1. INTRODUCTION

1.1 Project Overview

This project is an AI-powered medical assistant built using **Streamlit** and **IBM Watson Granite 13B Instruct v2**. It allows users to chat with an AI, predict diseases based on symptoms, generate treatment plans, and visualize health metrics.

1.2 Purpose

To create a helpful, intelligent healthcare assistant that offers medical advice and analytics, making basic healthcare accessible and AI-driven.

---

2. IDEATION PHASE

2.1 Problem Statement

Many people lack access to timely medical advice. This project aims to bridge that gap with an AI that provides symptom-based predictions, suggestions, and analytics.

2.2 Empathy Map Canvas

- **Think & Feel:** Wants accurate, private, and fast responses.
- **See:** Overloaded clinics, long wait times.
- **Hear:** Others complaining about slow healthcare.
- **Say & Do:** Seeks online help.
- **Pain:** Long delays, lack of access.
- **Gain:** Quick, AI-powered health guidance.

2.3 Brainstorming

- Use LLM for disease prediction.
- Add a chatbot.
- Visualize patient data.
- Suggest treatments using AI.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Step	Action	Feeling	Support Needed
1	Open app	Curious	Easy interface
2	Ask symptoms	Hopeful	Accurate response
3	View prediction	Informed	Clear info
4	Get treatment	Relieved	Trusted source

3.2 Solution Requirement

- LLM model (IBM Granite)
- UI using Streamlit
- Backend logic in Python
- APIs for predictions and responses

3.3 Data Flow Diagram

User → Streamlit UI → app.py (logic) → IBM Granite API → Result → UI

3.4 Technology Stack

- **Frontend:** Streamlit
- **Backend:** Python
- **Model:** IBM Granite 13B Instruct v2
- **Libraries:** requests, pandas, plotly
- **Hosting:** Localhost or Hugging Face Spaces

---

4. PROJECT DESIGN

4.1 Problem-Solution Fit

AI medical assistant offers accessible and affordable health support when real doctors are not immediately available.

4.2 Proposed Solution

Use IBM Watson Granite model to analyze user input and return useful medical insights.

4.3 Solution Architecture

[User]

↓

[Streamlit UI]

↓

[app.py Logic Layer]

↓

[IBM Watson Granite Model API]

↓

[Response → Display to User]

---

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Activity	**Timeline**
Model Selection	Week 1
Core Features	Week 2
Frontend & app.py	Week 3
Deployment	Week 4

---

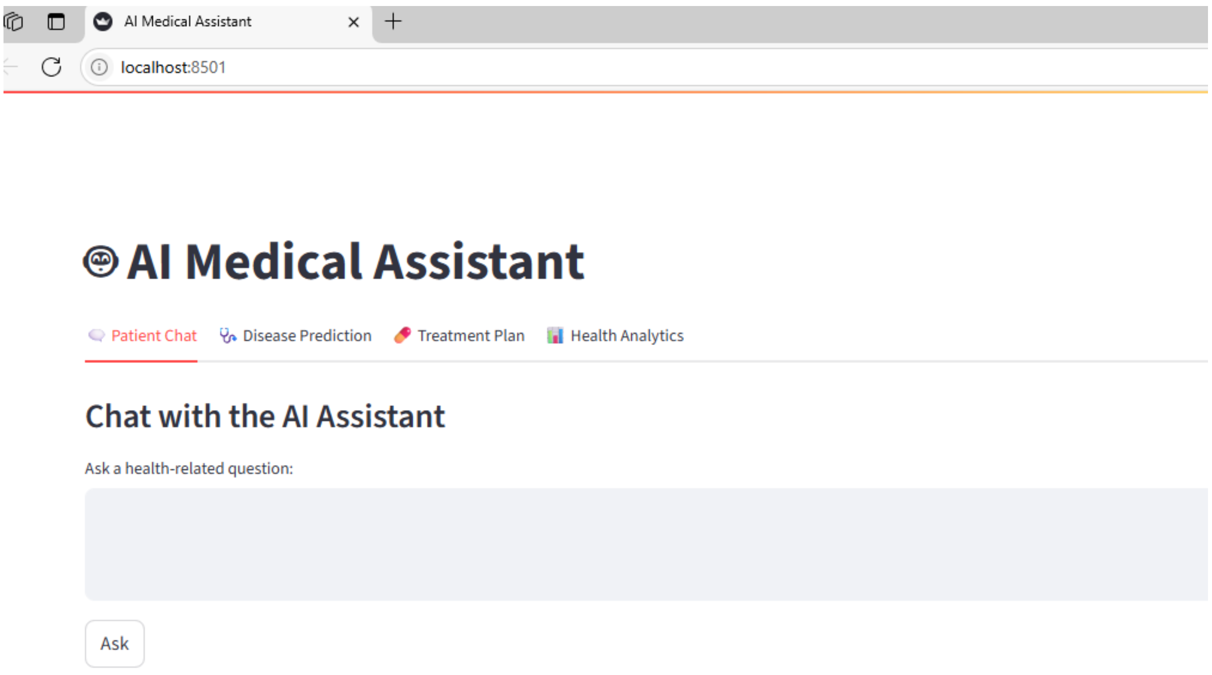
6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

- Tested response time of AI model.
- Checked if app works smoothly on different devices.
- Verified data handling and error messages.

7. RESULTS

7.1 Output Screenshots



8. ADVANTAGES & DISADVANTAGES

Advantages

- Accessible 24/7
- AI-based insights
- Easy UI
- Fast response

Disadvantages

- Not a replacement for real doctors
- Needs stable internet
- May not cover all rare diseases

---

9. CONCLUSION

The AI Medical Assistant successfully integrates IBM Granite with a simple Streamlit frontend to provide basic healthcare guidance, disease prediction, and treatment plans.

---

10. FUTURE SCOPE

- Add voice assistant
- Translate responses to regional languages
- Store patient history
- Improve accuracy with more data

---

11. APPENDIX

✅ Source Code

app.py contains main logic with function definitions for chat, prediction, analytics.

✅ Dataset Link

If used, include CSV/Excel file location (like: data/patient_data.csv)

✅ GitHub & Demo Links

GitHub Repo: [https://github.com/Radhika-CR/ai-medical-assistant/tree/main/Project%20files]

Demo: Hosted on [local]

🛠 Step-by-Step Guide to Create & Deploy the Project

1. Setup Project Folder

```
ai-medical-assistant/
 ├── app.py
 ├── requirements.txt
 ├── .env (not uploaded)
 ├── README.md
 ├── data/
|   └── patient_data.csv
```

Steps to Run the Demo Locally:

1. **Clone the Project Repository:**

Open a terminal/command prompt and run:

git clone https://github.com/your-username/ai-medical-assistant

cd ai-medical-assistant

2. **Set Up Environment Variables:**

Create a new file named .env in the project folder and paste the following content:

.env

IBM_API_KEY=your_ibm_api_key_here

DEPLOYMENT_URL=your_ibm_deployment_url_here

3. **Install Required Libraries:**

Make sure you have Python installed, then run:

pip install -r requirements.txt

4. **Run the Application:**

Start the app using Streamlit:

streamlit run app.py

python -m streamlit run app.py

5. **Access the Application:**

Open your web browser and visit:

http://localhost:8501

This will launch the AI Medical Assistant on your local system, where you can use features like patient chat, disease prediction, treatment suggestions, and health analytics.

9. Deployment Options

- **Hugging Face Spaces**
  - Create a new Space → Type: Streamlit → Upload your code
  - Add secrets or upload .env using Hugging Face interface
- **Others:** Use platforms like **Render**, **Streamlit Cloud**, or **Docker** if preferred.