INTELLIGENT HEALTHCARE ASSISTANCE

PROJECT DOCUMENTATION

INTRODUCTION

Project Title : Intelligent Healthcare Assistance

Team Members :

- **Muthu Mari . K**
- **Pavithra . J**
- **Mitrra A.V.R**
- **Prithika . S**

The motivation behind this project is the increasing demand for **reliable healthcare guidance that is accessible to everyone at any time**, especially in situations where immediate medical consultation is not available. While this platform does not replace professional medical advice, it provides **preliminary assistance in identifying possible health conditions**, offering home remedies, recommending lifestyle improvements, and guiding users with first-aid measures during emergencies. By combining **Artificial Intelligence with user-friendly Interfaces**, the system aspires to promote health awareness, early intervention, and wellness management in a safe and accessible way.

PROJECT OVERVIEW

The primary purpose of this project is to bridge the gap between healthcare accessibility and timely information. Many individuals resort to online searches when they experience symptoms, which often leads to confusion or misinformation. To address this issue, the **HealthAI** system provides structured and trustworthy responses that are tailored to the user's input.

The platform comes with a wide range of features. **Disease prediction** helps users identify possible health conditions based on their symptoms, while personalized treatment plans suggest remedies, lifestyle adjustments, or medical consultations depending on the severity of the condition. In addition, the system includes modules for natural remedies, diet and nutrition advice, yoga and exercise recommendations, and emergency first-aid guidance. These ensure that users are not only informed about their symptoms but also guided toward healthier living. Features such as **medicine reminders**, a **wellness tracker**, and a general health chatbot further add to its practical usability.

Looking toward the future, the project is designed to be scalable. Planned features include **policy summarisation** for digesting healthcare regulations, **resource forecasting** to predict medical demands, **anomaly detection** for unusual patterns in vitals, and multimodal input to

support voice and image-based diagnosis assistance. This makes the project adaptable to both personal and institutional healthcare contexts.

》PURPOSE

- Enhance healthcare accessibility using AI.
- Provide real-time advice on common symptoms.
- Reduce dependency on searching unreliable internet sources.
- Encourage healthy lifestyle through tips, diet plans, and exercise suggestions.

》FEATURES

1. **Disease Prediction:** Analyze symptoms → possible conditions with severity.
2. **Treatment Plans:** Tailored advice based on age, gender, and history.
3. **Natural Remedies:** Safe home remedies with alerts for when to see a doctor.
4. **Diet & Nutrition:** Condition-specific food recommendations.
5. **Yoga & Exercise:** Easy-to-follow fitness routines for health conditions.
6. **Emergency Steps:** First-aid checklists with clear guidance.
7. **Medicine Reminder:** Simulation of schedules with friendly reminders.
8. **Wellness Tracker:** Analysis of vitals (temperature, BP, glucose, etc.).
9. **Health Chatbot:** Conversational Q&A for general health queries.
10. **Daily Health Tips:** Random motivational/health tips displayed.

ARCHITECTURE

The architecture of the HealthAI system is modular, ensuring scalability, maintainability, and flexibility. The frontend is currently built using Gradio, which allows for rapid prototyping and interactive UI design. For more advanced customization, Streamlit is considered as an alternate frontend. The backend relies on FastAPI, a lightweight yet powerful framework suitable for high-performance APIs.

The project integrates with IBM Watsonx Granite for large language model capabilities, which power the natural language understanding and response generation. For efficient information retrieval and history management, Pinecone is planned as a vector search database. Furthermore, dedicated machine learning modules handle forecasting of medical needs and anomaly detection for unusual health readings. This layered architecture ensures

that the system can evolve from a basic chatbot to a sophisticated healthcare decision-support tool.

》Frontend:

- Gradio (current prototype).
- Streamlit (planned for more flexibility).

》Backend:

- FastAPI for API endpoints and modular integration.

》LLM Integration:

- IBM Watsonx Granite (granite-3.2-2b-instruct).

》Vector Search:

- Pinecone (for storing patient history, quick retrieval).

》ML Modules:

- Forecasting (predict future health trends).
- Anomaly Detection (alert on irregular vitals).

SETUP INSTRUCTIONS

To run the project, users must have Python 3.9 or above installed on their systems. It is recommended to create a virtual environment to manage dependencies efficiently. After cloning the repository, users should install the required libraries such as Transformers, Torch, Gradio, FastAPI, and Pinecone-client. The application can be executed by running python app.py for the frontend and starting the backend with the command uvicorn backend.api:app --reload. Although the system is optimized to run on normal CPUs, a GPU-enabled environment can improve response times, especially when handling complex queries.

》Prerequisites

- Python 3.9+
- Virtual environment recommended
- GPU optional (for faster model inference)

FOLDER STRUCTURE

The repository is organized for clarity and scalability. The root directory contains the app.py file, which is the entry point for the Gradio-based interface. The backend folder houses the FastAPI services, while the models folder includes the logic for LLM integration, forecasting, and anomaly detection. UI-related assets are stored in the static folder, whereas documentation and screenshots are placed in the docs folder. The requirements.txt file lists all dependencies required for installation. This modular design allows team members to work on different components without interfering with one another.

- **app.py → Gradio app entry point.**
- **backend/ → FastAPI backend logic.**
- **models/ → AI modules: LLM integration, forecasting, anomaly detection.**
- **static/ → CSS and UI assets.**
- **docs/ → Documentation and screenshots.**
- **requirements.txt → Project dependencies.**

RUNNING THE APPLICATION

The application can be launched in two main parts. First, the frontend interface is run through python app.py, which launches a Gradio-based web interface accessible through a browser. This interface provides users with interactive tabs for symptom entry, treatment advice, and wellness tracking. Second, the backend services powered by FastAPI can be run using uvicorn. This setup allows the project to be scaled in the future, where the backend may be hosted on a server while the frontend interacts remotely through APIs.

- **Frontend (Gradio):** python app.py → opens in browser at http://127.0.0.1:7860/.
- **Backend (FastAPI):** uvicorn backend.api:app --reload.

API DOCUMENTATION

The system defines several API endpoints that enable modularity. For instance, POST /predict_disease accepts symptoms and returns possible health conditions. Similarly, POST /treatment_plan suggests remedies and lifestyle adjustments, while POST /diet_plan and POST /exercise_plan provide nutritional and physical fitness guidance. Other endpoints handle emergency steps and vital analysis. This modular API design makes the system compatible with external applications, mobile platforms, or wearable devices.

- POST /predict_disease
- POST /treatment_plan
- POST /natural_remedies
- POST /diet_plan
- POST /exercise_plan
- POST /emergency_steps
- POST /analyze_vitals

AUTHENTICATION

Currently, the system uses a simple login mechanism with static credentials (admin/1234). While sufficient for a prototype, this approach is not secure for production use. Future improvements involve adopting OAuth2 or JWT authentication to ensure secure access and support for multiple user profiles. This would allow healthcare institutions to integrate the system safely while maintaining user privacy.

- **Current:** Basic login (admin / 1234).
- **Future:** OAuth2 or JWT for secure multi-user support.

USER INTERFACE

The user interface is designed to be intuitive and visually appealing. Login pages are available in different themes such as Glassmorphism, Split Screen, and Neon Dark, providing flexibility in presentation. After logging in, users can access multiple tabs, each dedicated to a healthcare function—such as disease prediction, treatment plans, emergency care, and health tracking. The emphasis on a simple but modern interface ensures that users with minimal technical knowledge can operate the system comfortably.

TESTING

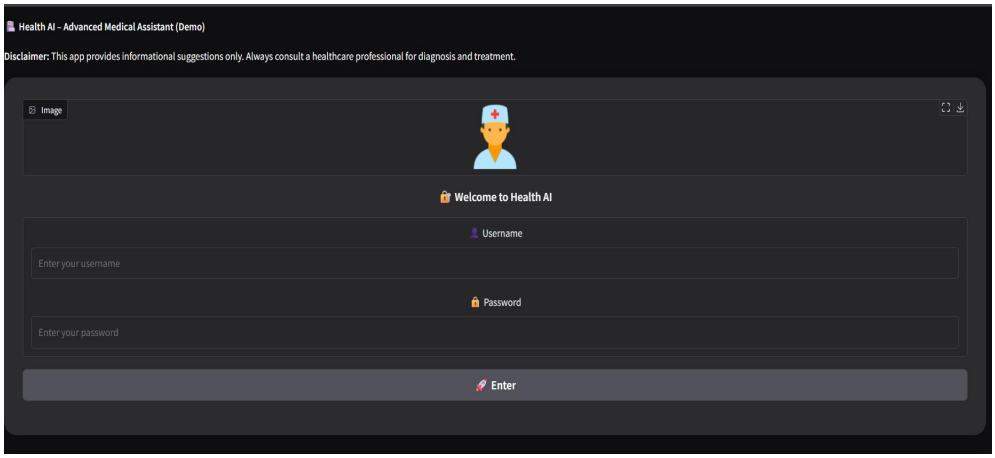
Testing has been conducted primarily through manual interaction using the Gradio interface. Users were able to input symptoms and verify whether the system produced consistent and relevant results. A basic unit test was also demonstrated, such as verifying that the disease prediction function returns expected keywords like “Possible Conditions.” In future iterations, automated testing with frameworks like PyTest will be integrated to ensure stability and reliability across updates.

assert "Possible Conditions" in disease_prediction("fever, cough, chest pain")

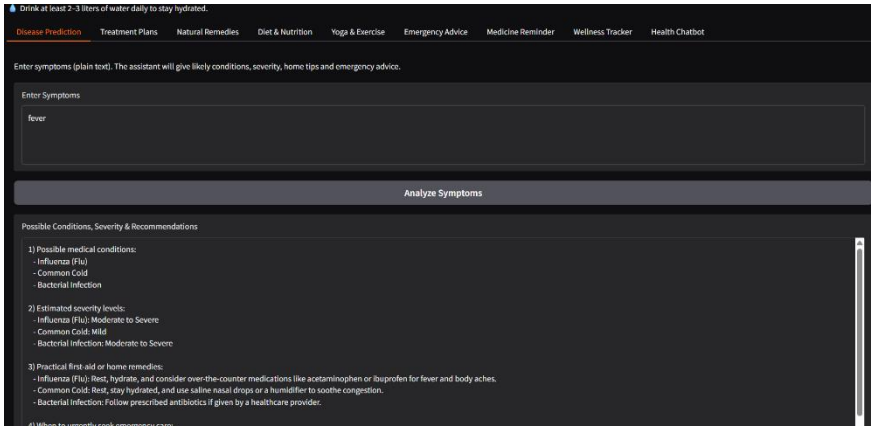
- **Login Styles:** Glassmorphism, Split Screen, Neon Dark.
- **Tabs:**
    - Disease Prediction
    - Treatment Plans
    - Natural Remedies
    - Diet & Nutrition
    - Yoga & Exercise
    - Emergency Advice
    - Medicine Reminder
    - Wellness Tracker
    - Health Chatbot

SCREENSHOTS

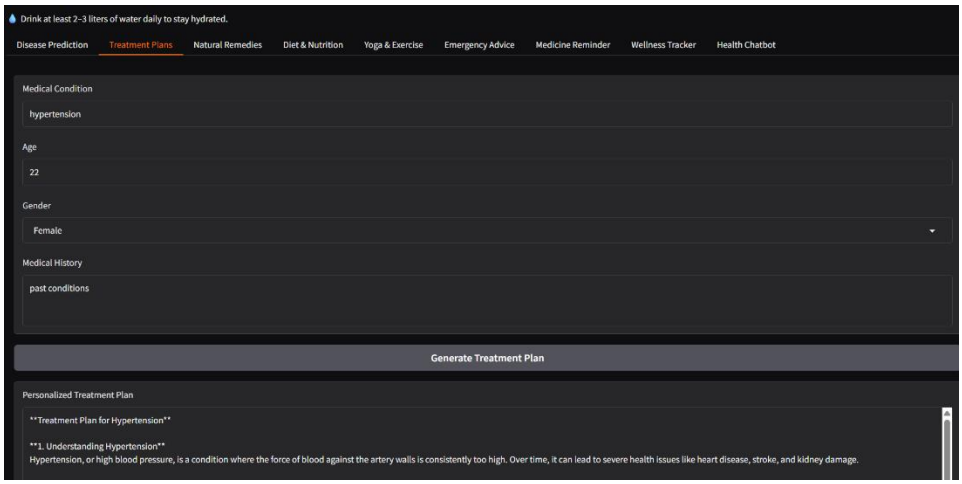
The documentation includes placeholders for screenshots of the interface.

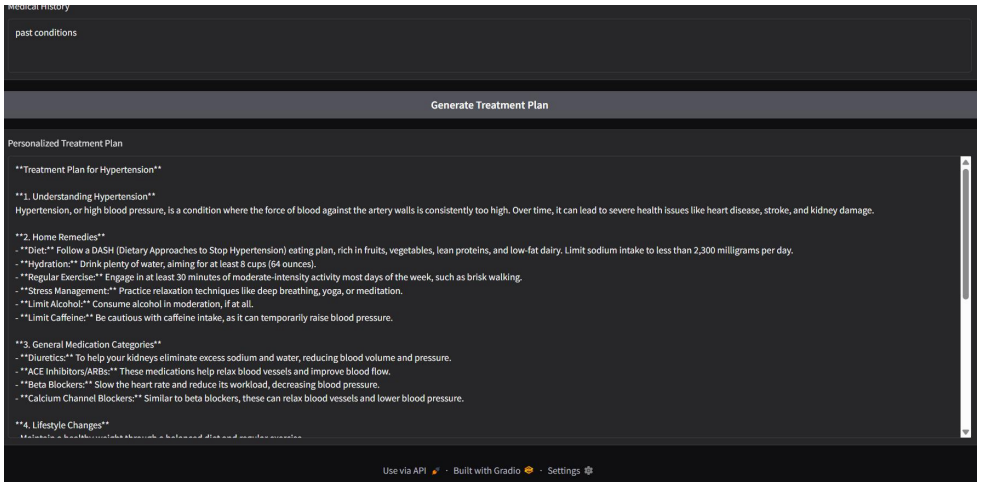


Login page

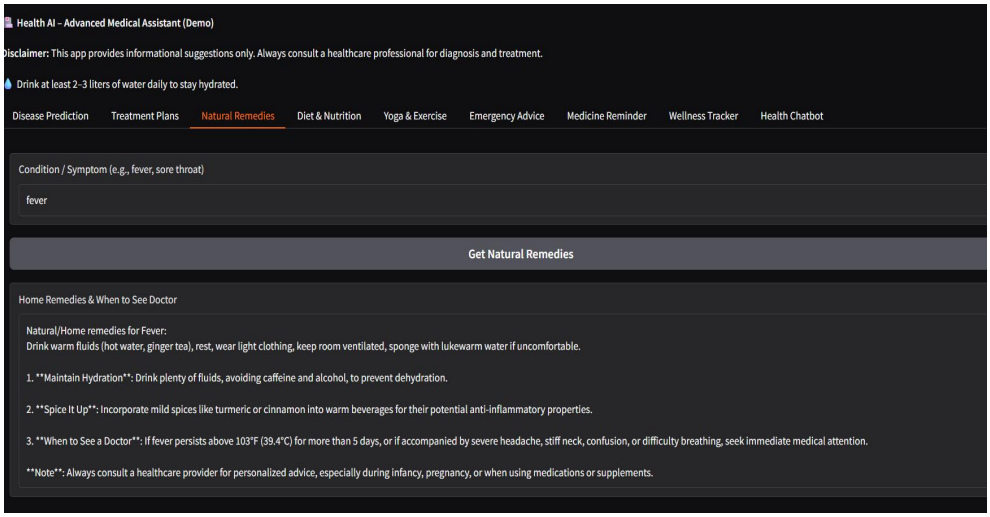


1. Disease Prediction





2.Treatment Plans



3. Natural Remedies

These screenshots showcase the login styles, the symptom input process, and the output of modules such as the Dieases Prediction, Treatment plans, Natural remedies. Visual documentation helps both developers and stakeholders understand the functionality at a glance and provides references for user training.

KNOWN ISSUES

Like any prototype, the system comes with limitations. It is not a substitute for professional medical consultation, and its advice should only be used for guidance. Responses may vary depending on the language model's interpretation, leading to occasional inconsistency. The authentication system is basic and not suitable for sensitive environments. Features like Pinecone-based history retrieval and persistent user storage are not yet implemented, limiting the project's ability to remember past interactions.

- Not a substitute for medical diagnosis.
- Output may vary due to LLM randomness.
- Authentication is basic → security risk in production.
- No persistent storage of users or history yet.
- Pinecone/vector search not yet implemented.

FUTURE ENHANCEMENTS

The project is designed with scalability in mind. Planned upgrades include a fully deployed FastAPI backend, Pinecone integration for memory and context, and advanced anomaly

detection models for better monitoring of vitals. There is also a roadmap for supporting multimodal input, where users can upload images or speak symptoms for analysis. Expanding to mobile applications and integrating with IoT devices like smartwatches or health trackers will further improve accessibility. Enhanced authentication and security will make the system suitable for institutional adoption.

- FastAPI backend deployment.
- Pinecone integration for memory and history.
- Advanced ML models for anomaly detection.
- Voice/image input (multimodal).
- Mobile version.
- IoT device integration (wearables, glucose monitors, oximeters).
- Secure login with OAuth2/JWT.