Health AI: health intelligent Healthcare Assistant

Project Documentation

1.Introduction

- Project title : Health AI:Health intelligent healthcare Assistant
- Team member : Varshini.S
- Team member : Suvetha.S
- Team member : Nivetha.P
- Team member : Sandhiya.E

2.project overview

- Purpose :

  The purpose of a Sustainable Smart City Assistant is to empower cities and their residents to thrive in a more eco-conscious and connected urban environment. By leveraging AI and real-time data, the assistant helps optimize essential resources like energy, water, and waste, while also guiding sustainable behaviors among citizens through personalized tips and services. For city officials, it serves as a decision-making partner—offering clear insights, forecasting tools, and summarizations of complex policies to support strategic planning. Ultimately, this assistant bridges technology, governance, and community engagement to foster greener cities that are more efficient, inclusive, and resilient.
- Features:

Conversational Interface

Key Point: Natural language interaction

Functionality: Allows citizens and officials to ask questions, get updates, and receive guidance in plain language

Policy Summarization

Key Point: Simplified policy understanding

Functionality: Converts lengthy government documents into concise, actionable summaries.

3. Selects treatment generator → gets suggestions.
4. Uses chat module for specific questions.
5. Enters vital signs or health data into analytics module → sees charts, insights.

---

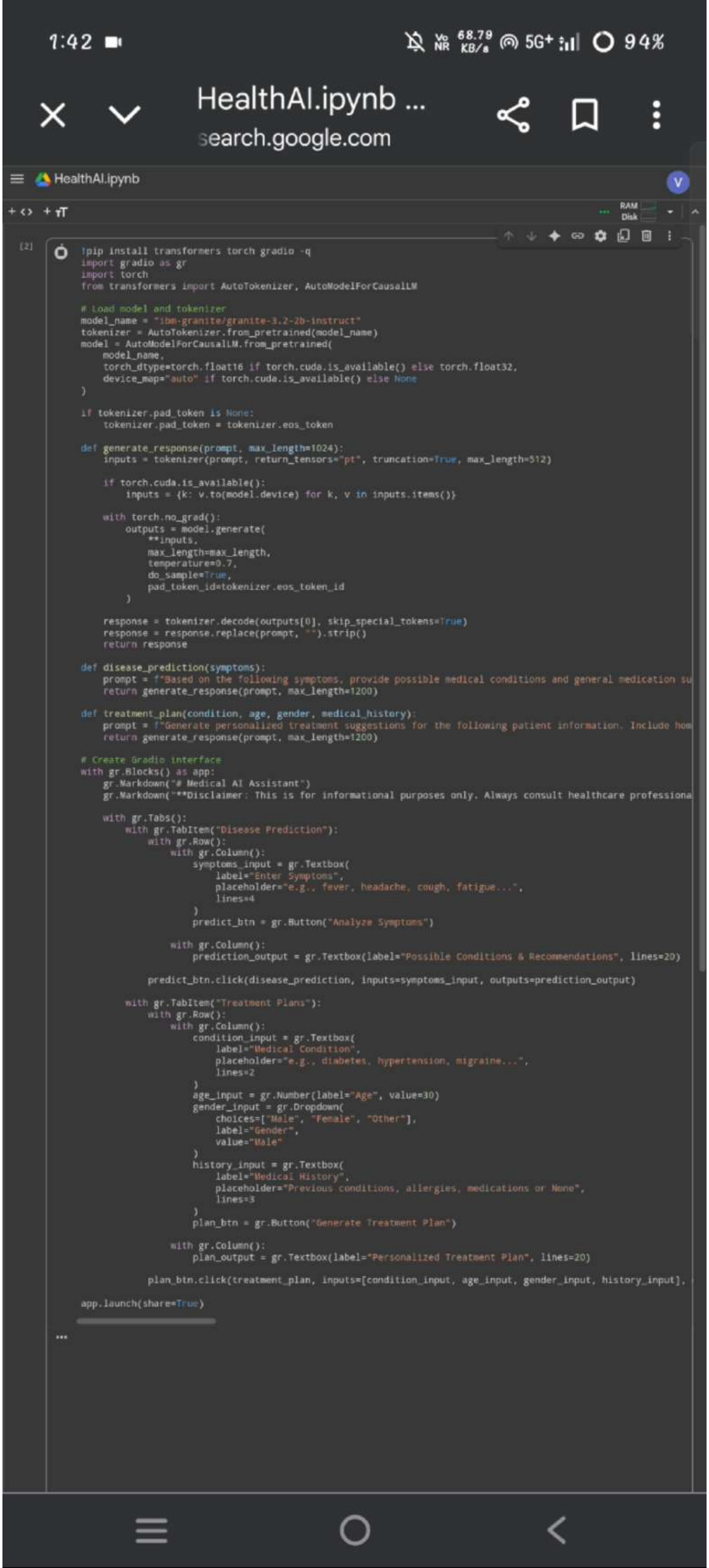
3. Features / Modules

Module	Input
Disease Prediction	User enters symptoms (fre text or form)
Treatment Generator	Diagnosed (or predicted) condition + possibly user profile
Patient Chat Module	Free text queries (health questions)
Health Analytics Dashboard	Vital signs/history (BP, sug pulse, etc.)

---

4. Architecture & Tech Stack

- **Programming Language**: Python
- **Frontend/UI Framework**: Streamlit for quick web UI



1:42 68.79 KB/s 5G+ 94%

HealthAI.ipynb ...
search.google.com

HealthAI.ipynb

RAM
Disk

[2]

```
!pip install transformers torch gradio -q
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,
            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 su
    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 hom
    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 professiona

    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(
                        choices=["Male", "Female", "Other"],
                        label="Gender",
                        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],

app.launch(share=True)
```

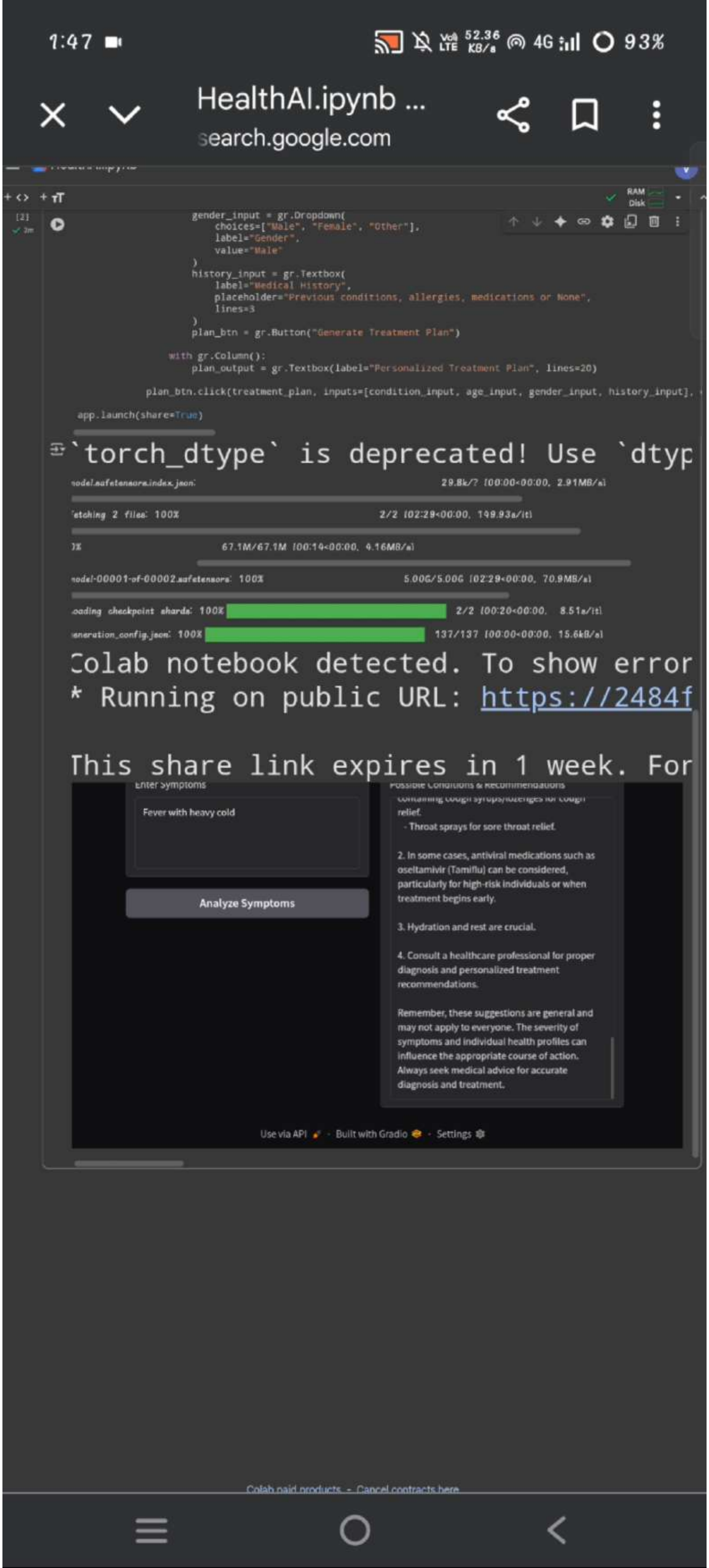
...

4. Architecture & Tech Stack

- **Programming Language**: Python
- **Frontend/UI Framework**: Streamlit for quick web UI prototyping and dashboards Scribd +1
- **AI / Model**: IBM Granite (Granite-13b-instruct-v2) for generative inference. Scribd +1
- **Data Handling**: Pandas, NumPy for data manipulation; Matplotlib / Seaborn for visualization. Scribd +2
- **Version Control & Hosting**: GitHub; possible use of hosting platform (e.g. cloud services) for deployment. Scribd +1
- **Folder / Code Structure** (based on what's reported):

Copy code

```
HealthAI/
  ├── main.py
  ├── disease_predictor.py
  ├── treatment_generator.py
  ├── chat_interface.py
  ├── analytics_dashboard.py
  ├── requirements.txt
  └── README.md
```



```
            gender_input = gr.Dropdown(
                choices=["Male", "Female", "Other"],
                label="Gender",
                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],

app.launch(share=True)
```

```
`torch_dtype` is deprecated! Use `dtyp
nodel.safetensors.index.json:                    29.8k/? [00:00<00:00, 2.91MB/s]
'etching 2 files: 100%                           2/2 [02:29<00:00, 149.93s/it]
)%                  67.1M/67.1M [00:14<00:00, 4.16MB/s]
nodel-00001-of-00002.safetensors: 100%           5.00G/5.00G [02:29<00:00, 70.9MB/s]
.oading checkpoint shards: 100%                  2/2 [00:20<00:00,  8.51s/it]
eneration_config.json: 100%                      137/137 [00:00<00:00, 15.6kB/s]
Colab notebook detected. To show error
* Running on public URL: https://2484f

This share link expires in 1 week. For
```

Enter Symptoms

Fever with heavy cold

Analyze Symptoms

Possible Conditions & Recommendations

containing cough syrups/lozenges for cough relief.
- Throat sprays for sore throat relief.

2. In some cases, antiviral medications such as oseltamivir (Tamiflu) can be considered, particularly for high-risk individuals or when treatment begins early.

3. Hydration and rest are crucial.

4. Consult a healthcare professional for proper diagnosis and personalized treatment recommendations.

Remember, these suggestions are general and may not apply to everyone. The severity of symptoms and individual health profiles can influence the appropriate course of action. Always seek medical advice for accurate diagnosis and treatment.

Use via API · Built with Gradio · Settings
Colab paid products - Cancel contracts here

5. API / Model Interaction

- **Model Endpoint**: Either local model loading or via Hugging Face API. E.g., IBM Granite model endpoint. Studocu
- **Authentication**: If using external API (like Hugging Face), use API tokens stored securely (e.g. via `.env`) and not committed in source. Studocu
- **Input Format**:
  - For disease prediction: symptoms in structured or free text form
  - For treatment plan: the condition name + possibly user profile or preferences
  - For chat: natural language questions
  - For analytics: numerical or historical data (vitals, time series)
- **Output Format**: Usually plain text / JSON (if API) + charts / visualization (for analytics)
- **Error Handling**: Deal with invalid inputs (missing fields, malformed symptoms), timeouts in model response, etc.

6. Installation & Deployment

Requirements

- Python 3.x
- Libraries: (as per `requirements.txt`) — streamlit, pandas, numpy, matplotlib, model-inference related libs, etc. Scribd +1
- Possible external dependencies: Hugging Face account / token if using their model; IBM Granite access.

Setup Steps

1. Clone repository from GitHub
2. Create virtual environment, install dependencies:

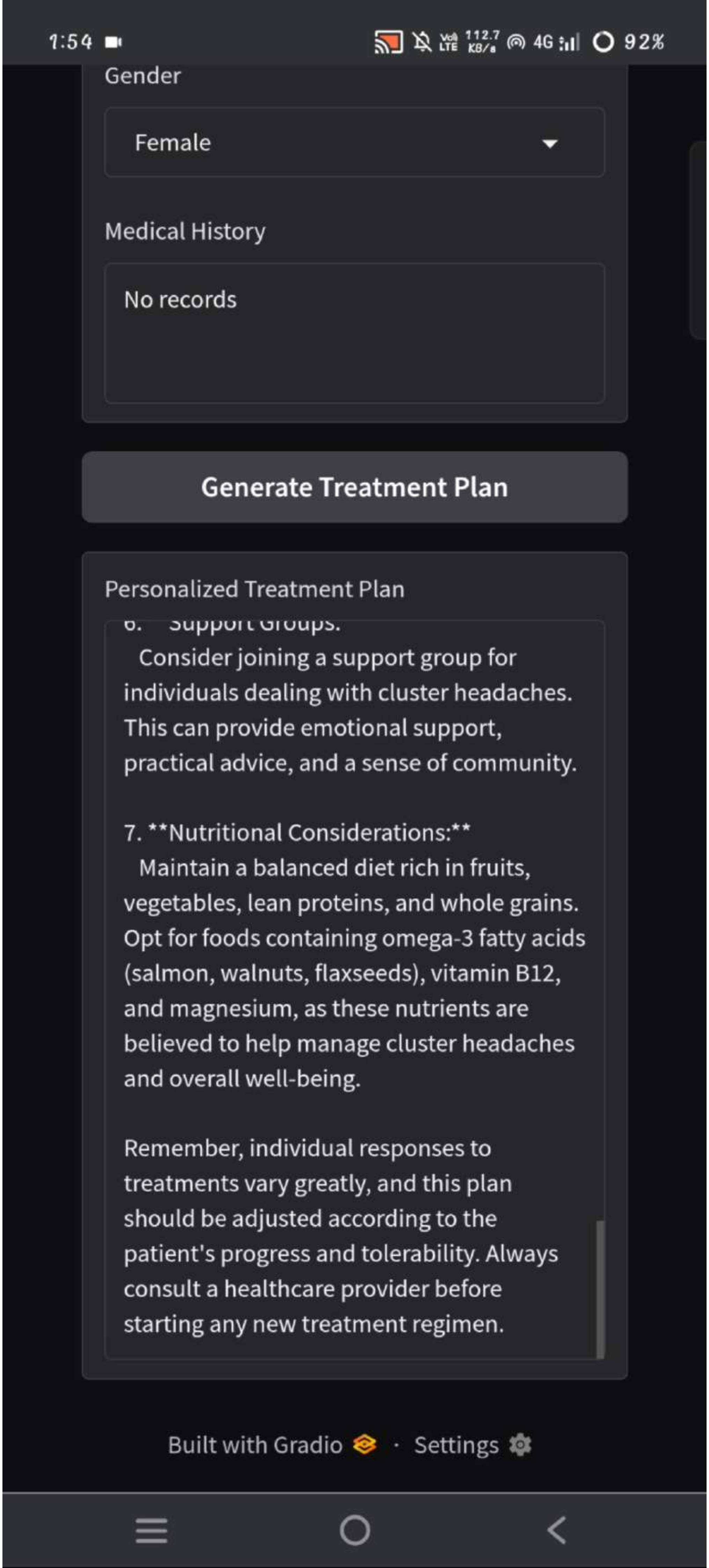
   Bash Copy code

   ```bash
   pip install -r requirements.txt
   ```

3. Set up API token / model files (`.env`)
4. Run locally:

   Bash Copy code

   ```bash
   streamlit run main.py
   ```



Gender

Female

Medical History

No records

Generate Treatment Plan

Personalized Treatment Plan

6. Support Groups:
  Consider joining a support group for individuals dealing with cluster headaches. This can provide emotional support, practical advice, and a sense of community.

7. **Nutritional Considerations:**
  Maintain a balanced diet rich in fruits, vegetables, lean proteins, and whole grains. Opt for foods containing omega-3 fatty acids (salmon, walnuts, flaxseeds), vitamin B12, and magnesium, as these nutrients are believed to help manage cluster headaches and overall well-being.

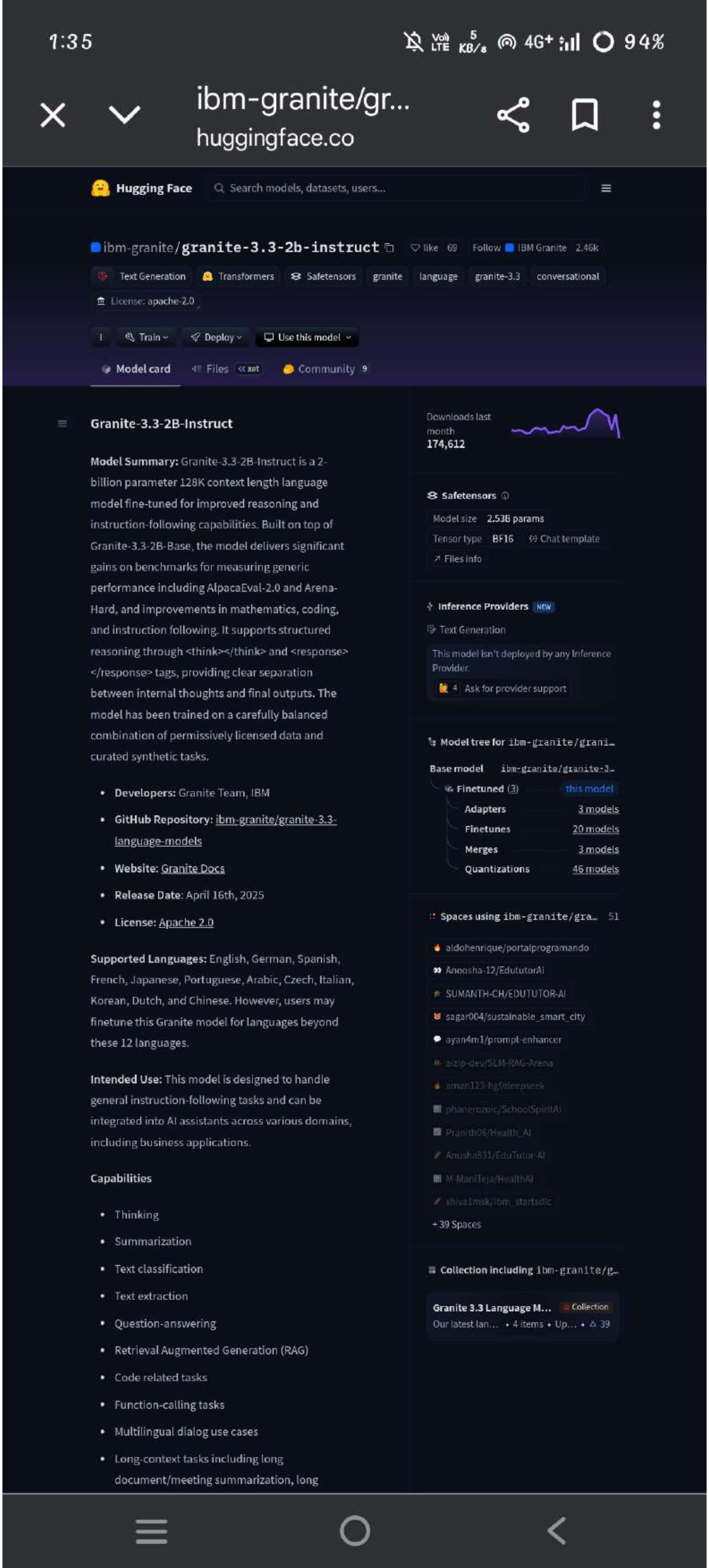
Remember, individual responses to treatments vary greatly, and this plan should be adjusted according to the patient's progress and tolerability. Always consult a healthcare provider before starting any new treatment regimen.

7. Testing & Validation

- **Unit Testing**: For modules (disease prediction logic, treatment generator) to check expected outputs for given inputs.
- **Prompt Testing**: Use varied prompts/questions to ensure model responses are coherent, medically reasonable.
- **Edge Cases**: No symptoms, contradictory inputs, rare disease symptoms — ensure system handles these gracefully.
- **User Acceptance**: (If possible) feedback from medical professionals / domain experts.

8. Limitations, Risks & Ethical Considerations

- **Non-clinical tool**: HealthAI is for guidance, not diagnosis. Must include disclaimers.
- **Bias & Data Quality**: Model may reflect biases from training data; predictions may be inaccurate.
- **Privacy & Security**: If taking personal health data, ensure secure transmission, storage, user consent.



1:35 94%

ibm-granite/gr...
huggingface.co

Hugging Face Search models, datasets, users...

ibm-granite/**granite-3.3-2b-instruct** like 69 Follow IBM Granite 2.46k

Text Generation · Transformers · Safetensors · granite · language · granite-3.3 · conversational

License: apache-2.0

Train · Deploy · Use this model

Model card · Files xet · Community 9

Granite-3.3-2B-Instruct

Model Summary: Granite-3.3-2B-Instruct is a 2-billion parameter 128K context length language model fine-tuned for improved reasoning and instruction-following capabilities. Built on top of Granite-3.3-2B-Base, the model delivers significant gains on benchmarks for measuring generic performance including AlpacaEval-2.0 and Arena-Hard, and improvements in mathematics, coding, and instruction following. It supports structured reasoning through <think></think> and <response></response> tags, providing clear separation between internal thoughts and final outputs. The model has been trained on a carefully balanced combination of permissively licensed data and curated synthetic tasks.

- **Developers:** Granite Team, IBM
- **GitHub Repository:** ibm-granite/granite-3.3-language-models
- **Website:** Granite Docs
- **Release Date**: April 16th, 2025
- **License:** Apache 2.0

Supported Languages: English, German, Spanish, French, Japanese, Portuguese, Arabic, Czech, Italian, Korean, Dutch, and Chinese. However, users may finetune this Granite model for languages beyond these 12 languages.

Intended Use: This model is designed to handle general instruction-following tasks and can be integrated into AI assistants across various domains, including business applications.

Capabilities

- Thinking
- Summarization
- Text classification
- Text extraction
- Question-answering
- Retrieval Augmented Generation (RAG)
- Code related tasks
- Function-calling tasks
- Multilingual dialog use cases
- Long-context tasks including long document/meeting summarization, long

Downloads last month
174,612

Safetensors
Model size 2.53B params
Tensor type BF16 · Chat template
Files info

Inference Providers NEW
Text Generation
This model isn't deployed by any Inference Provider.
4 Ask for provider support

Model tree for ibm-granite/grani...

Base model	ibm-granite/granite-3...
Finetuned (3)	this model
Adapters	3 models
Finetunes	20 models
Merges	3 models
Quantizations	46 models

Spaces using ibm-granite/gra... 51

- aldohenrique/portalprogramando
- Anoosha-12/EdututorAI
- SUMANTH-CH/EDUTUTOR-AI
- sagar004/sustainable_smart_city
- ayan4m1/prompt-enhancer
- aizip-dev/SLM-RAG-Arena
- aman123-hgf/deepseek
- phanerozoic/SchoolSpiritAI
- Pranith06/Health_AI
- Anusha831/EduTutor-AI
- M-ManiTeja/HealthAI
- shiva1msk/ibm_startsdic

+39 Spaces

Collection including ibm-granite/g...

Granite 3.3 Language M... Collection
Our latest lan... • 4 items • Up... • 39

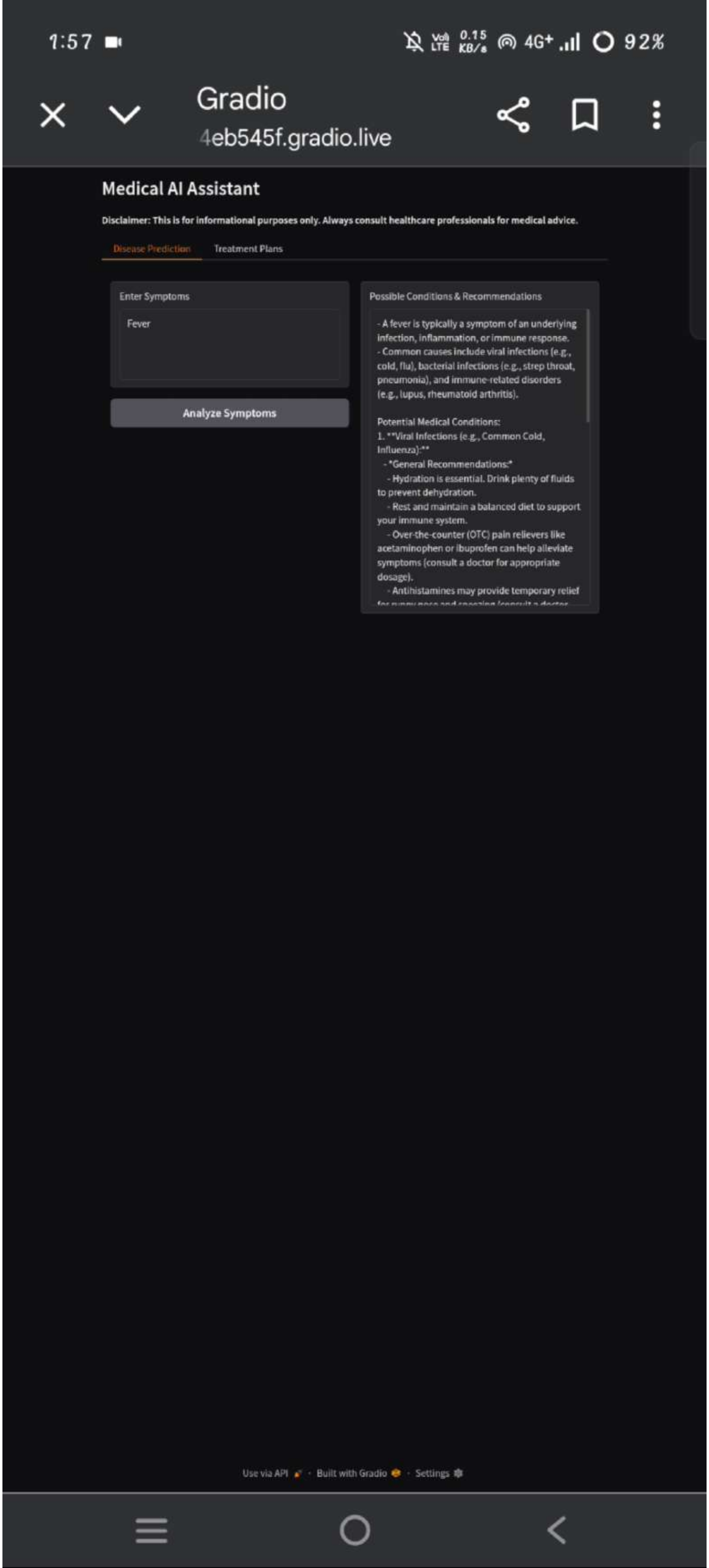
9. Future Work & Enhancements

- Real-time health monitoring (e.g. wearable integrations)
- Enhanced chatbot with better context memory + multilingual support
- More modules: image analysis, lab reports, etc.
- Backend improvements: more robust REST/API interface, user authentication, logging / audit trails
- Regulatory compliance, clinical validation studies

10. References & Glossary

References

- Original project documentation / academic write-ups (if any) Studocu +1
- IBM Granite model documentation
- Literature on AI in health applications



Medical AI Assistant

Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.

Enter Symptoms

Fever

Possible Conditions & Recommendations

```
- A fever is typically a symptom of an underlying infection, inflammation, or immune response.
- Common causes include viral infections (e.g., cold, flu), bacterial infections (e.g., strep throat, pneumonia), and immune-related disorders (e.g., lupus, rheumatoid arthritis).

Potential Medical Conditions:
1. **Viral Infections (e.g., Common Cold, Influenza):**
  - *General Recommendations:*
    - Hydration is essential. Drink plenty of fluids to prevent dehydration.
    - Rest and maintain a balanced diet to support your immune system.
    - Over-the-counter (OTC) pain relievers like acetaminophen or ibuprofen can help alleviate symptoms (consult a doctor for appropriate dosage).
    - Antihistamines may provide temporary relief
```