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.

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

API Documentation is a comprehensive, technical manual that explains how to effectively use and integrate with an **Application Programming Interface (API)**. It provides **instructions, examples, reference materials, and explanations** for developers who want to interact with an API.

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

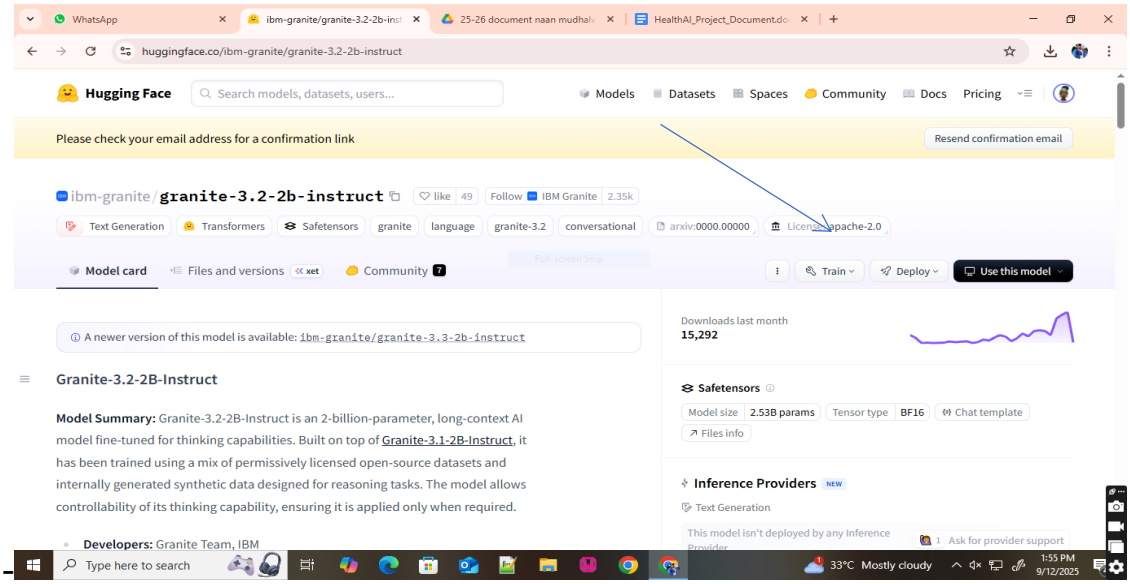
- Sidebar navigation
- Chat interface for Patient Chat
- Symptom input and prediction display
- Treatment recommendation output
- Interactive health dashboard with visualizations

10. Testing

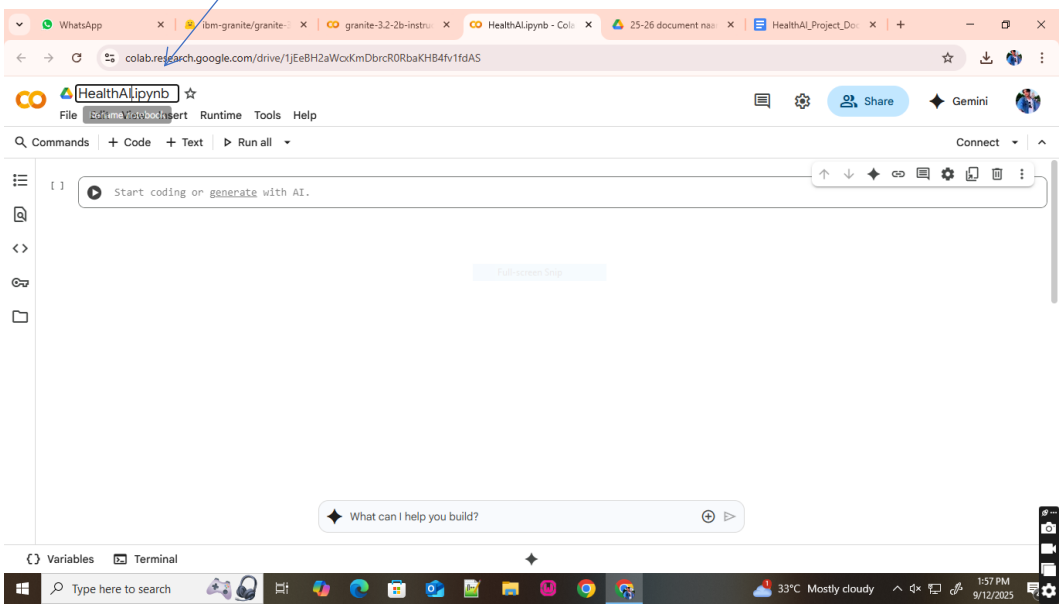
Health AI Testing refers to the process of evaluating and validating Artificial Intelligence (AI) systems designed for healthcare applications. These AI systems can range from diagnostic tools, treatment recommendation engines, patient monitoring, predictive analytics, and more.

- 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

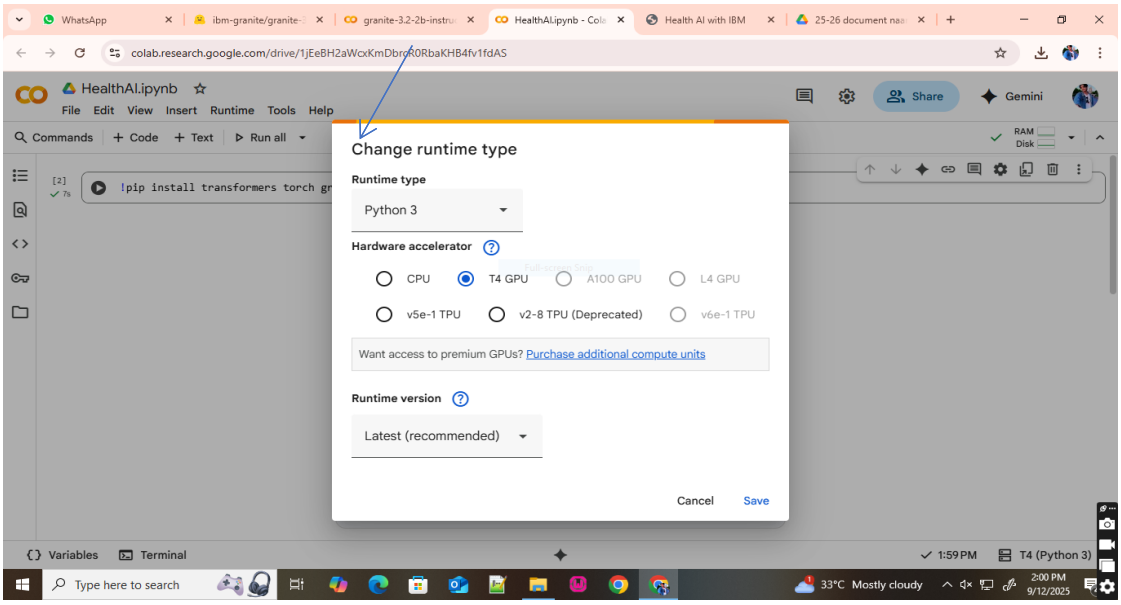
11. Screenshots



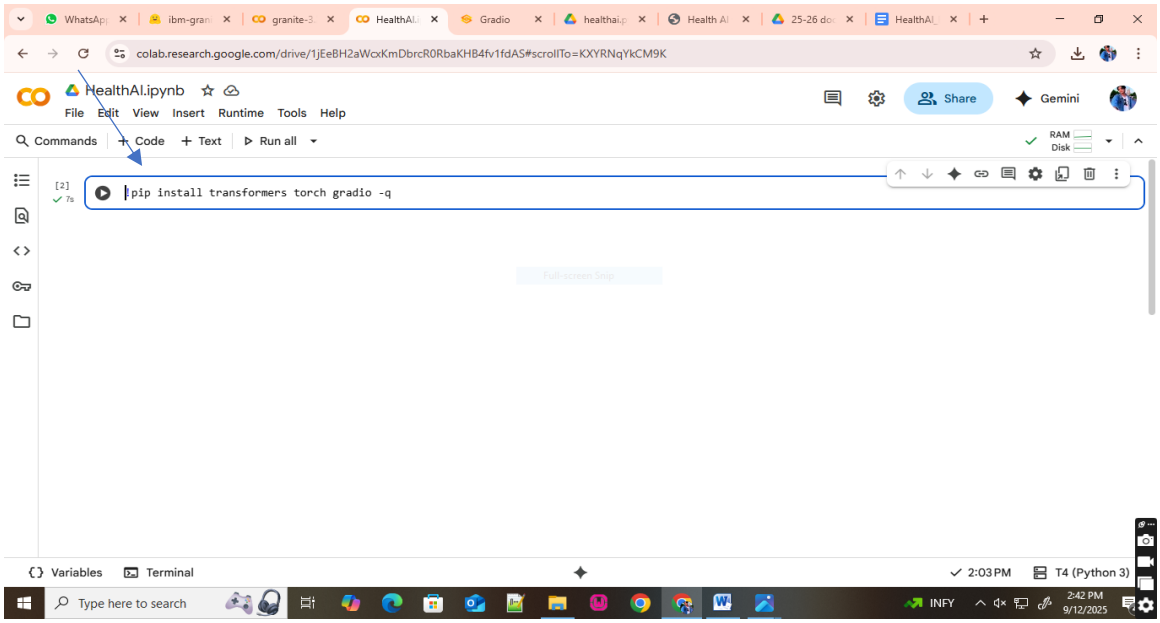
- Open Google Colab in any web browser, then click on the “Files” tab on the left sidebar, and select “Open notebook” to browse or upload your notebook.



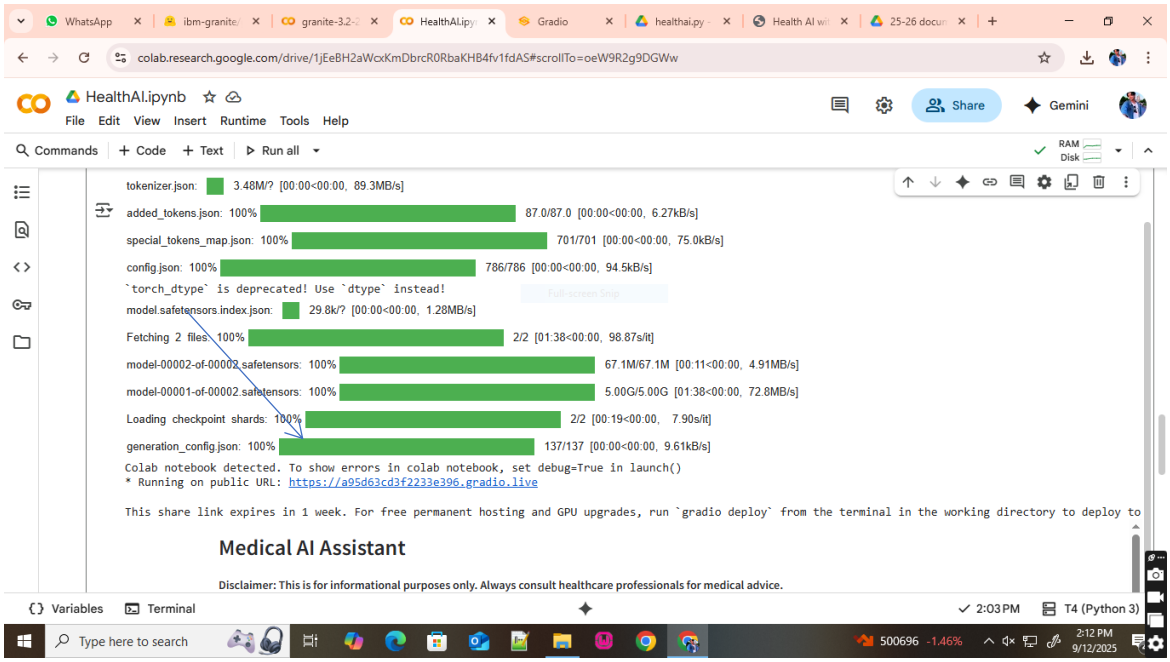
- Open Google Colab in any web browser, then click on the “Files” tab on the left sidebar, and select “Open notebook” to browse or upload your notebook.



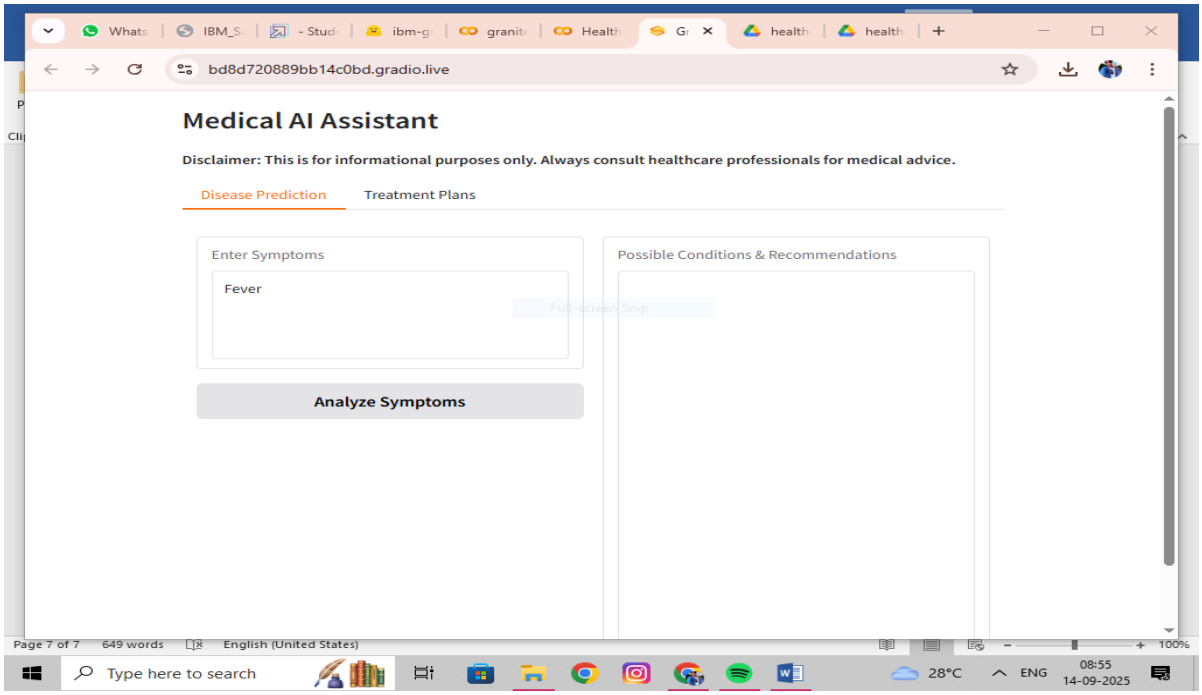
- Click on “Runtime” in the top menu, then select “Change runtime type”. In the hardware accelerator dropdown, choose “T4 GPU”, and click “Save” to apply the changes.

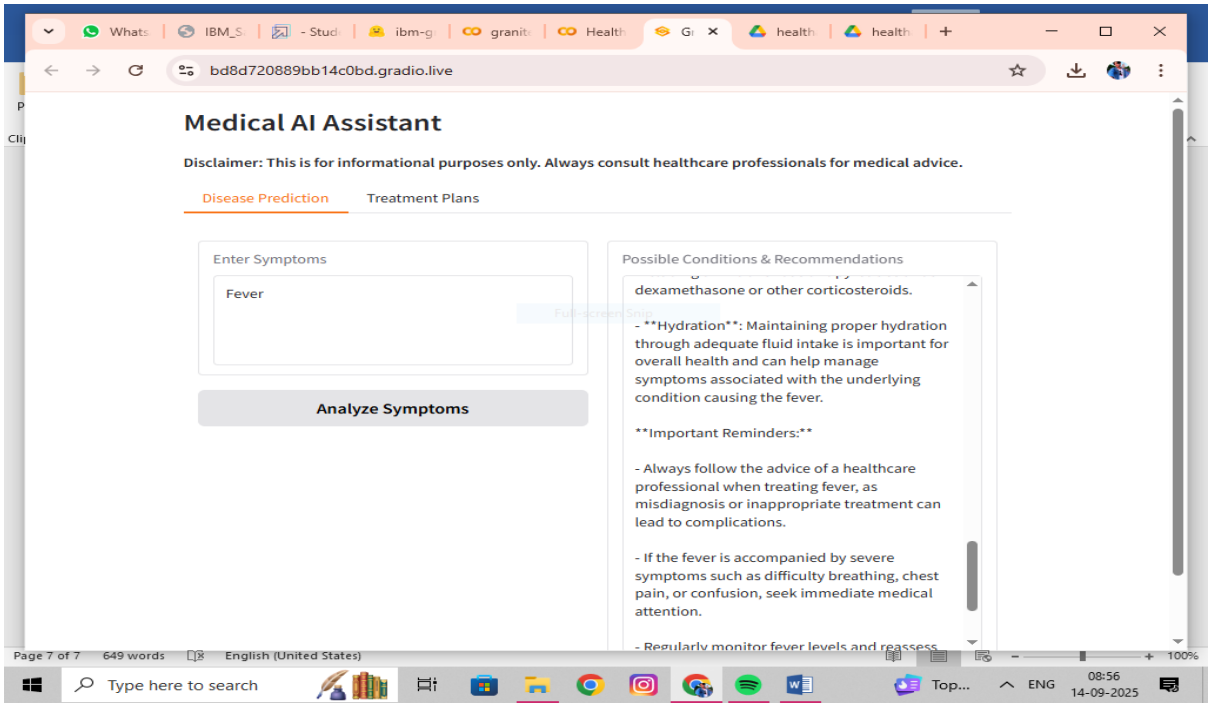


- Then, in the first code cell, run the following command to install the required libraries for the application: `!pip install transformers torch gradio -q`.

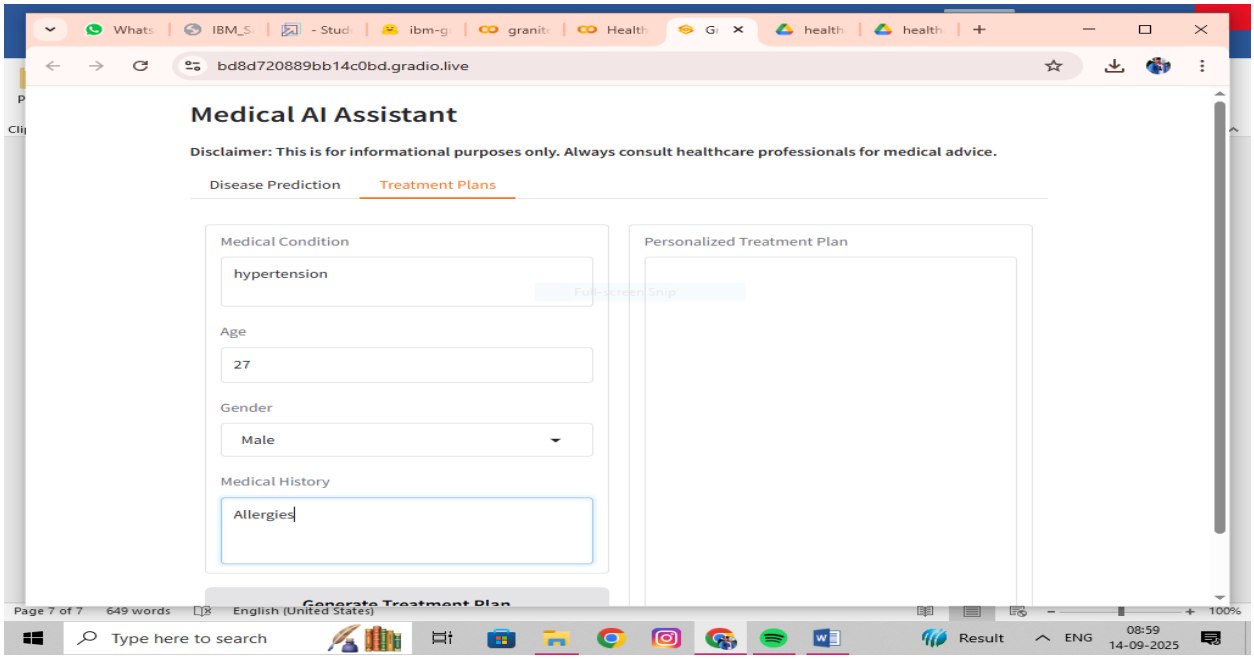


- Click on the URL generated in the output to open the Gradio application

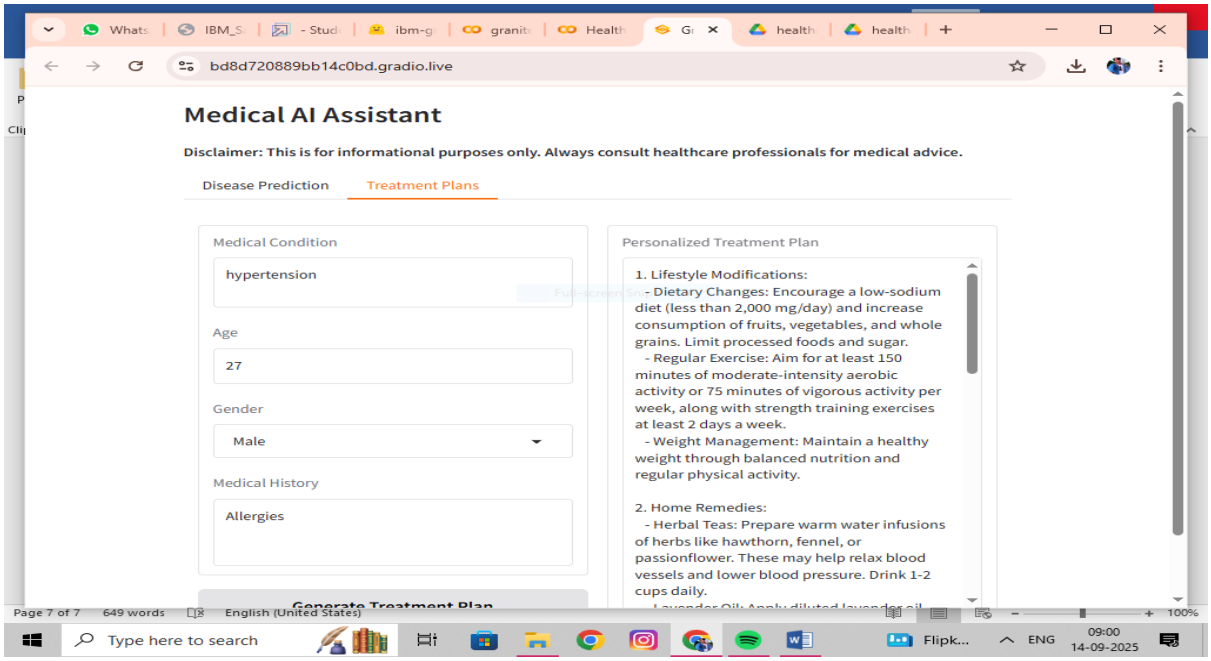




- You can view the application running in a new browser tab.



.



- Finally You can view the treatment plans running in a new browser tab.

12. System Limitations and Constraints

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

13. Future Enhancements

"**Future Enhancements**" refers to the planned or potential improvements, additions, or upgrades that can be made to a system, product, project, or service after its initial release. These enhancements are typically based on feedback, performance evaluations, market trends, or technological advancements.

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