**Health AI – Intelligent Healthcare Assistant**

Generative AI with IBM

Project Documentation

**1. Introduction**

Project Title: Health AI – Intelligent Healthcare Assistant

Team Leader: Veerapriya M

Team Members:

* Suganthi M
* Varsha S
* Sivadharshini M

**2. Project Overview**

**Purpose**:

The purpose of this project is to develop an AI-powered healthcare assistant that leverages IBM Watsonx generative AI models to enhance patient care, support doctors, and streamline hospital operations. The assistant provides real-time medical information, personalized health recommendations, and administrative support for both patients and healthcare professionals.

**Features:**

Conversational Interface – Natural language chat for patients and doctors.

Medical Knowledge Summarization – Converts lengthy medical research papers or reports into simplified insights.

Symptom Checker & Triage – Provides preliminary health advice based on symptoms.

Personalized Health Tips – Suggests diet, exercise, and lifestyle guidance tailored to the user.

Patient Feedback Loop – Collects patient feedback to improve healthcare services.

KPI Forecasting – Predicts hospital resource needs (beds, medicines, staff).

Anomaly Detection – Detects unusual patterns in patient vitals or hospital data.

Multimodal Input Support – Accepts text, medical reports (PDF), and lab results (CSV).

User-Friendly Interface (Streamlit/Gradio) – Dashboard for patients, doctors, and administrators.

**3. Architecture**

Frontend (Streamlit/Gradio): Patient portal, doctor dashboard, chatbot, report viewer.

Backend (FastAPI): Handles APIs for medical Q&A, symptom analysis, report summarization.

LLM Integration (IBM Watsonx Granite): For generative responses, summarization, and recommendations.

Vector Search (Pinecone): To enable semantic search across medical research, guidelines, and patient history.

ML Modules (Forecasting & Anomaly Detection): Predicting patient inflow, anomaly detection in vital signs.

**4. Setup Instructions**

Python 3.9+

pip + venv

API keys (IBM Watsonx, Pinecone)

Install dependencies → Run backend → Launch frontend

**5. Folder Structure**

app/ – FastAPI backend

app/api/ – APIs for chat, medical data, patient reports

ui/ – Streamlit/Gradio UI for patient-doctor interaction

granite\_llm.py – IBM Watsonx Granite integration

symptom\_checker.py – Symptom analysis and recommendations

health\_forecaster.py – Predicts hospital KPIs (beds, staff, meds)

anomaly\_detector.py – Flags anomalies in vitals or hospital data

report\_generator.py – Generates patient summaries and hospital reports

**6. Running the Application**

Start FastAPI server

Run Streamlit dashboard

Interact with chat assistant, upload reports, receive summaries & health tips

**7. API Documentation**

POST /chat/ask – Health Q&A

POST /upload-report – Upload & analyze reports (PDF, CSV)

GET /search-medical – Semantic medical literature search

GET /get-health-tips – Personalized wellness tips

POST /submit-feedback – Collects patient feedback

**8. Authentication**

JWT tokens for patients/doctors

Role-based access (patient, doctor, admin)

IBM Cloud OAuth2 integration

**9. User Interface**

Sidebar navigation

Patient health dashboard

Doctor dashboard with patient monitoring

Chat with AI medical assistant

Downloadable reports

**10. Testing**

Unit testing (symptom checker, summarizer)

API testing (Swagger/Postman)

Manual testing (chat, uploads, dashboard)

Edge cases (invalid input, large files, missing data)

**11. Screenshots** - A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**12. Known Issues**

**Accuracy and Reliability**

* + AI-generated outputs may sometimes produce **incomplete, misleading, or incorrect medical advice**.
  + Symptom checkers cannot provide a **definitive diagnosis** — only guidance.
* **Bias in Training Data**
  + If the AI model is trained on biased or incomplete medical datasets, it may produce **skewed recommendations** (e.g., underrepresenting certain populations, genders, or age groups).
* **Data Privacy and Security**
  + Handling sensitive patient health records requires strict compliance with standards like **HIPAA** and **GDPR**.
  + Any security breach could expose highly confidential patient data.
* **Over-Reliance on AI**
  + Patients might **treat AI suggestions as medical advice**, ignoring professional doctors.
  + Risk of **self-medication or delayed medical visits** due to AI overconfidence.
* **Integration Challenges**
  + Hospitals use different **Electronic Health Record (EHR) systems**, making **interoperability** a challenge.
  + Smooth integration with existing hospital IT infrastructure is still complex.
* **Resource Limitations**
  + Large-scale deployment may require **high computational power** and continuous internet access.
  + Rural or low-resource hospitals may struggle with implementation.
* **Explainability (Black Box Problem)**
  + Many generative AI models lack **clear reasoning pathways**, making it hard for doctors to trust their recommendations.
  + Explainable AI (XAI) techniques are still evolving.
* **Regulatory and Ethical Constraints**
  + Healthcare AI systems must be **approved by regulatory bodies** before clinical deployment.
  + Continuous updates in medical laws may create **compliance issues**.
* **Handling Multilingual and Low-Literacy Patients**
  + AI may fail to **interpret local dialects** or explain medical terms at an accessible level.
  + Risk of miscommunication with non-English-speaking patients.
* **Real-Time Responsiveness**
* Uploading large medical files or handling **complex queries** may lead to latency issues.
* In emergencies, delays can reduce system usefulness.

**13. Future Enhancements:**

Integration with wearable IoT devices

Voice-enabled medical assistant

Multilingual support for global healthcare