Project Report

Title: Health AI – Intelligent Healthcare Assistant with Artificial Intelligence

1. Introduction

Project Title: Health AI – AI-Powered Healthcare Assistant for Monitoring and Diagnosis

Team Members:

P.Saraswathi

M.Sreya

E.Shyamaladevi

K.Sowmiya

2. Project Overview

Purpose

The goal of Health AI is to enhance healthcare services using artificial intelligence. Instead of relying solely on manual check-ups and hospital visits, the system provides intelligent tools for patient monitoring, diagnosis support, predictive analysis, and virtual assistance.

This system supports:

AI-driven symptom checker for preliminary diagnosis

Patient monitoring dashboard with real-time health data tracking

Predictive analytics for disease risk assessment

Medical chatbot assistant for patient queries

Secure access for doctors, patients, and healthcare providers

Key Features

Symptom Analyzer (AI-Powered)

Uses NLP to analyze patient symptoms and suggest possible conditions.

Health Monitoring Dashboard

Displays vitals like heart rate, oxygen level, temperature, and blood pressure.

Predictive Disease Risk Assessment

Uses ML models to assess risks for conditions like diabetes, hypertension, and heart disease.

Medical Records Management

Secure storage and retrieval of patient data with role-based access.

Virtual Health Assistant (Conversational AI)

Answers patient queries, provides medication reminders, and connects to doctors if needed.

3. System Architecture

Frontend (Streamlit/React)

Intuitive dashboard for patients and doctors

Tabs for Symptom Analysis, Monitoring, Reports, and Assistant

Backend (FastAPI/Django)

REST APIs for patient data management, diagnosis predictions, and monitoring alerts

Asynchronous endpoints for fast response

Integrated with healthcare APIs (FHIR/HL7 standards)

AI/ML Models Integrated

NLP Models: Symptom and query analysis

Classification Models (Random Forest, XGBoost): Disease prediction

Time-Series Models (LSTM/Prophet): Health monitoring and anomaly detection

Chatbot Models: Conversational health guidance

4. Setup Instructions

Requirements

Python 3.9 or above

API keys (OpenAI/Hugging Face/Healthcare IoT APIs)

Git installed

Docker (optional)

Steps to Run

Clone the repository

Create and activate a virtual environment

Install dependencies: pip install -r requirements.txt

Add API credentials in .env file

Start backend: uvicorn app.main:app --reload

Run frontend: streamlit run ui/dashboard.py

Access app via browser

5. Folder Structure

```
health_ai/
├── app/
│   ├── api/
│   │   ├── symptoms.py
│   │   ├── monitoring.py
│   │   ├── predictions.py
│   │   ├── records.py
│   │   └── assistant.py
│   ├── models/
│   │   └── ml_models.py
│   ├── utils/
│   │   └── helpers.py
│   └── main.py
├── ui/
│   ├── dashboard.py
```

```
│   ├── pages/
│   │   ├── SymptomAnalyzer.py
│   │   ├── Monitoring.py
│   │   ├── RiskAssessment.py
│   │   ├── Records.py
│   │   └── Assistant.py
├── README.md
├── .env
└── requirements.txt
```

6. Running the Application

Start backend server with FastAPI

Open frontend dashboard in browser

Use workflow:

Enter symptoms → Get possible conditions

View patient monitoring reports

Generate risk predictions

Interact with AI health assistant

7. API Documentation

Endpoint	Method	Function
/analyze-symptoms	POST	Extracts symptoms and suggests conditions
/monitor-health	POST	Logs and analyzes patient health data
/predict-disease	POST	Predicts risk of specific diseases
/manage-records	POST	Stores/retrieves medical records
/chat-assistant	POST	AI assistant for patient queries

Swagger UI available at: http://localhost:8000/docs

8. Authentication

Demo Mode: Open access

Production Mode:

Role-based access (Doctor, Patient, Admin)

JWT token authentication

OAuth2/SSO for hospital integration

9. User Interface

Sidebar for navigation

Tab-based interface for each health module

Charts for vitals and risk predictions

Interactive chatbot assistant

Secure file download for medical reports

10. Testing Strategy

Unit Testing: ML models and API endpoints

Integration Testing: API + UI functionality

Security Testing: Data encryption and access control

Mock Testing: Simulated patient data

CI/CD: GitHub Actions for automation

11. Screenshots (to be added after implementation)

Health Dashboard

Symptom Analyzer Output

Risk Assessment Chart

Chatbot Interaction

Monitoring Vitals

12. Known Limitations

Limited accuracy if symptoms are vague or incomplete

IoT device integration may vary across brands

Predictions depend on quality of training data

Chatbot cannot replace professional medical advice

13. Future Scope

Integration with wearable devices (smartwatch, ECG, glucometers)

Multilingual support for patient queries

Voice-enabled health assistant

Advanced imaging diagnostics (X-ray, MRI analysis with AI)

Integration with telemedicine platforms

Blockchain-based medical record security