■ Health AI – Smart Healthcare Assistant

1. Introduction

Project Title: Health AI – Smart Healthcare Assistant
Team Members:

- Ramayee. C
- Sivakami. C
- Sivaranjani. K
- Sivaranjani. M

Problem Statement

Healthcare accessibility is a challenge in both rural and urban areas. Patients face delays, lack of immediate medical help, and difficulty in understanding complex reports. This leads to poor treatment outcomes.

Objectives

- Provide a 24×7 AI-powered assistant for patients.
- Assist doctors with risk prediction and report analysis.
- Deliver simplified health insights for patients.
- Build a feedback loop between doctors and patients.

2. Project Overview

Purpose

Health AI provides an intelligent assistant for patients, doctors, and hospitals, powered by Artificial Intelligence. It improves healthcare accessibility, monitoring, and decision-making.

Features

- Conversational Chat Interface – Natural language interaction for patients.
- Symptom Checker – AI analyzes symptoms and suggests possible causes.
- Medicine Reminder – Sends timely alerts for prescribed medicines.
- Health Report Summarization – Simplifies complex medical reports.
- Risk Prediction – Identifies early signs of diseases.
- Doctor–Patient Feedback Loop – Supports continuous monitoring.
- Anomaly Detection – Alerts doctors on unusual patterns.
- Multimodal Input Support – Accepts lab reports, images, CSV records.
- User-Friendly Dashboard – Easy interface for patients and doctors.

3. Architecture

- Frontend (Streamlit/Gradio): Dashboards, chat, report uploads, feedback.
- Backend (FastAPI): APIs for chat, predictions, reminders, and data.
- LLM Integration (IBM Watsonx / OpenAI): Chatbot and report summarization.
- Database (MongoDB / MySQL): Stores patient records and reminders.
- ML Modules (Scikit-learn, TensorFlow): Risk prediction & anomaly detection.

4. Setup Instructions

Prerequisites

- Python 3.9+
- API keys for AI model integration
- Internet access

Process

- Deploy backend services.
- Launch frontend dashboard.
- Upload medical reports or records.
- Interact with chatbot, predictions, and reminders.

5. Folder Structure

- app/ – Backend APIs (chat, prediction, summarization, reminders)
- ui/ – Frontend files (Streamlit/Gradio dashboard)
- report_summarizer.py – Handles report summarization
- health_predictor.py – Risk prediction module
- reminder_manager.py – Medicine reminder system
- anomaly_checker.py – Detects unusual data

6. Running the Application

- Start backend server
- Launch frontend dashboard
- Upload patient records
- Chat with assistant and view reports/reminders

7. API Documentation

- POST /chat/ask – Patient queries and chatbot responses
- POST /upload-report – Upload and summarize medical reports
- GET /get-reminders – Fetch upcoming medicine reminders
- POST /predict-risk – Predict disease risks
- POST /submit-feedback – Collect patient feedback

8. Authentication

- Secure login with Doctor / Patient / Admin roles
- Token-based authentication (JWT)
- Role-based permissions for privacy

9. User Interface

- Sidebar for navigation
- Tabs: Chat, Reports, Predictions, Reminders
- Dashboard with visualizations
- Option to download reports

10. Testing

- Unit Testing – Chatbot and prediction modules
- API Testing – Swagger UI, Postman

- Manual Testing – Upload reports and reminders
- Edge Cases – Wrong inputs, invalid files

11. Screenshots (to be added)

- Chat interface
- Report upload form
- Prediction results dashboard

12. Known Issues

- AI is a support tool, not a doctor replacement
- Needs stable internet
- Prototype supports only structured data

13. Future Enhancements

- Voice-based AI assistant
- Mobile application (Android & iOS)
- Integration with wearables
- Multi-language support for rural areas
- AI-powered emergency alerts
- Integration with government health schemes

14. Conclusion

Health AI makes medical help more accessible and faster. It connects patients and doctors through AI, supports decision-making, and improves communication for better healthcare outcomes.