**Project Title**

**HealthAI – Intelligent Healthcare Assistant**

**1. Introduction**

* **Project title:** HealthAI – AI-powered Healthcare Assistant
* **Team member:C.V.VISHWA(23CS1631)**
* **Team member:K.VIJAYASAN(23CS1630)**
* **Team member:S.TAMILARASAN(23CS1628)**
* **Team member:G.THULASI RAMAN(23CS1629)**

**2. Project Overview**

**Purpose:**  
The purpose of HealthAI is to revolutionize healthcare delivery by empowering patients, doctors, and hospitals with AI-driven insights. By leveraging medical data, AI, and real-time monitoring, HealthAI supports preventive care, early diagnosis, resource optimization, and patient engagement. For healthcare providers, it acts as a decision-making partner—summarizing medical policies, forecasting hospital resources, and generating actionable insights. Ultimately, HealthAI bridges technology, healthcare, and patient needs to foster a smarter, safer, and more connected health ecosystem.

**Features:**

* **Conversational Interface**
  + *Key Point:* Natural language interaction
  + *Functionality:* Patients and doctors ask health questions, receive guidance, and get reminders in plain language.
* **Medical Policy Summarization**
  + *Key Point:* Simplified medical guidelines
  + *Functionality:* Converts lengthy clinical protocols, insurance policies, and research into concise summaries.
* **Health Forecasting**
  + *Key Point:* Predictive analytics
  + *Functionality:* Estimates patient recovery, outbreak patterns, and hospital resource needs.
* **Personalized Health Tips**
  + *Key Point:* Preventive care support
  + *Functionality:* Provides tailored fitness, diet, and medication reminders.
* **Patient Feedback Loop**
  + *Key Point:* Service improvement
  + *Functionality:* Collects patient feedback to enhance healthcare delivery.
* **Anomaly Detection**
  + *Key Point:* Early warning system
  + *Functionality:* Detects abnormal patterns in vitals, lab results, or hospital data.
* **Multimodal Input Support**
  + *Key Point:* Flexible medical data handling
  + *Functionality:* Accepts lab reports, prescriptions, PDFs, and CSVs for analysis.
* **Healthcare Dashboard (Streamlit/Gradio)**
  + *Key Point:* User-friendly interface
  + *Functionality:* Provides an intuitive dashboard for patients, doctors, and administrators.

**3. Architecture**

**Frontend (Streamlit/Gradio):**  
Interactive web UI with dashboards, file uploads, chat interface, medical history, feedback forms, and reports.

**Backend (FastAPI):**  
Handles data processing, chat, medical summarization, forecasting, anomaly detection, and API communication.

**LLM Integration (IBM Watsonx Granite):**  
Used for natural language understanding—summarizing policies, generating patient-friendly advice, and medical reports.

**Vector Search (Pinecone):**  
Stores medical documents and supports semantic search for quick retrieval of medical references.

**ML Modules (Forecasting & Anomaly Detection):**  
Uses time-series patient data and hospital metrics to predict trends and detect abnormalities.

**4. Setup Instructions**

**Prerequisites:**

* Python 3.9+
* pip and virtual environment tools
* API keys for IBM Watsonx and Pinecone
* Secure access for patient data

**Installation Process:**

* Clone repository
* Install dependencies from *requirements.txt*
* Configure credentials in *.env*
* Run backend server with FastAPI
* Launch frontend with Streamlit/Gradio
* Upload medical records and interact with AI modules

**5. Folder Structure**

* **app/** – Backend logic (APIs, models, integrations)
* **app/api/** – Routes (chat, feedback, medical reports, document processing)
* **ui/** – Frontend Streamlit/Gradio components
* **health\_dashboard.py** – Main dashboard entry
* **granite\_llm.py** – Handles Watsonx LLM interactions
* **document\_embedder.py** – Converts reports into embeddings
* **health\_forecaster.py** – Forecasts patient/hospital trends
* **anomaly\_checker.py** – Flags abnormal patterns in data
* **report\_generator.py** – Generates AI-driven medical reports

**6. Running the Application**

* Launch FastAPI server for backend APIs
* Run Streamlit/Gradio dashboard for UI
* Navigate via sidebar
* Upload medical documents or reports
* Interact with chat assistant, view summaries, predictions, and reports
* All responses update in real-time

**7. API Documentation**

* **POST /chat/ask** – AI medical Q&A
* **POST /upload-doc** – Uploads and embeds medical documents
* **GET /search-docs** – Retrieves similar medical guidelines/research
* **GET /get-health-tips** – Provides personalized health advice
* **POST /submit-feedback** – Stores patient feedback

**8. Authentication**

* Token-based authentication (JWT/API keys)
* OAuth2 with healthcare credentials
* Role-based access (patient, doctor, admin)
* Planned: encrypted sessions, medical history tracking

**9. User Interface**

* Sidebar navigation
* Health KPI visualizations (vitals, hospital data)
* Tabs for chat, health tips, and forecasting
* Real-time forms
* PDF report download support

**10. Testing**

* **Unit Testing:** AI functions & utilities
* **API Testing:** Swagger, Postman
* **Manual Testing:** Uploads, chat, outputs
* **Edge Cases:** Large records, malformed data, invalid keys

**11. Screenshots**

*(Placeholder for system images)*

**12. Known Issues**

* Limited offline mode
* Initial setup complexity
* Requires secure handling of sensitive medical data

**13. Future Enhancements**

* Integration with wearable devices & IoT health sensors
* Advanced predictive analytics (disease risk scoring)
* Multi-language patient support
* Blockchain-based medical data security