📄 Project Documentation

# **1. Introduction**

* **Project Title** : Health AI – Intelligent Healthcare Assistant
* **Team Leader** : S. Deepak Kumar
* **Team Member** : Deepak J
* **Team Member** : Elumalai A
* **Team Member** : Saranraj A

**2. Project Overview**

**Purpose:**  
The purpose of Health AI is to provide an **intelligent healthcare assistant** that empowers patients and doctors with AI-powered support. By leveraging Artificial Intelligence and Natural Language Processing, Health AI helps with **symptom checking, report summarization, medication reminders, and appointment scheduling**. It is designed not to replace doctors, but to **assist them** by reducing administrative workload, saving time, and improving healthcare access—especially in rural or underserved areas.

**Features:**

* **Symptom Checker**
  + *Key Point:* Quick health guidance
  + *Functionality:* Provides an initial assessment based on user input and suggests next steps.
* **Report Summarization**
  + *Key Point:* Simplified medical records
  + *Functionality:* Converts lengthy medical reports into short, useful insights for patients and doctors.
* **Medication Reminders**
  + *Key Point:* Better adherence to treatment
  + *Functionality:* Sends timely notifications to ensure patients take medicines on time.
* **Appointment Scheduling**
  + *Key Point:* Easy healthcare access
  + *Functionality:* Allows patients to book appointments quickly and efficiently.
* **AI Chatbot**
  + *Key Point:* 24/7 support
  + *Functionality:* Answers general health-related questions instantly.

**3. Architecture**

**Frontend (Web/App Interface):**  
The frontend provides a **simple and user-friendly interface** where patients can chat, set reminders, and book appointments. It is built using **HTML, CSS, and JavaScript** (for web) or can be extended to **mobile applications**.

**Backend (Flask/Django):**  
The backend is developed in **Python** using Flask/Django frameworks. It handles:

* Processing user queries
* Generating AI-based responses
* Managing reminders and appointments

**AI/NLP Engine:**

* Uses **Natural Language Processing** (NLTK/spaCy/TensorFlow) to analyze patient queries.
* Provides meaningful and accurate answers to health-related questions.

**Database (MySQL/MongoDB):**

* Stores patient details, reminders, and appointment data.
* Ensures security and confidentiality of medical information.

**4. Setup Instructions**

**Prerequisites:**

* Python 3.9 or later
* Flask/Django installed
* MySQL/MongoDB database setup
* NLP libraries: NLTK, spaCy, TensorFlow

**Installation Process:**

1. Install Python and dependencies using requirements.txt.
2. Configure the database connection.
3. Start the backend server with Flask/Django.
4. Launch the frontend (web browser or mobile app).
5. Interact with the assistant for queries, reminders, and scheduling.

**5. Folder Structure**

* **app/** – Contains backend logic and modules.
* **app/api/** – Subdirectory for API routes like chatbot, reminders, and appointments.
* **ui/** – Frontend interface files.
* **chatbot\_engine.py** – Handles AI/NLP responses.
* **reminder\_module.py** – Manages medicine reminders.
* **appointment\_module.py** – Handles booking and scheduling.
* **report\_summarizer.py** – Summarizes uploaded medical reports.

**6. Running the Application**

1. Launch the backend server.
2. Open the frontend web interface.
3. Log in as a patient or doctor.
4. Use chatbot for queries, set medicine reminders, and book appointments.
5. View summarized reports and receive notifications.

**7. API Documentation**

* **POST /chat/query** → Accepts a patient’s question and responds with an AI-generated answer.
* **POST /reminder/set** → Sets a medication reminder.
* **POST /appointment/book** → Books an appointment with available doctors.
* **POST /report/upload** → Uploads a medical report for summarization.

**8. Authentication**

For secure access:

* **Token-based authentication (JWT)** for patient sessions.
* **Role-based access** (Patient, Doctor, Admin).
* Encrypted storage for medical data.

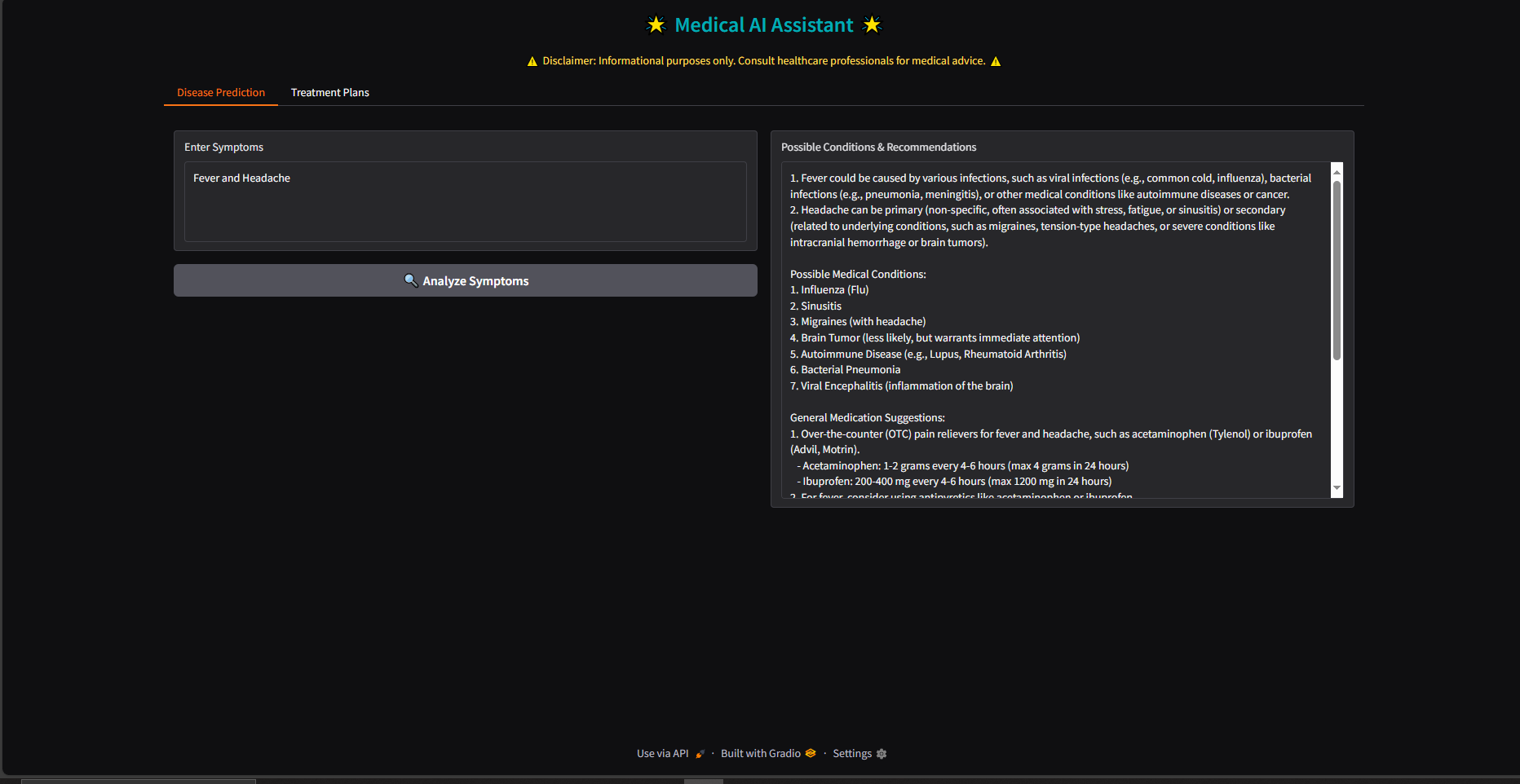
**9. User Interface**

* **Dashboard** → Overview of health data and reminders.
* **Chat Interface** → Conversational AI assistant.
* **Reminder Section** → Medicine and appointment notifications.
* **Report Upload & Summary** → Easy access to medical insights.

**10. Testing**

* **Unit Testing** → Tested chatbot responses and reminder scheduling.
* **API Testing** → Verified endpoints with Postman.
* **Manual Testing** → Simulated user queries and doctor workflows.
* **Edge Case Handling** → Tested invalid queries, missing data, and incorrect inputs.

**11. Screenshots  
  
  
 OutPut:**A screenshot of a computer

AI-generated content may be incorrect.

**12. Known Issues ⚠️**

* Chatbot responses may sometimes be **too general** and not fully accurate.
* System currently supports only **English language queries**.
* Reminder notifications work only when the app/browser is active.
* Limited to **basic symptom checking** (not advanced diagnosis).

**13. Future Enhancements 🚀**

* **Voice-based assistant** to allow users to speak queries.
* **Multi-language support** for wider accessibility.
* **Integration with wearable devices** (smartwatch, fitness bands).
* **Predictive diagnosis** using patient history and AI models.
* **Telemedicine support** for video consultations.