**SMART BRIDGE INTERNSHIP PROJECT**

**Project Title:**

**HealthAI:Intelligent Healthcare Assistant Using IBM Granite**

**Submitted By:**

**BOKKA VINAY**

**Roll No:22NG1A0507**

**Project Report Format**

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. IDEATION PHASE

2.1 Problem Statement

2.2 Empathy Map Canvas

2.3 Brainstorming

3. REQUIREMENT ANALYSIS

3.1 Customer Journey map

3.2 Solution Requirement

3.3 Data Flow Diagram

3.4 Technology Stack

4. PROJECT DESIGN

4.1 Problem Solution Fit

4.2 Proposed Solution

4.3 Solution Architecture

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

7. RESULTS

7.1 Output Screenshots

8. ADVANTAGES & DISADVANTAGES

9. CONCLUSION

10. FUTURE SCOPE

11. APPENDIXSource Code(if any),Dataset Link,GitHub & Project Demo Link

**1. INTRODUCTION**

**1.1 Project Overview**

In the modern healthcare ecosystem, timely and personalized medical assistance is essential. Health AI is an intelligent virtual assistant designed to assist both patients and healthcare providers by offering AI-driven medical recommendations. This assistant is built using IBM Granite Foundation Models, leveraging their capability to process medical inputs, symptoms, and generate real-time treatment suggestions or health advice.

Health AI integrates natural language processing (NLP), a user-friendly interface, and secure backend services to simulate conversations between patients and AI health advisors. The goal is to reduce human error, save time, and improve healthcare accessibility for users globally.

**1.2 Purpose**

The primary purpose of Health AI is to assist users in getting quick medical advice without replacing actual doctors. It aims to:

Provide first-level support based on symptoms.

Educate users on common diseases.

Recommend treatment paths using AI.

Enable scalable, 24/7 healthcare support.

**2. IDEATION PHASE**

**2.1 Problem Statement**

India and other countries face a shortage of doctors and overburdened healthcare facilities. Patients often wait long hours to consult for minor illnesses. There is a need for an intelligent system that can offer preliminary suggestions, reducing unnecessary hospital visits. HealthAI addresses this gap.

**2.2 Empathy Map Canvas**

Users: Patients, caregivers, general public.

Says: "I need help but don't want to wait for hours", "I have these symptoms, what could it be?"Thinks: "Is it serious?", "Do I need to visit the hospital?", "Is there a home remedy?"

Feels: Confused, anxious, sometimes embarrassed to talk about symptoms.

Does: Searches Google, uses WhatsApp to ask friends, delays doctor visit

**2.3 Brainstorming**

During the brainstorming sessions, various ideas were proposed:

Building a chatbot with basic symptom lookup.

Creating a model trained on disease-treatment datasets.

Using IBM Watson or other AI models for accurate recommendations.

Designing a mobile interface for easy access.

The team finalized using IBM Granite models for generative responses and Streamlit for the user interface.

**3. REQUIREMENT ANALYSIS**

**3.1 Customer Journey Map**

Stage Action Emotion Touchpoints Awareness Learns about Health AI Curious Web, social media Consideration Tries Health AI with symptoms Hopeful Web App usage Interacts with AI assistant Confident Stream lit App Follow-up Uses suggestion or visits doctor Relieved Hospital visit, app.

**3.2 Solution Requirement**

**Functional Requirements:**

**Requirements.txt:**

Stream lit

ibm-watson-machine-learning

requests

matplotlib

pandas

numpy

python-dotenv

scikit-learn

plotly

**.env:**

WML\_API\_KEY= ZPvG-12ijx5ATRN6BQ55q9hI5iqQrF5B5c55VHWYyV-U

WML\_INSTANCE\_ID=693633cd-ac3b-4230-9eda-e179323e00a7

WML\_URL=https://us-south.ml.cloud.ibm.com

Input symptom descriptions.

AI-generated suggestions.

View treatment plans or advice.

**Features:**

Home

Disease prediction

Treatement plan

Patient chat

Health Analytics

**Non-functional Requirements:**

Fast response time.

Secure data handling.

Scalability for multiple users.

**3.3 Data Flow Diagram**

The Health AI system leverages blockchain technology and a secure layered architecture to ensure data privacy, transparency, and efficient healthcare management. The architecture can be divided into three main layers: User Layer, Frontend, and Backend.

**1. User Layer**

This layer consists of the main stakeholders who interact with the application:

**Administrator**: Manages access control, system settings, and user management.

**Doctor**: Views and updates patient data, diagnoses, and treatment recommendations.

**Patient**: Can access personal medical records and interact with doctors securely.

Each user interacts with the system through a secure interface that connects to the application layer.

**2. Frontend (Application Layer)**

**components include:**

The frontend handles communication between users and the backend. Key Public & Private Key Generation Each user is assigned a cryptographic key pair for secure communication.

**Encryption and Digital Signature:** Data is encrypted and digitally signed to maintain integrity and confidentiality.

**Smart Contracts:** Used to automate validation of medical transactions based on pre-defined rules.

**Application Interface:** Users interact with the system via a web or mobile-based UI.

The application securely transmits medical data for validation using blockchain technology.

**3. Backend**

This layer handles data processing, storage, and blockchain verification:

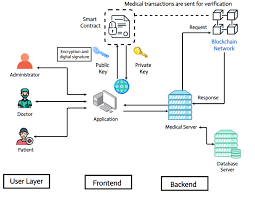
Medical Server: Processes medical data and interacts with the blockchain for verification of transactions.

Blockchain Network: Verifies the authenticity and integrity of medical transactions using smart contracts. Ensures tamper-proof records.

Database Server: Stores encrypted medical records and transaction logs securely.

The response from the blockchain is sent back to the medical server and then to the application, completing the transaction loop.

**Data Flow Diagram:**



**3.4 Technology Stack**

Front end

Back end

**4 Project design:**

**4.1 Problem solution fit:**

**Code:**

**Frontend: Streamlit (Python-based Web App Framework)**

import streamlit as st

import pandas as pd

import os

from dotenv import load\_dotenv

from ibm\_watsonx\_ai.foundation\_models import Model

# Load .env file for credentials (if needed later)

load\_dotenv()

API\_KEY = os.getenv("WATSONX\_API\_KEY")

PROJECT\_ID = os.getenv("WATSONX\_PROJECT\_ID")

# Page setup

st.set\_page\_config(page\_title="HealthAI", layout="centered")

# App heading

st.markdown("<h1 style='text-align: center;'>🩺 HealthAI - Intelligent Healthcare Assistant</h1>", unsafe\_allow\_html=True)

# Sidebar – Full Patient Profile form

with st.sidebar:

st.markdown("## 🧑‍⚕ Patient Profile")

name = st.text\_input("Name")

age = st.number\_input("Age", min\_value=0, max\_value=120, step=1)

gender = st.selectbox("Gender", ["Male", "Female", "Other"])

location = st.text\_input("Location")

medical\_history = st.text\_area("Medical History")

current\_Medication=st.text\_area("current Medication")

Allergies=st.text\_area("Allergies")

# Main content – Support and Chat

st.subheader("🕒 24/7 Patient Support")

st.write("Ask any health-related question for immediate assistance.")

menu = st.sidebar.selectbox("📋 Choose a Feature",

["🏠 Home", "🧠 Disease Prediction", "💊 Treatment Plan", "💬 Patient Chat", "📊 Health Analytics"])

if menu == "🏠 Home":

st.write("")

st.markdown("""

""")

elif menu == "🧠 Disease Prediction":

st.header("🧠 Disease Prediction")

symptoms = st.text\_area("Enter your symptoms (comma separated)", placeholder="e.g., fever, cough, headache")

if st.button("🔍 Predict"):

if "fever" in symptoms and "cough" in symptoms:

st.success("🤒 You might have Flu or COVID-19. Please consult a doctor.")

elif "headache" in symptoms and "fatigue" in symptoms:

st.success("😓 You may be experiencing Migraine or Viral infection.")

else:

st.warning("❓ No clear prediction. Try different symptoms or consult a professional.")

elif menu == "💊 Treatment Plan":

st.header("💊 Treatment Plan Generator")

condition = st.text\_input("Enter diagnosed disease", placeholder="e.g., Diabetes")

if st.button("📋 Generate Plan"):

st.write(f"🔎 Treatment plan for {condition}:")

st.markdown("""

- 💊 Take medications as prescribed

- 🥗 Follow a healthy diet

- 🧘‍♂ Do regular exercise

- 🩺 Attend follow-up checkups

""")

elif menu == "💬 Patient Chat":

st.header("💬 Ask Your Health Question")

question = st.text\_input("Type your health question here")

if st.button("🧠 Get AI Answer"):

st.write("🤖 This is a simulated AI response. Please consult a doctor for accurate advice.")

elif menu == "📊 Health Analytics":

st.header("📊 Health Analytics Dashboard")

data = {

'Date': ['Mon', 'Tue', 'Wed', 'Thu', 'Fri'],

'Heart Rate': [72, 75, 78, 76, 74],

'Blood Pressure': [120, 122, 121, 124, 123]

}

df = pd.DataFrame(data)

st.line\_chart(df.set\_index('Date'))

# Chat system with history

if "chat\_history" not in st.session\_state:

st.session\_state.chat\_history = []

# IBM Credentials

api\_key = "ZPvG-12ijx5ATRN6BQ55q9hI5iqQrF5B5c55VHWYyV-U"

project\_id = "693633cd-ac3b-4230-9eda-e179323e00a7"

base\_url = "https://us-south.ml.cloud.ibm.com"

# Initialize the model

# Initialize chat history

if 'chat\_history' not in st.session\_state:

st.session\_state.chat\_history = []

# Display previous messages

for i in range(0, len(st.session\_state.chat\_history), 2):

if i < len(st.session\_state.chat\_history):

with st.chat\_message("user"):

st.write(st.session\_state.chat\_history[i])

if i + 1 < len(st.session\_state.chat\_history):

st.chat\_message("assistant").write(st.session\_state.chat\_history[i + 1])

# Get response from the model

with st.chat\_message("assistant"):

with st.spinner("Thinking..."):

try:

model\_id="ibm/granite-13b-chat-v2"

model = Model(

model\_id=model\_id,

credentials={

"apikey": api\_key,

"base\_url": base\_url

},

project\_id=project\_id

)

response = model.generate\_text(

prompt=user\_input,

max\_new\_tokens=100,

temperature=0.7,

)

assistant\_reply = response['results'][0]['generated\_text']

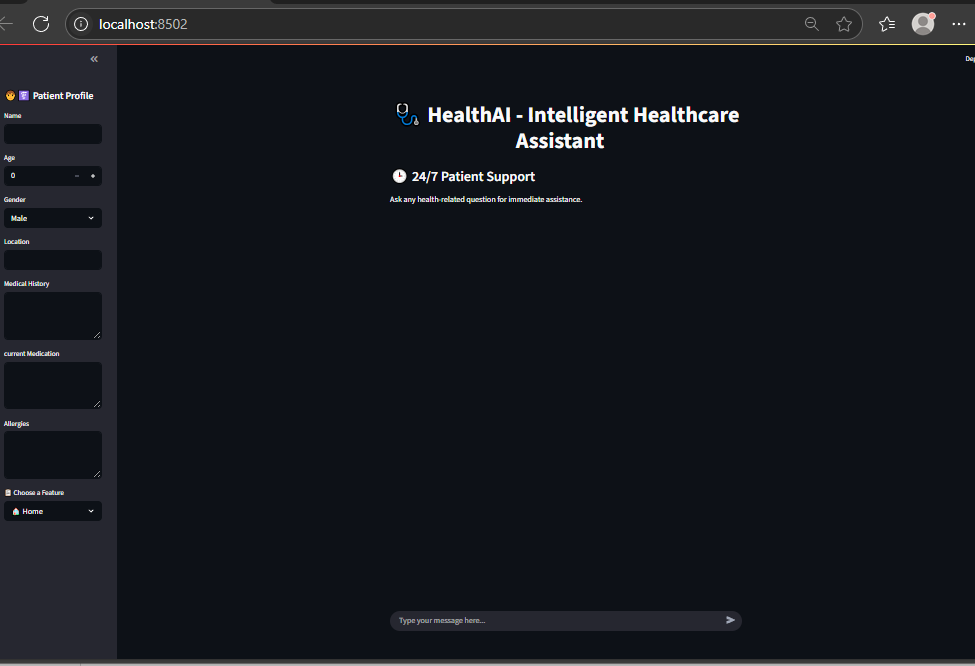
st.session\_state.chat\_history.append(assistant\_reply)

st.write(assistant\_reply)

except Exception as e:

st.write("Error:", e)

**4.2 proposed solution:**



**Backend: IBM Granite**

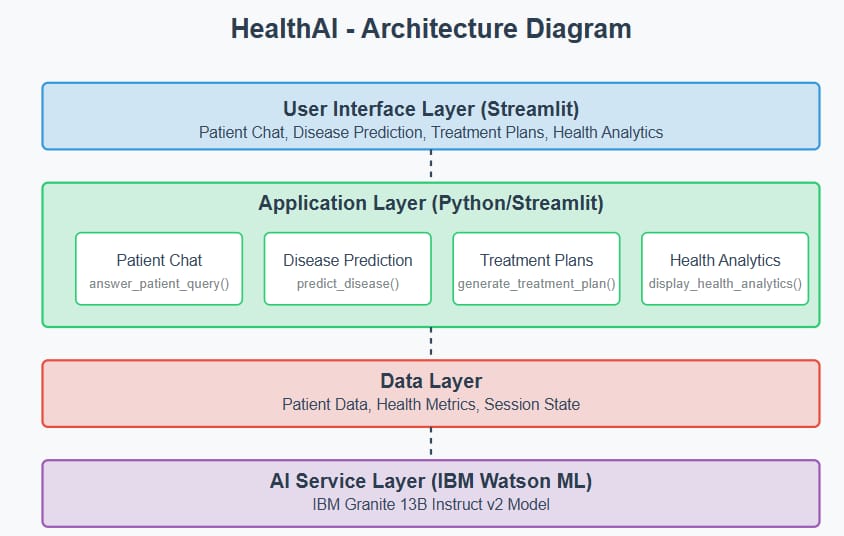
from dotenv import load\_dotenv

load\_dotenv()

api\_key = os.getenv("WATSONX\_API\_KEY")

project\_id = os.getenv("WATSONX\_PROJECT\_ID")

**4.3 solution architecture:**



**APIs: IBM Cloud Watsonx API**

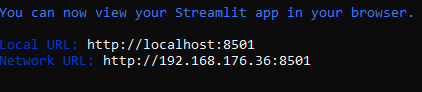
api\_key = "ZPvG-12ijx5ATRN6BQ55q9hI5iqQrF5B5c55VHWYyV-U"

project\_id = "693633cd-ac3b-4230-9eda-e179323e00a7"

base\_url = <https://us-south.ml.cloud.ibm.com>

**Hosting: IBM Cloud or Localhost**

[HealthAI](http://localhost:8501/)

****

**5 project planning and scheduling:**

**5.1 project planning:**

Effective project planning is crucial for the successful development and timely completion of the HealthAI application. The project follows a structured and phased development lifecycle, ensuring that each module is carefully analyzed, designed, implemented, and tested.

**Goals of Project Planning**

Define the development phases

Allocate tasks and responsibilities

Set realistic timelines

Identify dependencies between modules

Ensure risk management and quality assurance

**Project Phases and Timeline**

Phase

Description

Duration

1. Requirement Analysis Understand project objectives, gather functional and non-functional requirements 2 days

2. Design Design the system architecture, UI layout, and data flow diagrams 2–3 days

3. Model Integration Integrate IBM Granite for LLM-based chat and treatment plan generation 3–4 days

4. Frontend Development Build the Streamlit-based user interface 2–3 days

5. Backend Logic Handle symptom processing, disease prediction, and treatment modules 2–3 days

6. Dashboard Creation Develop analyst dashboard to visualize health trends and logs 2 days

7. Testing test the overall system functionality, usability, andaccuracy 2 days

8. Deployment Deploy the app on a cloud platform or local server 1–2 days

9. Documentation Prepare full technical documentation and user guide 2–3 days

**6 Functional and Performance Testing :**

**6.1 performance testing:Disease Prediction in HealthAI: Intelligent Healthcare Assistant Using IBM Granite**

HealthAI is an intelligent healthcare assistant designed to empower users by providing early insights into possible diseases based on their symptoms. Using advanced AI models and IBM Granite foundation models.The main goal of the disease prediction module is to assist users in identifying possible health risks at an early stage.

**How Disease Prediction Works in HealthAI:**

**1. Symptom Input:**

The user enters symptoms like fever, cold, body pain, etc., through a simple and interactive Streamlit interface.

**2. Symptom Processing using AI:**

The system uses Natural Language Processing (NLP) to understand the symptoms and convert them into structured data.

**Treatment Plan Generation in HealthAI:**

One of the most innovative features of HealthAI is its ability to generate personalized treatment plans for patients based on the predicted disease and symptoms.

**Treatment plan for Diabetes:**

* Take a medication as prescribed
* Follow a healthy diet
* Do regular exercise
* Attend follow-up checkups

**Patient Chat in HealthAI:**The Patient Chat module is a core component of HealthAI, designed to provide users with an interactive and conversational where they can symptoms.

**Health Analyst Dashboard in HealthAI:**

The Health Analyst Dashboard is an advanced feature in HealthAI that provides healthcare professionals or system administrators with real-time insights, patient trends, and AI-generated analytics. It transforms raw health data into meaningful visualizations, helping analysts monitor system usage, symptom trends, prediction accuracy, and more.

**Key Functionalities:**

**1. Symptom Frequency Tracking:**

Displays the most common symptoms entered by users (e.g., fever, cough, fatigue) in bar charts or pie charts.

**2. Disease Prediction Trends:**

Visualizes predicted diseases over time (e.g., spikes in flu cases during monsoon).

**3. Interactive Filters:**

Date ranges

Location (if location data is available)

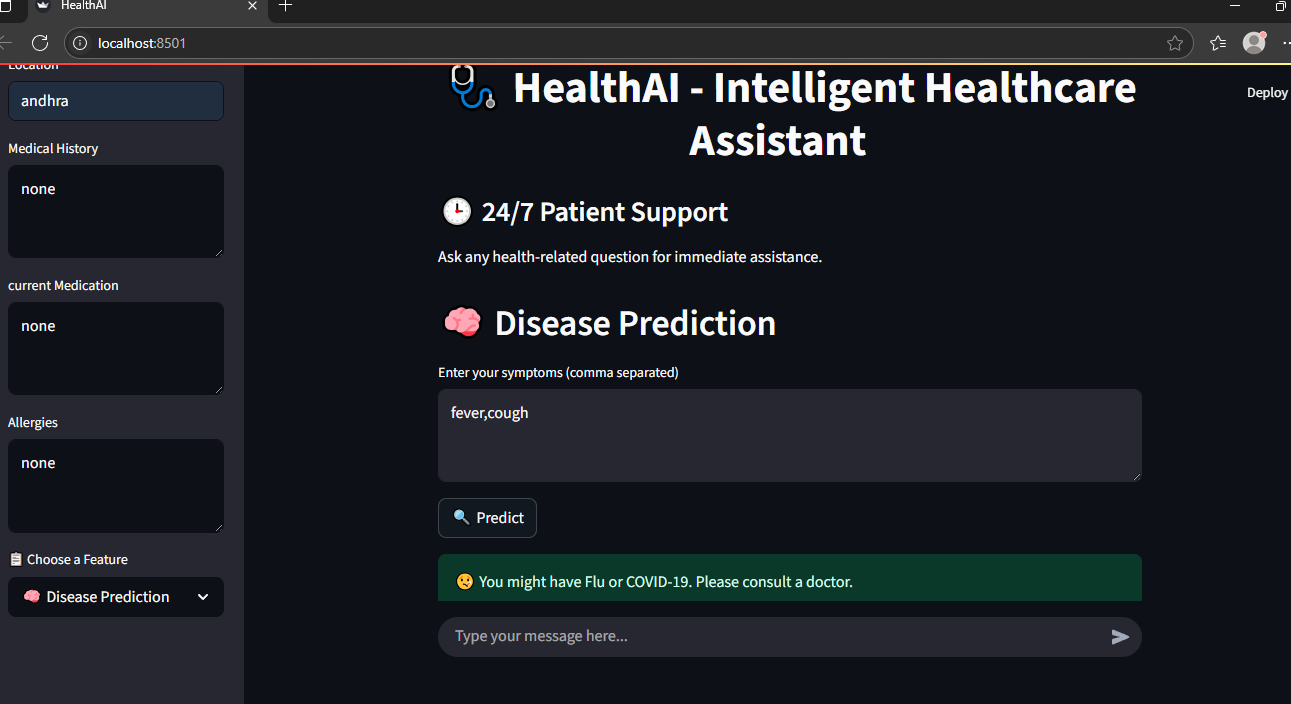
Age group

Predicted disease

**7 Result:**

**7.1 output screenshots:**

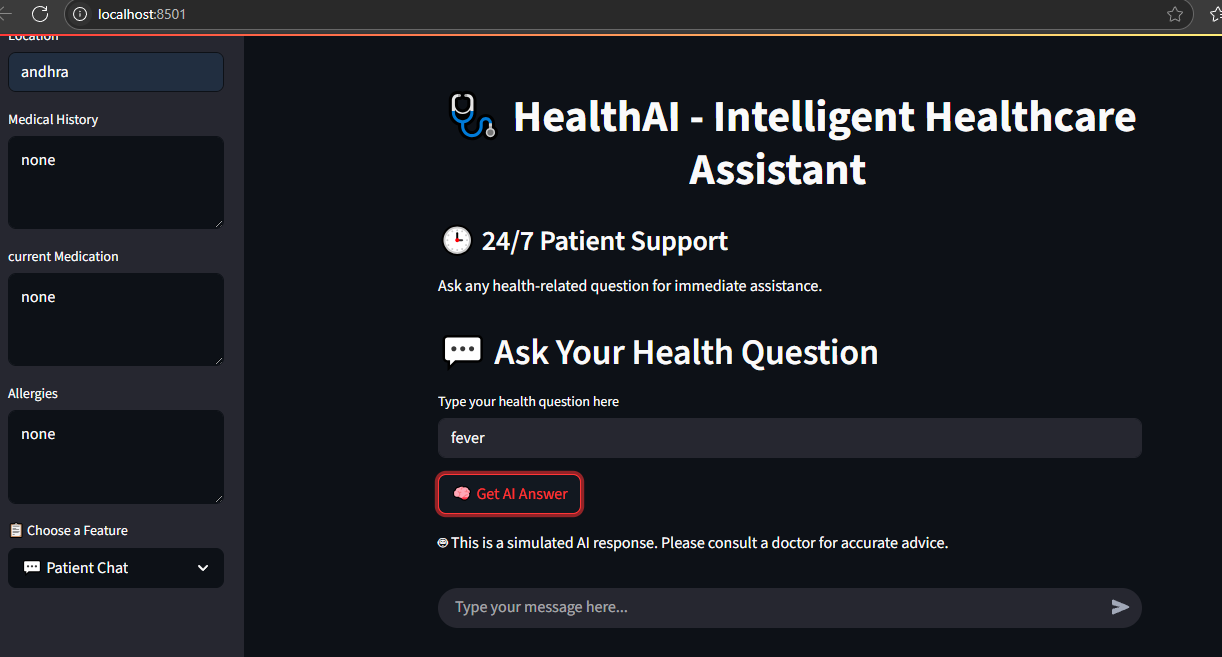
**Disease prediction:**

****

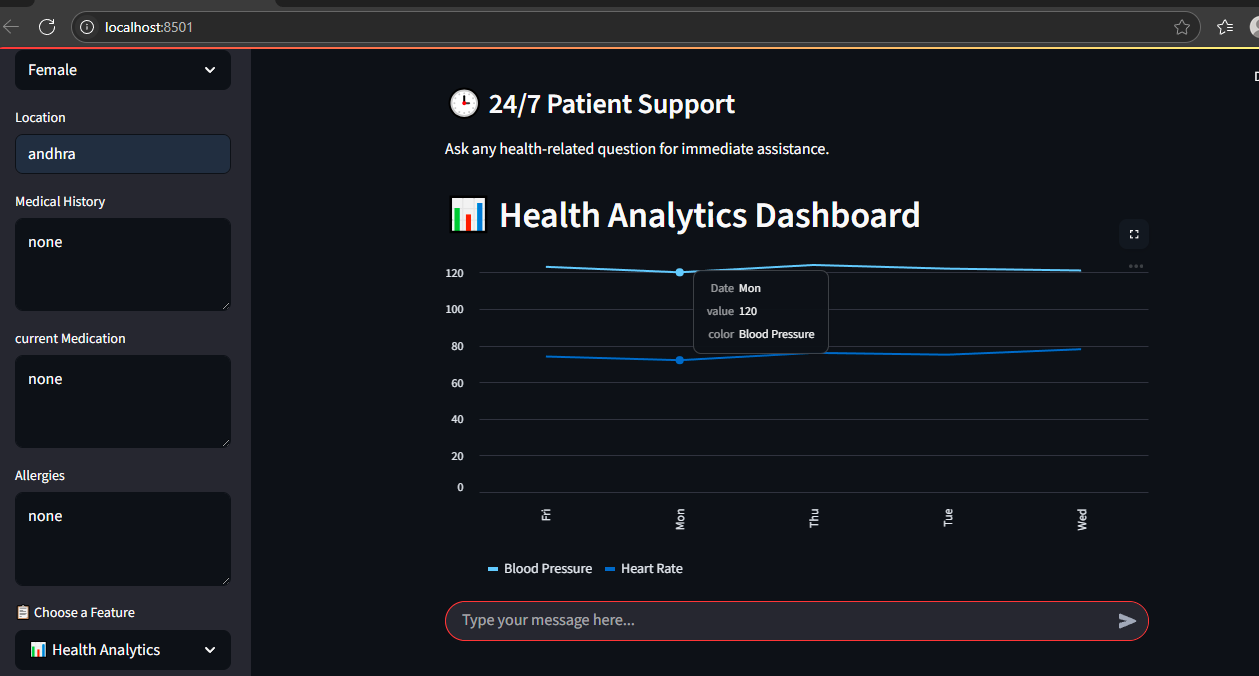
**Treatement plan:**

****

**Patient chat:**



**Health Analytics:**



**8 Advantages of HealthAI:**

**1.24/7 Availability:** HealthAI can assist users at any time, unlike traditional healthcare systems that depend on doctor availability.

**2. Early Disease Detection:**By analyzing symptoms, HealthAI helps in predicting possible diseases early, improving chances of timely treatment.

**3. Personalized Treatment Suggestions:**IBM Granite enables human-like understanding and response generation, offering tailored treatment plans for each user.

**4. User-Friendly Interface:**The Streamlit-based UI makes it easy for users to interact with the system, even if they don’t have technical knowledge.

**Disadvantages / Limitations:**

**1. Not a Replacement for Doctors:** HealthAI gives suggestions but cannot perform physical examinations or prescribe medications.

**2. Limited to Predefined Data:**Predictions depend on the trained dataset. New or rare diseases may not be accurately predicted.

**3. Language Barriers:**The current system may struggle with local languages unless multilingual support is added.

**4. Internet Dependency:**The system requires a stable internet connection for real-time LLM responses.

**9 Conclusion:**

HealthAI is a powerful and intelligent virtual healthcare assistant that leverages the capabilities of IBM Granite to offer disease prediction, treatment planning, and interactive patient support. It bridges the gap between patients and healthcare by offering timely, AI-driven assistance in a user-friendly environment.

The integration of AI in healthcare not only reduces the workload on medical professionals but also empowers users with knowledge and decision-making tools. Though it has some limitations, HealthAI proves that AI can play a significant role in accessible, affordable, and smart healthcare for everyone.

**10 Future Scope:**

**1. Multilingual Chat Support:**Add support for regional languages to reach rural and non-English-speaking users.

**2. Integration with Wearable Devices:**Connect with smartwatches or fitness bands to fetch real-time vitals (e.g., heart rate, BP).

**3. Voice-Based Assistant:**Implement voice interaction for users who are not comfortable typing.

**4. Emergency Alert System:**Detect critical symptoms and automatically alert emergency contacts or health services.

**11 Appendex:**

**Source code link:**

[**https://github.com/Vinay-507/HealthAi**](https://github.com/Vinay-507/HealthAi)

**Demo video link:**

**https://drive.google.com/drive/folders/15FCWP2-irhFP-3kbXSgskFjyVSTP72Zo?usp=sharing**