# Health AI: Intelligent Healthcare Assistant

## Generative AI with IBM

## 1. Introduction

Project Title:

Health AI: Intelligent Healthcare Assistant

Team Members:

- D. Kaviya
- K. Karthika

## 2. Project Overview

Purpose:

The purpose of the Health AI: Intelligent Healthcare Assistant is to enhance healthcare delivery by leveraging advanced AI technologies like IBM Watsonx Granite to offer personalized and accessible healthcare solutions. The assistant helps patients, caregivers, and healthcare professionals by providing real-time information, predictive analytics, and decision-making support. It aims to improve health outcomes, patient engagement, and clinical efficiency through a seamless and intuitive interface.

## Features:

Conversational Interface

Key Point: Natural language interaction

Functionality: Allows users to ask health-related questions, receive medication guidelines, and access preventive care tips using plain language.

## Symptom Analysis & Recommendation

Key Point: Personalized care suggestions

Functionality: Analyzes symptoms based on user input and provides possible causes, recommended actions, and alerts for serious conditions.

## Medical Report Summarization

Key Point: Simplified healthcare documentation

Functionality: Converts lengthy medical reports, prescriptions, or discharge summaries into concise, understandable formats for users.

## Health Forecasting

Key Point: Predictive analytics

Functionality: Uses historical health data and patterns to predict trends such as blood sugar fluctuations, medication adherence, or potential health risks.

## Lifestyle Guidance

Key Point: Preventive health support

Functionality: Provides personalized recommendations for diet, exercise, and mental wellness based on user health profiles.

## Emergency Alerts

Key Point: Safety and intervention

Functionality: Detects anomalies in vital signs or health reports and sends alerts to users and healthcare providers.

## Multimodal Input Support

Key Point: Versatile data processing

Functionality: Accepts text input, PDF reports, and structured health data in CSV formats for analysis.

## User-Friendly Interface (Streamlit or Gradio)

Key Point: Accessibility for all users

Functionality: Offers intuitive dashboards for patients, caregivers, and medical professionals to interact with the assistant.

# 3. Architecture

Frontend (Streamlit):

The interface is built using Streamlit to create an interactive web dashboard. It includes pages for health queries, document uploads, symptom input forms, and reports. Navigation is handled through an easy sidebar layout for quick access.

## Backend (FastAPI):

FastAPI serves as the backend framework that manages API endpoints for document processing, symptom analysis, forecasting, and user interaction. The system is optimized for asynchronous performance and easy integration with AI services.

## LLM Integration (IBM Watsonx Granite):

The Granite LLM models from IBM Watsonx are employed to understand natural language queries, generate health recommendations, and summarize medical documents effectively.

## Vector Search (Pinecone):

Uploaded medical documents and historical health data are converted into embeddings using Sentence Transformers and stored in Pinecone. Semantic search allows users to retrieve relevant health information quickly.

## ML Modules (Forecasting and Anomaly Detection):

Lightweight machine learning models in Scikit-learn are used for health trend forecasting and detecting anomalies in vital signs or medical reports. Visualization is powered by pandas and matplotlib.

# 4. Setup Instructions

## Prerequisites:

* Python 3.9 or later
* pip and virtual environment tools
* API keys for IBM Watsonx and Pinecone
* Internet access

## Installation Process:

1. Clone the project repository

2. Install dependencies using requirements.txt

3. Create a .env file with necessary credentials

4. Launch the backend server via FastAPI

5. Open the frontend using Streamlit

6. Upload medical reports or health data and explore interactive modules

# 5. Folder Structure

app/ – Contains backend logic including routers, models, and integrations

app/api/ – Subdirectory for APIs like symptom analysis, report generation, and feedback

ui/ – Contains frontend components for the dashboard and forms

health_dashboard.py – Entry point for the Streamlit interface

granite_llm.py – Handles communication with IBM Watsonx Granite for queries and summaries

document_embedder.py – Embeds medical documents into Pinecone

health_forecaster.py – Models trends like blood pressure or glucose levels

anomaly_checker.py – Detects irregularities in health data

report_generator.py – Generates personalized health summaries and recommendations

# 6. Running the Application

1. Launch the FastAPI backend server

2. Start the Streamlit frontend dashboard

3. Navigate through pages for symptom input, document upload, and health insights

4. View summaries, health predictions, and recommendations in real-time

# 7. API Documentation

## Available Endpoints:

* POST /chat/ask – Process health queries and provide responses.

* POST /upload-doc – Upload medical files for embedding.

* GET /search-docs – Retrieve documents based on symptoms or queries.

* GET /get-health-tips – Provides wellness and lifestyle advice.

* POST /submit-feedback – Allows users to submit feedback.

All endpoints are documented using Swagger UI for testing and inspection.

## 8. Authentication

For demonstration purposes, the system operates in an open environment. Future versions will implement:

* JWT or API key-based authentication

* OAuth2 with IBM Cloud credentials

* Role-based access (patient, caregiver, doctor)

* Session tracking and secure data access

## 9. User Interface

The interface focuses on accessibility and ease of use:

* Sidebar navigation

* Health KPI visualizations

* Real-time chat for symptom analysis

* Forms for document upload

* PDF reports with summaries and alerts

# 10. Testing

Testing includes:

* Unit testing of AI prompts and logic

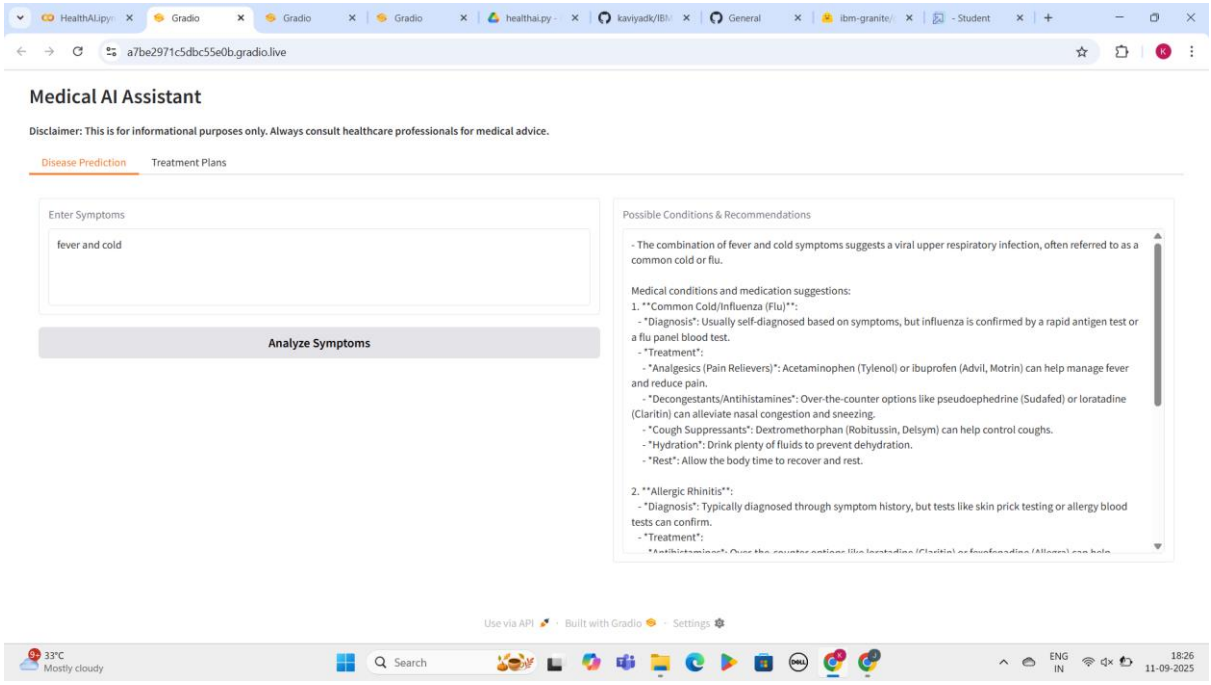
* API testing using Swagger and Postman

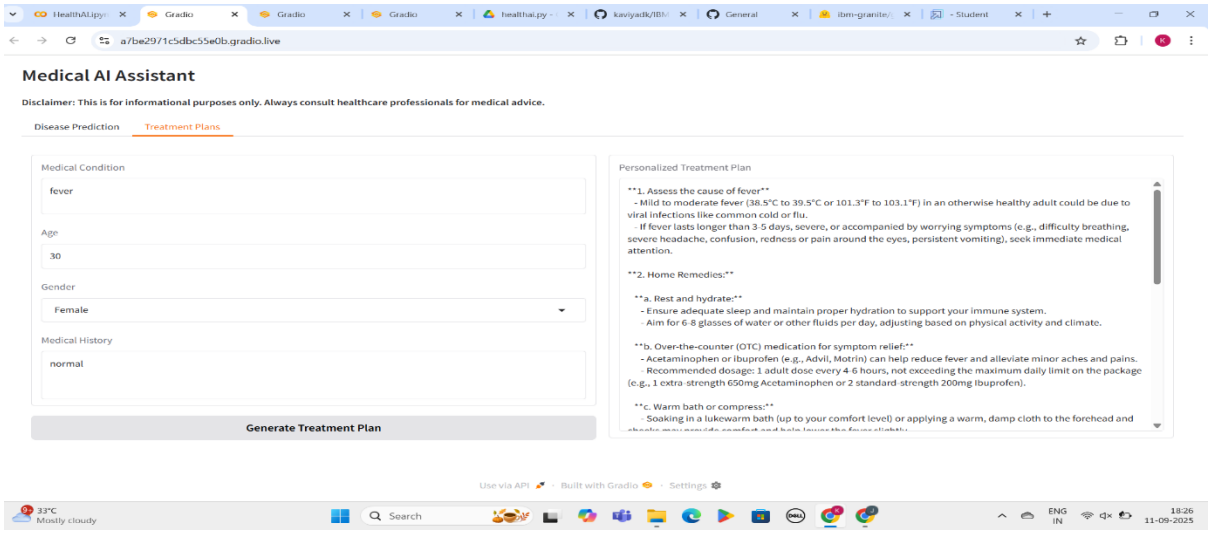
* Manual testing for symptom inputs and uploads

* Handling edge cases like incorrect files or invalid data

# 11. Screenshot

OUTPUT: The program has been executed the output will given the link to search about Health and it suggested a solution of the issue.






# 12. Known Issues

* Accuracy depends on user input and available data

* Limited dataset for rare medical conditions

* Privacy features to be integrated in future versions

# 13. Future Enhancements

* Integration with wearable health devices

* Support for voice input and emergency call triggers

* Advanced diagnostics using imaging data

* Multilingual support for regional healthcare access

* Secure deployment with encryption and compliance standards.