**Health AI: Intelligent Healthcare Assistant Using IBM Granite**

**Project Documentation**

**1. Introduction**

**Project title:** HealthAI:Intelligent Healthcare Assistant Using IBM Granite

**Team members:**

* Team member 1: SARAN RAJ K
* Team member 2: VISHNU S
* Team member 3: DAVID CHANDRU S
* Team member 4: SONIYA T

**2. Project Overview**

**Purpose:**  
The purpose of Health AI is to provide users with an intelligent, AI-driven medical assistant capable of analyzing symptoms, predicting possible diseases, and generating personalized treatment plans. By leveraging IBM’s Granite LLM and real-time AI processing, the assistant aims to deliver accurate, informative, and accessible healthcare guidance while emphasizing the importance of consulting professional doctors.

**Features:**

**Conversational Interface**

* **Key Point:** Natural language interaction
* **Functionality:** Users can input symptoms or conditions and receive AI-generated responses.

**Disease Prediction**

* **Key Point:** Symptom-based health analysis
* **Functionality:** Provides possible medical conditions and general medication suggestions based on user-provided symptoms.

**Treatment Plan Generation**

* **Key Point:** Personalized healthcare guidance
* **Functionality:** Generates treatment plans based on condition, age, gender, and medical history.

**Voice Output (Text-to-Speech)**

* **Key Point:** Audio assistance
* **Functionality:** Converts AI responses into voice using gTTS for accessibility and enhanced user experience.

**Gradio UI**

* **Key Point:** User-friendly interface
* **Functionality:** Simple tab-based dashboard with disease prediction and treatment plan modules.

**3. Architecture**

**Frontend (Gradio):**

* Tab-based web interface with two main tabs: Disease Prediction and Treatment Plans.
* Inputs: Textbox for symptoms, condition, medical history, age, gender.
* Outputs: AI response textbox and audio player for voice output.

**Backend (Python + Transformers):**

* Processes user inputs, generates AI responses using Granite LLM.
* Generates voice output using gTTS.
* Handles model loading and GPU optimization if available.

**LLM Integration (IBM Granite – Hugging Face Model):**

* Model: ibm-granite/granite-3.2-2b-instruct
* Performs natural language understanding and generation.
* Generates disease predictions and treatment plans.

**Audio Generation (gTTS):**

* Converts AI text responses into MP3 audio files.
* Plays directly in Gradio interface.

**Deployment:**

* The app can be deployed locally or shared publicly using app.launch(share=True).

**4. Setup Instructions**

**Prerequisites:**

* Python 3.9 or later
* pip for package installation
* Internet connection (for downloading model and TTS)
* GPU recommended for faster response (optional)

**Installation Process:**

1. Clone the repository.
2. Install dependencies:

pip install gradio torch transformers gtts

1. Run the Gradio app:

python app.py

1. Open the provided local URL or use the public share link.
2. Interact with the Disease Prediction and Treatment Plan modules.

**5. Folder Structure**

app.py -Main Gradio app and UI layout

model/ - Optional folder to store downloaded model weights

requirements.txt - Python dependencies

audio/ - Generated TTS audio files (doctor\_voice.mp3, treatment\_voice.mp3)

**6. Running the Application**

1. Launch the Python script app.py.
2. Gradio interface opens in the browser.
3. Navigate through tabs:
   * **Disease Prediction:** Enter symptoms → Click “Analyze Symptoms” → View AI response and doctor voice.
   * **Treatment Plans:** Enter condition, age, gender, medical history → Click “Generate Treatment Plan” → View AI response and voice.
4. Interact with AI modules in real-time; outputs are displayed as text and audio.

**7. API Documentation**

*(Note: This Gradio app runs as a local interface; no separate REST API is implemented. Optional future enhancement could add FastAPI backend.)*

* **Inputs:** Textboxes (symptoms, condition, age, gender, medical history)
* **Outputs:** Text (AI response), Audio (voice)

**8. Authentication**

* Current version runs in an open environment.
* No authentication is implemented.
* Future enhancement: add login system, role-based access, and secure API endpoints.

**9. User Interface**

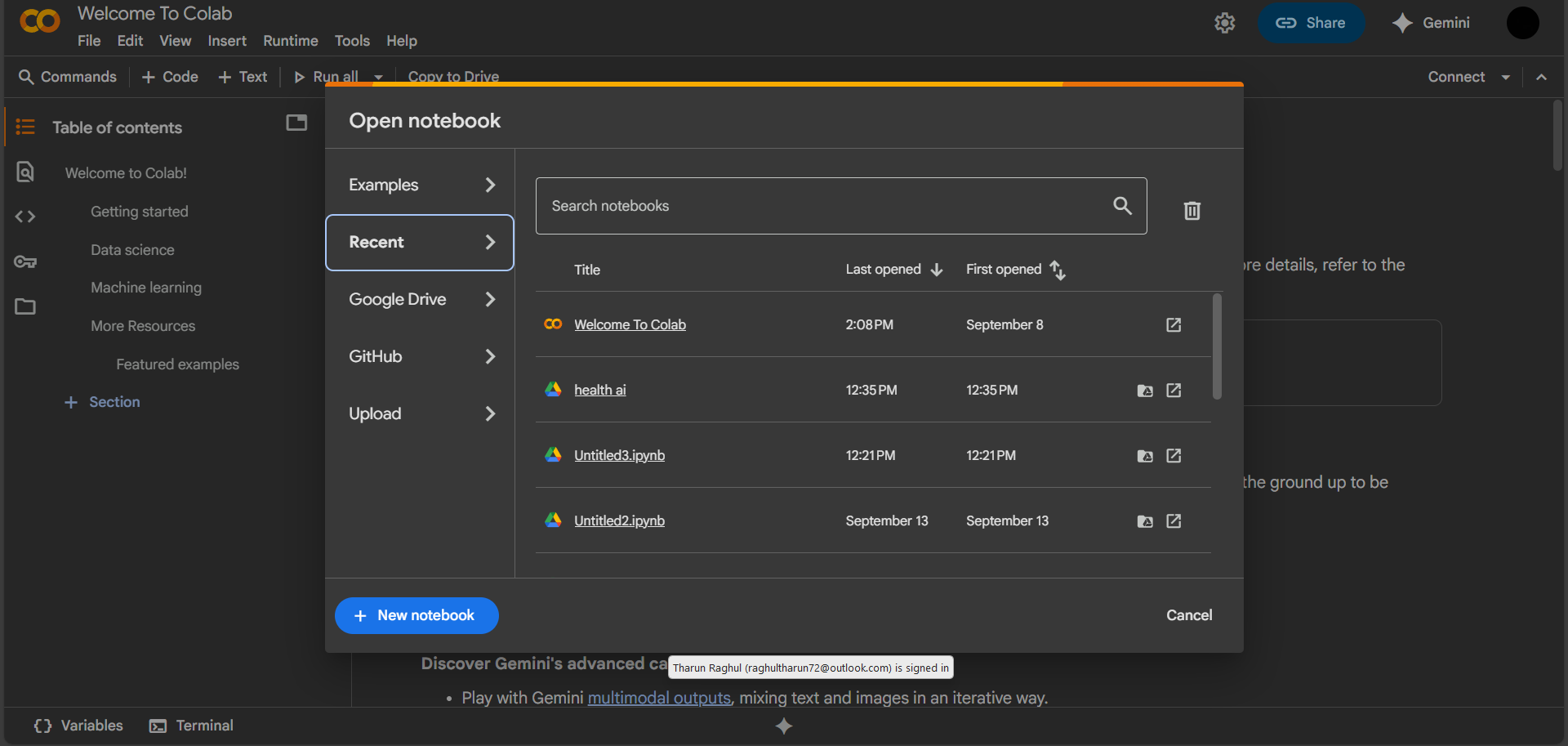
* Minimalist, tab-based interface with Gradio.
* Tabs: Disease Prediction, Treatment Plans
* Each tab contains:
  + Input fields (textbox, number, dropdown)
  + Action button (Analyze Symptoms / Generate Treatment Plan)
  + Output areas (text response + audio player)
* Clear disclaimer displayed at the top: **For informational purposes only. Always consult healthcare professionals.**

**10. Testing**

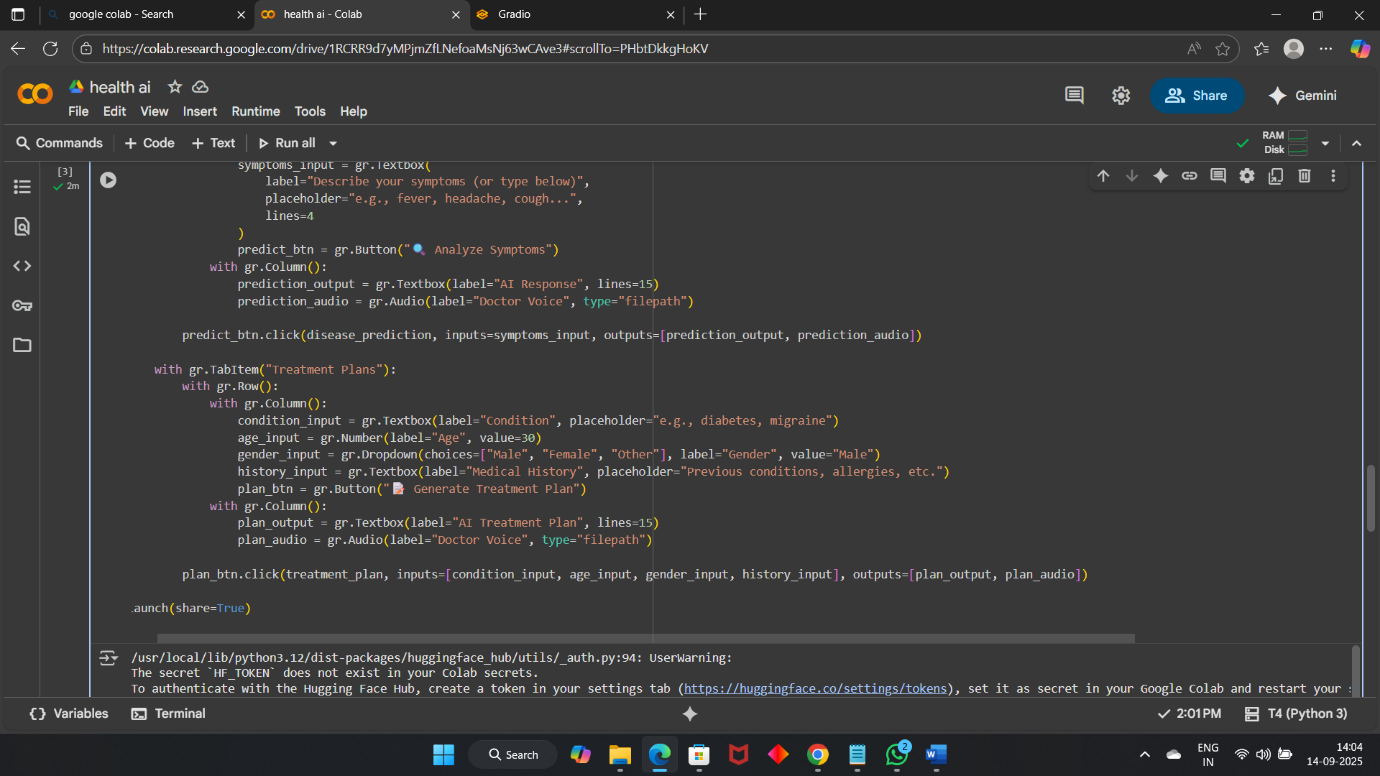
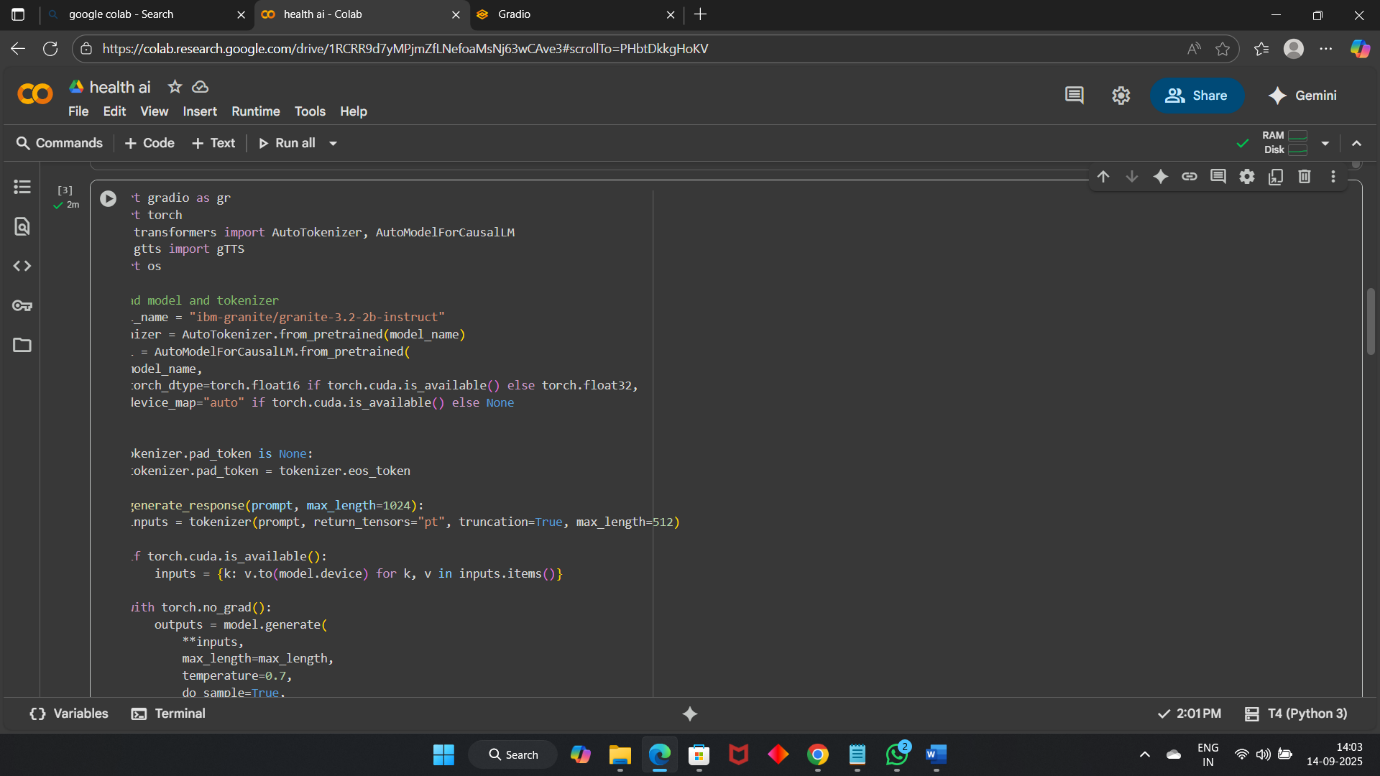
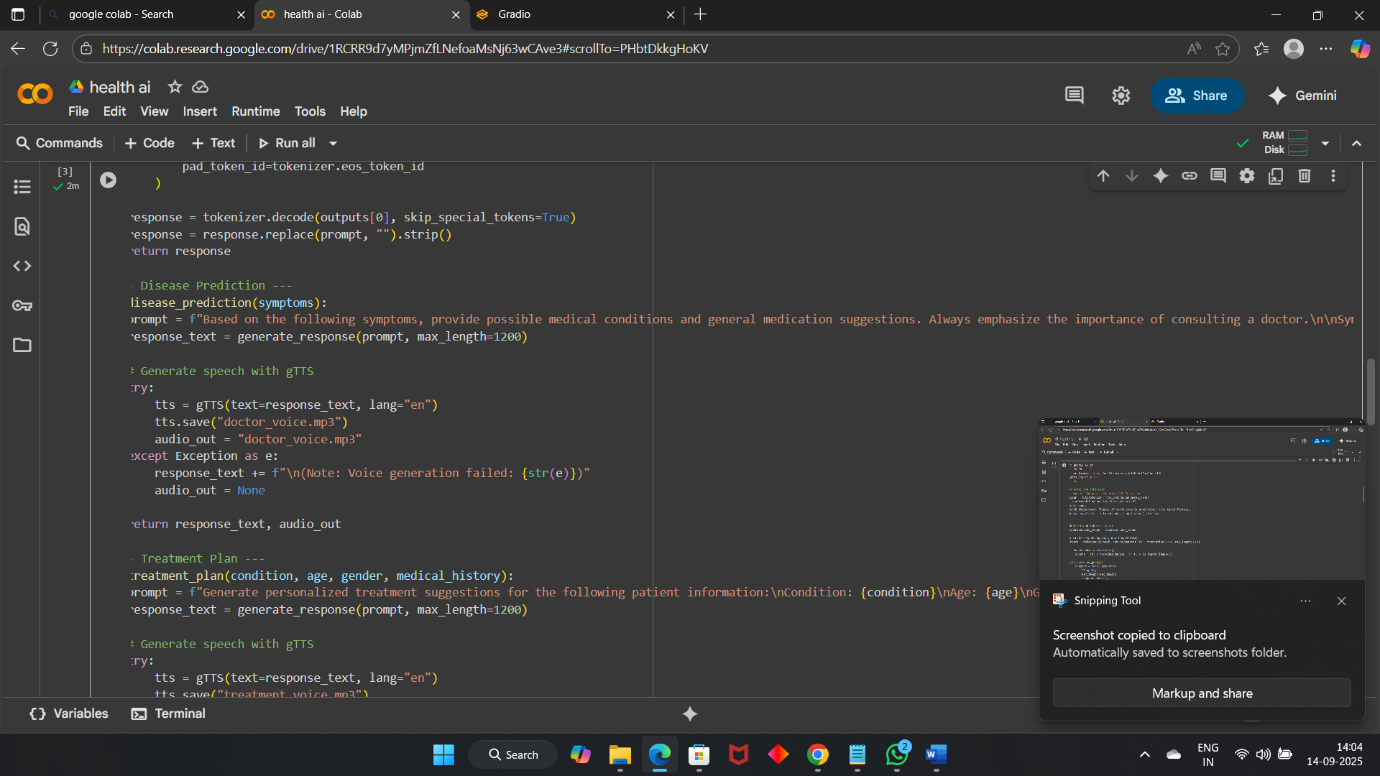
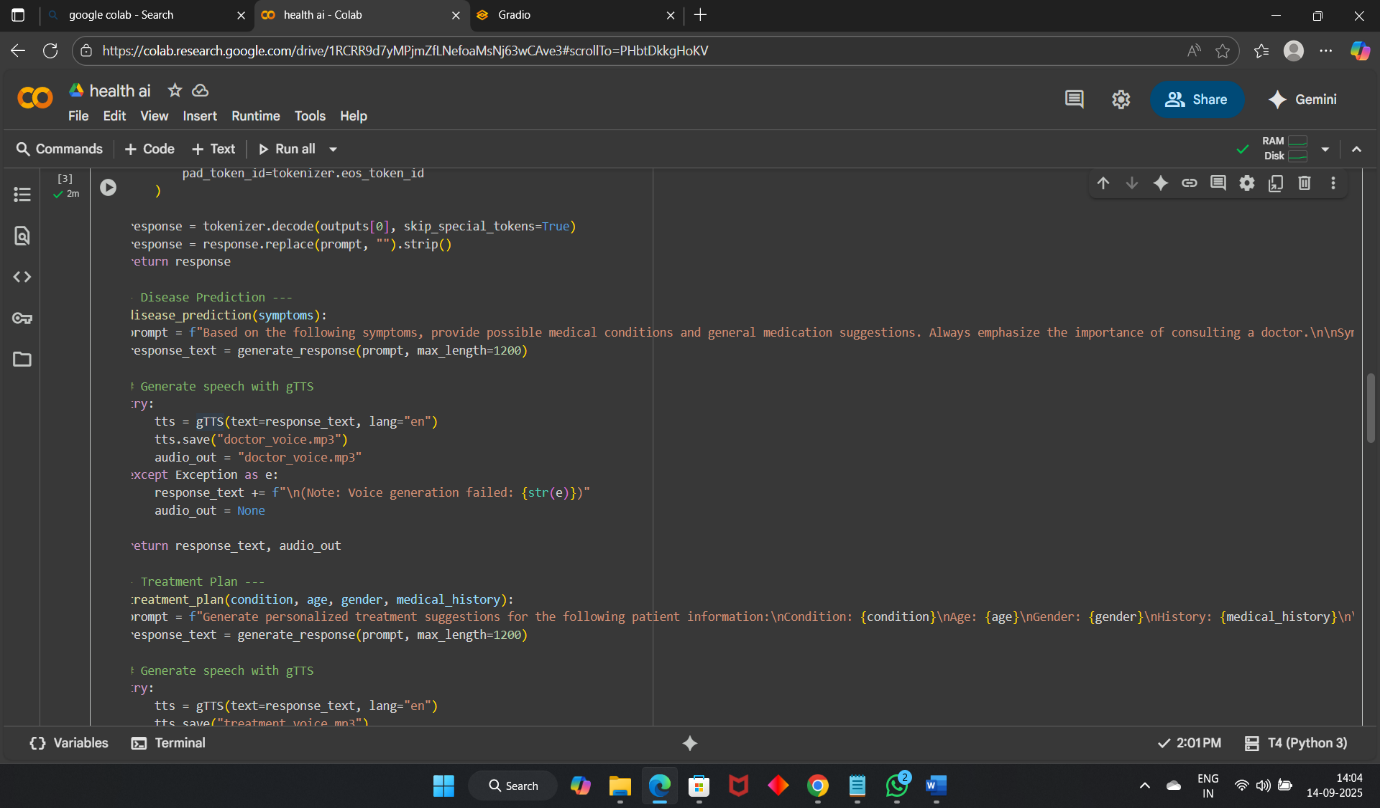
* **Unit Testing:** Validate prompt inputs and model response generation.
* **Manual Testing:** Check symptom analysis and treatment plan outputs for common cases.
* **Edge Case Handling:** Empty inputs, very long text, unusual symptom combinations.
* **Audio Testing:** Validate that gTTS generates and plays MP3 correctly.

**11. Screenshots**

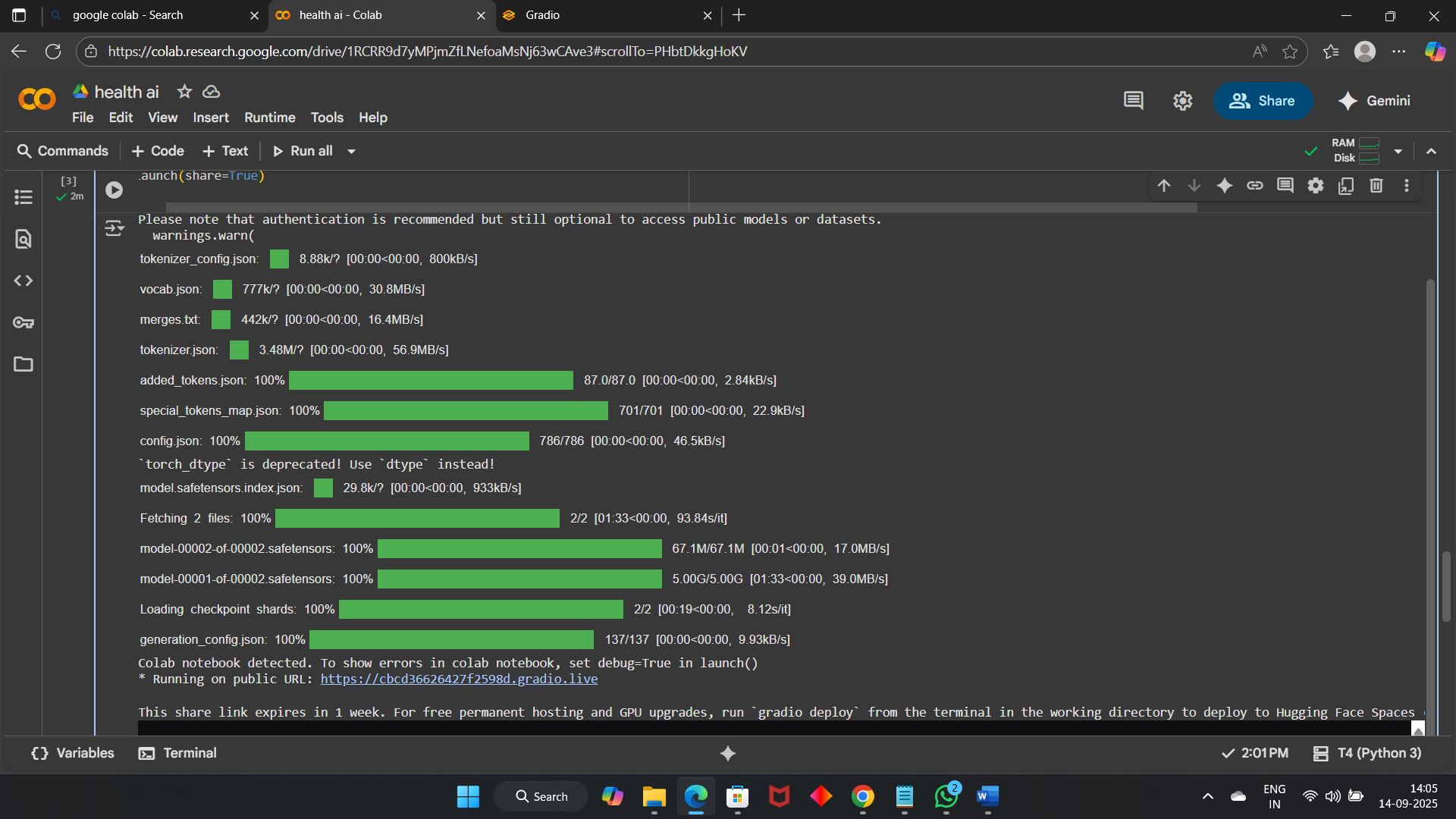
**TOOLS**



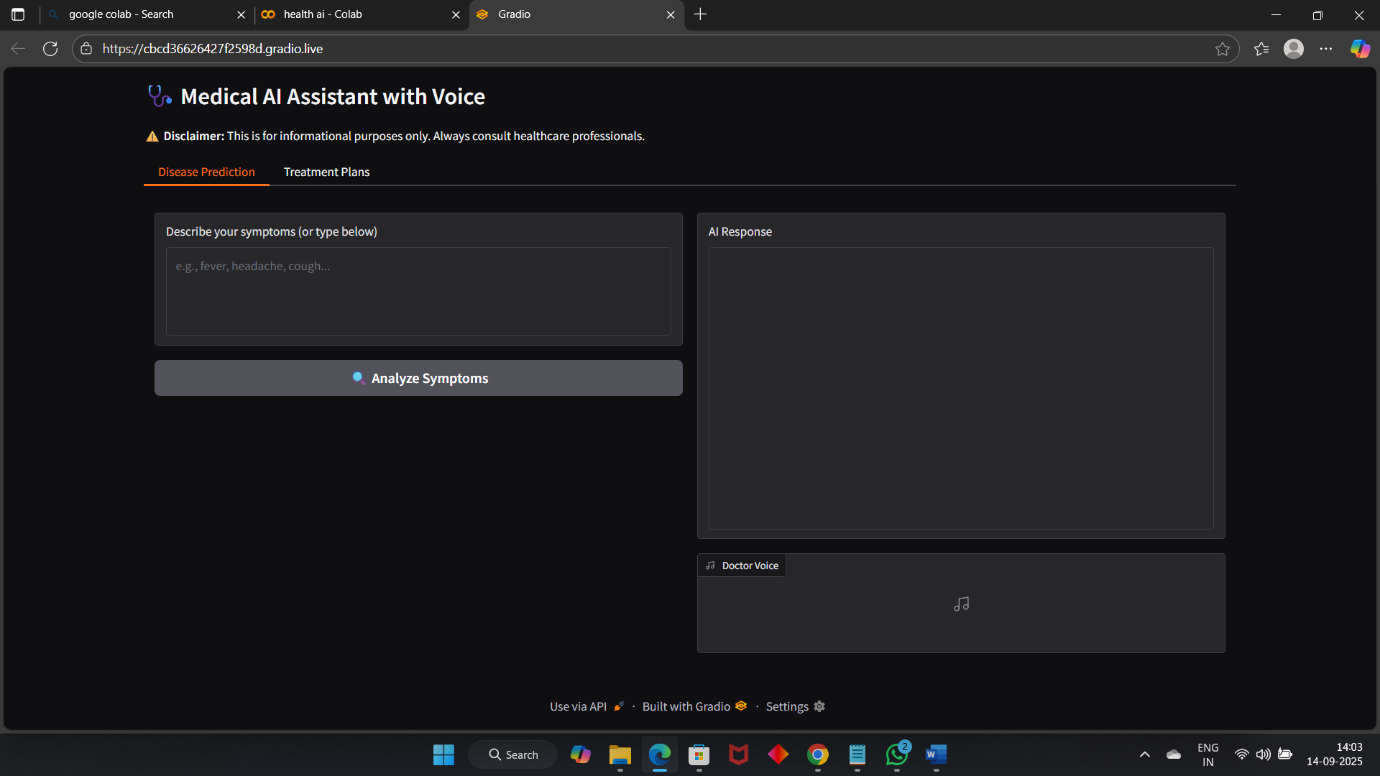
**CODING**



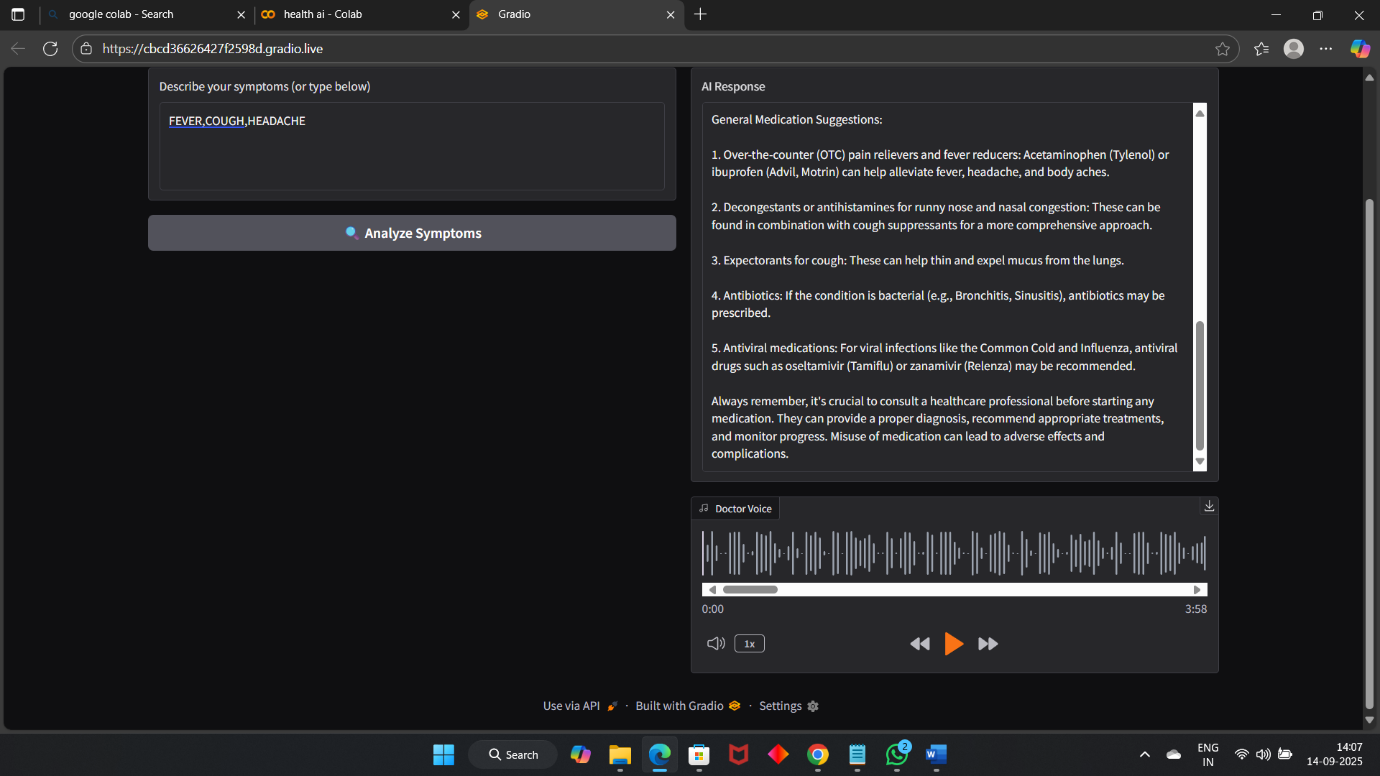
**OUTPUT LINK**



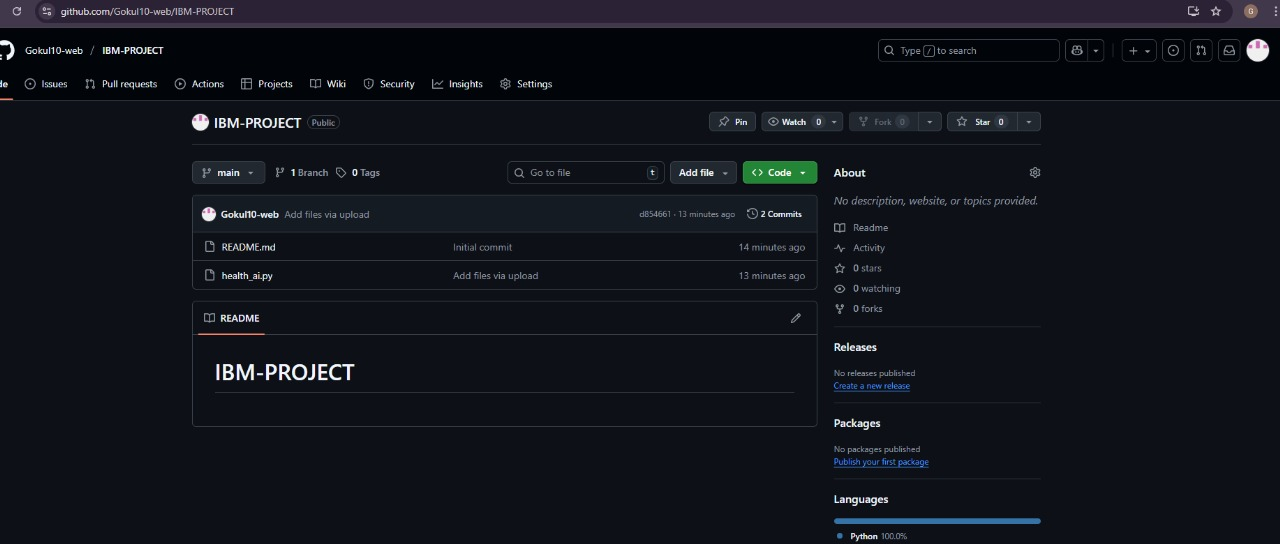
**FINAL OUTPUT**



**GITHUB SCREENSHOT:**



**GITHUB SCREENSHOT:**



**12. Known Issues**

* Large prompts may exceed model max token limit.
* Some complex medical queries may be inaccurate.
* Voice generation may fail if internet connection is unstable.
* Model requires significant memory; GPU recommended.

**13. Future Enhancements**

* Convert Gradio app into REST API with FastAPI backend.
* Add user authentication and history tracking.
* Integrate with cloud TTS services for better voice quality.
* Add additional health modules (e.g., medication reminders, symptom checker).
* Mobile-friendly web interface.