HealthAI: Intelligent Healthcare Assistant Using IBM Granite

Team Leader:

- D.Parameswari

Team Members:

- S.Varsha
- A.Mohaneshwari
- S.Karnikasri

1.Introduction:

Health AI (or AI in healthcare) applies machine learning and other cognitive technologies to medical and healthcare data to mimic human cognition, improve decision-making, personalize treatments, and automate administrative tasks. It works by analyzing vast amounts of complex medical and patient data from various sources to provide quicker, more accurate

diagnoses, assist in treatment planning, optimize hospital operations, and ultimately enhance patient care and outcomes. Key applications include medical imaging analysis, predictive analytics, robotic surgery, and virtual assistants, all aiming to improve efficiency, reduce costs, and augment human capabilities in healthcare.

2.How Health AI Works

Data Analysis:

Health AI utilizes machine learning algorithms to analyze enormous, complex medical datasets, including patient records, medical images, and scientific literature.

Pattern Recognition:

It identifies patterns and insights within this data that might not be obvious to human observers, leading to more accurate diagnoses and effective treatments.

Predictive Capabilities:

AI models can forecast disease risk, patient volume, and resource needs, allowing for proactive interventions and better resource management.

Automation:

AI automates repetitive administrative tasks like scheduling and billing, freeing up healthcare professionals to focus on patient care.

3.Key Benefits of Health AI

Diagnostics Improved:

Faster and more accurate diagnoses of conditions like cancer, heart disease, and eye disease by analyzing medical images and patient data.

Personalized Treatment:

Development of tailored treatment plans based on individual patient data and real-time insights.

Enhanced Patient Care:

Increased access to care, improved patient satisfaction, and more timely and personalized treatments.

Operational Efficiency:

Streamlined hospital operations, optimized resource allocation, and reduced wait times.

Cost Reduction:

Potential to lower healthcare costs through more efficient operations and preventative care.

4.Applications in Healthcare

Medical Imaging:

Analyzing X-rays, CT scans, and MRIs for early disease detection.

Drug Discovery:

Accelerating the development of new treatments and therapies.

Personalized Medicine:

Tailoring treatments and interventions based on a patient's unique genetic makeup and lifestyle.

5.Project Description:

HealthAI harnesses IBM Watson Machine Learning and Generative AI to provide intelligent healthcare assistance, offering users accurate medical insights. The platform includes a Patient Chat for answering health-related questions, Disease Prediction that evaluates user-reported symptoms to deliver potential condition details, Treatment Plans that provide personalized medical recommendations, and Health Analytics to visualize and monitor patient health metrics.

Utilizing IBM's Granite-13b-instruct-v2 model, HealthAI processes user inputs to deliver personalized and data-driven medical guidance, improving accessibility to healthcare information. Built with Streamlit and powered by

IBM Watson, the platform ensures a seamless and user-friendly experience. With secure API key management and responsible data handling, HealthAI empowers users to make informed health decisions with confidence.

Pre-requisites:

1. Streamlit Framework Knowledge: Streamlit Documentation
2. IBM Watson Machine Learning: IBM Watson ML Documentation
3. Python Programming Proficiency: Python Documentation
4. Data Visualization Libraries: Plotly Documentation
5. Version Control with Git: Git Documentation
6. Development Environment Setup: Flask Installation Guide

6.Authentication:

AI authentication leverages artificial intelligence and machine learning to verify user identities by analyzing unique biometric characteristics, behavioral patterns, and contextual information. This adaptive approach provides continuous and real-time authentication, enhances multi-factor authentication (MFA) systems, and improves security outcomes by detecting suspicious activities and adapting security measures based on risk levels.

7.Key Enhancements and Future Prospects

Diagnostics and Prediction:

AI can analyze medical images (like X-rays and MRIs) to detect diseases like cancer and heart conditions at earlier stages, often before symptoms appear. AI-driven predictive analytics can forecast patient outcomes, identify high-risk individuals, and predict disease progression, allowing for proactive and personalized interventions.

Personalized Medicine:

By analyzing vast amounts of data, including patient records and genetic information, AI enables the creation of highly tailored treatment plans that are more effective for each individual.

Drug Discovery:

AI helps accelerate the drug development process by analyzing chemical compounds and predicting potential drug candidates, which can lead to more efficient treatment options.

Robotic Surgery and Automation:

AI-powered robotics enhances surgical precision and control, leading to quicker recovery times and less patient distress. AI also automates administrative tasks, such as patient scheduling and data coding, improving efficiency and reducing errors.

Patient Engagement:

Virtual health assistants, chatbots, and AI-powered virtual assistants can educate patients, help them manage their medications, and provide real-time support, fostering greater patient involvement in their own well-being.

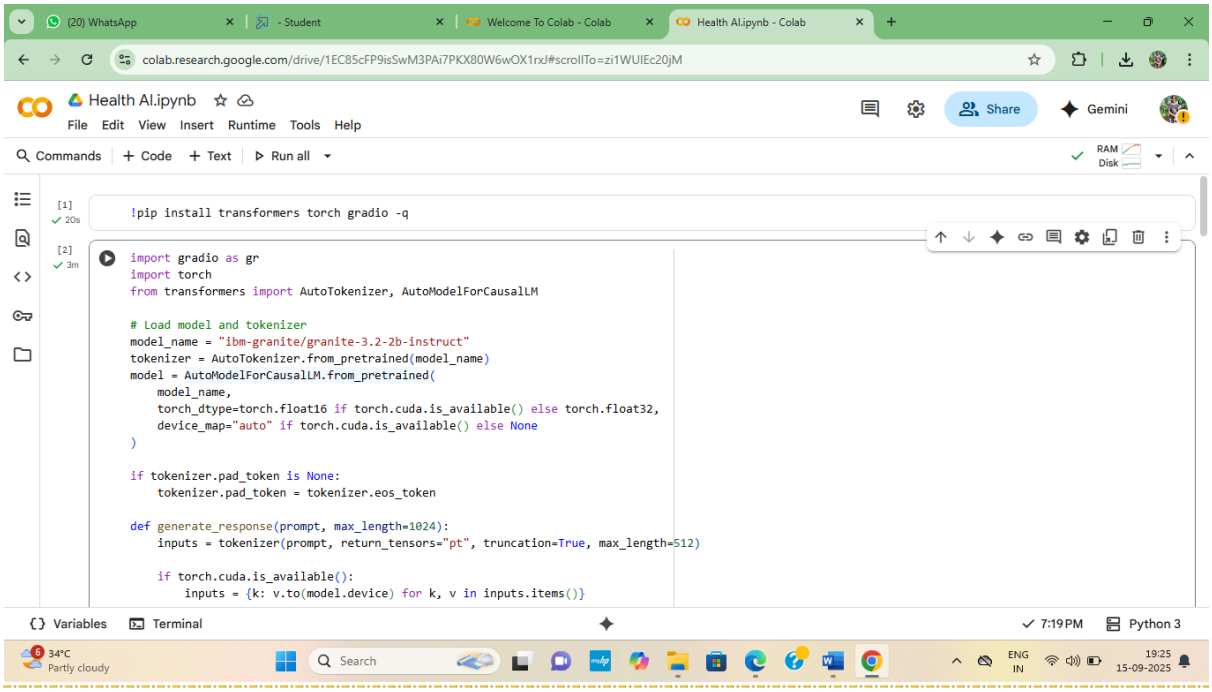
Integration with Wearables and Monitoring:

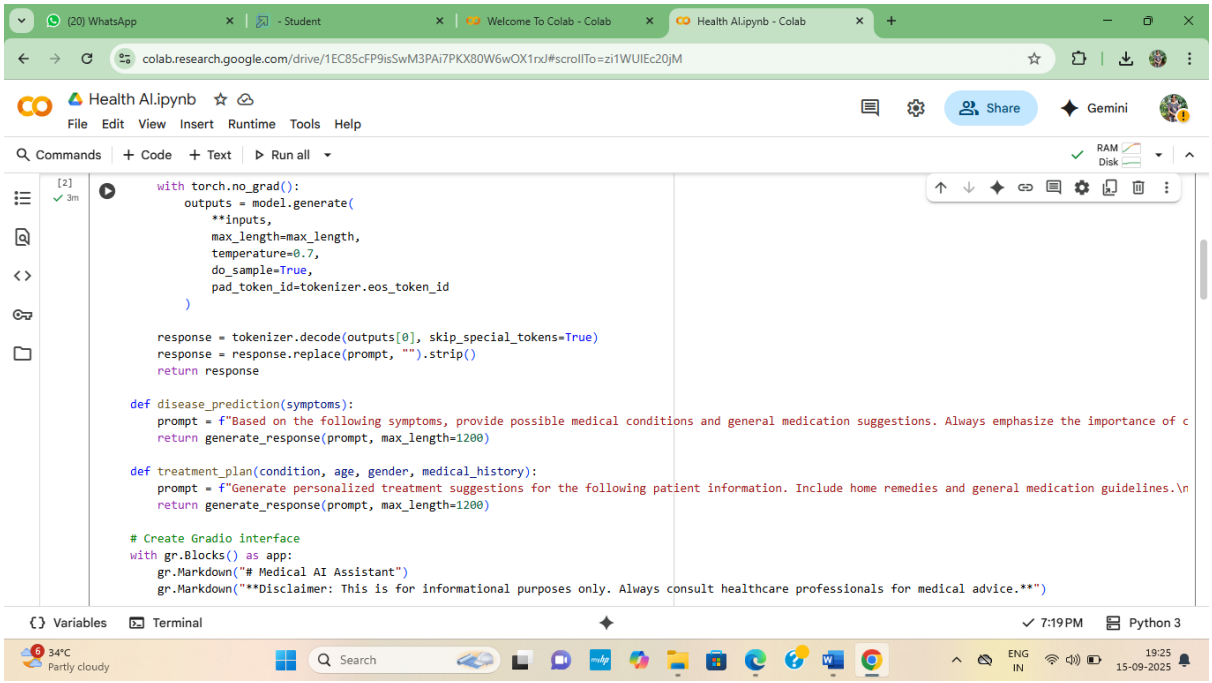
Future AI technologies will integrate more closely with wearable devices and real-time monitoring systems, allowing for continuous data collection and enabling more proactive healthcare interventions.

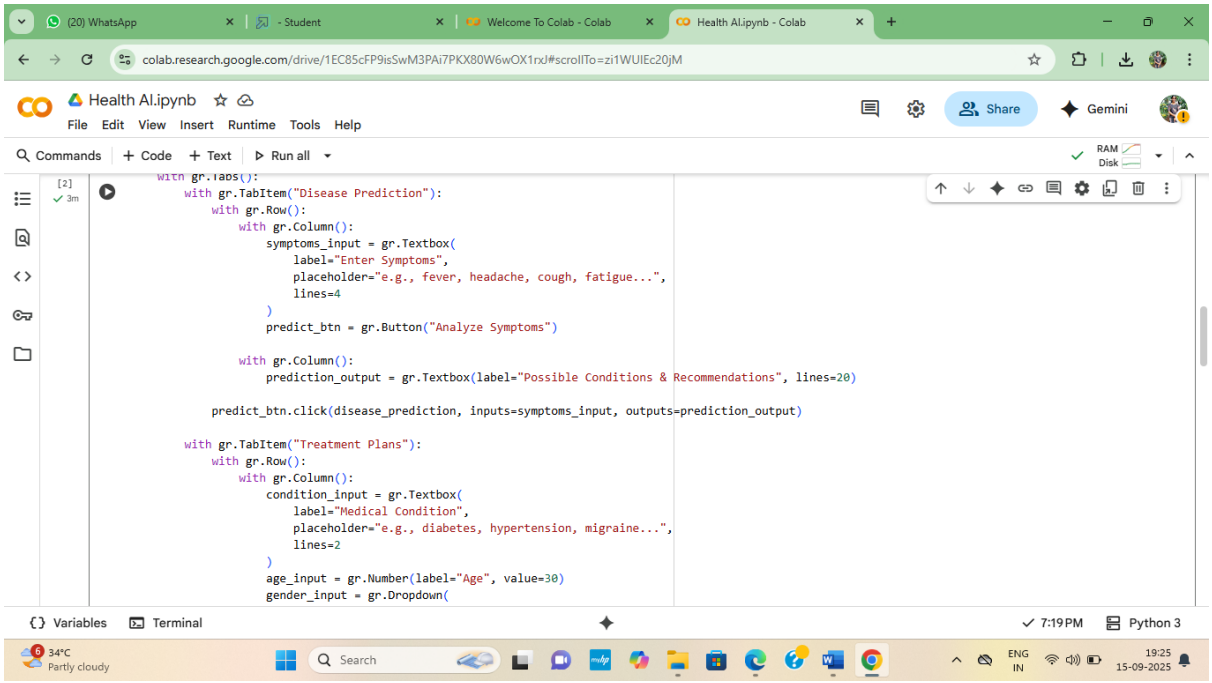
8.SETUP INSTRUCTIONS:

1. Open Google Colab → create a new notebook.
2. Change runtime to T4 GPU (Runtime → Change Runtime Type → GPU).
3. Install dependencies: !pip install transformers torch gradio PyPDF2 -q
4. Load IBM Granite model from Hugging Face (granite-3.2-2b instruct).
5. Run the provided application code (from Drive/GitHub).
6. A Gradio link will be generated → open it to access the assistant.

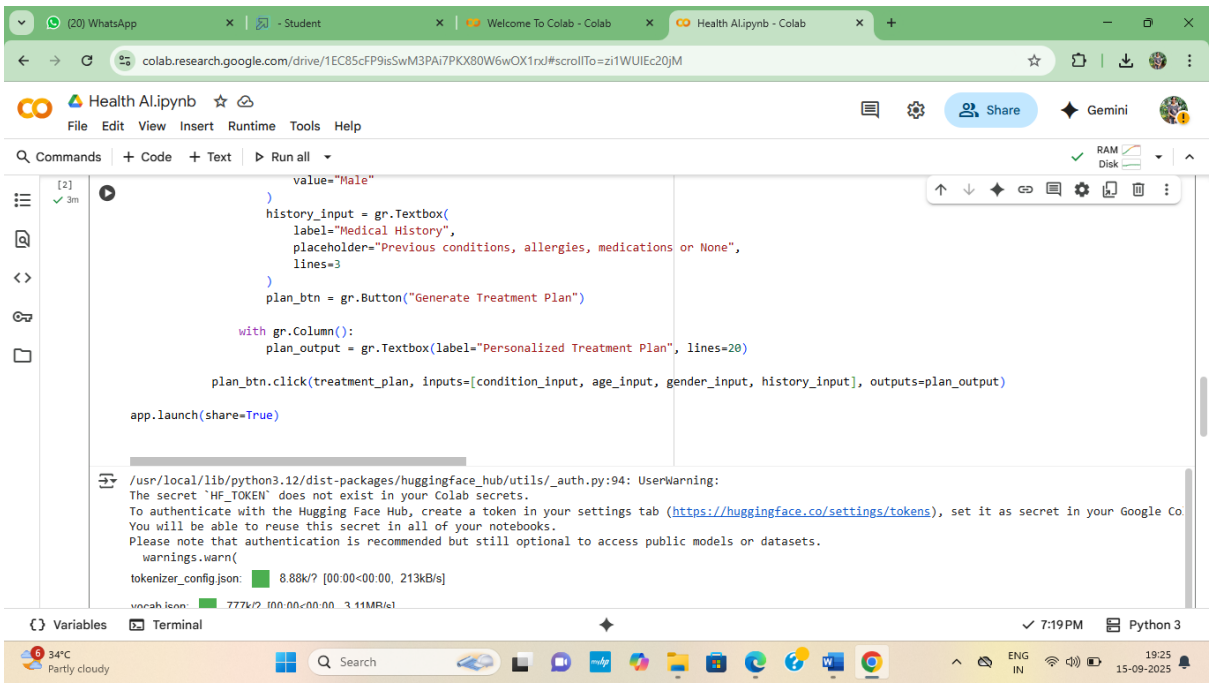
9.PROJECT SCREENSHORT:


(20) WhatsApp
- Student
Welcome To Colab - Colab
Health AI.ipynb - Colab
colab.research.google.com/drive/1EC85cFP9isSwM3PAi7PKX80W6wOX1rxJ#scrollTo=zi1WUIEc20jM
Health AI.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands + Code + Text Run all
RAM Disk
[1] 20s
!pip install transformers torch gradio -q
[2] 3m
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=1024):
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()}
Variables Terminal
7:19 PM Python 3
34°C Partly cloudy
Search
ENG IN
19:25 15-09-2025


(20) WhatsApp
- Student
Welcome To Colab - Colab
Health AI.ipynb - Colab
colab.research.google.com/drive/1EC85cFP9isSwM3PAi7PKX80W6wOX1rxJ#scrollTo=zi1WUIEc20jM
Health AI.ipynb
File Edit View Insert Runtime Tools Help
Share
Gemini
Commands + Code + Text Run all
RAM Disk
[2] 3m
with torch.no_grad():
outputs = model.generate(
**inputs,
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 c
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
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.**")
Variables Terminal
7:19 PM Python 3
34°C Partly cloudy
Search
ENG IN
19:25 15-09-2025

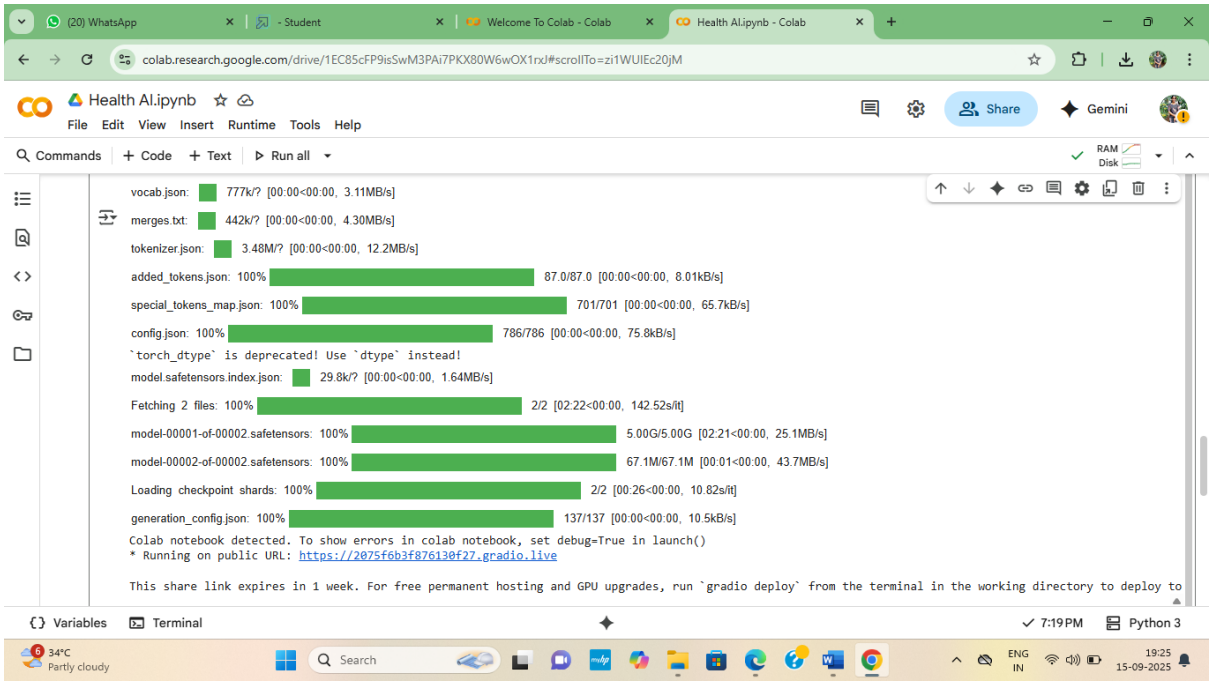


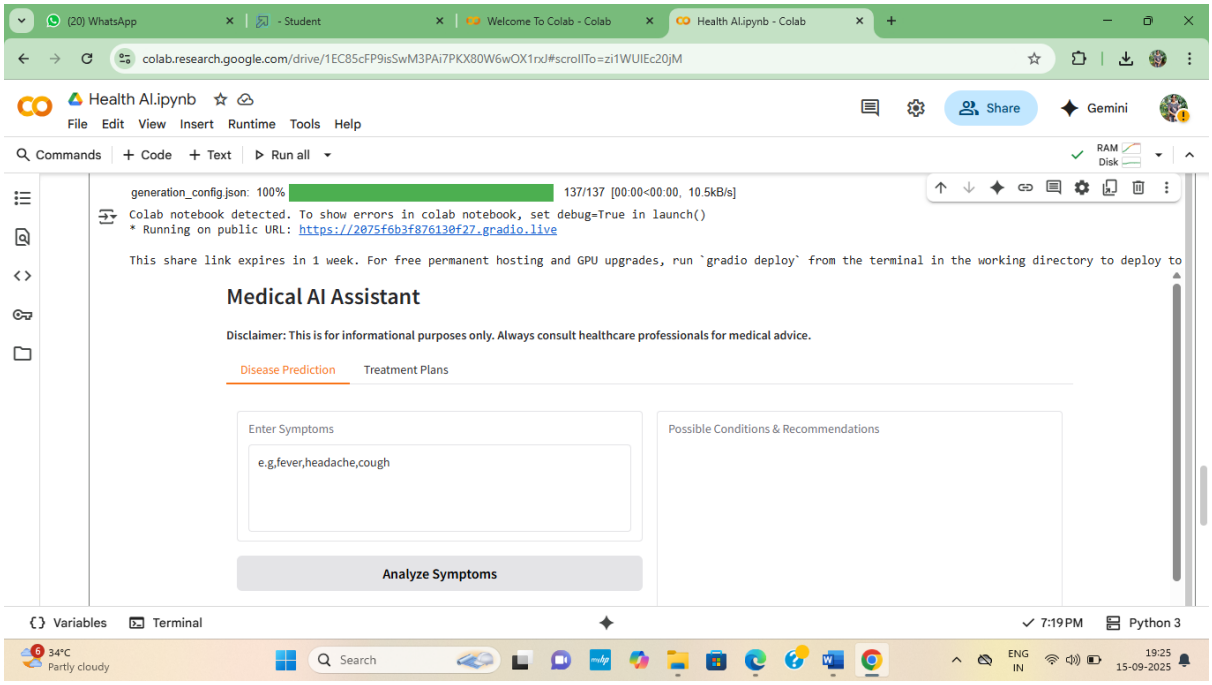
```
        with gr.Tabs():
            with gr.TabItem("Disease Prediction"):
                with gr.Row():
                    with gr.Column():
                        symptoms_input = gr.Textbox(
                            label="Enter Symptoms",
                            placeholder="e.g., fever, headache, cough, fatigue...",
                            lines=4
                        )
                        predict_btn = gr.Button("Analyze Symptoms")

                    with gr.Column():
                        prediction_output = gr.Textbox(label="Possible Conditions & Recommendations", lines=20)

                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(
```



```
                            value="Male"
                        )
                        history_input = gr.Textbox(
                            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 Co
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, 213kB/s]
vocab.json: 777k/? [00:00<00:00, 3.11MB/s]
```





10.CONCLUSION:

HealthAI: Intelligent Healthcare Assistant Using IBM Granite LLM project was successfully done by: 3rd Computer Science 1st Section, D.Parameswari,SVarsha,A.Mohaneshwari and S.Karnikasri.

Demo video link

https://youtube.com/shorts/6BZH6AW9vxI?si=Ah8fnepUMd0ue31F