colab.research.google.com/drive/1fcgsFTvQVsbVArTzpGw0sPEszJVXGMEv

Commands | + Code | + Text | Run all

RAM Disk

[8]

```
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()}

    with torch.no_grad():
        outputs = model.generate(
            **inputs,
            max_length=max_length,
            temperature=0.7,
```

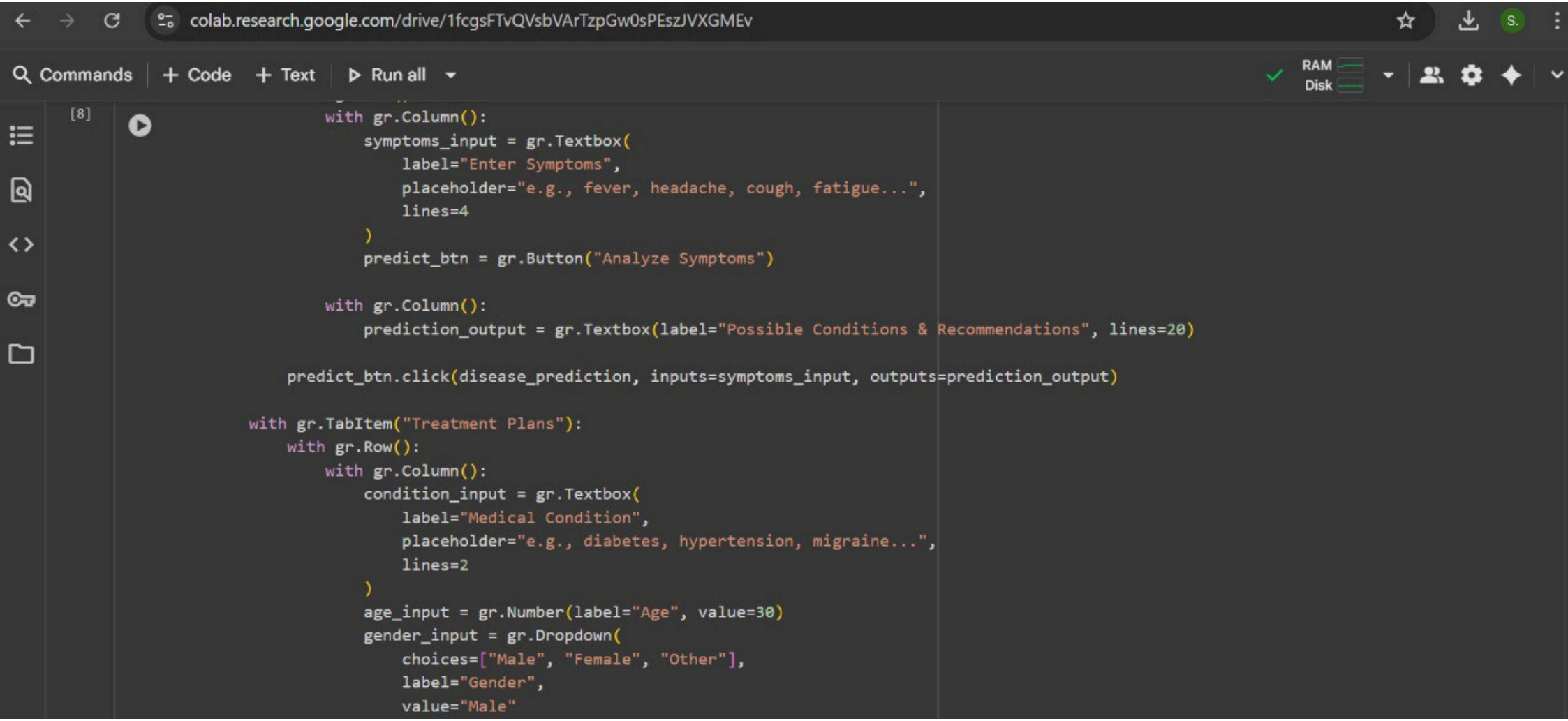
colab.research.google.com/drive/1fcgsFTvQVsbVArTzpGw0sPEszJVXGMEv

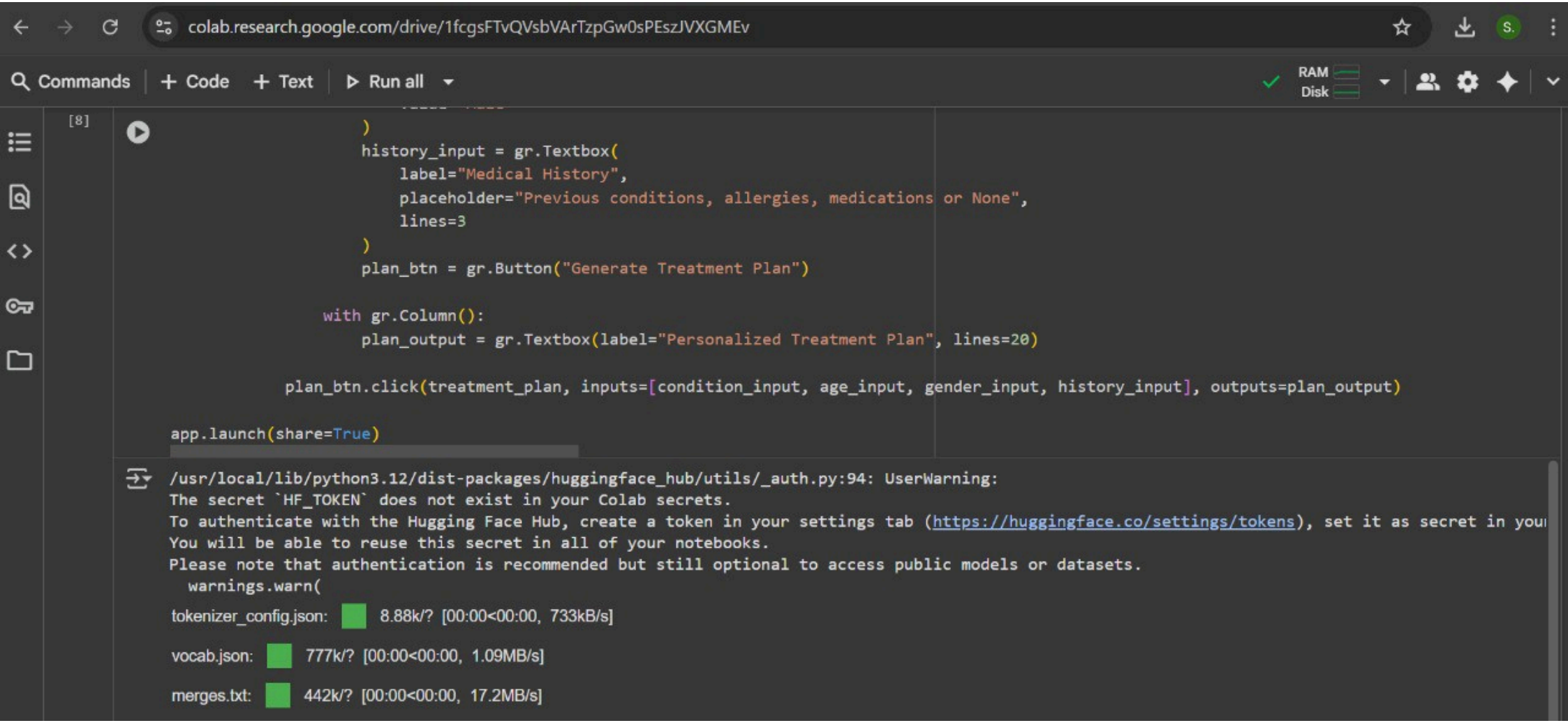
Commands | + Code | + Text | Run all

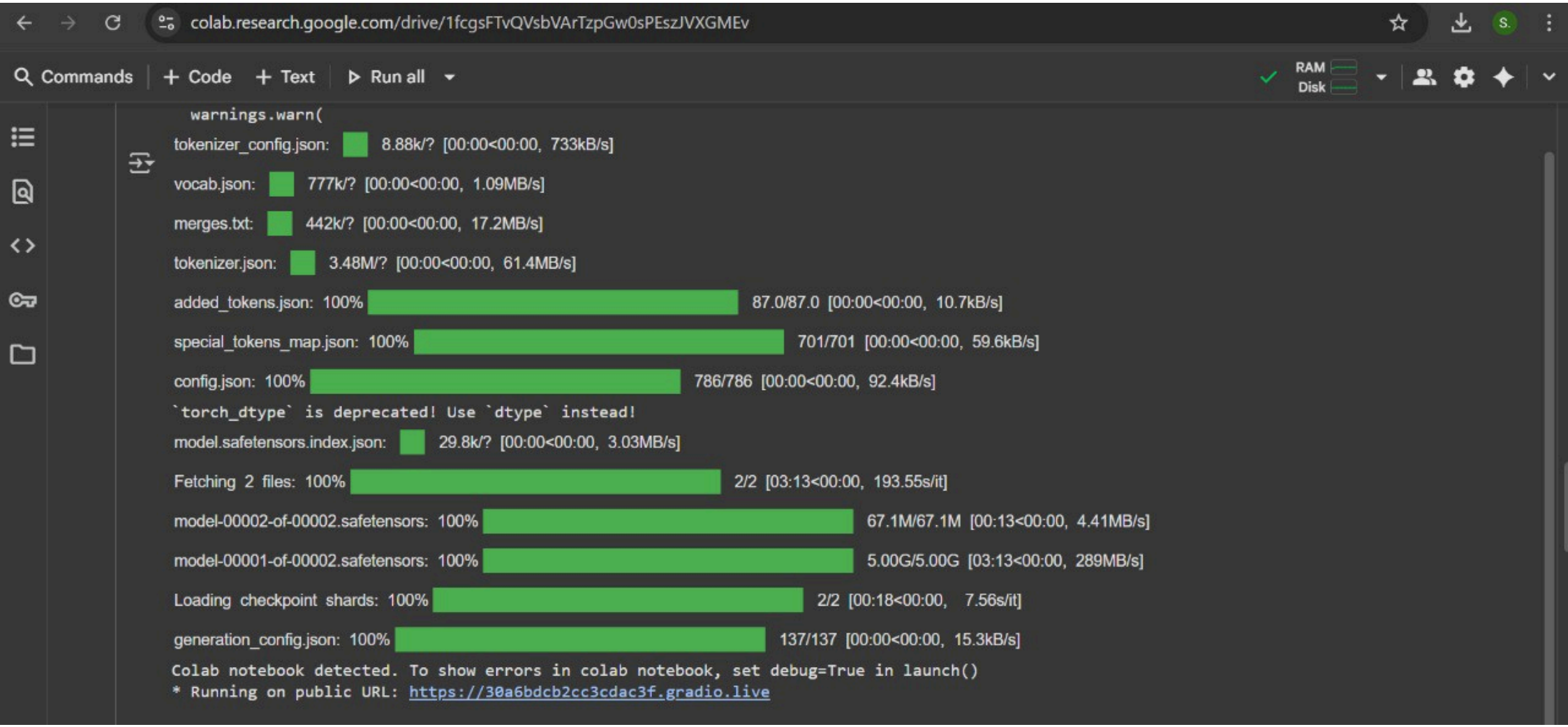
RAM Disk

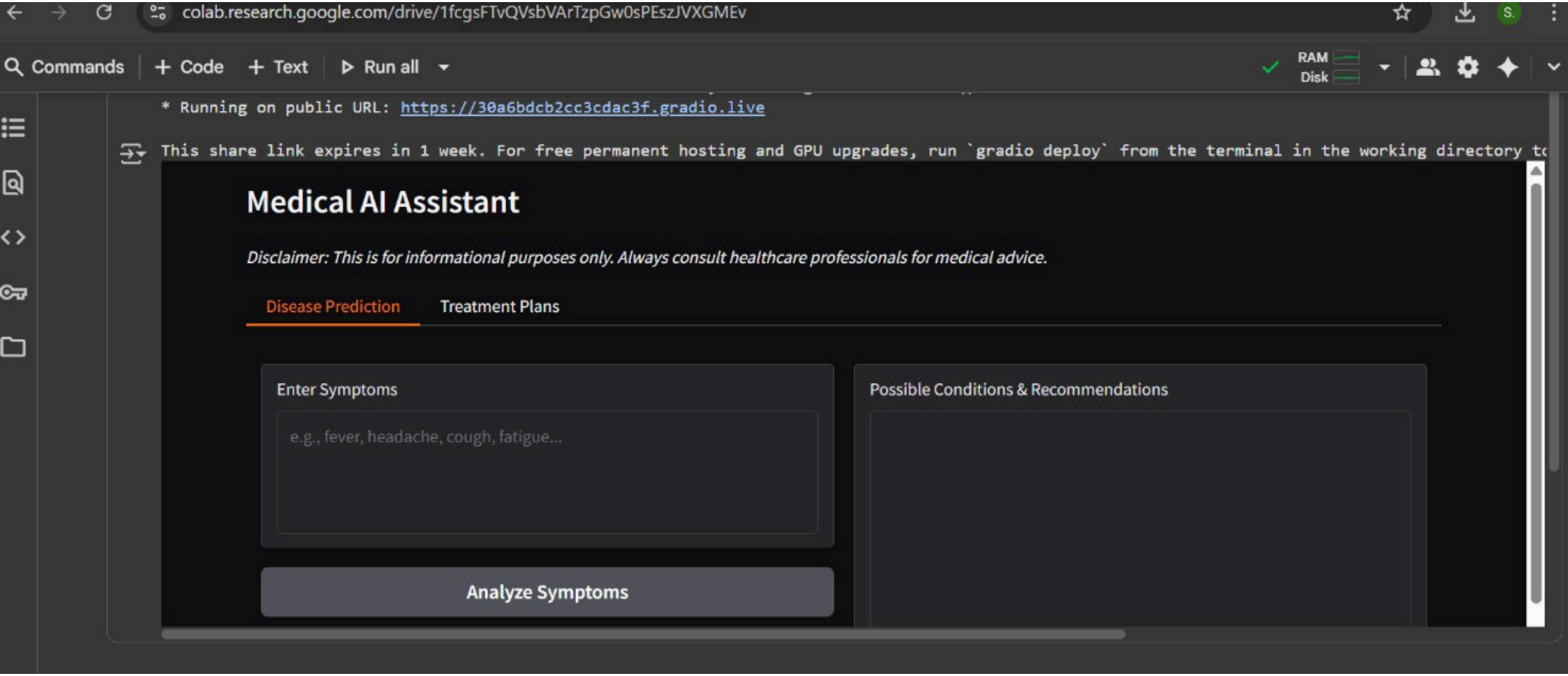
[8]

```
            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 impo
    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 gu
    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():
```


colab.research.google.com/drive/1fcgsFTvQVsbVArTzpGw0sPEszJVXGMEv
Commands
+ Code
+ Text
Run all
RAM
Disk
[8]
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(
choices=["Male", "Female", "Other"],
label="Gender",
value="Male"


colab.research.google.com/drive/1fcgsFTvQVsbVArTzpGw0sPEszJVXGMEv
Commands
+ Code
+ Text
Run all
RAM
Disk
[8]
)
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
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, 733kB/s]
vocab.json: 777k/? [00:00<00:00, 1.09MB/s]
merges.txt: 442k/? [00:00<00:00, 17.2MB/s]


colab.research.google.com/drive/1fcgsFTvQVsbVArTzpGw0sPEszJVXGMEv
Commands
+ Code
+ Text
Run all
RAM
Disk
warnings.warn(
tokenizer_config.json: 8.88k/? [00:00<00:00, 733kB/s]
vocab.json: 777k/? [00:00<00:00, 1.09MB/s]
merges.txt: 442k/? [00:00<00:00, 17.2MB/s]
tokenizer.json: 3.48M/? [00:00<00:00, 61.4MB/s]
added_tokens.json: 100% 87.0/87.0 [00:00<00:00, 10.7kB/s]
special_tokens_map.json: 100% 701/701 [00:00<00:00, 59.6kB/s]
config.json: 100% 786/786 [00:00<00:00, 92.4kB/s]
`torch_dtype` is deprecated! Use `dtype` instead!
model.safetensors.index.json: 29.8k/? [00:00<00:00, 3.03MB/s]
Fetching 2 files: 100% 2/2 [03:13<00:00, 193.55s/it]
model-00002-of-00002.safetensors: 100% 67.1M/67.1M [00:13<00:00, 4.41MB/s]
model-00001-of-00002.safetensors: 100% 5.00G/5.00G [03:13<00:00, 289MB/s]
Loading checkpoint shards: 100% 2/2 [00:18<00:00, 7.56s/it]
generation_config.json: 100% 137/137 [00:00<00:00, 15.3kB/s]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://30a6bdcb2cc3cdac3f.gradio.live


colab.research.google.com/drive/1fcgsFTvQVsbVArTzpGw0sPEszJVXGMEv
Commands
+ Code
+ Text
Run all
RAM
Disk
* Running on public URL: https://30a6bdcb2cc3cdac3f.gradio.live
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
Medical AI Assistant
Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.
Disease Prediction
Treatment Plans
Enter Symptoms
e.g., fever, headache, cough, fatigue...
Analyze Symptoms
Possible Conditions & Recommendations