1. INTRODUCTION

1.1 Project Overview

HealthAI is a web-based intelligent healthcare assistant built using **Streamlit**, **IBM Granite LLM via Hugging Face**, and **Python**. It enables users to:

- Ask health-related questions.
- Predict diseases based on symptoms.
- Generate treatment plans.
- Analyze health data (e.g., heart rate, glucose levels) via CSV uploads.

> ⚠️ Disclaimer: HealthAI is an informational tool and not a substitute for professional medical advice.

1.2 Purpose

To create a user-friendly AI-powered assistant that:

- Educates users about symptoms, conditions, and treatments.
- Supports medical professionals with treatment suggestions.
- Visualizes personal health data for insight generation.

---

2. IDEATION PHASE

2.1 Problem Statement

People often struggle with understanding symptoms and navigating basic healthcare decisions due to limited access to medical professionals or trusted resources.

2.2 Empathy Map Canvas

- **Think/Feel:** Need health clarity, fear of conditions.
- **Hear:** Advice from peers, online health myths.
- **See:** Conflicting info online, delayed diagnosis.
- **Say/Do:** Google symptoms, self-diagnose.
- **Pain Points:** Uncertainty, anxiety, misinformation.
- **Gain:** Clarity, faster info, AI-assisted suggestions.

2.3 Brainstorming

- AI for health Q&A
- Disease prediction from symptoms
- CSV-based health analytics
- Natural remedy suggestions
- Voice/Multilingual support

---

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

1. User visits HealthAI.
2. Selects a module (Q&A, Prediction, etc.).
3. Inputs data (text/symptoms/CSV).
4. Receives AI-driven response.
5. Gains insight or takes informed action.

3.2 Solution Requirement

- Frontend: Streamlit UI
- Backend: Python + Hugging Face API
- Data Input: Text, CSV upload
- Output: AI-generated suggestions

3.3 Data Flow Diagram

text
CopyEdit

```
User Input (Symptom/CSV/Text) →
API Request to Hugging Face Model →
Receive Prediction/Insight →
Display in Streamlit Interface
```

3.4 Technology Stack

- **Frontend:** Streamlit
- **Backend:** Python
- **AI/ML:** IBM Granite LLM (via Hugging Face)
- **Data Handling:** Pandas
- **APIs:** Hugging Face Inference API
- **Environment Variables:** `dotenv`

---

4. PROJECT DESIGN

4.1 Problem-Solution Fit

Users want accessible, quick, and accurate health-related information without relying entirely on search engines or waiting for appointments.

4.2 Proposed Solution

HealthAI addresses this by offering a conversational, interactive AI platform with multiple modules for personalized insights and health analytics.

4.3 Solution Architecture

- **Frontend:** Streamlit UI
- **Model Query:** Prompt → IBM Granite via API
- **Modules:** Q&A, Disease Prediction, Treatment Plan, CSV Analytics

---

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Task	Timeline
Ideation & Research	2 Days
UI/UX Design (Streamlit)	2 Days
API Integration (LLM)	3 Days
Feature Implementation	4 Days
Testing & Debugging	2 Days
Final Review & Documentation	2 Days

---

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

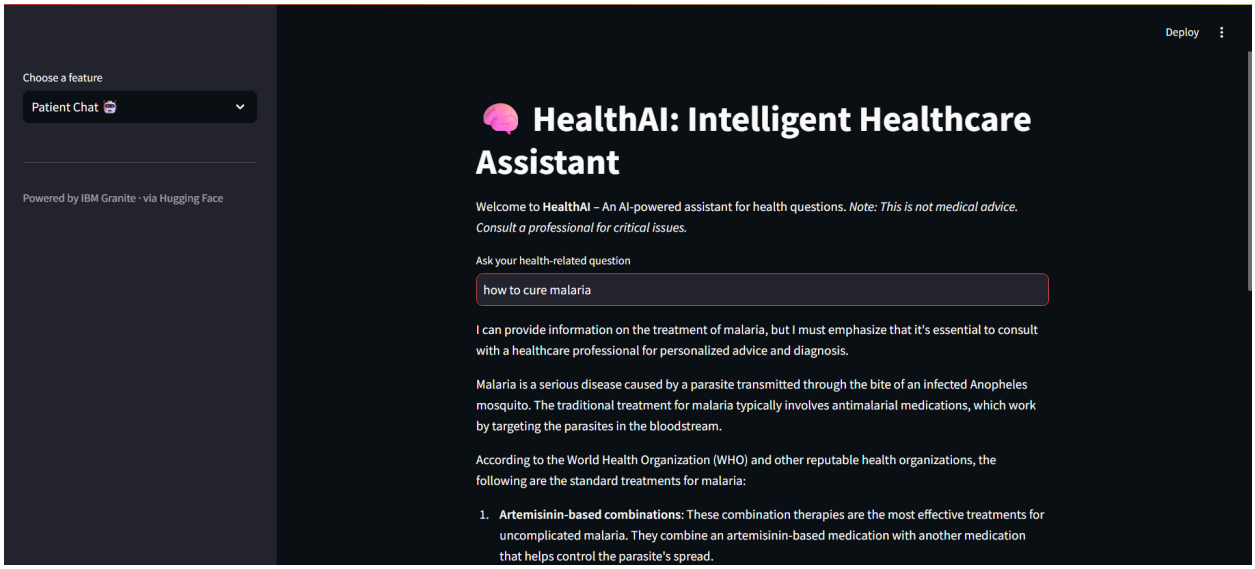
- **Response Time:** Avg. < 3 sec for model queries.
- **Scalability:** Can handle simultaneous users via Streamlit Cloud.
- **Accuracy:** Dependent on LLM's trained knowledge base.

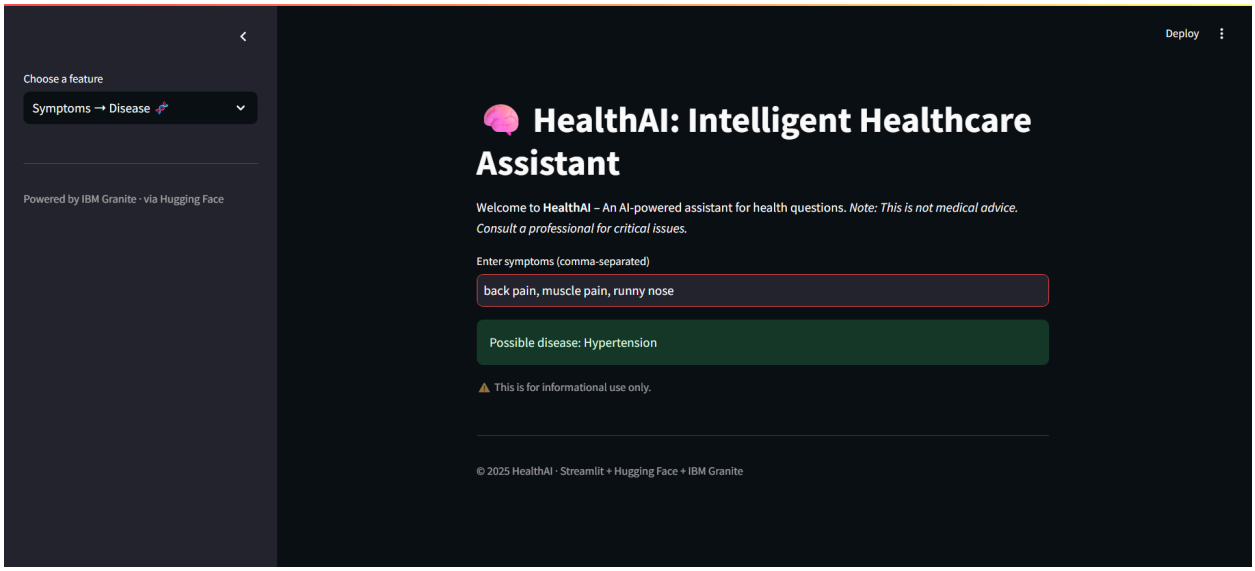
---

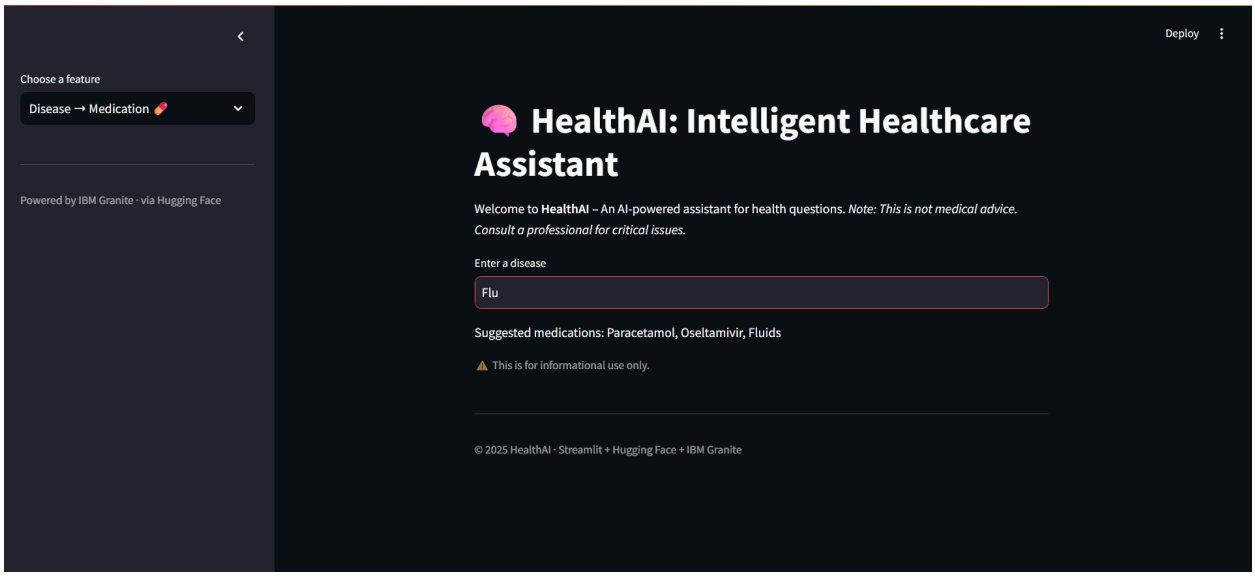
7. RESULTS

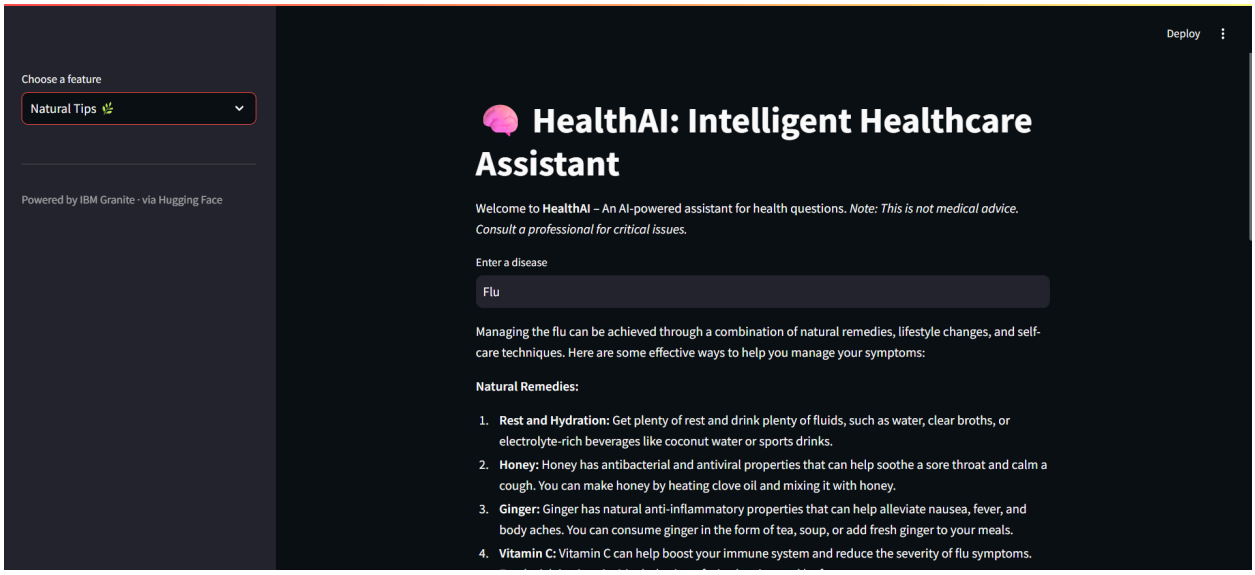
7.1 Output Screenshots

- Health Q&A response
- Disease prediction result
- Treatment plan generation
- CSV analytics chart + AI summary




Choose a feature
Symptoms → Disease
Powered by IBM Granite · via Hugging Face
Deploy
HealthAI: Intelligent Healthcare Assistant
Welcome to HealthAI – An AI-powered assistant for health questions. Note: This is not medical advice. Consult a professional for critical issues.
Enter symptoms (comma-separated)
back pain, muscle pain, runny nose
Possible disease: Hypertension
This is for informational use only.
© 2025 HealthAI · Streamlit + Hugging Face + IBM Granite


Choose a feature
Disease → Medication
Powered by IBM Granite · via Hugging Face
Deploy
HealthAI: Intelligent Healthcare Assistant
Welcome to HealthAI – An AI-powered assistant for health questions. Note: This is not medical advice. Consult a professional for critical issues.
Enter a disease
Flu
Suggested medications: Paracetamol, Oseltamivir, Fluids
This is for informational use only.
© 2025 HealthAI · Streamlit + Hugging Face + IBM Granite



---

8. ADVANTAGES & DISADVANTAGES

Advantages	Disadvantages
Easy to use and interactive UI	Relies on third-party API (internet needed)
AI-based fast suggestions	Cannot provide real-time clinical validation
Multi-feature platform (Q&A, Prediction)	CSV file format must match required fields

---

9. CONCLUSION

HealthAI successfully demonstrates how LLMs can be integrated with intuitive interfaces to create intelligent healthcare tools. It empowers users with insights and supports healthcare awareness and monitoring.

---

10. FUTURE SCOPE

- 🔊 Voice Input & TTS Output
- 🌐 Multilingual Support (Hindi, Spanish, etc.)
- 🗂️ EMR/EHR Integration
- 🔐 User Authentication & History
- 💻 Offline Mode using lightweight models

11. APPENDIX

- **Source Code:**

```
import streamlit as st
import requests
import os
import pandas as pd
from dotenv import load_dotenv

# Load .env variables
load_dotenv()
HF_API_KEY = os.getenv("HF_API_KEY")

API_URL =
"https://api-inference.huggingface.co/models/ibm-granite/granite-3.3-2b-in
struct"
HEADERS = {"Authorization": f"Bearer {HF_API_KEY}"}
# Function to query Hugging Face Inference API
def query_model(prompt, max_tokens=300, temperature=0.7):
    payload = {
        "inputs": prompt,
        "parameters": {
            "max_new_tokens": max_tokens,
            "temperature": temperature,
            "do_sample": True
        }
    }
```

```
    response = requests.post(API_URL, headers=HEADERS, json=payload)
    if response.status_code == 200:
        result = response.json()
        return result[0]["generated_text"]
    else:
        return f"⚠️ API Error: {response.status_code} - {response.text}"

# Streamlit layout
st.set_page_config(page_title="HealthAI", layout="wide")
st.title("🧠 HealthAI: Intelligent Healthcare Assistant")

menu = st.sidebar.selectbox("Choose a Module", [
    "👨💎 Patient Chat",
    "🔍 Disease Prediction",
    "💊 Treatment Plans",
    "📊 Health Analytics"
])

st.sidebar.markdown("---")
st.sidebar.caption("Powered by IBM Granite · via Hugging Face")

# 1. Patient Chat
if menu == "👨💎 Patient Chat":
    st.header("💬 Ask a Health Question")
    question = st.text_input("What would you like to ask?")
    if st.button("Get Answer") and question.strip():
        prompt = f"User: {question}\nHealthAI:"
        with st.spinner("Thinking..."):
            result = query_model(prompt)
            response = result.split("HealthAI:")[-1].strip()
        st.success(response)
        st.caption("⚠️ This is for informational use only.")

# 2. Disease Prediction
elif menu == "🔍 Disease Prediction":
    st.header("🩺 Symptom-Based Disease Prediction")
    symptoms = st.text_area("Enter your symptoms (comma separated)")
    age = st.slider("Age", 0, 100, 30)
    gender = st.selectbox("Gender", ["Male", "Female", "Other", "Prefer
not to say"])
```

```
    if st.button("Predict Disease") and symptoms:
        prompt = (
            f"A patient reports the following symptoms: {symptoms}. "
            f"Age: {age}, Gender: {gender}. "
            f"Based on this, what are the top 3 likely conditions with
brief explanations?"
        )
        with st.spinner("Analyzing..."):
            result = query_model(prompt, max_tokens=400, temperature=0.8)
            response = result.split("conditions")[-1].strip()
        st.success("🔍 Predicted Conditions:")
        st.write(response)
        st.caption("📌 Consult a doctor for confirmation.")

# 3. Treatment Plans
elif menu == "💊 Treatment Plans":
    st.header("📋 Treatment Plan Generator")
    condition = st.text_input("Enter a diagnosed condition")
    if st.button("Generate Plan") and condition:
        prompt = (
            f"Create a detailed treatment plan for {condition}. "
            f"Include medication, lifestyle changes, and recommended
tests."
        )
        with st.spinner("Generating plan..."):
            result = query_model(prompt, max_tokens=400)
            response = result.split("plan")[-1].strip()
        st.success("🧾 Suggested Treatment:")
        st.write(response)

# 4. Health Analytics
elif menu == "📊 Health Analytics":
    st.header("📈 Upload & Analyze Health Data")
    uploaded_file = st.file_uploader("Upload your health data CSV (date,
heart_rate, bp, glucose)", type="csv")

    if uploaded_file:
        df = pd.read_csv(uploaded_file, parse_dates=["date"])
        df.set_index("date", inplace=True)
```

```
        st.subheader("🧪 Metrics Summary")
        metric = st.selectbox("Select a metric to visualize", df.columns)
        st.line_chart(df[metric])
        st.write(df[metric].describe())

        # AI Insight
        prompt = f"Analyze this time series health data:
{df[metric].tolist()[:20]}. What patterns or risks do you notice?"
        with st.spinner("Analyzing trend..."):
            result = query_model(prompt, max_tokens=200, temperature=0.6)
            insight = result.split("data:")[-1].strip()
        st.success("🧠 AI Insight:")
        st.write(insight)

# Footer
st.markdown("---")
st.caption("© 2025 HealthAI · Streamlit + Hugging Face + IBM Granite")
```

GitHub:https://github.com/Charancholaveti/HealthAI