**Health AI Project - Detailed Documentation**

**HealthAI-Intelligence HealthCare Using IBM Granite**

**Project Report**

**Project Team Members:**

1.NAJITHA.M

2.JUHAINA.M

3.LALITHA.R

4.KARTHIKA.R

5.NALIFA RUMAN.S

**1. Introduction**

**1.1 Project Title: HealthAI-Intelligence HealthCare Using IBM Granite**

**1.2 Background:**

HealthAI is an innovative healthcare solution leveraging IBM's Granite technology to provide intelligent, personalized, and data-driven healthcare assistance.

1. AI-powered Healthcare: HealthAI utilizes artificial intelligence to enhance patient care, diagnosis, and treatment planning.

2. IBM Granite Integration: IBM Granite's advanced capabilities enable HealthAI to process complex healthcare data and deliver insightful recommendations.

3. Personalized Medicine: HealthAI focuses on tailoring healthcare solutions to individual patient needs and profiles.

**1.3 Objectives:**

1. Enhance Patient Care: Improve healthcare outcomes through personalized and data-driven approaches.

2. Support Clinical Decision-Making: Provide healthcare professionals with AI-assisted insights for diagnosis and treatment planning.

3. Streamline Healthcare Processes: Leverage AI to optimize efficiency in healthcare delivery and management.

4. Predict and Prevent Health Risks: Utilize predictive analytics to identify potential health issues early.

5. Improve Patient Engagement: Empower patients with personalized health information and recommendations.

6. Ensure Data Security and Compliance: Protect patient data and adhere to healthcare regulations like HIPAA.

7. Facilitate Research and Innovation: Support healthcare research through advanced data analysis and AI capabilities.

8. Augment Healthcare Professional Capabilities: Assist doctors and healthcare providers in delivering effective care.

9. Deliver Personalized Treatment Plans: Generate tailored treatment recommendations based on patient profiles and data.

10. Continuously Improve and Adapt: Evolve HealthAI based on feedback, data, and advancements in AI and healthcare practices.

**2. Project Overview**

**2.1 Purpose:**

\*HealthAI aims to help people get smart healthcare guidance using AI (powered by IBM Granite) through an easy-to-use interface (built with Gradio).

\* To support better healthcare decisions with AI-powered assistance, making health info more accessible and useful.

**2.2 Features:**

- Conversational Interface: Natural language interaction.

- Symptom-Based Analysis: HealthAI analyzes input symptoms to suggest possible conditions.

- Personalized Recommendations: Suggests treatment options tailored to patient profile and condition.

- Health AI Query Response:Answering Questions adout the Disease prediction and Treatment plans

- User-Friendly Interface: Uses Gradio for dashboards.

**3. Architecture**

3.1 Frontend (Gradio): Provides interactive web-based UI with tabs.

3.2 Backend (Hugging Face Transformers + PyTorch): Runs AI model for responses.

3.3 Model Integration: IBM Granite model is used for text generation and analysis.

**3.4 Functions:**

- generate\_response: Generates AI responses.

-Disease\_prediction: HealthAI's disease prediction operation involves analyzing patient symptoms and data using AI (IBM Granite) to suggest possible conditions and guide next steps.

-Treatment\_Plan: HealthAI's treatment plan operation generates personalized recommendations for patient care based on their condition, symptoms, and data, using AI-powered insights from IBM Granite.

**4.Setup Instructions**

**4.1 Prerequisites:**

- Python 3.8 or later

- Google Colab (T4 GPU preferred)

- Libraries: transformers, torch, gradio

**4.2 Installation Process:**

1. Open Google Colab.

2. Change runtime to GPU (T4 preferred).

3. Install libraries with !pip install transformers torch gradio -q.

4. Paste the project code into Colab.

5. Run all cells to launch.

6. Access the public link for interaction.

**5. Folder Structure**

app/ – Backend logic

app/api/ – API routes

ui/ – Frontend components

smart\_dashboard.py – Entry script

granite\_llm.py – Model integration

document\_embedder.py – Document embeddings

**6. Running the Application**

**6.1 Steps to Run:**

➢ Launch Colab notebook.

➢ Run installation and model setup.

➢ Start Gradio interface.

➢ Use Disease Prediction To Analysis the What Disease is Affact the patient to cure the disease the healthAI gives a treatement plans to solve the disease

➢ Use Treatment Plans tab for queries.

**7. API Documentation**

**7.1 Available Functions:**

 generate\_response(prompt)

 disease\_prediction(symptoms)

 treatment\_plan(condition,age,gender,medical\_history)

**8. Authentication & Security**

**Future deployments can include:**

- Token-based authentication (JWT, API Keys).

- OAuth2 integration.

- Role-based access control (Admin, HealthAI, Researcher).

**9. User Interface**

9.1 Disease Prediction Tab: Input Symptoms → Fever.

9.2 Treatment plan Tab: Input (condition,age,gender,medical\_history) → Medical Condition-Hypertension,Age-30,Gender-Male,Medical History-He Has a Stress so his condition In Hypertension

9.3 Output: Clear textboxes with results.

**10. Testing**

10.1 Unit Testing: Functions tested independently.

10.2 API Testing: Using test inputs.

10.3 Manual Testing: Validating analysis and queries.

10.4 Edge Case Handling: Empty (condition ,age,gender,medical\_history), invalid symptoms.

**11. Screenshots**

[Placeholder for Gradio interface screenshots]

**12. Known Issues**

- Requires stable internet connection.

- AI-generated outputs may vary.

- Limited accuracy without real-time datasets.

**13. Future Enhancements**

- Integration with real-time Health data.

- Advanced analytics and forecasting.

- Multilingual support.

- Mobile app interface

**14. Conclusion**

**HealthAI, powered by IBM Granite and built with Gradio, aims to enhance healthcare through AI-driven insights, offering personalized disease prediction and treatment guidance. It supports informed healthcare decisions by analyzing patient data and symptoms, complementing professional medical judgment.**