**Project Documentation – Health AI Assistant**

**1. Introduction**

* **Project Title**: Health AI Assistant
* **Team Members**:  
  • Member : Tamilselvi D  
  • Member : Rajarajeswari S  
  • Member : Saidhanya J  
  • Member : Yuvashri D

**2. Project Overview**

**Purpose:**  
The Health AI Assistant is designed to provide users with quick, AI-driven medical insights for symptoms, conditions, and general wellness recommendations. It is not a replacement for doctors, but an **informational tool** that promotes awareness, encourages early consultation, and provides users with personalized guidance.

By leveraging AI and natural language models, the assistant can:

* Predict possible medical conditions based on symptoms
* Suggest general treatment and home remedies
* Provide personalized treatment plan drafts based on patient details

**Important**: All outputs emphasize the need for professional medical consultation.

**Features:**

* **Conversational Symptom Analysis**
  + Accepts symptoms in plain text and returns possible conditions.
* **Treatment Plan Generator**
  + Creates a customized treatment plan with home remedies and lifestyle tips based on user info (age, gender, history).
* **Disclaimer Integration**
  + Ensures every response reminds users to consult a healthcare professional.
* **Gradio-based UI**
  + Clean, accessible web interface for disease prediction and treatment planning.
* **LLM Integration**
  + Uses IBM Watsonx Granite (granite-3.2-2b-instruct) for high-quality medical text generation.

**3. Architecture**

**Frontend (Gradio):**

* Built with **Gradio Blocks & Tabs**.
* Provides two main modules:
  + *Disease Prediction* (symptom input → conditions & recommendations)
  + *Treatment Plans* (condition + patient details → personalized plan).
* Simple, user-friendly interface with disclaimers.

**Backend (Python + Transformers):**

* Uses Hugging Face Transformers with IBM Watsonx Granite model.
* Handles:
  + Symptom → prompt → AI response
  + Condition + patient details → prompt → AI-generated treatment plan

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

* Provides natural language understanding and generation.
* Prompts carefully structured to:
  + Return medical insights
  + Enforce safety disclaimers

**4. Setup Instruction**

**Prerequisites:**

* Python 3.9+
* pip and venv
* GPU (optional, for faster inference)
* Internet access (to fetch models)

**Run the Application:**

python health\_ai.py

**6. Running the Application**

1. Launch the Gradio interface (python health\_ai.py).
2. Open the given localhost/Share link.
3. Navigate between **Disease Prediction** and **Treatment Plan** tabs.
4. Enter symptoms or patient details.
5. Receive AI-generated suggestions with medical disclaimer.

**7. API Documentation**

If integrated with FastAPI, the following APIs could be exposed:

* **POST /predict-symptoms** – Returns conditions & recommendations
* **POST /generate-plan** – Returns personalized treatment plan.

**8. Authentication**

Currently runs in **open mode** for demo. Future secure deployments can add:

* API key-based access
* Role-based usage (doctor vs. patient mode)

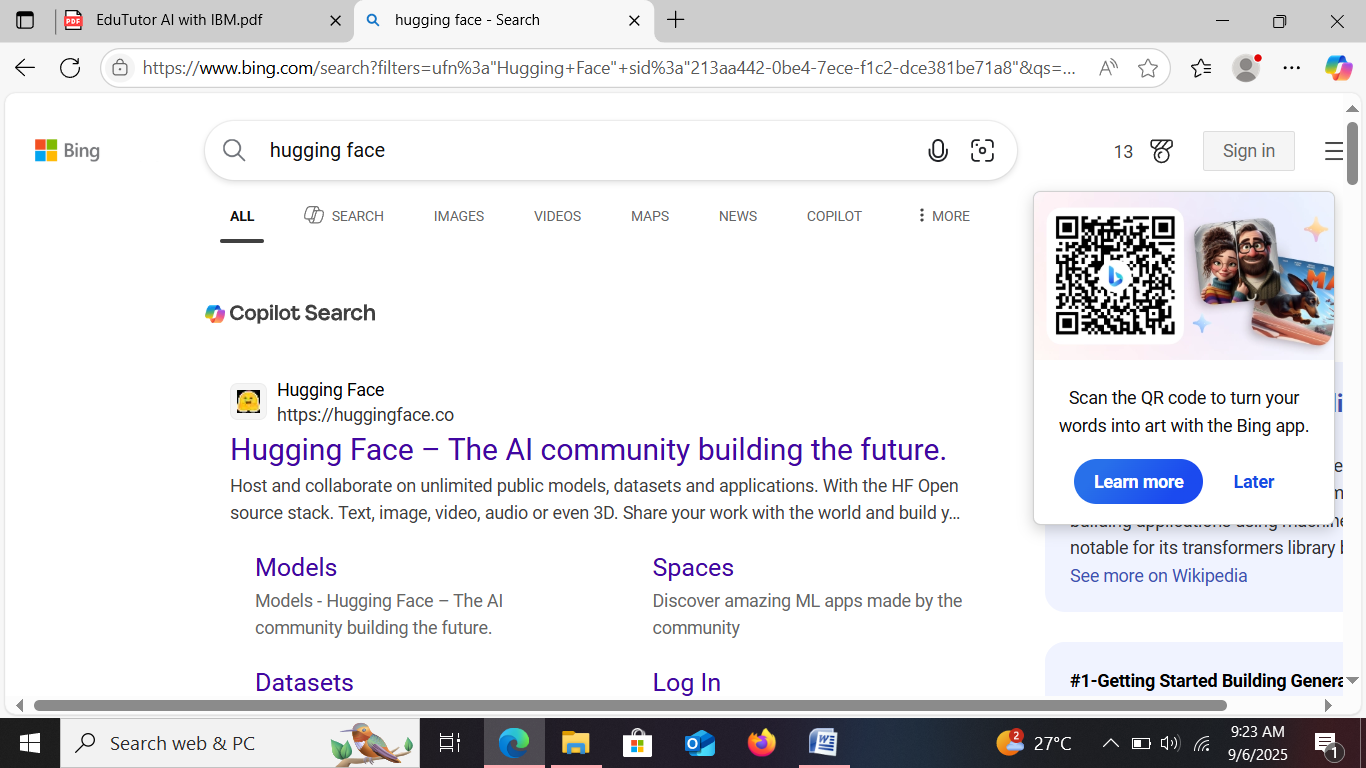
**9. User Interface**

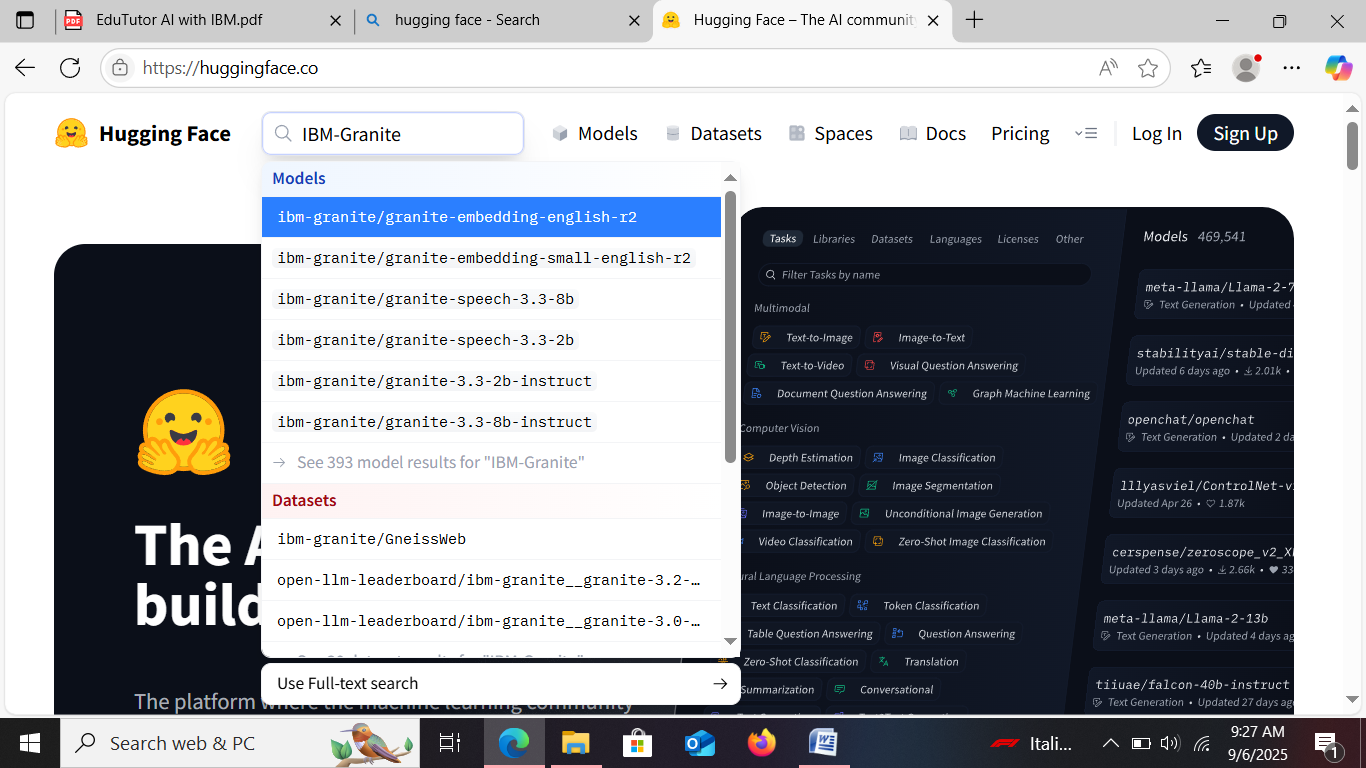
* **Tabs:** Disease Prediction | Treatment Plans
* **Inputs:** Textboxes, dropdowns, number inputs
* **Outputs:** Multi-line text with AI response
* **Design Priority:** Minimalist, clarity-first, with disclaimers at every step

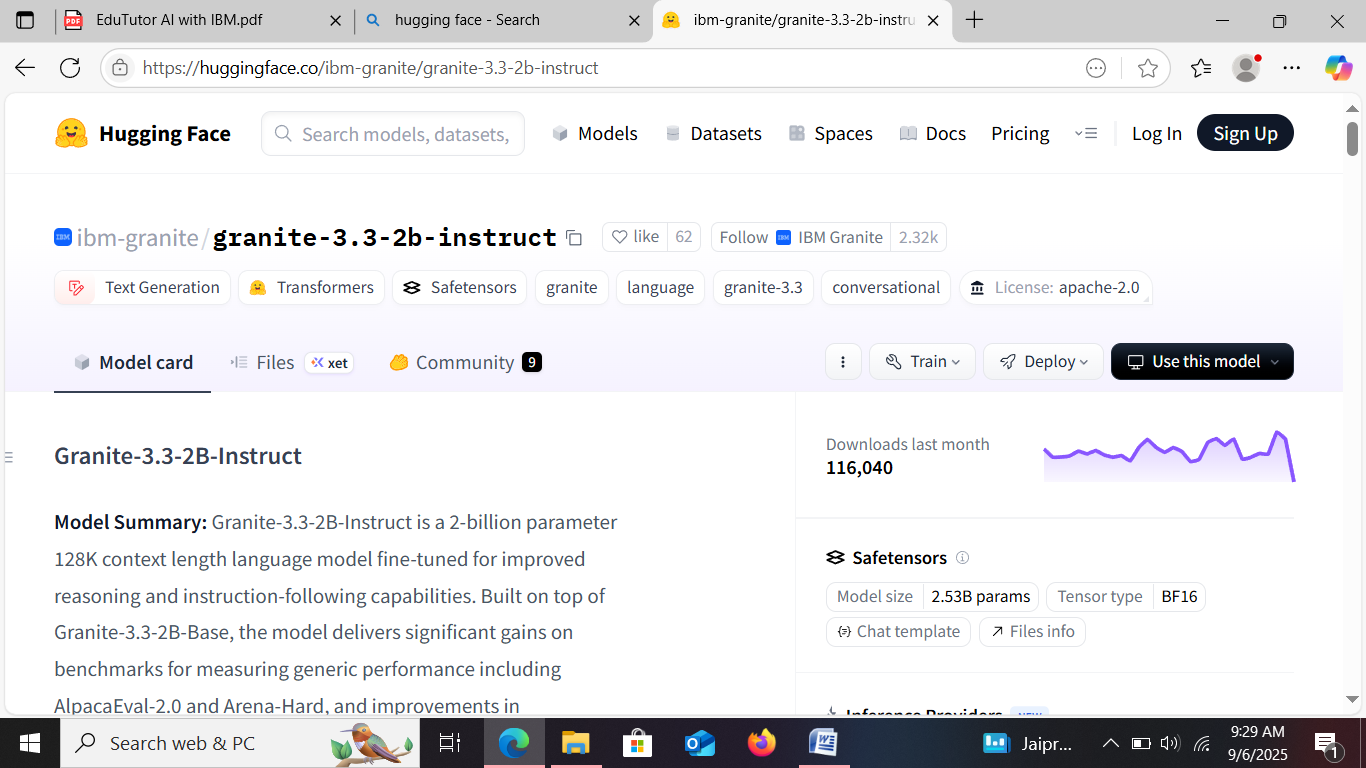
**10. Testing**

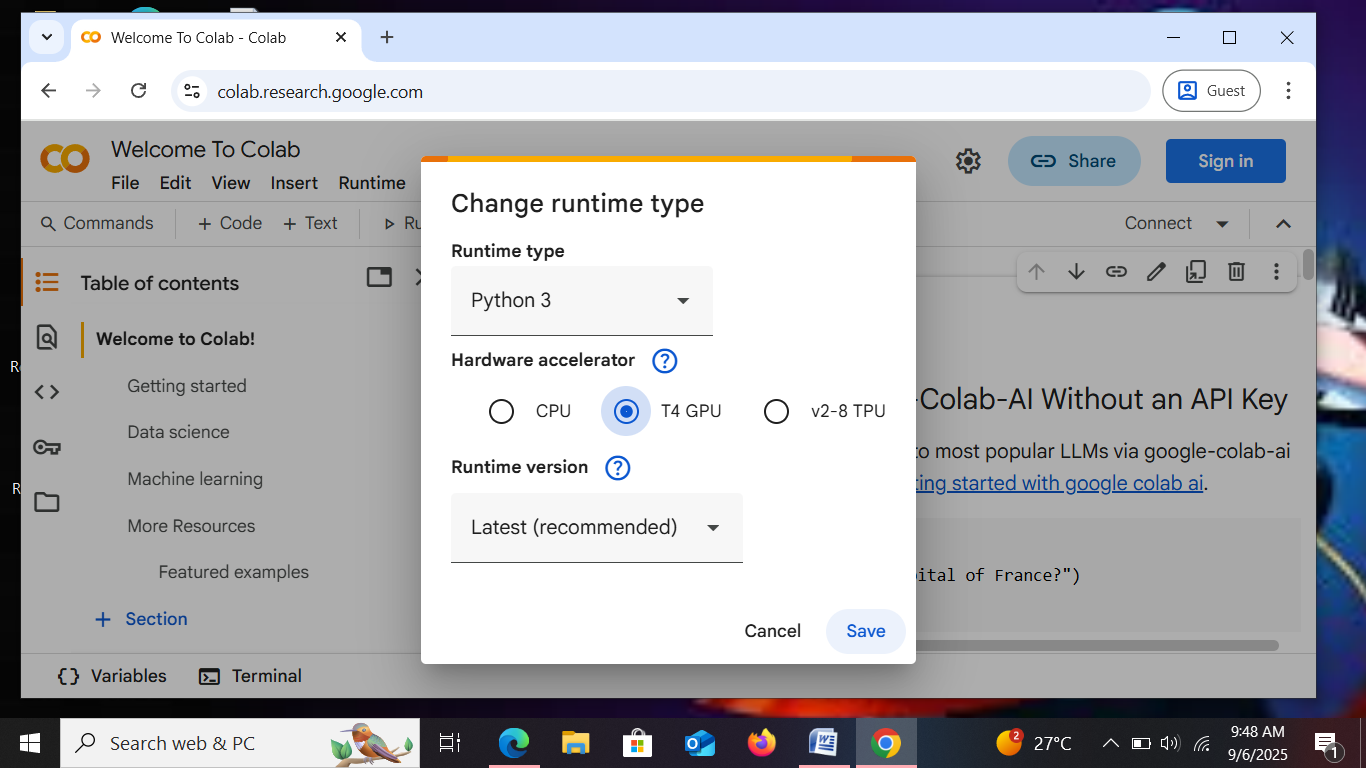
* **Unit Testing:** Prompt functions tested for valid AI responses
* **Manual Testing:** UI tested for different symptoms and conditions
* **Edge Cases:** Empty input, long symptom lists, invalid formats
* **Safety Validation:** Ensured disclaimer appears in every output

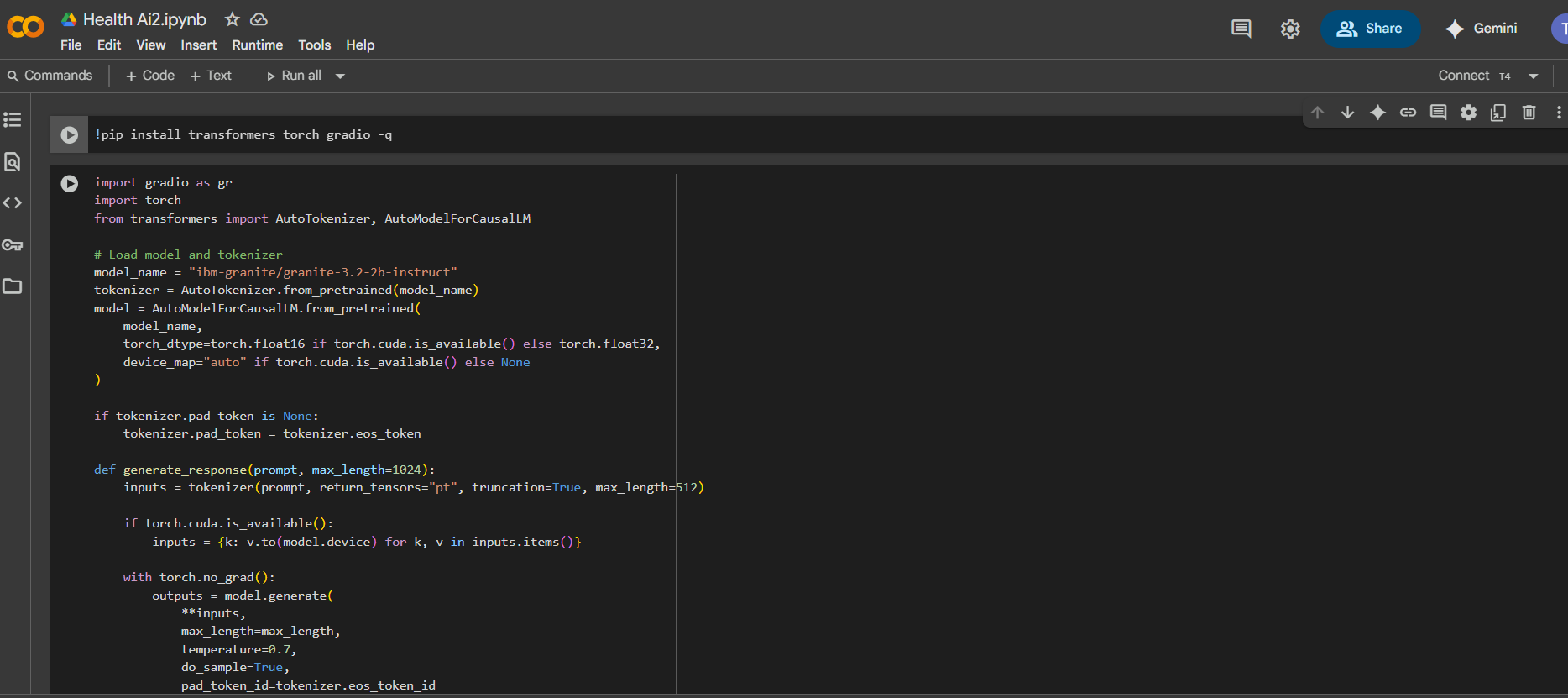
**11. Screenshots**

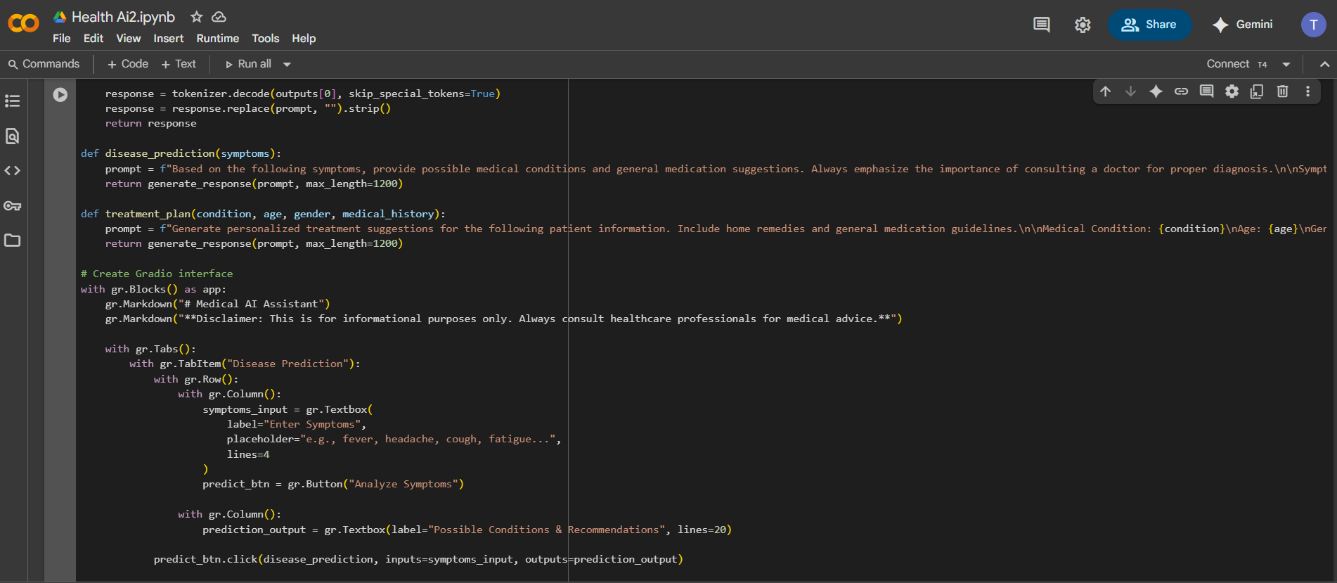


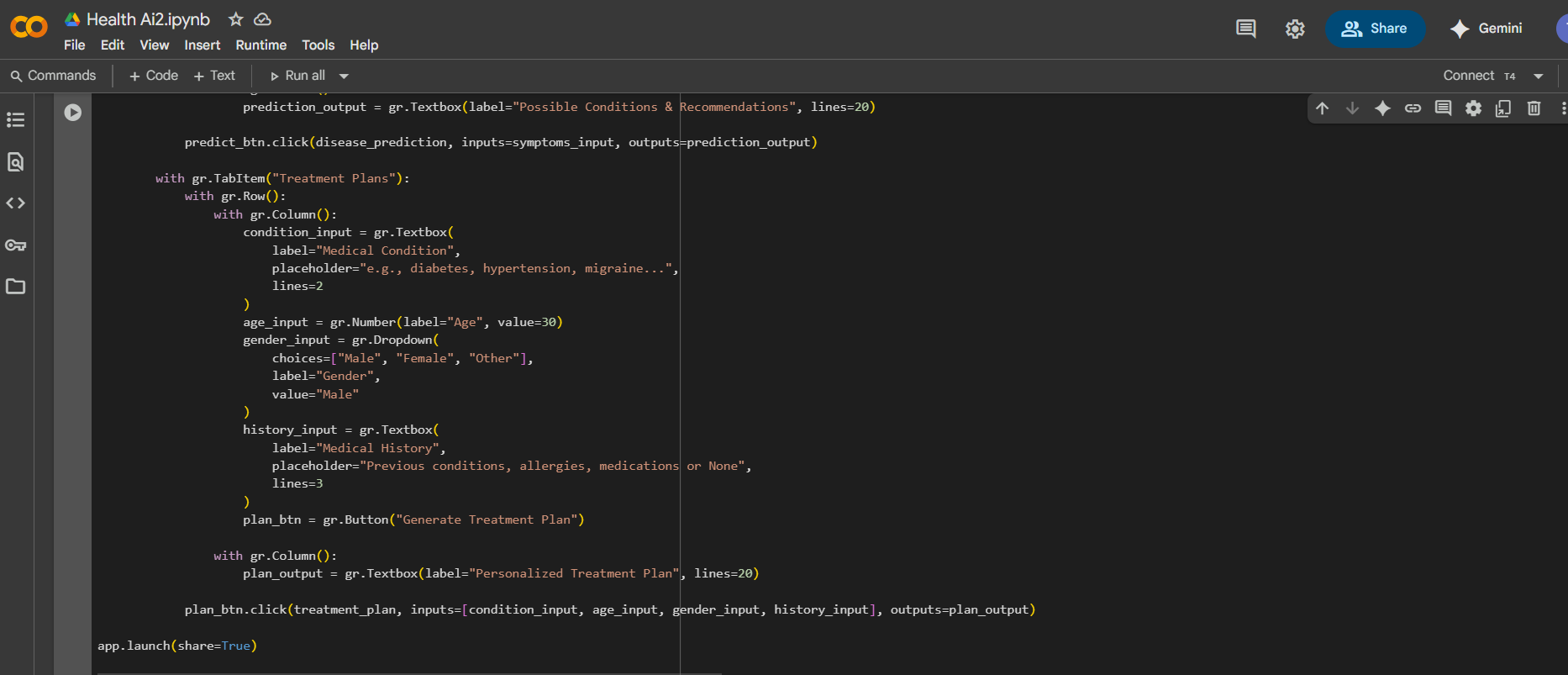
****

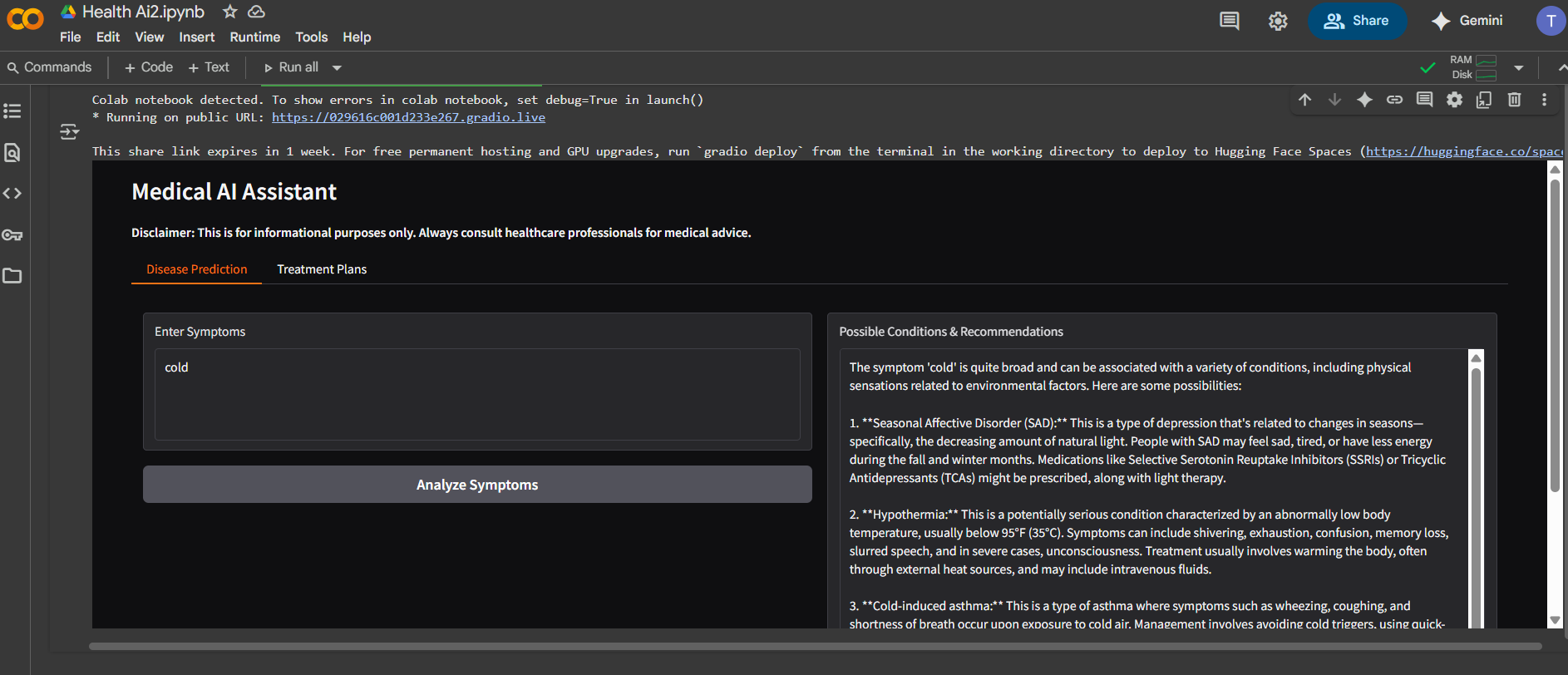
****

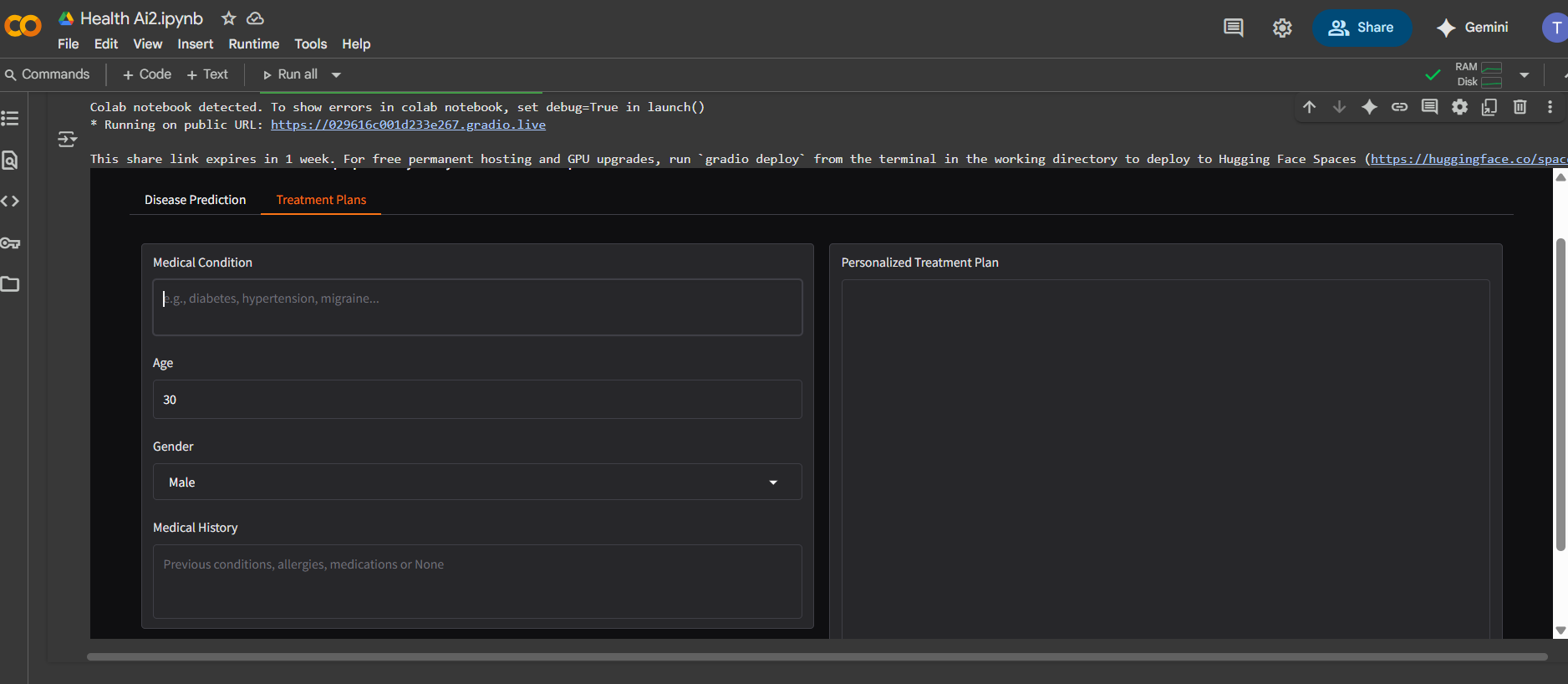




****

****





**12. Known Issues**

* Responses may vary in accuracy since model is not a certified medical system.
* No real-time medical data integration yet.
* Limited medical terminology handling for rare conditions.

**13. Future Enhancements**

* Integration with **FastAPI** for API-based deployment
* **Medical knowledge base integration** (PubMed, WHO, etc.)
* **Speech-to-text support** for accessibility
* **Secure authentication** for patient data privacy
* **Multilingual support** for non-English speakers