Health AI- Intelligent Healthcare Assistant

Generative AI with IBM

1.INTRODUCTION:

Health AI – Intelligent Healthcare Assistant

Team Members

1. ***K. Mahendrabupathi - (88472D5E26B209EA06B4C6E173DE899B)***
2. ***R. Pradeepraja - (AC97844F35595E9813B44B55BB6D70A9)***
3. ***P. Murugesan - (E3D4E9CF22BF393834542FB2C1F30F77)***
4. ***S. Naresh - (3C5B5E72979BBE6B9A04C946801155D5)***

2.PROJECT OVERVIEW :

Health AI-intelligent healthcare Assistant is an AI-driven system designed to improve healthcare delivery through intelligent support. The solution integrates AI/ML models with LLMs and vector databases, user-friendly healthcare assistance.

- Conversation Interface

The conversational interface allows users to interact with the healthcare natural text or voice. It supports health queries, policy guidance and human experts.

- Policy summarization

The system automatically extracts and condenses healthcare policies into simple,

easy-to-read summaries. It helps patients, citizens and staff quickly understand key update.

- Eco-Tip Generator

The eco-tip generator provides simple, actionable health and environment friendly suggestions. It promotes sustainable practices like waste reduction, energy, saving methods for community health.

- Citizen feedback loop

The citizen feedback loop collects patient and public opinion through survey, chat, or forms. It analyses feedback using sentiment and topic detection to improve system enhancement.

- KPI Forecasting

KPI forecasting predicts key healthcare performance indicators such as bed occupancy, patients' inflow, staff availability, and resource usage. It helps in planning, decision-making, improving overall Healthcare efficiency.

- Anomaly Detection

Anomaly detection identifies unusual pattern in healthcare data, such as sudden diseases spikes or abnormal resources usage. It enables early alerts and quick action to improve patient safety.

- Multimodal input support

Multimodal input support allows a healthcare assistant to receive and process text, voice, image and sensor data. This enables more natural, accurate, and efficient interaction for patients and medical staff.

- Stream lit to Gradio UI

Stream lit to Gradio UI allows converting python apps into interactive web interface with easy deployment. Gradio provide simple drag-and-drop components for inputs and outputs, user interaction.

3.ARCHITECTURE:

- Front-end Architecture of Health AI-Intelligent Healthcare assistant interaction such as web, mobile or voice app, allowing patients or citizens to accessible way.
- Back-end Architecture of Health AI-Intelligent Healthcare assistant handles business logic, connect LLM for prediction and recommendations provide APLs for secure, real-time communication with the front end

- LLM Integration:

The LLM Processes user queries, interpret intent, and generate response while interacting with for knowledge retrieval and personalized recommendations.

- Vector sector:

The vector sector in health care AI stores embedding of medical data, policies, records to enable fast semantic search, allowing the AI to Retrieve contextually relevant information guidance.

- ML Modules:

The ML Modules provide analytics, anomaly detection and personalized health recommendations by analysing patient data, trends medical knowledge to support.

4.SETUP INTRODUCTION:

- Prerequisites:

- Install python, node and code editor
- Access open AI API and vector database
- Basic ML knowledge and libraries
- Front end skills (react, angular or flutter)
- Medical database or pre-trained models

- Installation process:

- Install python, node and set up a virtual environment.
- Install backend and AI libraries
- Setup vector database and create index.
- Run back end and front end test queries and AI responses.

5.FOLDER STRUCTURE:

- App: application code
- Data: Storage data files
- Docs: Documentation
- Test: Unit test
- Ven: Virtual environment
- Txt: Text
- Utils: Utility function

6.RUNNING THE APPLICATION:

To start the project:

- Open terminal and clone the project from GitHub.
- Navigate to the project folder

• Health AI-Intelligent healthcare assistant-AI.

• Create and activate a virtual environment.

• Install dependencies using pip install- r requirements. Txt.

• Set up database and vector store (initialize configs).

• Configure API keys (OpenAI/LLM, database, etc.).

• Start backend server using python backend/ API/main.py.

• Run frontend UI using stream lit run frontend/ app. Py Or gradio app.

Py. Frontend (Stream lit):

Enter frontend folder - run stream lit – open browser- use Health AI-Intelligent healthcare assistant UI.

Backend (fast API):

Enter backend folder – run fast API server – open API docs – backend ready for health AI-Intelligent healthcare assistant UI integration.

7.API DOCUMENTATION:

API Documentation provides endpoint for patient interaction, health record management, and AI – drive recommendations, enabling seamless healthcare support through conversational and analytical modules.

8.AUTHENTICATION:

Register user/client – user signs up or application register to receive API credentials. Generate API key/token – system issues a unique API key or JWT token. Send authentication request – client sends login request with credentials. Receive access token – server validates and responds with an access token.

Use token in API calls – includes authorization: bearer <token>in header for every request.

9.USER INTERFACE:

- Provides a simple login for secure access.
- Chat window support text and voice queries.
- Dashboard shows health records and reports.
- AI gives personalized tips and recommendations.

10.TESTING:

- Testing in the health AI – intelligent healthcare assistant ensures the system works reliably and safety for patients and doctors. First, unit testing is

performed to verify individual modules like authentication, chat, and health records access.

- User acceptance testing (UAT) then validate that real user finds the system useful, accurate, easy to use. This step-by-step testing process guarantees a robust and trustworthy healthcare assistant.

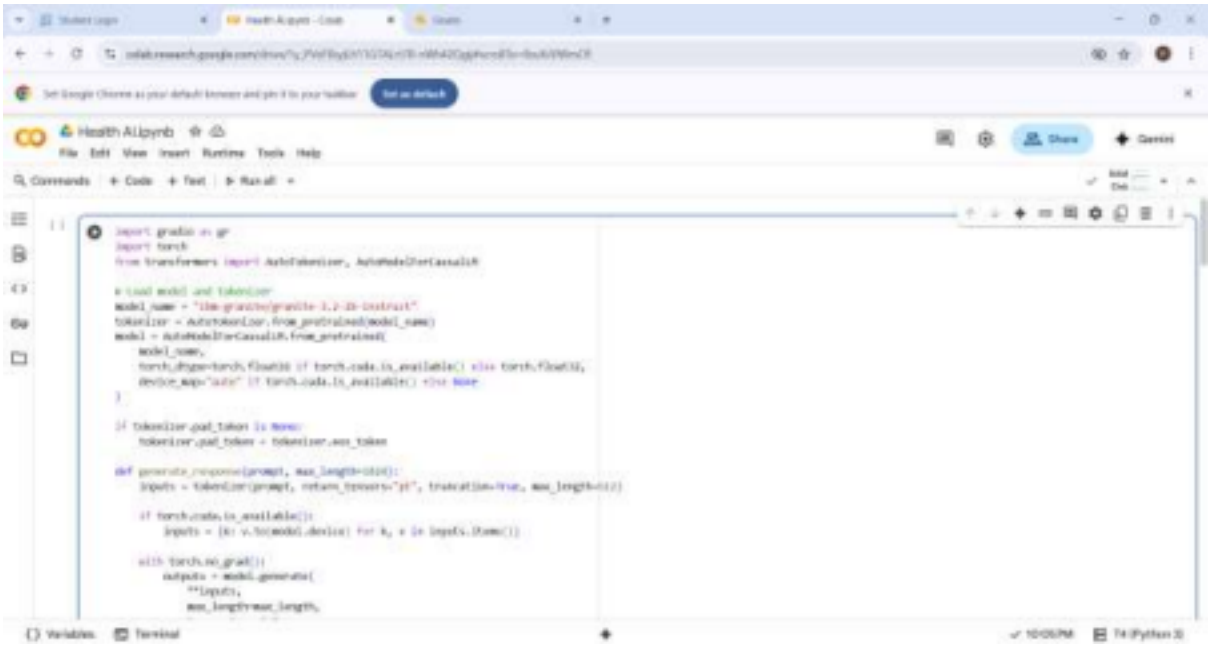
11.KNOWN ISSUES:

The Health AI-Intelligent healthcare assistant, while highly efficient, currently faces certain limitations. Occasional inaccuracies in symptom analysis may occur due to limited training data for rare conditions. Multimodal input processing, such as interpreting images and text together, may experience delay or misinterpretation under poor-quality inputs. Some users may encounter minor UI inconsistencies across different device these issues, improve reliability and enhance overall user experience. testing and initial deployment, several issues were identified that may affected the system performance or user experience. Some feature may response slowly under heavy load and occasionally UI observed on different device. Certain data inputs may trigger unexpected errors if they do not match the expected format additionally, integration with external APIs can server downtime will be provided in future releases to stability and functionality.

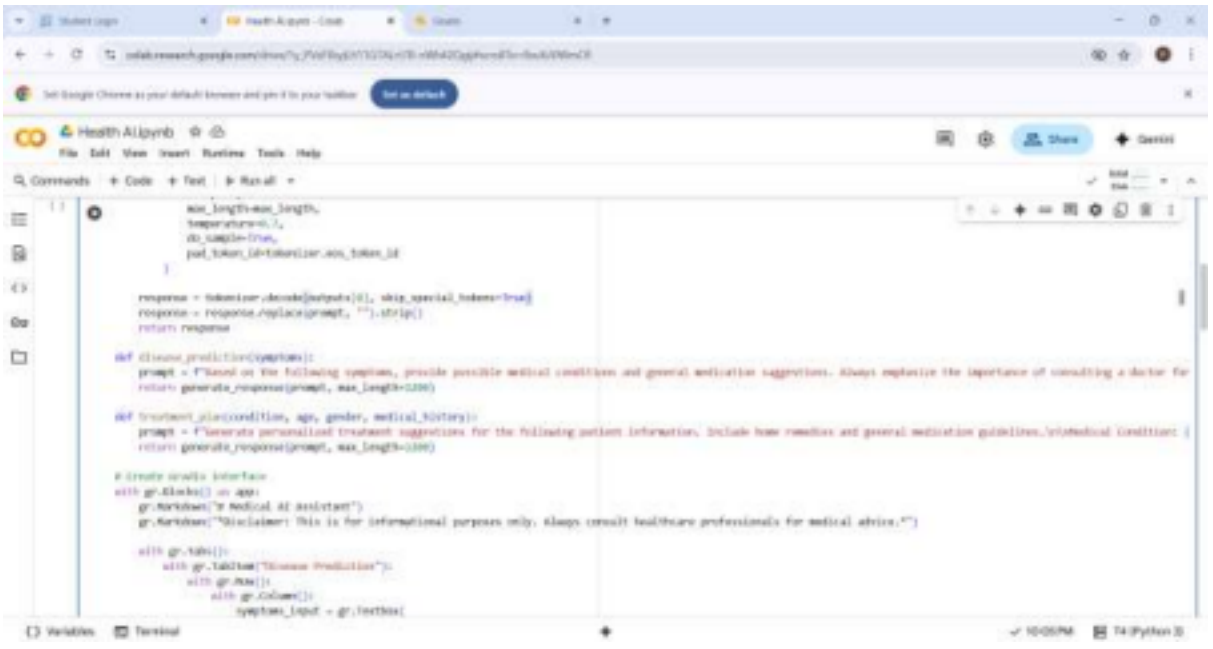
12.FUTURE ENHANCEMENTS:

In future as Health AI-Intelligent healthcare assistant update the system improvement for the health AI assistant will focus on expanding its capabilities and improving patient care. Planned enhancement include incorporating more advanced diagnosis tools, supporting multimodal input such as voice and image, and integrating with additional health data sources for personalized recommendations. The system will also implement predictive analytics for early disease detection, real-time health
monitoring and enhance natural language understanding for more accurate and empathetic interactions. The user interface will be refined for greater accessibility and responsibility while automated error handling and notification system will be implemented to ensure smoother operations. These enhancements aim to provide a more robust, efficient and user- friendly solution.

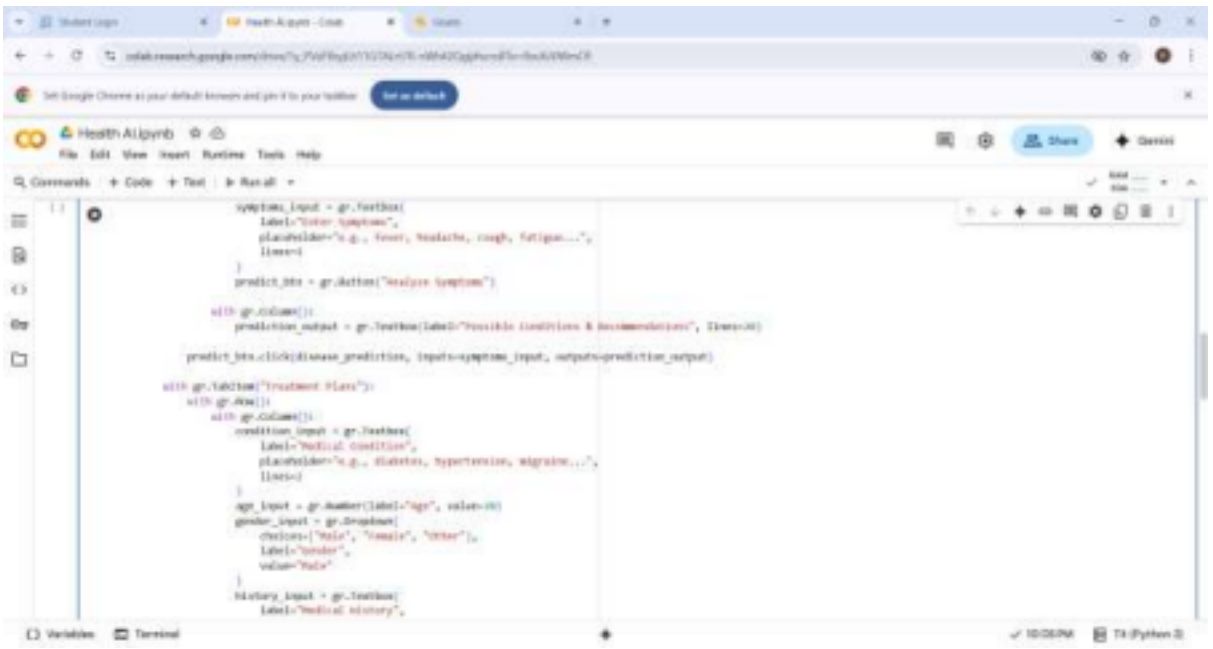
. 13PROJECT SCREENSHOT:



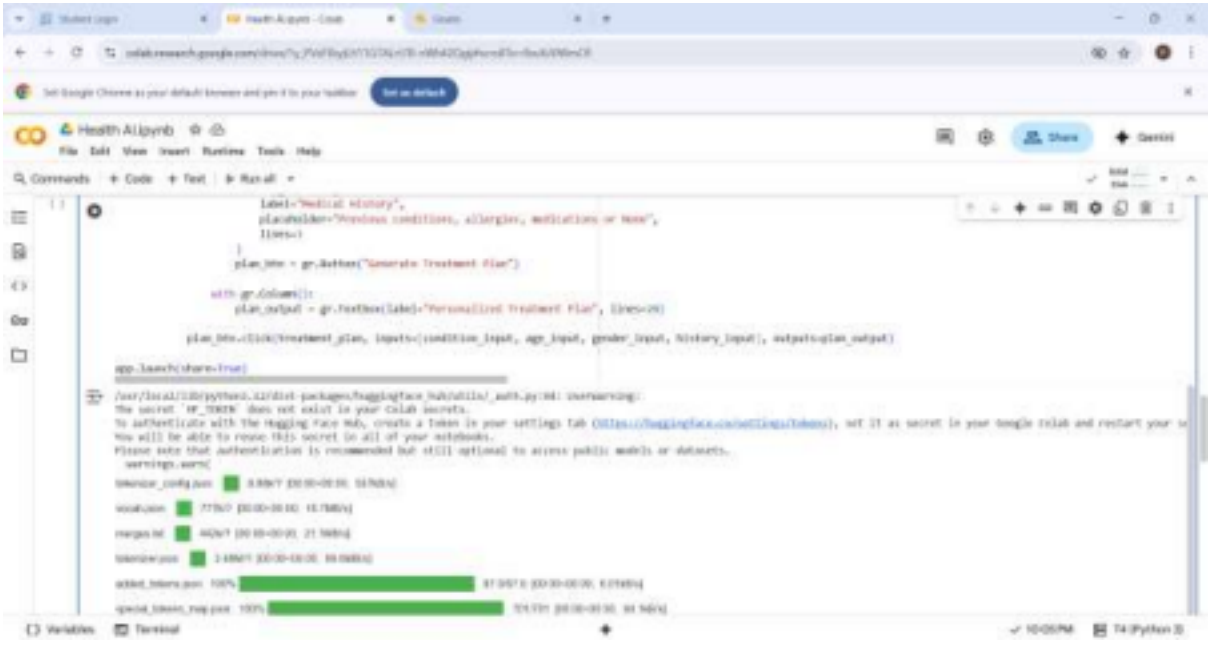
```
import gradio as gr
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

# Load model and tokenizer
model_name = "ibm-granite/granite-3.2-2b-instruct"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    torch_dtype=torch.float16 if torch.cuda.is_available() else torch.float32,
    device_map="auto" if torch.cuda.is_available() else None
)

if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token

def generate_response(prompt, max_length=512):
    inputs = tokenizer(prompt, return_tensors="pt", truncation=True, max_length=512)

    if torch.cuda.is_available():
        inputs = {k: v.to(model.device) for k, v in inputs.items()}

    with torch.no_grad():
        outputs = model.generate(
            **inputs,
            max_length=max_length,
```

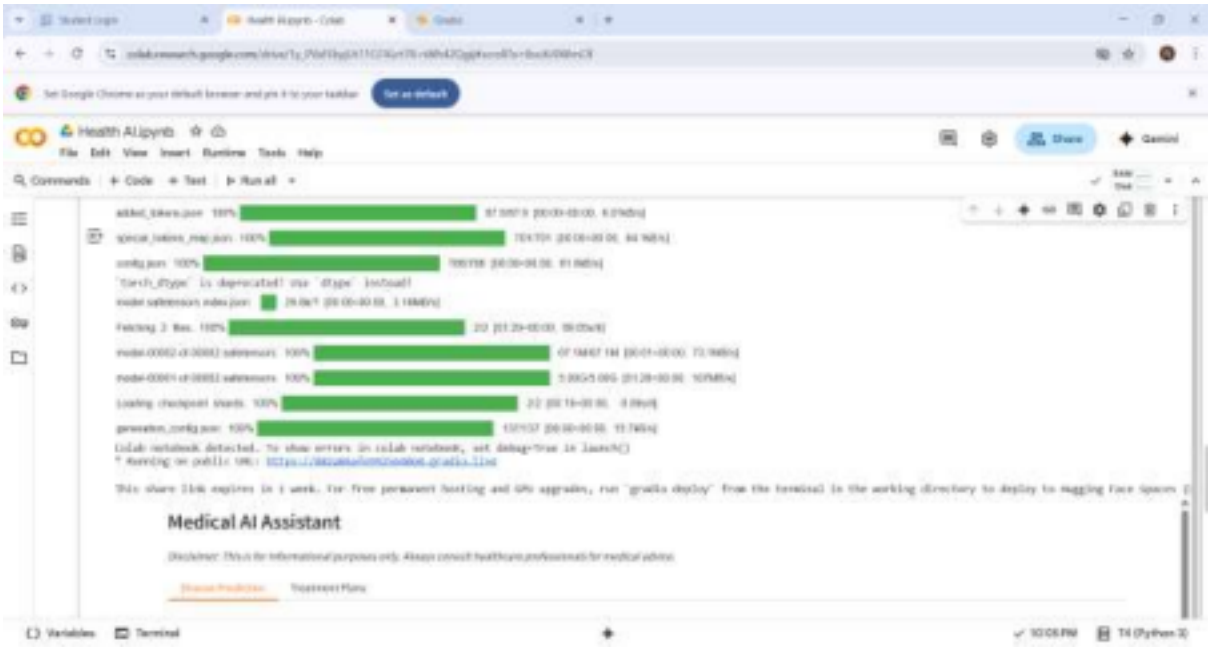


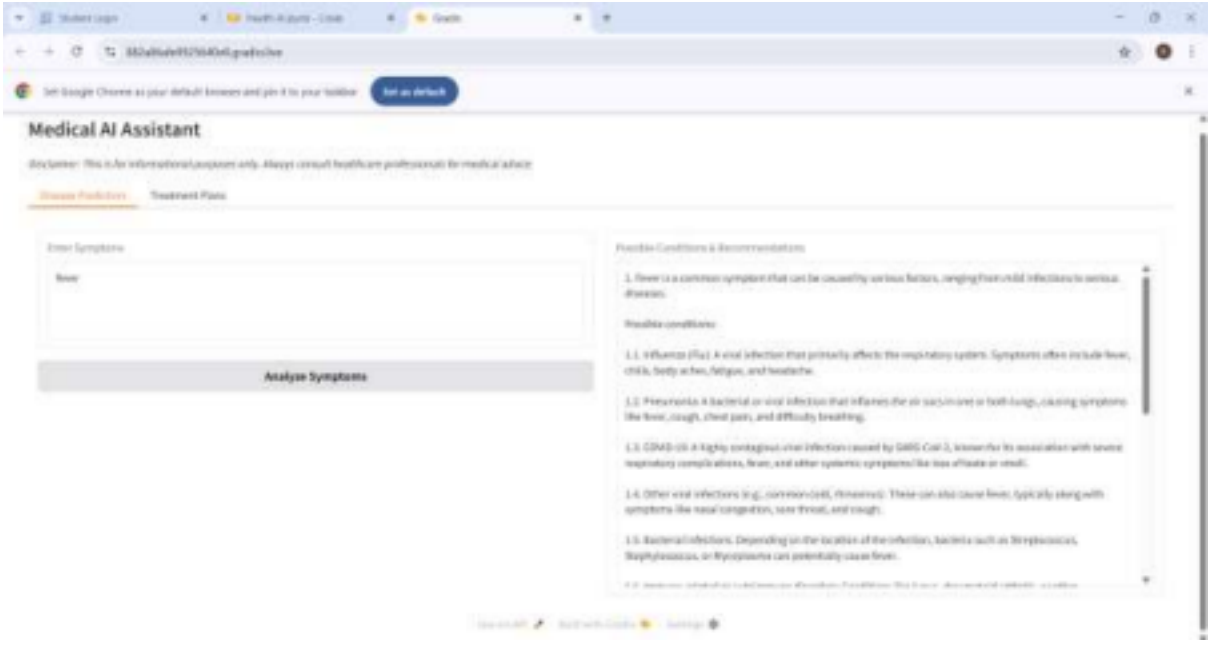
```
            max_length=max_length,
            temperature=0.7,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id
        )

    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = response.replace(prompt, "").strip()
    return response

def disease_prediction(symptoms):
    prompt = f"Based on the following symptoms, provide possible medical conditions and general medication suggestions. Always emphasize the importance of consulting a doctor for
    return generate_response(prompt, max_length=1200)

def treatment_plan(condition, age, gender, medical_history):
    prompt = f"Generate personalized treatment suggestions for the following patient information, include home remedies and general medication guidelines.\n\nMedical Condition: {
    return generate_response(prompt, max_length=1200)

# Create Gradio interface
with gr.Blocks() as app:
    gr.Markdown("# Medical AI Assistant")
    gr.Markdown("⚠️ Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.")

    with gr.Tabs():
        with gr.TabItem("Disease Prediction"):
            with gr.Row():
                with gr.Column():
                    symptoms_input = gr.Textbox(
```

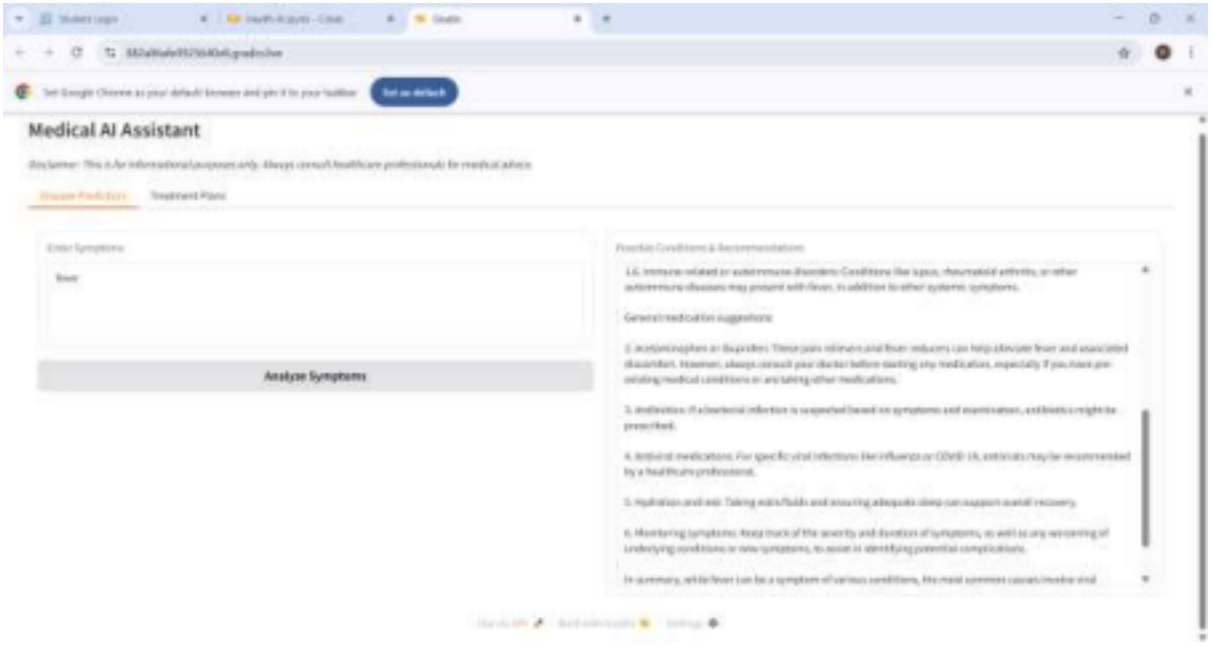


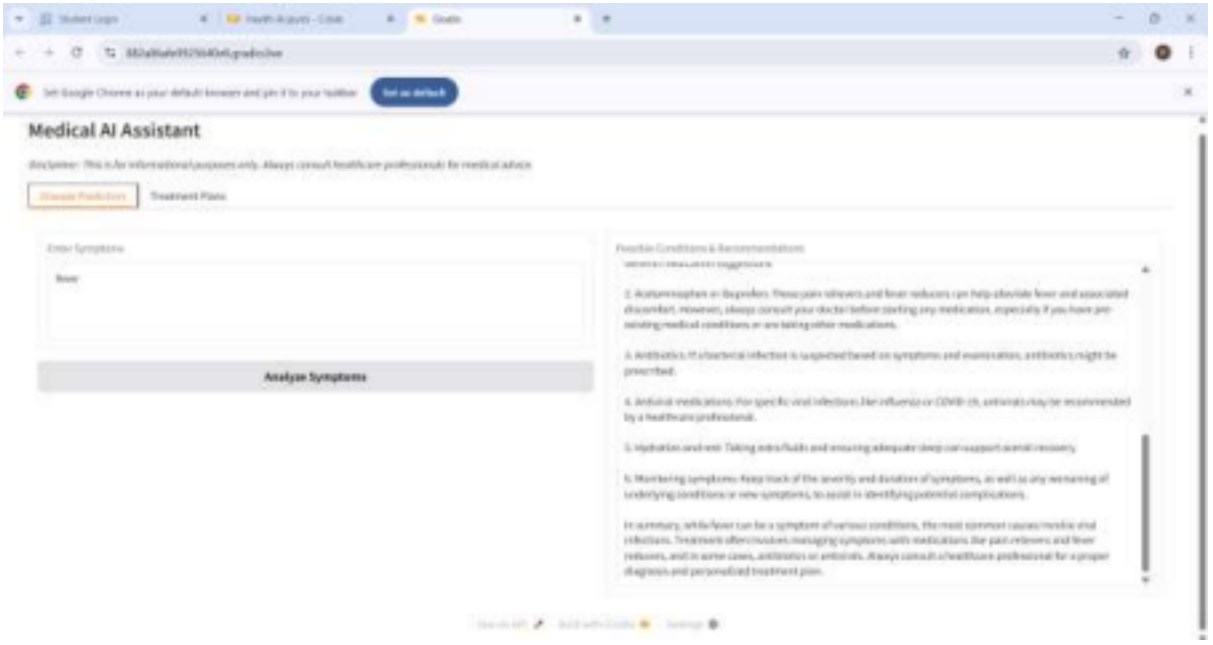
```
                    symptoms_input = gr.Textbox(
                        label="Enter Symptoms",
                        placeholder="e.g., fever, headache, cough, fatigue...",
                        lines=4
                    )
                    predict_btn = gr.Button("Analyze Symptoms")

                with gr.Column():
                    prediction_output = gr.Textbox(label="Possible Conditions & Recommendations", lines=15)

            predict_btn.click(disease_prediction, inputs=symptoms_input, outputs=prediction_output)

        with gr.TabItem("Treatment Plans"):
            with gr.Row():
                with gr.Column():
                    condition_input = gr.Textbox(
                        label="Medical Condition",
                        placeholder="e.g., diabetes, hypertension, migraine...",
                        lines=2
                    )
                    age_input = gr.Number(label="Age", value=30)
                    gender_input = gr.Dropdown(
                        choices=["Male", "Female", "Other"],
                        label="Gender",
                        value="Male"
                    )
                    history_input = gr.Textbox(
                        label="Medical History",
```

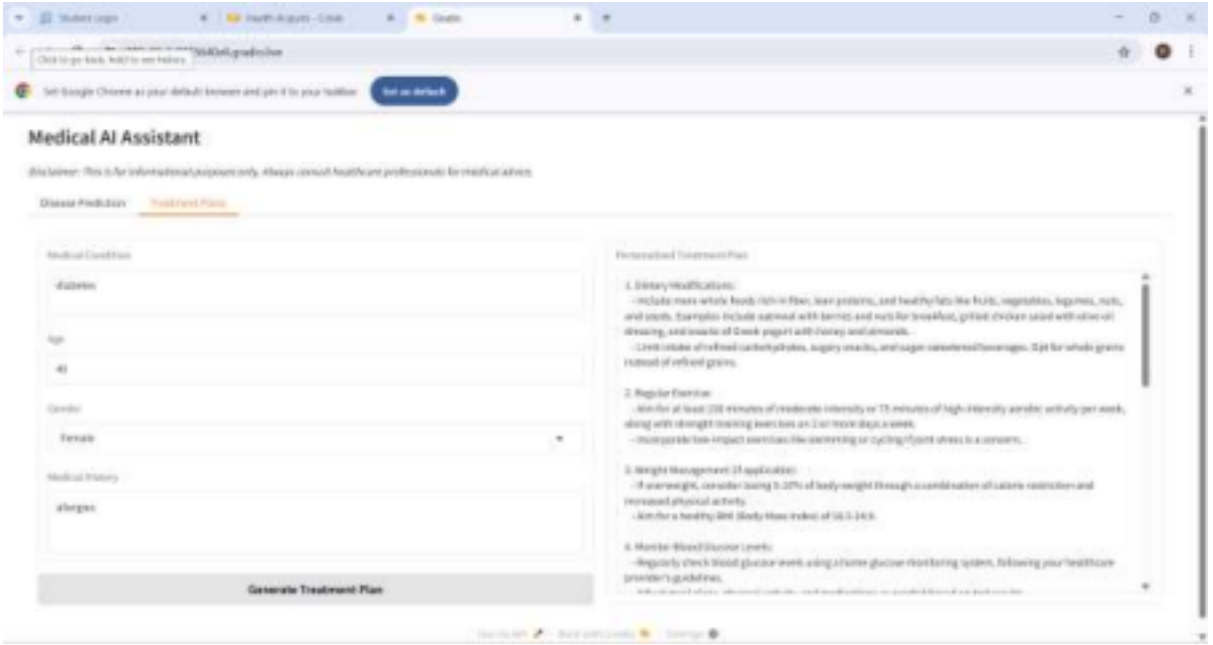

Health AI.ipynb
Label="Medical History",
placeholder="Previous conditions, allergies, medications or None",
lines=3
plan_btn = gr.Button("Generate Treatment Plan")
with gr.Column():
plan_output = gr.Textbox(label="Personalized Treatment Plan", lines=20)
plan_btn.click(treatment_plan, inputs=[condition_input, age_input, gender_input, history_input], outputs=plan_output)
app.launch(share=True)
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your s
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
warnings.warn(
tokenizer_config.json: 8.88k/? [00:00<00:00, 557kB/s]
vocab.json: 777k/? [00:00<00:00, 15.7MB/s]
merges.txt: 442k/? [00:00<00:00, 21.9MB/s]
tokenizer.json: 2.48M/? [00:00<00:00, 99.0MB/s]
added_tokens.json: 100%
special_tokens_map.json: 100%
Variables
Terminal
T4 (Python 3)

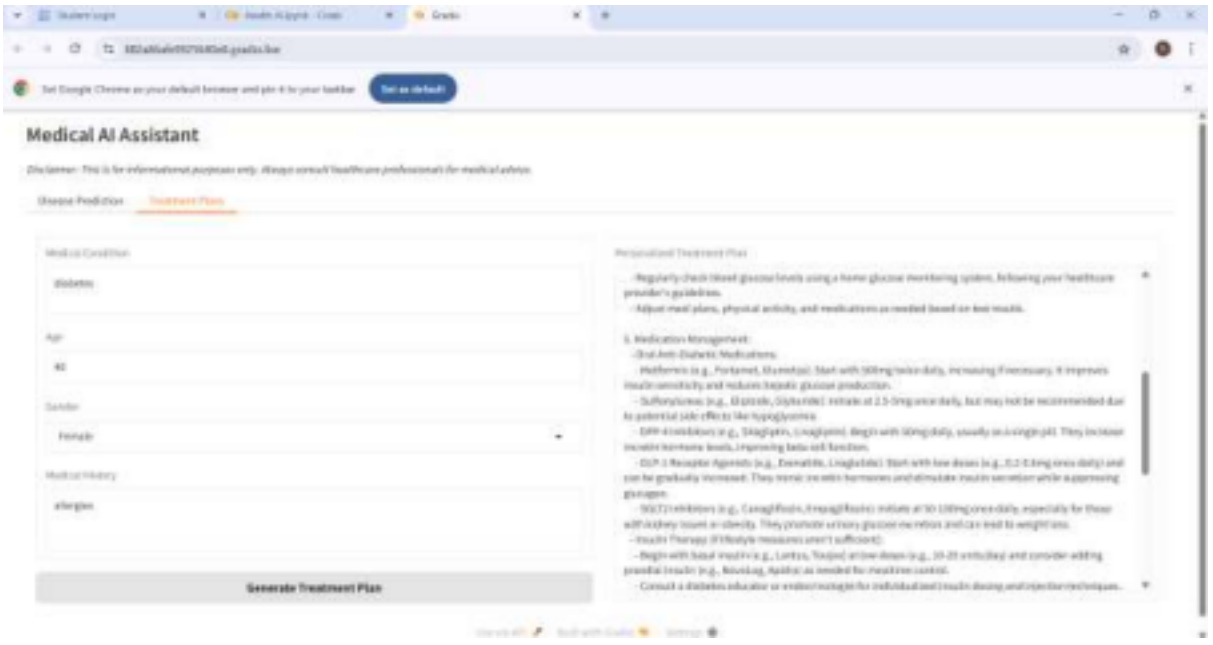

Health AI.ipynb
added_tokens.json: 100%
special_tokens_map.json: 100%
config.json: 100%
`torch_dtype` is deprecated! Use `dtype` instead!
model.safetensors.index.json: 26.0k/? [00:00<00:00, 1.18MB/s]
Fetching 2 files: 100%
model-00002-of-00002.safetensors: 100%
model-00001-of-00002.safetensors: 100%
Loading checkpoint shards: 100%
generation_config.json: 100%
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL:
This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the Terminal in the working directory to deploy to Hugging Face Spaces
Medical AI Assistant
Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.
Disease Prediction
Treatment Plans
Variables
Terminal
T4 (Python 3)

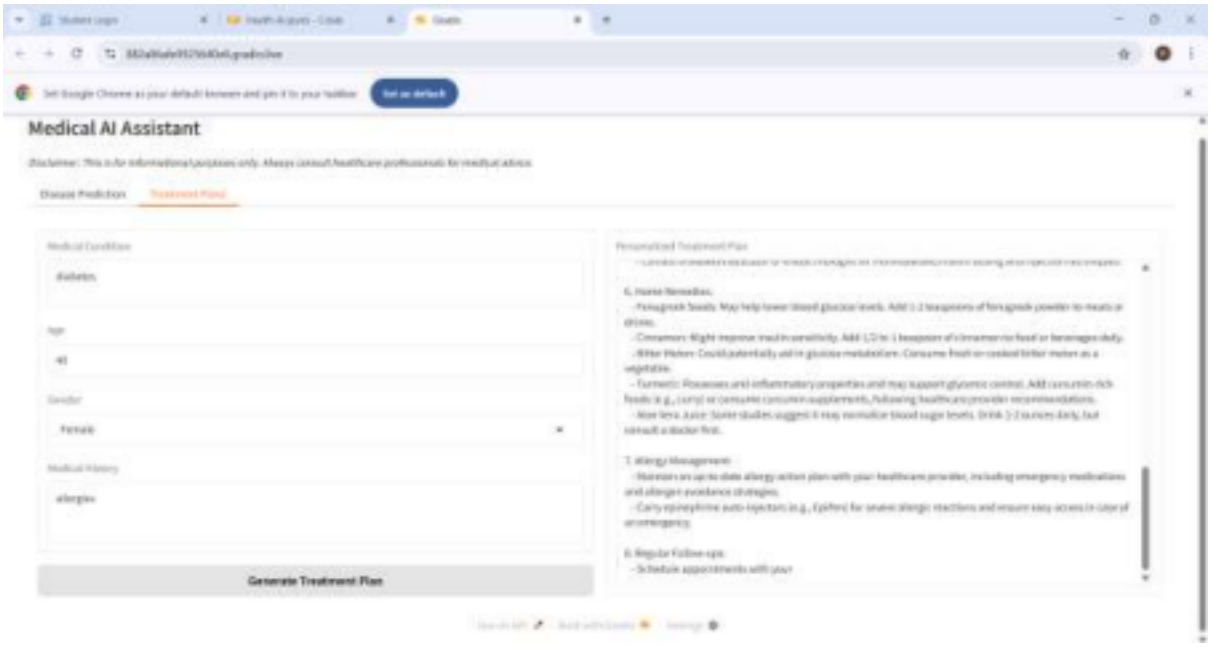

Medical AI Assistant
Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.
Disease Prediction
Treatment Plans
Enter Symptoms
fever
Analyze Symptoms
Possible Conditions & Recommendations
1. Fever is a common symptom that can be caused by various factors, ranging from mild infections to serious illnesses.
Possible conditions:
1.1. Influenza (Flu): A viral infection that primarily affects the respiratory system. Symptoms often include fever, chills, body aches, fatigue, and headache.
1.2. Pneumonia: A bacterial or viral infection that inflames the air sacs in one or both lungs, causing symptoms like fever, cough, chest pain, and difficulty breathing.
1.3. COVID-19: A highly contagious viral infection caused by SARS-CoV-2, known for its association with severe respiratory complications, fever, and other systemic symptoms like loss of taste or smell.
1.4. Other viral infections (e.g., common cold, rhinovirus): These can also cause fever, typically along with symptoms like nasal congestion, sore throat, and cough.
1.5. Bacterial infections: Depending on the location of the infection, bacteria such as Streptococcus, Staphylococcus, or Mycoplasma can potentially cause fever.


Medical AI Assistant
Disease Prediction
Treatment Plans
Enter Symptoms
fever
Analyze Symptoms
Possible Conditions & Recommendations
Generate Treatment Plan


Medical AI Assistant
Disease Prediction
Treatment Plans
Enter Symptoms
fever
Analyze Symptoms
Possible Conditions & Recommendations


Medical AI Assistant
Disease Prediction
Treatment Plans
Medical Condition
diabetes
Age
45
Gender
Female
Medical History
allergies
Generate Treatment Plan
Personalized Treatment Plan





14.VIDEO LINK:

https://drive.google.com/file/d/1tfZugVNZsBl8KKwDaEexnMUk5M2wpYI/view?usp=sharing

THANK YOU