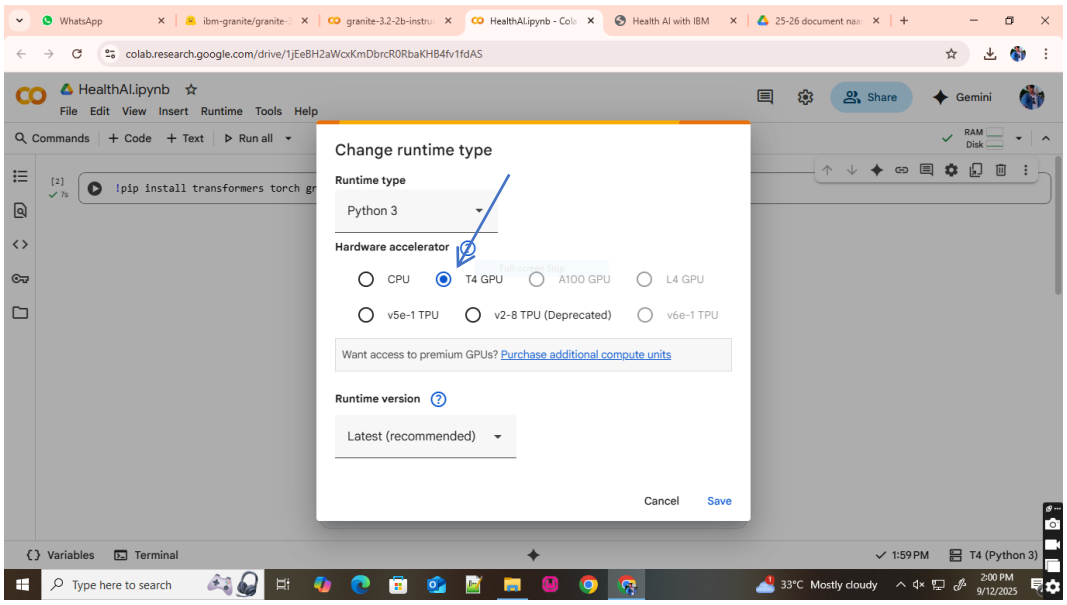
11. Screenshots



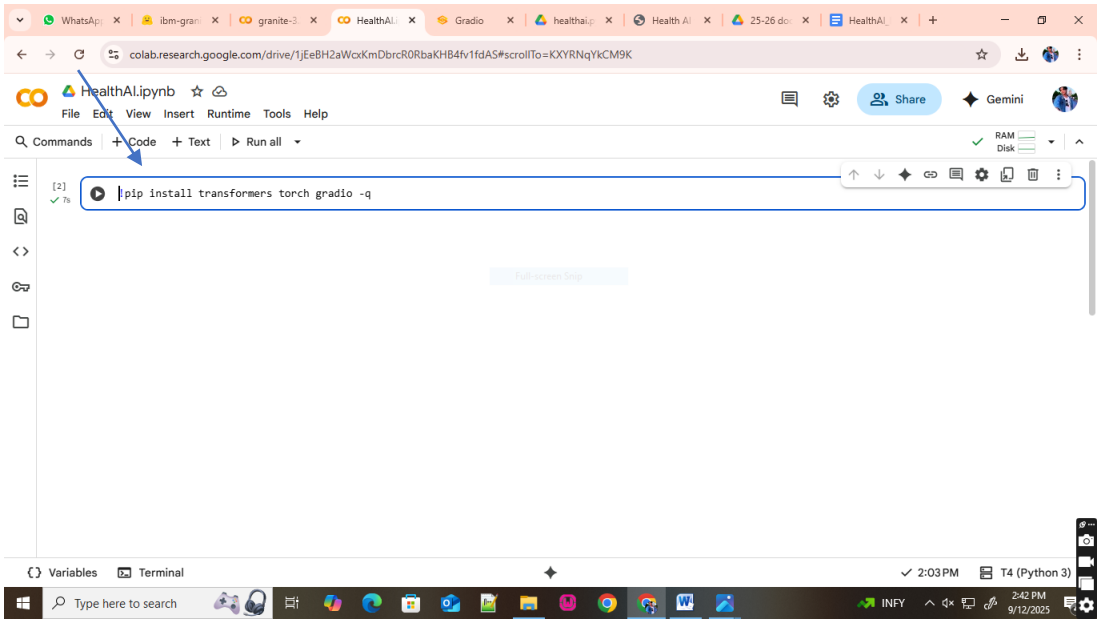
- Click on the first link (Google Colab), then click on “Files” and then “Open Notebook”.



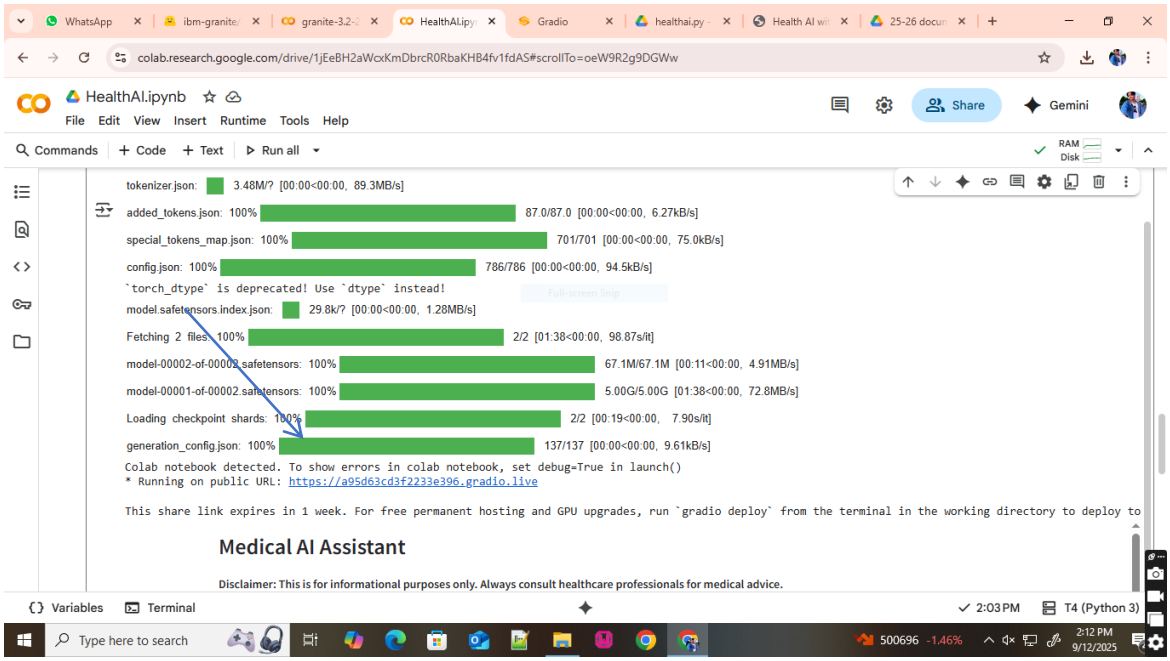
- Change the title of the notebook “Untitled” to “Health AI”.



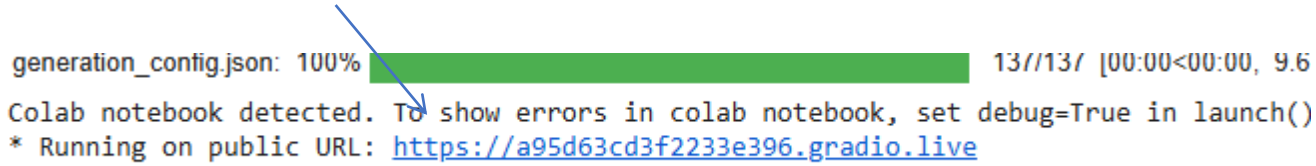
- Then click on “Runtime”, then go to “Change Runtime Type”.
- Choose “T4 GPU” and click on “Save”

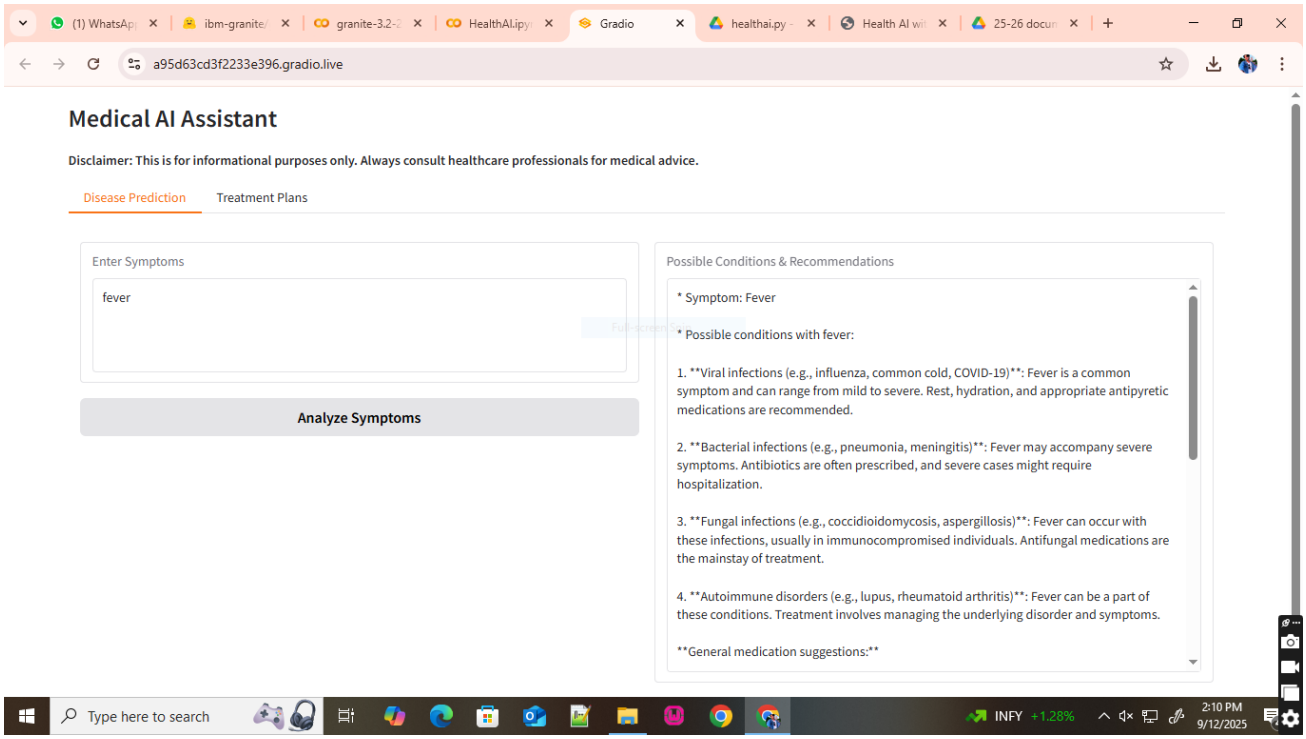


● Then run this command in the first cell "!pip install transformers torch gradio -q". To install the required libraries to run our application.



● Click on the URl to open the Gradio Application click on the link.





- You can View the Application is running in the other tab.

12. Known Is

- Limited coverage of rare medical conditions
- Requires stable internet for real-time AI queries
- Dependent on IBM Watson API quota

13. Future Enhancements

- Integration with wearable health devices
- Expanded medical condition coverage
- Doctor-verified treatment plans
- Multi-language support
- Advanced anomaly detection in patient data