Medical AI Assistant

1. Introduction

Project Title: Medical AI Assistant

Team Members: JAYSON R S

MOHAMED IRFAN j

NITHEESH KANNA P

ESAKKIMUTHU SANJAY E

2. Project Overview

• Purpose:

The Medical AI Assistant is designed to serve as an accessible healthcare information support tool. It leverages natural language processing (NLP) and large language models to deliver accurate, concise, and user-friendly responses to health-related queries. It empowers users to understand their symptoms, receive health tips, and gather insights into possible conditions and medications while ensuring that users are reminded that professional consultation is mandatory.

• Features:

- Conversational Interface: Natural language chat with the AI model.
- Disease Prediction: Predicts possible medical conditions based on symptoms.
- Treatment Plans: Suggests home remedies and general medication guidelines.
- Health Tips: Provides 5 actionable tips for a selected topic.
- Medication Information: Details about usage, side effects, and precautions.
- Authentication System: Login/Logout mechanism for controlled access.
- Gradio Web UI: Simple and responsive user interface with multiple tabs.

3. Architecture

Frontend (Gradio):

The frontend is implemented using Gradio Blocks, Tabs, Rows, and Columns. It provides a modern, user-friendly interface with distinct sections for login, disease prediction, treatment plans, health tips, and medication information. It also supports a clean layout with markdown headers, textboxes, dropdowns, and buttons.

Backend (Transformers + PyTorch):

The backend integrates Hugging Face's Transformers library along with PyTorch. The IBM Granite LLM is used for conversational AI, capable of understanding natural language queries and generating human-like responses.

Authentication:

A basic authentication system is implemented where users must log in using predefined credentials (user/password). Once authenticated, the user is redirected to the main application. Logout resets the session.

4. Setup Instructions

Prerequisites:

- Python 3.9 or higher

- pip installed

- Installed libraries: gradio, torch, transformers

- Internet access to download models

Installation Process:

- Clone or download the project repository.

- Run `pip install -r requirements.txt` to install dependencies.

- Start the application using `python app.py`.

- Access the Gradio interface via localhost or the shareable link.

5. Folder Structure

medical_ai_assistant/

├── app.py # Main Gradio application

├── requirements.txt # Required dependencies

├── README.md # Project documentation

6. Running the Application

- Run the script using `python app.py`.

- Use demo credentials (Username: user, Password: password).

- Explore functionalities:

  * Enter symptoms for disease prediction.

  * Provide condition, age, gender, and history for a treatment plan.

  * Enter a health topic to get 5 practical tips.

  * Search for information about medications.

7. Authentication

The application uses a very simple authentication method for demonstration:

- Username: user

- Password: password

Once logged in, the user can access all features of the app. The logout button ends the session and redirects back to the login page.

8. User Interface

The UI is divided into sections:

- Login Page: Provides username and password fields.

- Disease Prediction Tab: Input symptoms and view possible conditions.

- Treatment Plans Tab: Input medical details to generate a treatment plan.

- Health Tips Tab: Input topic and get useful tips.

- Medication Information Tab: Input medicine name and get details.

Each output is displayed in a text area for easy readability.

9. Testing

Testing was carried out in different scenarios:

- Entering valid and invalid login credentials.

- Providing different sets of symptoms to test disease prediction.

- Inputting different conditions and ages for treatment plan outputs.

- Asking for health tips on diet, exercise, and stress management.

- Searching medication info for Paracetamol, Ibuprofen, and antibiotics.

- Verifying logout functionality returns to login page.

10. Future Enhancements

- Enhanced authentication (JWT, OAuth).

- Secure storage of user health history.

- Integration with official medical databases.

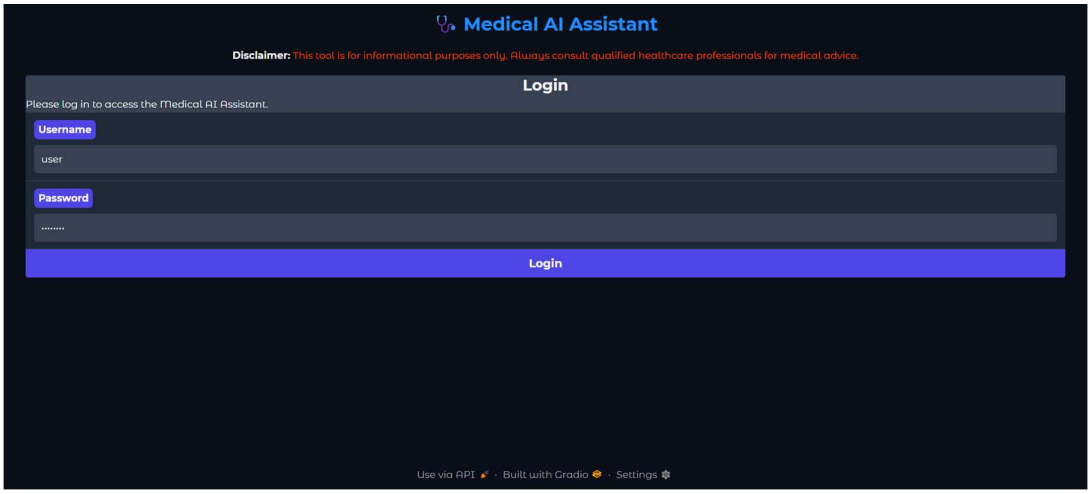
- Multi-language support for accessibility.

- Voice-based interaction for hands-free use.

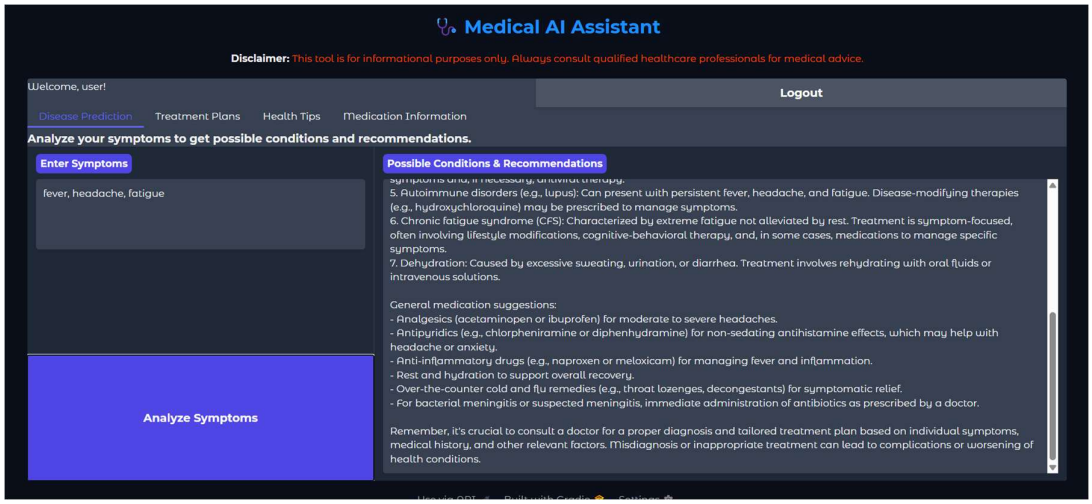
- Deployment on cloud with scalability (Docker/Kubernetes).

11. Screenshots

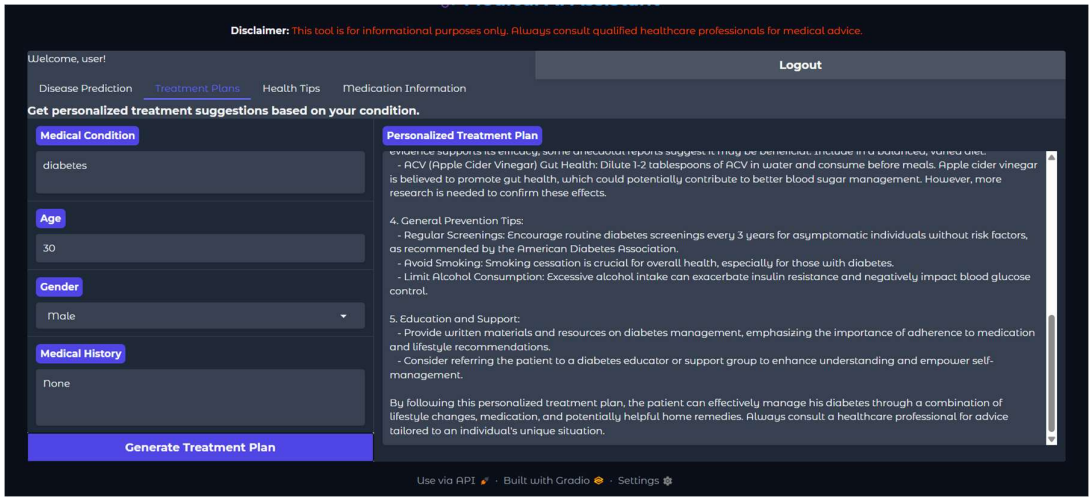
Login screen:



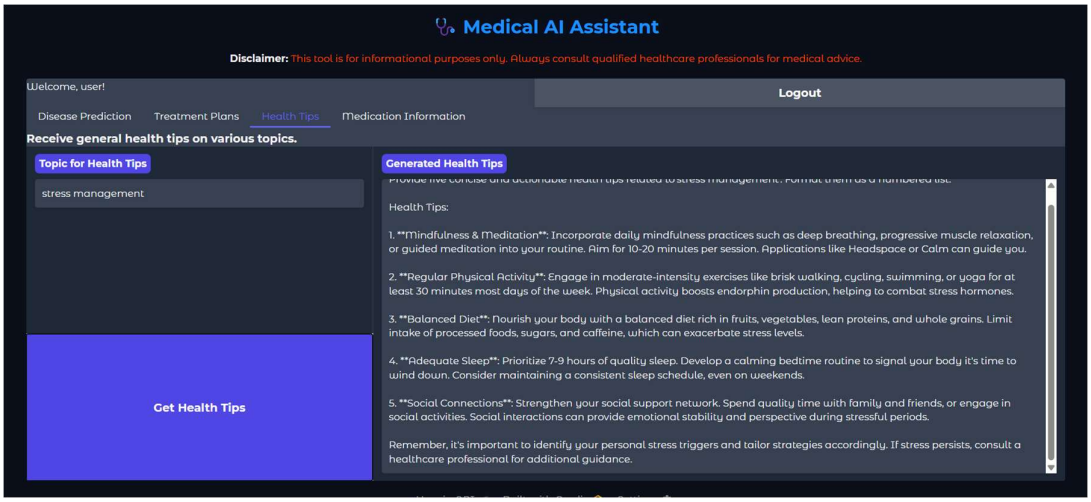
Disease Prediction:



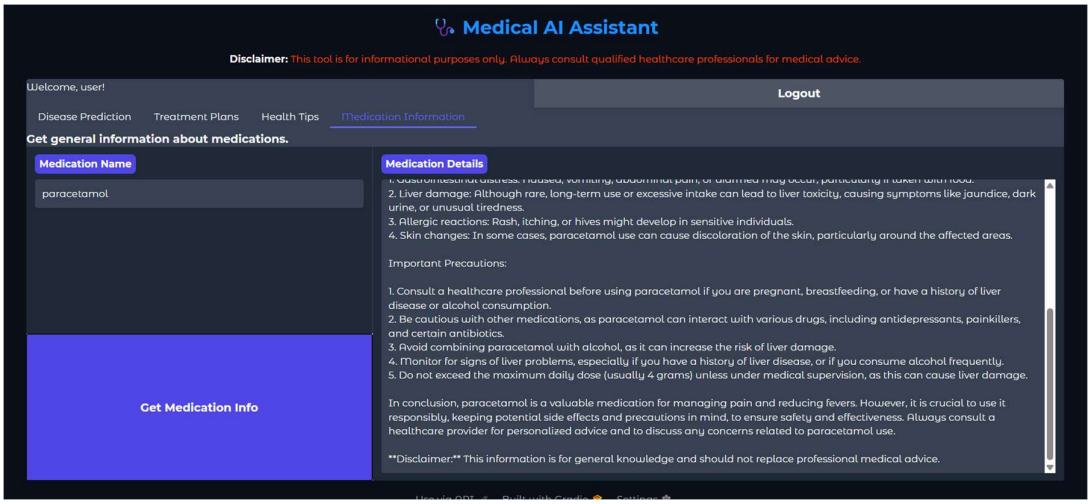
Treatment Plans:



Health Tips:



Medication Information:



12. Known Issues

- Model response may vary due to randomness in text generation.

- Login credentials are hardcoded (not secure).

- No database integration; all data is session-based.

- Cannot replace professional medical advice.

13. Conclusion

The Medical AI Assistant demonstrates how large language models can be applied to healthcare information systems. It shows how AI can support patients and general users by offering quick access to medical knowledge. With future improvements, it has the potential to become a robust tool for health awareness and patient engagement.