

HealthAI: Intelligent Healthcare Assistant Using IBM Granite

Introduction:

Team ID : NM2025TMID06786

Team Size : 4




Team Leader : MONIKA

Team member : KASTHURI G

Team member : KARPAGAM S

Team member : JAMUNA A

Sure, Here a comprehensive explanation of the **HealthAi-Intelligent healthcare assistant using IBM granite** project, covering the following,

1.  Project Overview
2.  Architecture
3.  Setup Instructions
4.  Folder Structure
5.  Running the Application
6.  API Documentation
7.  Authentication
8.  User Interface
9. Testing

Project Description:

HealthAI harnesses IBM Watson Machine Learning and Generative AI to provide intelligent healthcare assistance, offering users accurate medical insights. The platform includes a Patient Chat for answering health-related questions, Disease Prediction that evaluates user-reported symptoms to deliver potential condition details, Treatment Plans that provide personalized medical recommendations, and Health Analytics to visualize and monitor patient health metrics.

Utilizing IBM's Granite-13b-instruct-v2 model, HealthAI processes user inputs to deliver personalized and data-driven medical guidance, improving accessibility to healthcare information. Built with Streamlit and powered by IBM Watson, the platform ensures a seamless and user-friendly experience. With secure API key management and responsible data handling, HealthAI empowers users to make informed health decisions with confidence.

Scenarios:

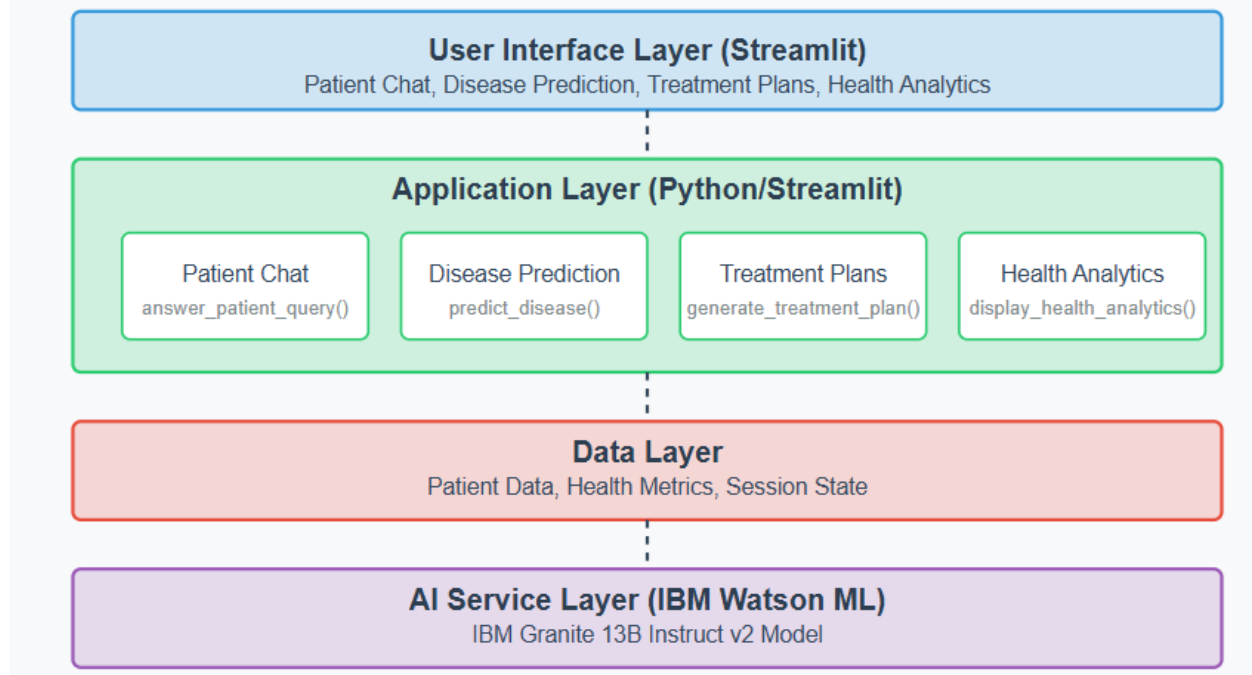
Scenario 1: A user inputs their symptoms into the Disease Prediction system, describing issues like persistent headache, fatigue, and mild fever. The system analyzes the symptoms along with the patient's profile and health data to provide potential condition predictions, including likelihood assessments and recommended next steps.

Scenario 2: A user needs personalized treatment recommendations for a diagnosed condition. By entering their condition in the Treatment Plans generator, the AI processes the information along with patient data to create a comprehensive, evidence-based treatment plan that includes medications, lifestyle modifications, and follow-up testing.

Scenario 3: A user wants insights about their health trends. Using the Health Analytics dashboard, they can visualize their vital signs over time (heart rate, blood pressure, blood glucose, etc.) and receive AI-generated insights about potential health concerns and improvement recommendations.

Architecture

HealthAI - Architecture Diagram



Pre-requisites

1. Streamlit Framework Knowledge: [Streamlit Documentation](#)
2. IBM Watson Machine Learning: [IBM Watson ML Documentation](#)
3. Python Programming Proficiency: [Python Documentation](#)
4. Data Visualization Libraries: [Plotly Documentation](#)
5. Version Control with Git: [Git Documentation](#)
6. Development Environment Setup: [Flask Installation Guide](#)

Activity 1: Model Selection and Architecture

- **Activity 1.1:** Research and select the appropriate AI model from IBM Watson for medical assistance (IBM Granite 13B Instruct v2).
- **Activity 1.2:** Setup and Access your IBM WatsonX API key.
- **Activity 1.3:** Define the architecture of the application, detailing interactions between the frontend, backend, and AI integration.
- **Activity 1.4:** Set up the development environment, installing necessary libraries and dependencies for Streamlit and IBM Watson ML.

Activity 2: Core Functionalities Development

- **Activity 2.1:** Develop the core functionalities: Patient Chat, Disease Prediction, Treatment Plan Generation, and Health Analytics.
- **Activity 2.2:** Implement patient data utilities to manage and visualize health metrics.

Activity 3: App.py Development

- **Activity 3.1:** Write the main application logic in app.py, establishing functions for each feature and integrating AI responses.
- **Activity 3.2:** Create prompting strategies for the IBM Granite model to generate high-quality medical content.

Activity 4: Frontend Development

- **Activity 4.1:** Design and develop the user interface using Streamlit components, ensuring a responsive and intuitive layout.
- **Activity 4.2:** Create dynamic visualizations with Plotly to display health metrics and trends.

Activity 5: Deployment

- **Activity 5.1:** Prepare the application for deployment by configuring environment variables for API credentials.
- **Activity 5.2:** Deploy the application on a suitable hosting platform to make it accessible to users.

Milestone 1: Model Selection and Architecture

- In this milestone, we focus on selecting the appropriate AI model from IBM Watson for our medical assistance needs. This involves researching the capabilities and performance of various models, ensuring that the chosen model aligns well with our application's objectives of creating a Patient Chat system, Disease Prediction, Treatment Plan Generation, and Health Analytics.

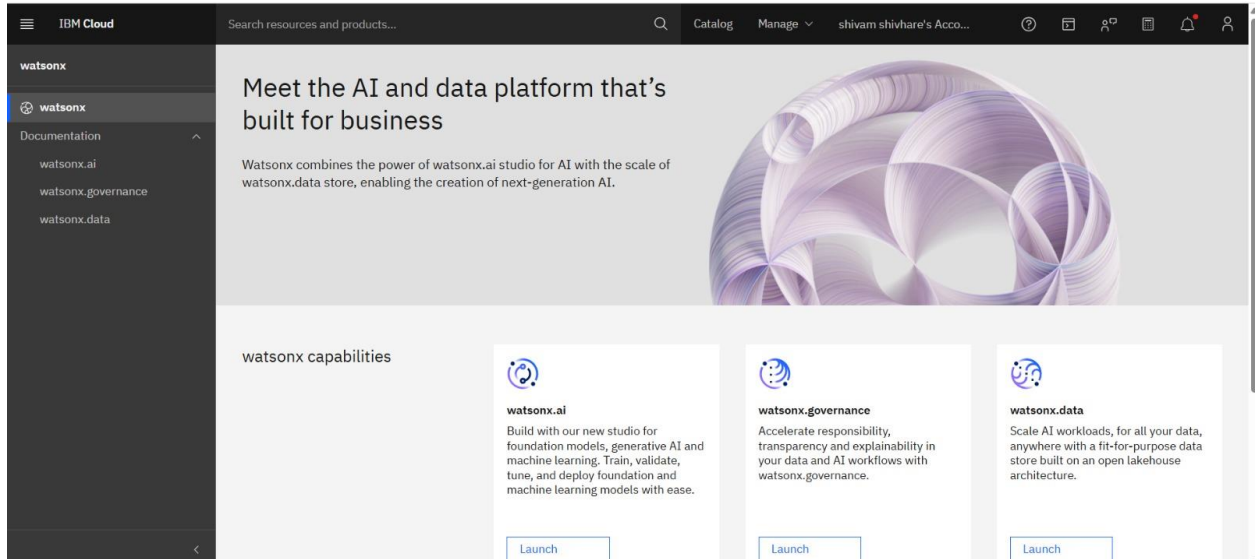
Activity 1.1: Research and select the appropriate AI model

1. **Understand the Project Requirements:** Review the specific needs of the healthcare application.
2. **Explore IBM Watson ML Documentation:** Examine the various models available, including their functionalities and limitations.

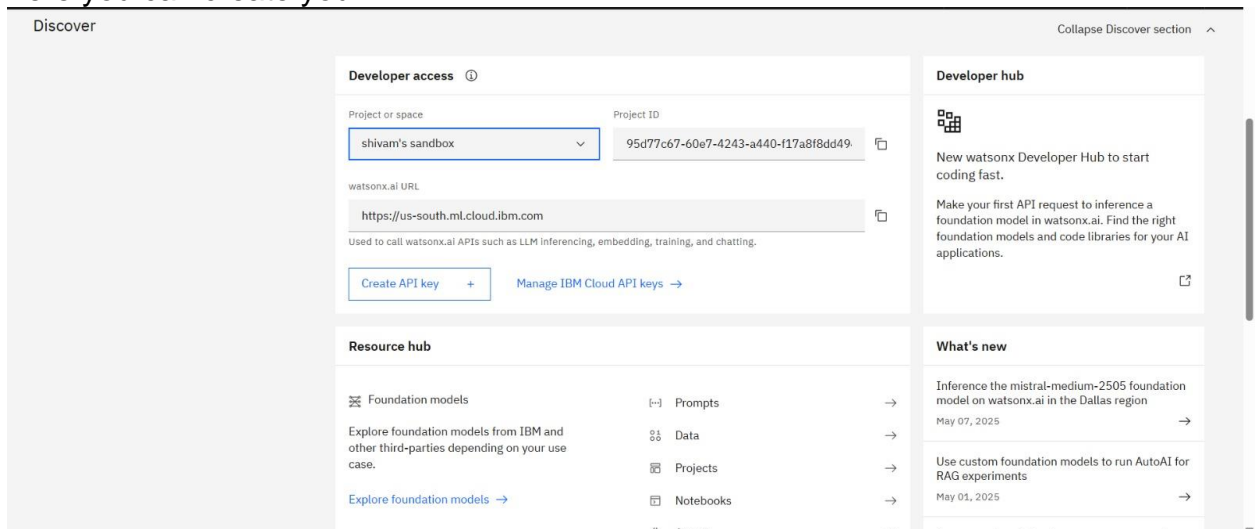
3. Select the Optimal Model: Choose IBM's Granite 13B Instruct v2 model for its strong performance with healthcare-related content.

Activity 1.2: Setup and Access your IBM WatsonX API key.

1. Go to IBM WatsonX
2. Then, Launch watsonx.ai



3. Here you can create your API



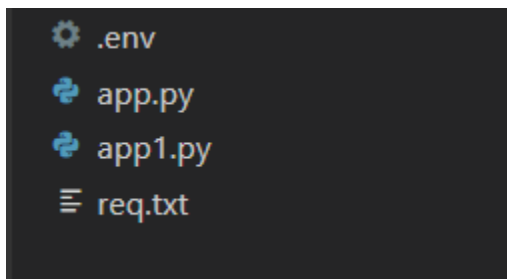
Activity 1.3: Define the architecture of the application

1. Draft an Architectural Diagram: Create a visual representation of the application architecture.

2. **Detail Frontend Functionality:** Outline how users will interact with the application through a Streamlit interface.
3. **Outline Backend Responsibilities:** Specify how the backend will process user input and communicate with the IBM Granite model.
4. **Describe AI Integration Points:** Define how the application will make API calls to IBM Watson ML.

Activity 1.4: Set up the development environment

1. **Install Python and Pip:** Ensure Python is installed along with pip for managing dependencies.
2. **Create a Virtual Environment:** Set up a virtual environment to isolate project dependencies.
3. **Install Required Libraries:**
`pip install streamlit pandas numpy plotly ibm-watson-machine-learning python-dotenv`
4. **Set Up Application Structure:** Create the initial directory structure for the HealthAI application.



Milestone 2: Core Functionalities Development

Activity 2.1: Develop core functionalities

1. **Patient Chat System:**
 - Implement conversational interface for answering health questions
 - Create prompting system for the IBM Granite model to provide medical advice
 - Develop session-based chat history management
2. **Disease Prediction System:**
 - Create symptom input interface
 - Develop prediction function using patient data and reported symptoms

- Structure output format to show potential conditions with likelihood and next steps
- 3. **Treatment Plan Generator:**
 - Build input interface for condition and patient details
 - Create prompting system for personalized treatment plans
 - Structure output to include medications, lifestyle changes, and follow-up care
- 4. **Health Analytics Dashboard:**
 - Implement patient data visualization with interactive charts
 - Create metrics summary with trend indicators
 - Develop AI-generated insights based on health trends

Activity 2.2: Implement data management utilities

1. **Patient Data Generation:**
 - Create sample data with realistic health metrics
 - Implement date-based trend generation for visualization
 - Structure data for efficient analysis and display
2. **Patient Profile Management:**
 - Develop interface for managing patient details
 - Create session state handling for persistent data
 - Implement profile update functionality
 - and return JSON responses.
2. **Integrate the Gemini AI Responses in Each Function**
 - Implement function calls within each route to leverage Gemini's models based on user-provided details.
 - Ensure each response from Gemini API is processed in a way that enhances readability and displays clearly to the use.
 - Implement profile update functionality

Milestone 3: App.py Development

Activity 3.1: Write the main application logic

The app.py file is organized into several key sections:\

1. **Imports and Setup:**
 - Import necessary libraries (Streamlit, pandas, plotly, IBM Watson ML)
 - Load environment variables for API keys
 - Initialize IBM Granite model connection
2. **Core Functions:**
 - `init_granite_model()`: Set up connection to IBM Watson ML
 - `predict_disease()`: Analyze symptoms for potential diagnoses

- `generate_treatment_plan()`: Create personalized treatment recommendations
 - `answer_patient_query()`: Process health questions with AI responses
3. **UI Components:**
- Main application layout with sidebar navigation
 - Tab-based interface for different features
 - Custom CSS styling for enhanced user experience
4. **Feature Implementation:**
- `display_patient_chat()`: Chatbot interface for health questions
 - `display_disease_prediction()`: Symptom analysis system
 - `display_treatment_plans()`: Treatment plan generator

`display_health_analytics()`: Interactive health dashboard

Activity 3.2: Create prompting strategies

Patient Query Prompting:

```
def answer_patient_query(query):  
    """Use IBM Granite to answer patient health questions"""  
    model = init_granite_model()  
  
    # Create prompt for answering patient query  
    query_prompt = f"""  
    As a healthcare AI assistant, provide a helpful, accurate, and evidence-based response to the following patient question:  
  
    PATIENT QUESTION: {query}  
  
    Provide a clear, empathetic response that:  
    - Directly addresses the question  
    - Includes relevant medical facts  
    - Acknowledges limitations (when appropriate)  
    - Suggests when to seek professional medical advice  
    - Avoids making definitive diagnoses  
    - Uses accessible, non-technical language  
  
    RESPONSE:  
    """  
  
    answer = model.generate_text(prompt=query_prompt)  
    return answer
```



```

prediction_prompt = f"""
As a medical AI assistant, predict potential health conditions based on the following patient data:

Current Symptoms: {symptoms}
Age: {age}
Gender: {gender}
Medical History: {medical_history}

Recent Health Metrics:
- Average Heart Rate: {avg_heart_rate} bpm
- Average Blood Pressure: {avg_bp_systolic}/{avg_bp_diastolic} mmHg
- Average Blood Glucose: {avg_glucose} mg/dL
- Recently Reported Symptoms: {recent_symptoms}

Format your response as:
1. Potential condition name
2. Likelihood (High/Medium/Low)
3. Brief explanation
4. Recommended next steps

Provide the top 3 most likely conditions based on the data provided.
"""

prediction = model.generate_text(prompt=prediction_prompt)
return prediction

```

Treatment Plan Prompting:

```

treatment_prompt = f"""
As a medical AI assistant, generate a personalized treatment plan for the following scenario:

Patient Profile:
- Condition: {condition}
- Age: {age}
- Gender: {gender}
- Medical History: {medical_history}

Create a comprehensive, evidence-based treatment plan that includes:
1. Recommended medications (include dosage guidelines if appropriate)
2. Lifestyle modifications
3. Follow-up testing and monitoring
4. Dietary recommendations
5. Physical activity guidelines
6. Mental health considerations

Format this as a clear, structured treatment plan that follows current medical guidelines while being personalized to this patient's specific needs.
"""

treatment_plan = model.generate_text(prompt=treatment_prompt)
return treatment_plan

```

```

treatment_prompt = f"""
As a medical AI assistant, generate a personalized treatment plan for the following scenario:

Patient Profile:
- Condition: {condition}
- Age: {age}
- Gender: {gender}
- Medical History: {medical_history}

Create a comprehensive, evidence-based treatment plan that includes:
1. Recommended medications (include dosage guidelines if appropriate)
2. Lifestyle modifications
3. Follow-up testing and monitoring
4. Dietary recommendations
5. Physical activity guidelines
6. Mental health considerations

Format this as a clear, structured treatment plan that follows current medical guidelines while being personalized to this patient's specific needs.
"""

treatment_plan = model.generate_text(prompt=treatment_prompt)
return treatment_plan

```

Milestone 4: Frontend Development

Activity 4.1: Design and develop the user interface

1. **Main Application Layout:**
 - Configure page title, icon, and layout preferences
 - Implement a sidebar for patient profiles and feature selection
 - Create custom CSS for enhanced visual appearance
2. **Feature-Specific Interfaces:**
 - **Patient Chat:** Chat-style interface with message history
 - **Disease Prediction:** Symptom input form and prediction display
 - **Treatment Plans:** Condition input and treatment plan output
 - **Health Analytics:** Interactive charts and metrics summary

Activity 4.2: Create dynamic visualizations

1. **Health Metric Charts:**
 - Heart rate trend line chart
 - Blood pressure dual-line chart
 - Blood glucose trend line chart with reference line
 - Symptom frequency pie chart
2. **Metrics Summary:**
 - Key health indicators with trend deltas
 - Color-coded metrics to indicate normal/abnormal ranges
 - Interactive tooltip information

Milestone 5: Deployment

Activity 5.1: Prepare for deployment

1. Environment Variable Configuration:

Create a `.env` file for IBM Watson API credentials:

`WATSONX_API_KEY=your_api_key_here`

- `WATSONX_PROJECT_ID=your_project_id_here`

- Implement secure loading of credentials

2. Dependency Management:

- Create `requirements.txt` file with all necessary packages
- Document installation process for deployment

Activity 5.2: Deploy the application

1. Local Deployment Testing:

- Run the application using `streamlit run app.py`
- Test all features for functionality
- Verify responsive design and performance

2. Cloud Deployment Options:

- Deploy on Streamlit Cloud for public access
- Configure environment variables in the deployment platform
- Set up monitoring and error logging

Milestone 6: functional testing and verify

Patient Chat Page

Patient Profile

Name

Age

0

—

+

Gender

Male

▼

Medical History

↗

Current Medications



↗

Allergies

Penicillin

Deploy

:

 **HealthAI - Intelligent Healthcare Assistant** 

24/7 Patient Support

Ask any health-related question for immediate assistance.

Ask your health question...

➤

Description: Here user have access to responsive healthcare communication platform enabling seamless dialogue about wellness concerns, with chronological message tracking for context retention. The system delivers intelligent, algorithmically-generated wellness guidance while providing verifiable medical insights supported by authoritative healthcare sources, creating a comprehensive virtual consultation experience.

Patient Chat output:

Patient Profile

Name

Rithvik

Age

22

Gender

Male

Medical History

None


Current Medications

None

Allergies


None


Deploy

 **HealthAI - Intelligent Healthcare Assistant**




24/7 Patient Support

Ask any health-related question for immediate assistance.

 I have been suffering from a fever for 2 days, my symptoms are running nose, cough, headache, and joint pain. So give me some medications

 Running nose, cough, headache, and joint pain are symptoms of the common cold. For most people, the common cold will resolve on its own within 7 to 10 days. However, if your symptoms persist or get worse, you should consult your doctor. If you are taking any other medications, please check with your doctor to see if they are safe to take with other medications.

Ask your health question...

Disease Prediction Page:

Treatment Plans Page:

Patient Profile

Name

Rithvik

Age

22

Gender

Male

Medical History

None


Current Medications


None

Allergies

None

Deploy

 **HealthAI - Intelligent Healthcare Assistant**

 **Personalized Treatment Plan Generator**

Generate customized treatment recommendations based on specific conditions.

Medical Condition

Mouth Ulcer

Generate Treatment Plan

Personalized Treatment Plan

1. Recommended medications (include dosage guidelines if appropriate) 2. Lifestyle modifications 3. Follow-up testing and monitoring 4. Dietary recommendations 5. Physical activity guidelines 6. Mental health considerations

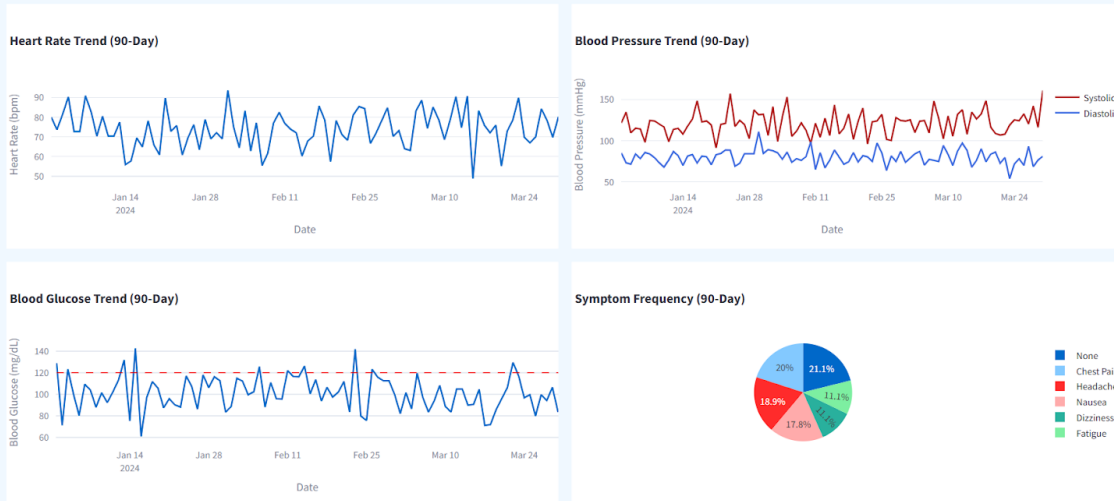
Description: This page contains medical condition entry area where users can specify their health concerns, complemented by a prominent action button for plan creation. Upon activation, the system produces comprehensive, organized therapeutic recommendations tailored to the specified condition. The interface includes essential medical advisories clarifying the informational nature of the provided guidance.

Health Analytics Dashboard:

HealthAI - Intelligent Healthcare Assistant

Health Analytics Dashboard

Visualize and analyze patient health data trends.



Health Metrics Summary

Avg. Heart Rate

74.0 bpm

↑ 6.1

Avg. Blood Pressure

120.8/79.9

↑ 40.0

Avg. Blood Glucose

101.2 mg/dL

↓ -17.8

Avg. Sleep

6.8 hours

↓ -0.6

AI-Generated Health Insights

- Heart rate, blood pressure, and blood glucose levels are all within normal range. - No current symptoms reported. - No current medications.

Description: The interface features dynamic visualizations including heart rate trends, blood pressure patterns, and blood glucose fluctuations over time. A color-coded pie chart illustrates symptom occurrence frequency, while the metrics summary section provides key health indicators with directional trend markers. The dashboard is enhanced with AI-powered insights that analyze collected data to offer personalized health recommendations and observations.

Conclusion

The HealthAI project effectively demonstrates the potential of AI in revolutionizing healthcare assistance. By integrating IBM's Granite language model, the platform enables users to receive personalized health insights through Patient Chat, Disease

Prediction, Treatment Plan Generation, and Health Analytics, making healthcare information more accessible.

Utilizing IBM Watson Machine Learning, the application ensures accurate health question answering, detailed disease prediction, personalized treatment recommendations, and insightful health trend analysis. The structured development process—spanning model selection, core feature implementation, backend and frontend development, and deployment—led to the creation of an interactive, user-friendly platform.

Built with Streamlit, HealthAI facilitates seamless visualization of health data and AI-generated insights, ensuring an efficient and responsive experience. This project highlights how targeted AI models and a well-structured framework can enhance healthcare accessibility. With future scalability in mind, HealthAI has the potential to expand its capabilities, incorporating more advanced diagnostics and broader medical applications.