# Health AI: Intelligent Healthcare Assistant

# Generative AI with IBM

# Project Documentation

## 1. Introduction

• Project title: Health AI - Intelligent Healthcare Assistant

• Team member: Latha Maheshwari.S

• Team member: Mahalakshmi.N.B

• Team member: Mahima.B

• Team member: Manju.A

## 2. Project Overview

**• Purpose:**

The purpose of HealthAI is to provide an intelligent healthcare assistant using IBM Granite models.

It empowers patients and healthcare providers with conversational support, disease prediction,

treatment recommendations, and secure access to medical guidance.

The assistant integrates generative AI with medical resources to simplify healthcare queries and

support better decision-making.

**• Features:**

* Disease Prediction  
   Key Point: Symptom analysis with severity levels  
   Functionality: Suggests possible conditions and warnings
* Treatment Plans  
   Key Point: Personalized patient care guidance  
   Functionality: Generates general treatment options, home remedies, and precautions
* Lifestyle & Wellness Tips  
   Key Point: Preventive care and healthy living  
   Functionality: Provides short, actionable tips for diagnosed conditions
* Conversational Interface  
   Key Point: Natural language interaction  
   Functionality: Interactive Gradio interface with multiple tabs

## 3. Architecture

-Frontend (Gradio):  
Provides an intuitive interface with tabs for Disease Prediction, Treatment Plans, and Lifestyle Tips.  
Textbox inputs, dropdowns, and buttons allow user interaction.  
  
-Backend (Transformers + PyTorch):  
Uses the ibm-granite/granite-3.2-2b-instruct model for natural language understanding and generation.  
Handles prompts for predictions, treatment plans, and tips.  
  
-Model Integration:  
Uses Hugging Face AutoTokenizer and AutoModelForCausalLM.  
Runs on GPU (Google Colab T4 GPU supported).  
  
Workflow:  
1. User enters symptoms or conditions.  
2. Model processes input via prompt engineering.  
3. Gradio displays results interactively.

## 4. Setup Instructions

**Prerequisites**:

o Python 3.9 or later

o Gradio framework installed

o IBM Granite models from Hugging Face

o Git for version control

o Google Colab environment with GPU access

**Installation Process:**

o Clone the project repository

o Install dependencies from requirements.txt

o Configure Hugging Face and IBM Granite credentials

o Launch the backend with FastAPI

o Run the Gradio interface on Google Colab

o Start interacting with the healthcare assistant

## 5. Folder Structure

app/ – Backend FastAPI logic for disease prediction, summarization, and chat

ui/ – Gradio components for chat and dashboards

health\_assistant.py – Main entry script for Gradio interface

granite\_llm.py – Handles communication with IBM Granite models

predictor.py – Symptom and disease prediction module

summarizer.py – Summarizes medical documents

treatment\_advisor.py – Provides treatment suggestions

## 6. Running the Application

➢ Run: python app.py  
➢ Access the Gradio UI link  
➢ Navigate through tabs:  
 - Disease Prediction: Enter symptoms & severity  
 - Treatment Plans: Provide condition, age, gender, history  
 - Lifestyle Tips: Generate preventive wellness guidance

## 7. API Documentation

This version primarily runs via Gradio UI. However, APIs could be exposed in future:

POST /predict-disease – Predicts possible diseases from symptoms

GET /get-treatment – Provides treatment suggestions based on conditions

POST /lifestyle-tips-Provides tips to enhance our health by changing our lifestyle

## 8. Authentication

Currently, the app is for demo and runs in an open environment. For secure deployments, authentication mechanisms like API keys, OAuth2, and role-based access can be integrated.

## 9. User Interface

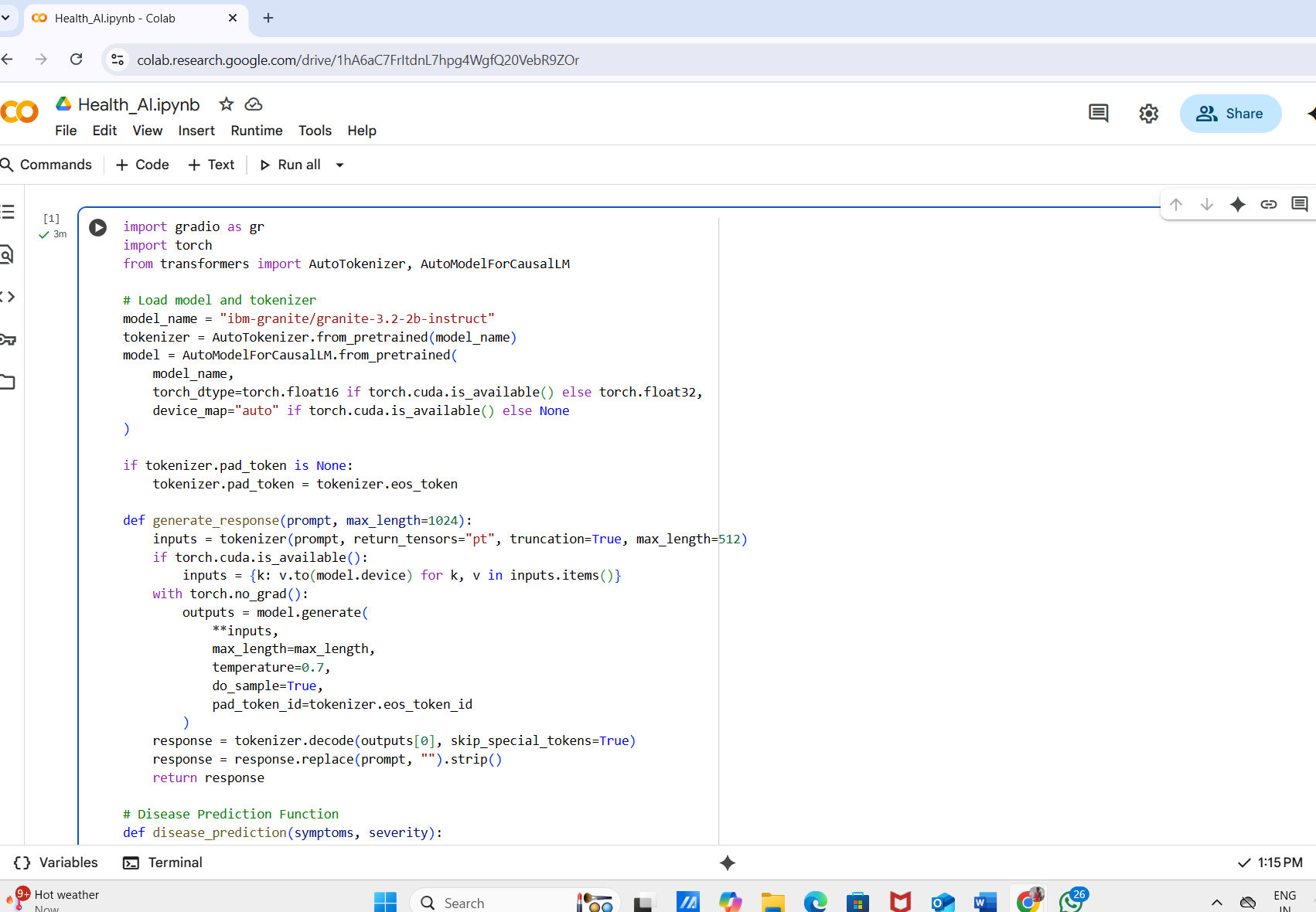
The Gradio interface includes:  
 - Tabs for predictions, treatment, and tips  
 - Multi-line text inputs  
 - Dropdowns for severity and gender  
 - Large text output areas for responses  
Design focuses on clarity, simplicity, and accessibility.

## 10. Testing

Testing was carried out in phases:  
• Unit Testing: Functions like generate\_response, disease\_prediction.  
• Manual Testing: UI interactions via Gradio.  
• Edge Cases: Invalid inputs, missing severity, large symptom lists.  
All functions validated for reliability.

## 11. Screenshots

CODING:



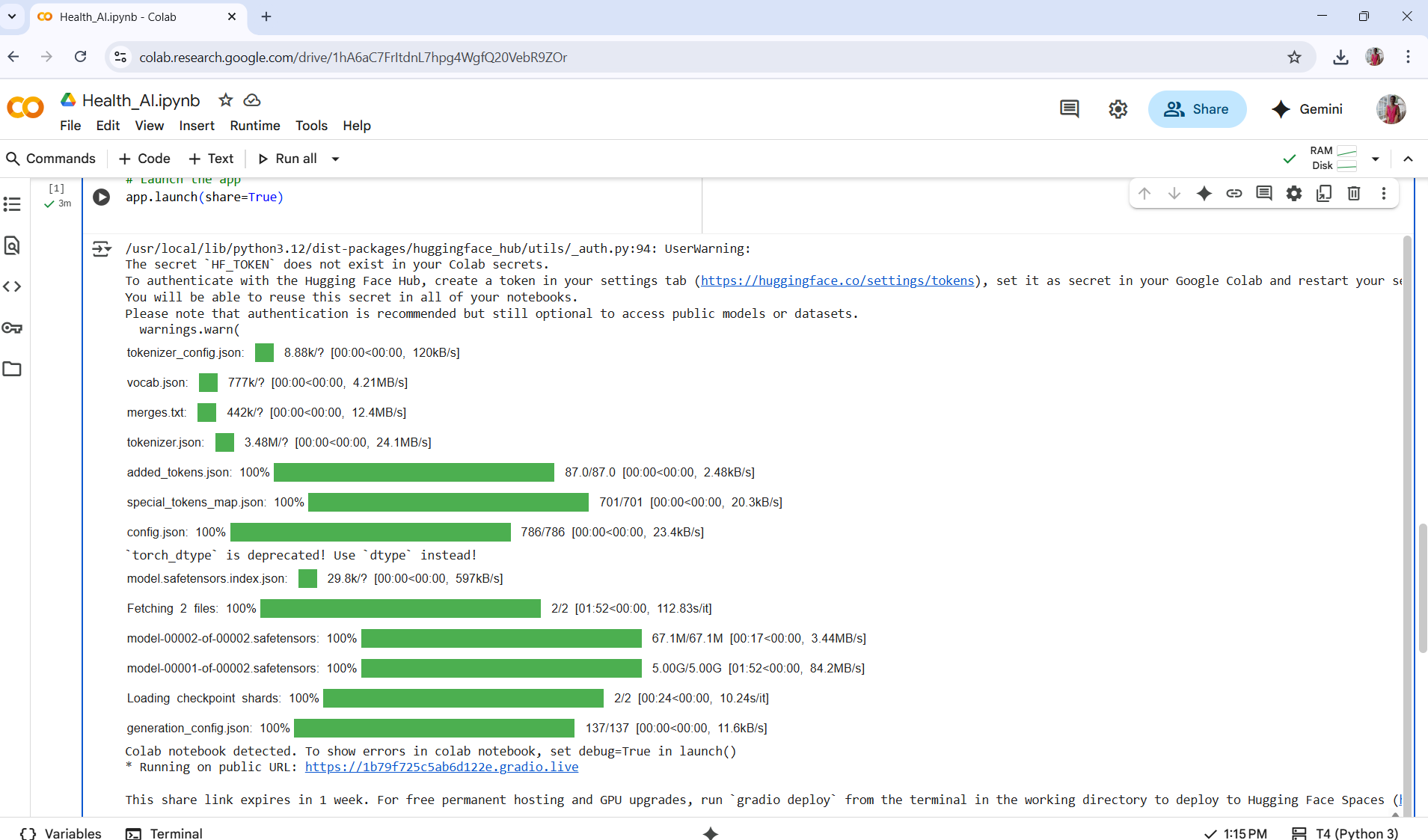
A screenshot of a computer

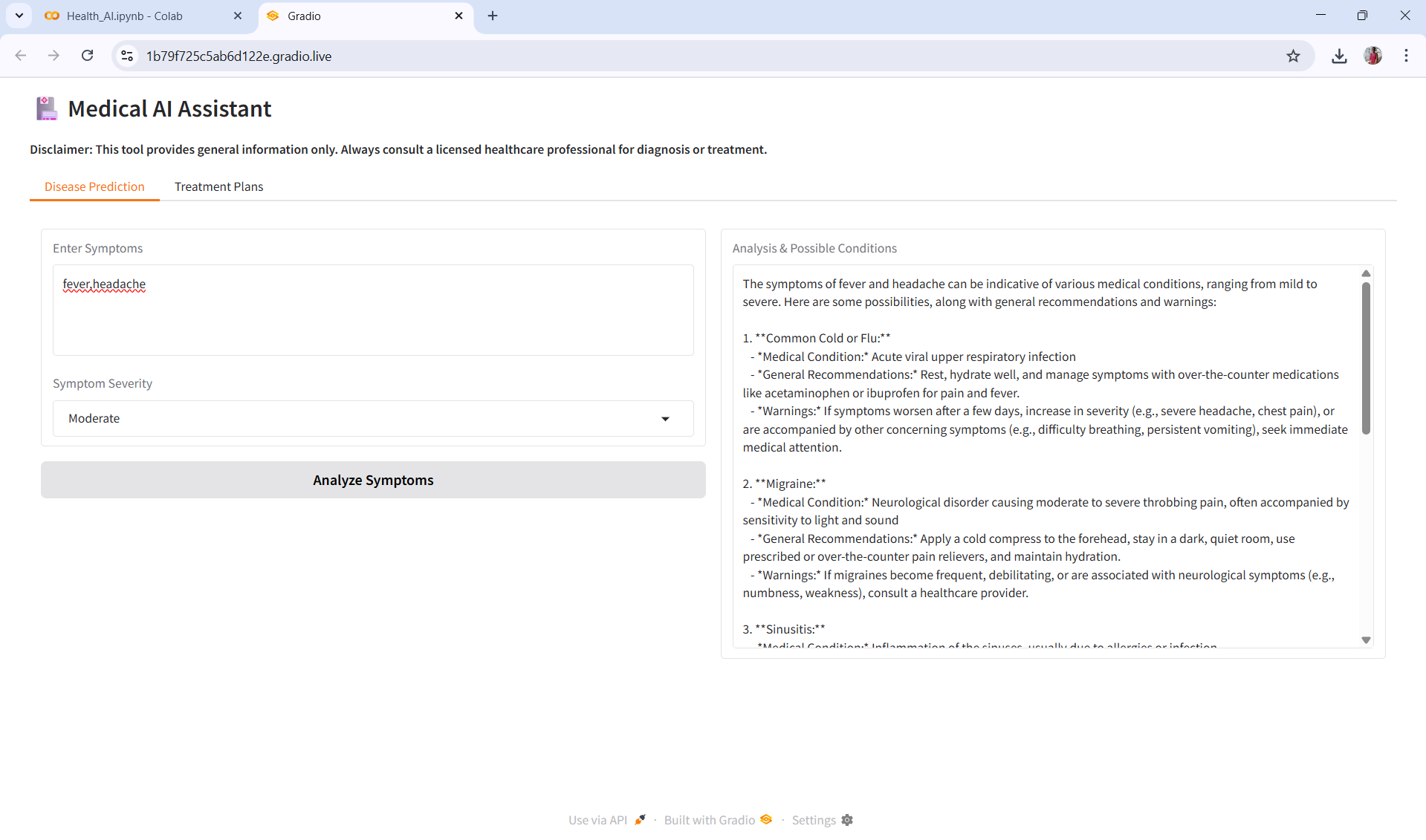
AI-generated content may be incorrect.

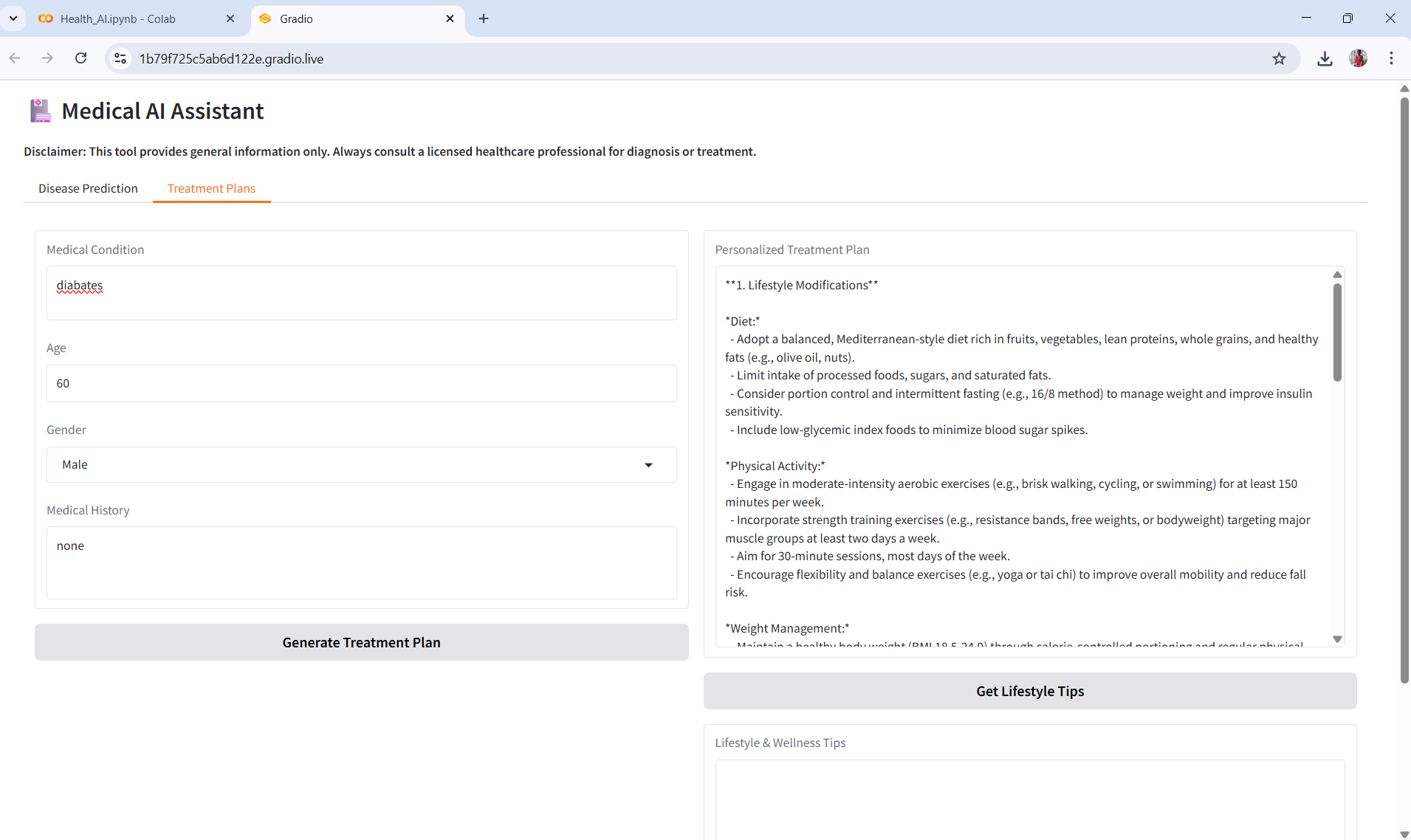
A screenshot of a computer

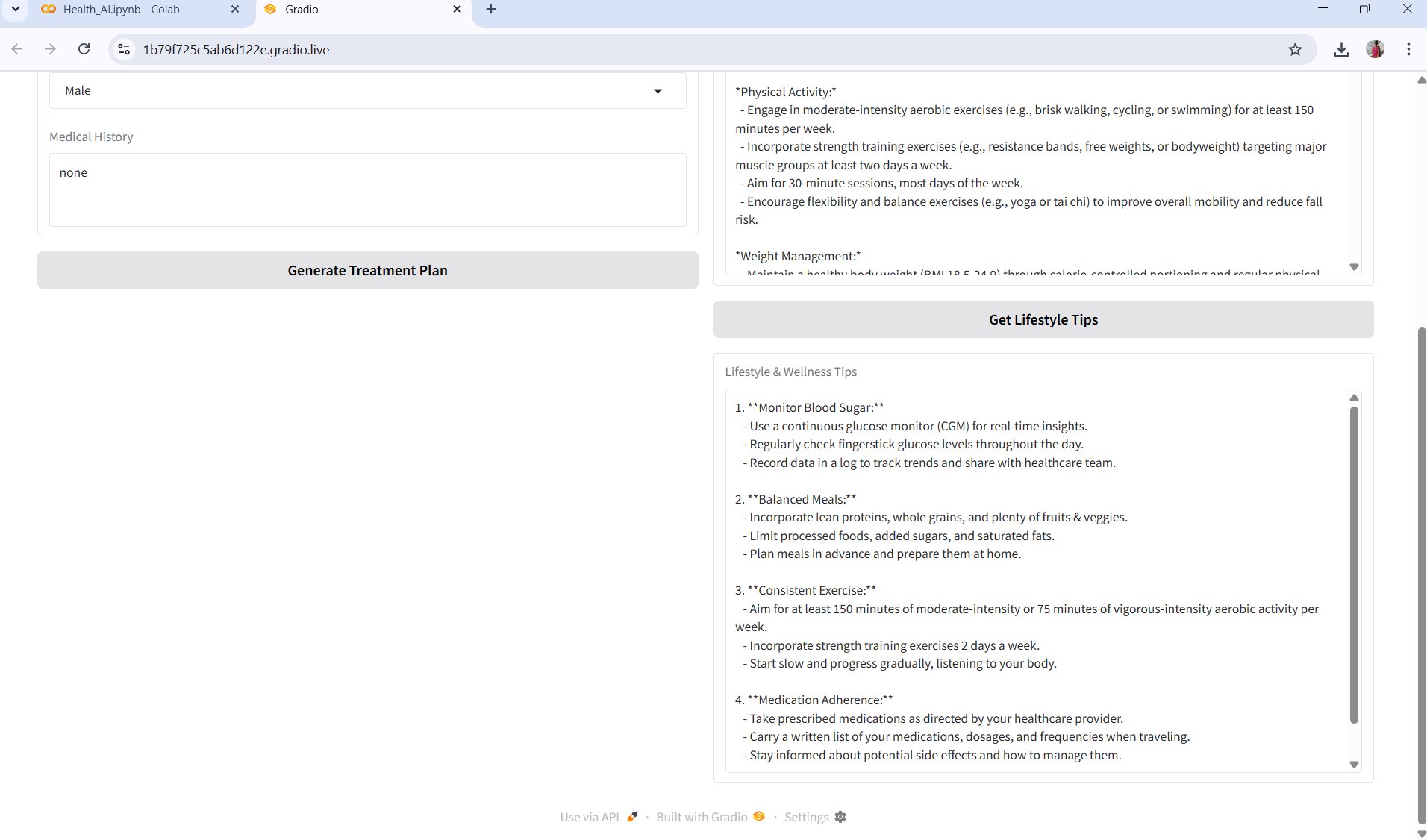
AI-generated content may be incorrect.

OUTPUT:









## 12. Known Issues

• Model may generate lengthy or less precise outputs.  
• Medical recommendations are generalized, not diagnostic.  
• Requires internet access for Hugging Face model download.

## 13. Future Enhancements

- Integration with wearable devices for real-time health monitoring

- Multilingual support for wider accessibility

- Secure cloud deployment with role-based access

- Expansion of disease prediction to chronic illnesses

- Database storage for patient session history

- Integration with FastAPI for API-based access.