**HEALTH AI**

**INTELLIGENT HEALTH CARE ASSISTANT USING IBM GRANITE**

**TEAM MEMBER 1 : MOPIDEVI DEEPTHI PRIYA**

**GMAIL :** [**deepthipriya562004@gmail.com**](mailto:deepthipriya562004@gmail.com)

**TEAM MEMBER 2 : BATHULA MOHANA SAMPATI NAGAMANI**

**GMAIL : sampatha2679@gmail.com**

**TEAM MEMBER 3 : GITTHADAPPA GARI VENU**

**GMAIL : venuvenug811@gmail.com**

**TEAM MEMBER 4 : KUNAPAREDDY LALITHA PRASANNA**

**GMAIL : lalithaprasannakunapareddy@gmail.com**

**INTRODUCTION :**

1. HealthAI is an intelligent healthcare assistant powered by IBM Watson Machine Learning and the Granite-13b-instruct-v2 generative AI model.
2. Designed to enhance accessibility to medical information,

HealthAI offers users accurate and personalized health

insights through four core features: Patient Chat, Disease Prediction, Treatment Plans, and Health Analytics.

1. Built with Streamlit and backed by secure, responsible AI integration, the platform empowers users to make informed health decisions in an interactive and user-friendly environment.

## Project Overview:

* HealthAI is an AI-driven healthcare assistant platform that leverages IBM Watson Machine Learning and the Granite-13b- instruct-v2 model to provide users with accurate, data-backed medical insights.
* The application is built using Streamlit for an intuitive user interface and integrates multiple healthcare functionalities into one cohesive system.

Key modules of HealthAI include:

* Patient Chat – An intelligent conversational interface for

answering health-related questions with empathetic and fact- based responses.

* Disease Prediction – A symptom analysis tool that evaluates user inputs to suggest potential conditions with associated probabilities and recommendations.
* Treatment Plan Generator – Delivers personalized treatment guidance, including medications, lifestyle adjustments, and follow-up care based on user profiles.
* Health Analytics Dashboard – Offers dynamic visualizations of vital signs and trends, along with AI-generated insights for proactive health monitoring.
* HealthAI ensures secure handling of sensitive data through environment-based API key management and focuses on providing medically informed, responsible AI support.
* The system is designed for scalability, with potential future integrations including EHR systems, wearable device data, and multilingual support.

## Purpose:

* The purpose of HealthAI is to enhance accessibility to reliable healthcare information by providing users with intelligent, AI- powered assistance for understanding symptoms, generating treatment plans, tracking health metrics, and answering

medical questions.

* By leveraging IBM’s Granite-13b-instruct-v2 model through Watson Machine Learning, HealthAI aims to support users in making informed health decisions, promote early awareness of potential health issues, and bridge

## IDEATION PHASE

* The ideation phase is where the concept for HealthAI was born, shaped, and refined. It involved identifying the core problem—limited access to immediate, reliable, and personalized healthcare information—and envisioning an AI- powered solution to address it.
* During this phase, the team:
* Researched real-world healthcare challenges faced by individuals, especially those without easy access to professional medical advice.
* Brainstormed potential features, such as symptom analysis, treatment suggestions, and health data tracking, that could empower users with meaningful insights.
* Explored AI capabilities, evaluating different models and ultimately selecting IBM’s Granite-13b-instruct-v2 for its strength in natural language understanding within healthcare contexts.
* Outlined the user journey and imagined practical use cases (e.g., Disease Prediction, Patient Chat), ensuring that the platform would be intuitive and helpful.
* Defined success criteria, such as accuracy, usability, and scalability, to guide design and development decisions.
* The ideation phase was essential to ensure that HealthAI was not just technically feasible but also truly valuable, addressing user needs with purpose-driven innovation.

## Problem Statement

* + - Access to timely, reliable, and personalized healthcare information remains a significant challenge across the globe. Many individuals struggle to obtain accurate medical guidance due to barriers such as:
    - Limited availability of healthcare professionals
    - Geographic or economic constraints
    - Lack of awareness or understanding of medical symptoms and treatment options
    - Difficulty tracking personal health trends over time
    - Traditional healthcare systems are often overburdened and cannot offer round-the-clock consultation or personalized attention for non-emergency concerns. Moreover, the absence of intelligent digital tools leaves patients without real-time

insights into their health data, leading to delayed or misinformed decisions.

* + - Need for the Solution
    - There is a critical need for an AI-powered healthcare assistant that can:
    - Provide medically-informed responses to user queries
    - Predict possible health conditions based on reported symptoms
    - Generate personalized treatment plans
    - Visualize and interpret health metrics over time
    - Such a system must be user-friendly, accessible from any

device, and capable of ensuring data privacy while delivering reliable and empathetic assistance.

## Empathy Map Canvas – HealthAI End User (Patient) SEES

* + - Complex and often confusing medical information online
    - Long queues or delays at hospitals and clinics Health apps that are difficult to use or not personalized
    - News and posts about AI in healthcare but unsure how to trust them

SAYS

* + - “I’m not sure if this symptom is serious
    - “I wish I could talk to a doctor now.”
    - “I don’t understand these medical terms.”
    - “Is this medicine safe for me?”
    - “It’s hard to track my health all the time.” THINKS
    - “What if I’m missing something important about my health?”
    - “I hope this app gives me the right advice.”
    - “Will this be private and secure?”
    - “How accurate is this AI in diagnosing?”
    - Is this information trustworthy?” FEELS
    - Worried, anxious about health issues
    - Overwhelmed by medical jargon and too much information
    - Powerless due to lack of access to expert medical helpCurious about technology’s ability to help
    - Relieved when they find clear, simple, helpful guidance

HEARS

* + - Friends or family suggesting health apps or “Dr. Google”
    - Social media posts about health hacks or AI doctors
    - News about rising healthcare costs overburdened systems
    - Caution from others about trusting AI with medical advice PAINS
    - Uncertainty about symptoms or self-diagnosis
    - Inaccessibility of real-time medical advice
    - Difficulty managing or understanding personal health data
    - Lack of personalized treatment guidance
    - Language and technical barriers in existing tools GAINS
    - Instant, AI-driven medical support anytime
    - Clear, understandable health insights
    - Personalized disease predictions and treatment plans
    - Visual health tracking and smart analytics
    - Confidence in making informed health decisions
    - A sense of empowerment and autonomy in managing theirhealth

## Brainstorming

1. User Needs C Problems
   * Users need instant medical help when doctors aren’t available.
   * Many people are confused by symptom checkers that aren’t personalized.
   * Health data (e.g., heart rate, BP) is often scattered and hard to interpret.
   * Language barriers and technical complexity in apps limit usability.
   * Users want trustworthy answers but fear AI mistakes or privacy issues.
2. Core Feature Ideas Patient Chat
   * Natural language conversation with empathetic tone
   * "Urgency detector" to suggest when professional help is critical
   * Language translation for multilingual support
   * Audio input/output for accessibility Disease Prediction
   * Dynamic symptom input: add/remove symptoms easily
   * Use past medical history for more accurate predictions
   * Risk scoring system with red/yellow/green indicators
   * Option to download/share results with doctor Treatment Plans
   * Personalized based on age, gender, allergies, etc.
   * Explain *why* each recommendation is given
   * Include alternative treatments (home remedies, diet)
   * Calendar-based plan with reminders Health Analytics
   * Daily/weekly/monthly view of vitals
   * AI trend detection: “Your blood pressure is rising steadily.”
   * Predictive alerts: “You may be at risk for X in the next 2 weeks.”
   * Integration with Fitbit, Apple Health, etc.
3. Technical Ideas
   * Prompt tuning or chain-of-thought prompting for more reliable AI outputs
   * Streamlit + Plotly for fast prototyping and visualization
   * Use .env and python-dotenv for secure API key handling
   * Cache frequently asked health questions to reduce API calls
   * Use secure backend logging for debugging without exposing health data
4. UX/UI Ideas
   * Tabbed layout with icons for features (Chat , Diagnosis , Plan , Analytics )
   * Progress bar or status indicator for long predictions
   * “Explain Like I’m 5” mode for simplified medical language
   * Dark mode for late-night usage
   * Feedback system: “Was this advice helpful?”
5. Scalability C Future Features
   * Add voice assistant integration (Alexa, Google Assistant)
   * Support elder care mode with large fonts and audio
   * Enable doctor login portal for reviewing AI predictions
   * Integrate emergency assistance button
   * Add mental health module for stress, anxiety, sleep, etc.
   * Implement data export to PDF or CSV for offline use
6. Privacy C Ethics
   * Transparent AI disclaimer: “This is not a replacement for a doctor.”
   * User consent screen for health data usage
   * Explain model limitations clearly (e.g., not trained for rare diseases)
   * Implement data anonymization for analytics
7. Engagement C Gamification
   * Daily health check-in reminders
   * Badges or progress trackers for consistency
   * “AI Health Coach” that motivates positive habits
   * Weekly health quiz or learning tips
   * A mind map?
   * A team workshop whiteboard template (Miro/FigJam style)?A prioritized task list.

## REQUIREMENT ANALYSIS

1. Functional Requirements

These define what the system *must do* to fulfill its purpose:

* 1. Patient Chat System
     + Users can input health-related queries in natural language.
     + System responds with AI-generated, empathetic, medically sound answers.
     + Conversation history is displayed chronologically.
     + Responses include disclaimers and suggest professional help when needed.
  2. Disease Prediction
     + Users can input symptoms through an interactive interface.
     + System analyzes symptoms and predicts possible conditions with likelihood scores.

Displays explanation and next recommended steps (e.g., consu

* + - lt doctor, monitor vitals).
    - Supports multiple symptoms and user profiles.
  1. Treatment Plan Generator
     + Users can enter a known condition (e.g., diabetes).
     + System provides a structured treatment plan including:
       - Medications
       - Lifestyle modifications
       - Follow-up testing
     + Guidance is tailored based on patient data (age, gender, health history).
  2. Health Analytics Dashboard
     + Visualizes vital signs (heart rate, blood pressure, glucose) over time.
     + Includes charts, summaries, and symptom frequency breakdowns.
     + Offers AI-generated trend insights and health recommendations.
     + Allows users to update and manage their health metrics.
  3. User Profile Management
     + Create/update user profile with name, age, gender, medical history.
     + Store health data securely for analytics and personalized AI responses.

1. Non-Functional Requirements

These describe how the system performs and operates:

* 1. Usability
     + Intuitive and responsive UI with clear navigation.
     + Accessible across devices (mobile, tablet, desktop).
     + Includes visual aids, tooltips, and simplification features.
  2. Performance
     + AI responses must be returned within 3–5 seconds.
     + Graphs and dashboards should load dynamically and efficiently.
  3. Reliability
     + The system must function reliably during health-related queries.
     + Local fallback for when API is unreachable (e.g., message: “AI unavailable, try again later”).
  4. Security
     + API keys securely handled using .env and python-dotenv.
     + Health data stored temporarily, not retained unless user consents.
     + Data anonymized when used for analytics.
  5. Scalability
     + Backend designed to support multiple concurrent users.
     + Architecture should support expansion to include more features (e.g., wearable integration, voice input).
  6. Compliance
     + Adheres to privacy regulations like HIPAA/GDPR (as applicable).
     + Includes user consent prompt before collecting personal health data.

1. User Requirements Primary Users

* General users seeking medical advice or self-diagnosis.
* Users managing chronic conditions needing daily health tracking.

Needs

* Immediate responses to health queries.
* Easy-to-use dashboard for health metric tracking.
* Personalized and non-technical explanations of health issues. Constraints
* Varying levels of tech proficiency.
* Language barriers (solution should ideally support multilingual interaction).

1. System Requirements Software:

* Programming Language: Python 3.x
* Frameworks: Streamlit, Flask (for backend if needed)
* Libraries: Plotly, Pandas, NumPy, python-dotenv
* AI Integration: IBM Watson ML API (Granite 13B v2)
* Environment: Virtualenv or conda
* Version Control: Git, GitHub Deployment:
* Local Testing: streamlit run app.py
* Hosting: Streamlit Cloud or IBM Cloud
* Configuration: .env for secure API keys Would you like this in:
* A tabular format for reports?
* A Software Requirements Specification (SRS) document?
* A diagram (like use-case or system architecture

## Customer Journey map

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stage | Custom er  Actions | Touchpoin ts | Emotions | Pain Points | Opportunitie s |
|  |  |  |  |  |  |
|  | - |  |  | Uncertainty about  trusting AI Not sure if the app is reliable | Clear communicat ion of  benefits and AI model credibility (e.g., IBM  Granite) |
|  | Searche |  |  |
|  | s online |  |  |
|  | for |  |  |
|  | health | Website, |  |
| 1.  Awarenes s | help  - Hears about | blog, app store,  social | Curious, Hopeful |
|  | HealthAI | posts |  |
|  | via |  |  |
|  | friend or |  |  |
|  | social |  |  |
|  | media |  |  |
|  | - Visits |  |  |  |  |
|  | website |  |  |  | Simple |
|  | or app | Homepag |  |  | UI/UX, |
| 2. | - Reads | e, Signup | Intereste | Technical | explain |
| Onboardi | about | page, | d, Slightly | terms, | features in |
| ng | features | About | skeptical | Long forms | layman |
|  | - | section |  |  | terms, guest |
|  | Creates |  |  |  | demo option |
|  | a profile |  |  |  |  |
| 3. Query Initiation | - Uses Patient Chat or Disease Predictio  n | Chat UI, Symptom form | Hopeful, Anxious | Fear of wrong  diagnosis, slow response | Friendly UI, show AI is “thinking,” include  disclaimers |
|  | - Enters |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stage | Custom er  Actions | Touchpoin ts | Emotions | Pain Points | Opportunitie s |
|  |  |  |  |  |  |
|  | sympto m |  |  |  |  |
|  | Receives | Chat thread,  prediction result, treatment plan section |  |  |  |
|  | AI- |  |  |  |
|  | generate |  |  |  |
|  | d |  |  |  |
|  | respons |  |  |  |
| 4. AI  Interactio n | e  - Asks follow- up  question  s | Reassure d, Informed | Medical language too complex | Simplified explanations  , “What this means for you” section |
|  | - |  |  |  |
|  | Reviews |  |  |  |
|  | insights |  |  |  |
|  | or next |  |  |  |
|  | steps |  |  |  |
|  | - Enters |  |  |  |  |
|  | health |  |  |  |  |
|  | data |  |  |  |  |
|  | - Views | Health |  | Charts may | Use color |
| 5. Health Tracking | analytic s  dashboa | dashboard  , charts, metrics | Motivated  , In control | confuse non-  technical | codes,  tooltips, daily/weekly |
|  | rd | summary |  | users | tips |
|  | - Reads |  |  |  |  |
|  | AI trend |  |  |  |  |
|  | insights |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stage | Custom er  Actions | Touchpoin ts | Emotions | Pain Points | Opportunitie s |
|  |  |  |  |  |  |
|  | - |  |  |  |  |
|  | Decides |  |  |  |  |
|  | on |  |  |  |  |
|  | actions: |  |  |  |  |
| 6.  Decision Making | take medicin e, call  doctor, etc.  - | Export section, follow-up prompts | Empower ed,  Confident | Doubts about  accuracy, no real  doctor | Reinforce “information al use,” option to email to real  doctor |
|  | Downloa |  |  |  |  |
|  | ds |  |  |  |  |
|  | treatme |  |  |  |  |
|  | nt plan |  |  |  |  |
|  | - |  |  |  |  |
|  | Returns |  |  |  |  |
|  | to log |  |  |  |  |
|  | new |  |  |  |  |
|  | sympto |  |  |  | Add |
| 7. Re-  engagem ent | ms  - Tracks  progress over | Notificatio ns,  Emails,  App alerts | Consiste nt,  Curious | Forget to use app,  low follow-  up | gamification (badges, streaks),  reminder |
|  | time |  |  |  | notifications |
|  | - |  |  |  |  |
|  | Receives |  |  |  |  |
|  | reminde |  |  |  |  |
|  | rs |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stage | Custom er  Actions | Touchpoin ts | Emotions | Pain Points | Opportunitie s |
|  |  |  |  |  |  |
| 8.  Feedback C Sharing | * Rates experien ce * Shares app with friends or family | Feedback form, share button | Grateful, Loyal | Feedback not  acknowled ged | Thank-you message, showcase “Your input  improves AI” |

Key Takeaways:

* + - Emotional Peak: When users receive a helpful and clearly explained response.
    - Emotional Dip: When users fear the AI might be wrong or when language feels too technical.
    - Critical Moments
    - First AI answer (build trust
    - First chart or treatment plan display (clarity is key) Would you like this journey map in:
    - Diagram form (PDF/Image)?
    - PowerPoint Slide?
    - Included in a Product Design Document
  1. Solution Requirement

Functional Solution Requirements

These define the core capabilities your solution must offer:

1. Patient Chat
   * + Natural language interface to receive and respond to health- related questions.
     + Integration with IBM Granite 13B to generate accurate, empathetic responses.
     + Conversation history tracking per session.
     + Clear AI disclaimer: “This is not a substitute for medical advice.”
2. Disease Prediction
   * + User interface for multi-symptom entry.
     + AI processing that returns:
       - Probable diagnoses
       - Likelihood/confidence scores
       - Recommended next steps
     + Visual indication (e.g., risk severity: low, moderate, high).
     + Input form for diagnosed condition and patient profile.
     + AI-generated treatment plan including:
       - Medications
       - Lifestyle changes
       - Diet and follow-ups

Option to export plan as PDF or share with healthcare provider.

D. Health Analytics

* + - Visual dashboards for key health metrics:
      * Heart rate, Blood pressure, Glucose levels
    - AI-generated health insights and trend warnings.
    - Interactive charts using Plotly with color-coded status.

E. User Profile C Data Management

* + - Form to capture personal data (age, gender, history, etc.).
    - Session state for storing input data securely.
    - Ability to update profile and metrics dynamically.

1. Non-Functional Requirements
   1. Performance
      * Response time for AI queries: under 5 seconds.
      * Smooth and responsive UI for all features.
   2. Usability
      * Mobile and desktop compatibility (responsive design).
      * Simple language options and minimal form complexity.
      * “Explain in simple terms” toggle for all results.
   3. Security
      * Use .env to store and load IBM Watson credentials securely.
      * Ensure patient data is processed temporarily and not stored without consent.
      * Anonymize health metrics before any analytics are shown or stored.
   4. Reliability C Availability
      * System should continue functioning even if AI service is down (basic error messages or fallback).
      * All modules should be tested for edge cases (e.g., invalid input, empty forms).
   5. Scalability
      * Modular architecture that can support:
        + Multiple concurrent users
        + Future integration with wearables or doctor access portals
2. Technical Requirements
   1. Technology Stack
      * Frontend: Streamlit (for UI)
      * Backend Logic: Python
      * Visualization: Plotly
      * AI Integration: IBM Watson Machine Learning (Granite-13B- instruct-v2)
      * Environment Config: python-dotenv for .env loading
   2. Libraries/Tools
      * streamlit, pandas, numpy, plotly, ibm-watson-machine- learning, python-dotenv
   3. Project Structure bash

CopyEdit HealthAI/

│

├── app.py

├── utils/

│ └── prompt\_builder.py, model\_connector.py

├── components/

│ └── chat.py, prediction.py, analytics.py

├── data/

│ └── sample\_health\_data.csv

├── .env

├── requirements.txt

└── README.md

* 1. Deployment Requirements
     + Hosted on Streamlit Cloud (or IBM Cloud if preferred)
     + .env configured securely
     + App runs using: streamlit run app.py

## AI Prompt Strategy Requirements

* Design prompts to extract accurate, medically relevant, and empathetic responses.
* Use structured templates for:
  + Chat: “You are a virtual health assistant...”
  + Diagnosis: “Based on the symptoms: {symptoms}...”
  + Treatment: “Generate a treatment plan for {condition} in a format like…”

Let me know if you’d like this formatted into a:

* Software Requirements Specification (SRS) document
* Checklist format for team implementation
* Technical presentation slide deck for stakeholders I'm happy to prepare it in your preferred format.

## 3.3 Data Flow Diagram

Entities

* User – Inputs symptoms, questions, health data, and retrievesinsights.
* HealthAI System – Main processing unit (frontend + backend).
* IBM Watson ML (Granite Model) – External AI service.
* Visualization Engine – Handles health metric graphs and analytics.
* Data Store (Temporary) – Manages session data and user profile temporarily.
* Processes C Data Flow Description

|  |  |  |
| --- | --- | --- |
| Process No. | Process Name | Description |
| P1 | Patient Chat Handler | Accepts health-related user queries, sends them to AI, and displays answers. |
| P2 | Disease Prediction Engine | Takes user symptoms, calls AI model, and returns likely conditions. |
| P3 | Treatment Plan Generator | Takes condition + user data, generates tailored  recommendations. |
| P4 | Health Analytics Processor | Fetches user data, processes trends, displays graphs + AI  insights. |
| P5 | Session C Profile Manager | Stores user session data temporarily for  analytics/personalization. |

Data Stores

* D1: User Profile C Session Data – Temporary storage (in- memory or cache)
* D2: Sample Medical Knowledge Base – Reference info for fallback or UI content
* External Systems
* IBM Watson Granite-13B (via API) – Handles language model responses
* Plotly Engine – For data visualization of metrics Visual Description of the DFD

sql

CopyEdit

+ + + +

| User | ------> | HealthAI (Streamlit) | < +

+ + + + |

| | |

| v |

| +------------+ |

| | P1: Chat | ---Query > [IBM Watson AI]|

| +------------+ <---Response--+ |

| | |

| v |

| +------------+ |

| | P2: Predict| ---Symptoms > [Granite AI]|

| +------------+ <---Diagnoses--+ |

| | |

|  |  |  |  |
| --- | --- | --- | --- |
| | | v | | | |
| | | +------------+ | | | |
| | | | P3: Plan | ---Condition---> [Granite AI]| | | |
| | | +------------+ <---Treatment--+ | | | |
| | | | | | |  |
| | | v | | |  |
| | | + + |  | | |
| | | P4: Analytics + | | | | |
| | | + + |  | | | |
| | | | | | | |  |
| | | v | | | |  |
| | | + + |  | | | |
| | | | Plotly UI | <--- Data ---- [D1] <---+ | | | |
| | | +------------+ | | | |
| | | ^ | | | |
| | | | | | | |

+----------------------> P5: Session/Profile Management |

+ +

Key Notes:

All AI interactions are routed through backend functions using API calls to IBM Watson (Granite).

* Data is not stored permanently to protect privacy — it resides in session state.
* Plotly handles dynamic chart rendering from health data (e.g., CSV or user input).
* The system must handle failures gracefully (e.g., API errors, invalid input).

Would you like:

* This DFD designed as a diagram (PDF/image)?
* A Level 0 DFD (high-level overview only)?
* A UML diagram (Component or Activity Diagram)?

## 3.4 Technology Stack

1. Frontend

|  |  |
| --- | --- |
| Technology | Purpose |
| Streamlit | Rapid UI development for data-centric web apps (used for the entire interface) |
| Plotly | Interactive and responsive data visualizations (health charts and dashboards) |
| HTML/CSS (within Streamlit) | Custom styling and layout formatting |
| Streamlit  Components | Used for sidebar, tabs, forms, chat UI, and interactive inputs |

|  |  |
| --- | --- |
| Python 3.x | Main programming language for backend and AI integration |
| Flask (optional) | Can be used to manage backend logic if separated from Streamlit |
| Pandas / NumPy | Data handling, transformation, and metrics calculations |
| IBM Watson Machine Learning SDK | Interface with IBM’s Granite 13B Instruct v2 model |
| python-dotenv | Load secure environment variables (.env file for API keys) |

1. AI / ML Integration

|  |  |
| --- | --- |
| Component | Purpose |
| IBM Granite-13B-Instruct- v2 (Watsonx.ai) | Generative AI model for medical QCA, treatment planning, and predictions |
| Prompt Engineering | Custom prompts for different modules (chat, diagnosis, treatment) |
| IBM Watson Machine Learning API | Facilitates AI model access via secure cloud endpoints |

## Security s Configuration

|  |  |
| --- | --- |
| Tool | Purpose |
| .env + python-dotenv | Secure API key and project ID management |
| Session State (Streamlit) | Temporary data storage for user interactions |
| Input Validation | Ensure clean and safe data processing before API call |

1. Deployment

|  |  |
| --- | --- |
| Platform | Purpose |
| Streamlit Cloud | Quick and free deployment for Streamlit apps |
| IBM Cloud (optional) | Hosting AI components or full app for enterprise scalability |
| GitHub | Version control, collaboration, and CI/CD (if configured) |

1. Dependencies

Defined in requirements.txt: plaintext

CopyEdit streamlit

pandas numpy plotly

ibm-watson-machine-learning python-dotenv

Optional Future Stack Additions

|  |  |
| --- | --- |
| Tool/Tech | Use Case |
| LangChain | Advanced prompt chaining or multi- step AI workflows |
| Whisper API /  SpeechRecognition | Voice input for accessibility |
| Firebase / MongoDB | Persistent storage for user data or long-term analytics |
| Docker | Containerized deployment setup for portability |
| JWT/Auth0 | Secure user authentication for profile- based use |

## 4.PROJECT DESIGN

1. System Architecture Architecture Overview

Csharp CopyEdit [User]

↓

[Streamlit Frontend UI]

↓

[Backend Logic (Python Functions)]

↓

[AI Layer – IBM Granite via Watson Machine Learning API]

↔

[Data Layer – Patient Profiles, Metrics, Session State] Components:

* Frontend (UI): Built with Streamlit
* AI Integration: IBM Granite 13B Instruct v2 via Watsonx API
* Backend Logic: Python modules (diagnosis, treatment, chat, analytics)
* Data Handling: Local session state, optional file-based storage
* Visualization: Plotly for interactive health charts

1. Module Design

|  |  |
| --- | --- |
| Module | Functionality |
| app.py | Main entry point, handles layout, routing, and rendering |
| chat.py | Patient Chat UI, prompt building, AI response handling |
| diagnosis.py | Symptom input form, API call to Granite for prediction |
| treatment.py | User enters condition, system generates treatment plan |
| analytics.py | Visualizes health data and provides AI-generated insights |
| utils.py | Prompt templates, IBM API setup, session management |

1. AI Prompt Design Prompt Templates:

* Patient Chat Prompt: css

CopyEdit

You are a helpful healthcare assistant. Answer the following medical question with empathy and clarity:

Question: {user\_input}

* Disease Prediction Prompt: pgsql

CopyEdit

A user reports the following symptoms: {symptoms}. Provide

possible diagnoses with confidence levels and recommendations.

* Treatment Plan Prompt: pgsql

CopyEdit

Provide a treatment plan for the condition {condition} based on a 32- year-old male with no chronic history. Include medications, lifestyle tips, and follow-up steps.

1. Data Handling Design Temporary Data

* Stored using Streamlit’s st.session\_state
* Includes chat history, profile data, and symptom inputs Optional Persistent Data
* Simulated using sample\_health\_data.csv or file-based JSON
* Could scale to Firebase/MongoDB in future Data Flow Example (Disease Prediction):

1. User inputs symptoms
2. Backend builds structured prompt
3. AI model returns diagnoses + confidence
4. Output formatted and displayed on UI
5. UI/UX Design Layout
   * Sidebar: Patient Profile, Navigation Tabs
   * Main Tabs:
     + Chat <C
     + Disease Prediction C\*˛
     + Treatment Plan • ¸ ‘ )
     + Health Analytics ç# ¡/

Styling

* + Streamlit native + custom CSS
  + Responsive layout for mobile/tablet
  + Tooltips and “explain in simple terms” options Accessibility
  + Easy form inputs, large buttons
  + AI output simplified for general users
  + Clear visual feedback (status, loading indicators)

1. Visualization Design (Plotly)

|  |  |  |
| --- | --- | --- |
| Metric | Chart Type | Details |
| Heart Rate | Line Chart | Shows trend over time |

|  |  |  |
| --- | --- | --- |
| Metric | Chart Type | Details |
| Blood Pressure | Dual Line Chart | Systolic C Diastolic |
| Blood  Glucose | Line Chart + Range Band | Shows if values are out of range |
| Symptoms | Pie Chart | Frequency distribution |

1. Scalability Considerations

|  |  |  |
| --- | --- | --- |
| Area | Current | Future Option |
| Data Storage | Session-based | Firebase / MongoDB |
| AI Model | IBM Granite API | Fine-tuned LLM hosted locally or on cloud |
| User Auth | Basic / optional | OAuth2 / Auth0 / JWT |
| Deployment | Streamlit Cloud | Dockerized on AWS/GCP/IBM Cloud |

1. Security Design
   * .env for IBM API key and project ID
   * Use python-dotenv to securely load credentials
   * Data anonymized before visualization
   * Optional user consent checkbox for storing health data Testing Strategy

|  |  |
| --- | --- |
| Type | What to Test |
| Unit Testing | AI prompt formatting, API response parsing |
| UI Testing | Form validation, chat input/output |
| Integration Testing | AI + Backend + Frontend flow |
| Performance Testing | API response time, chart rendering delay |

* 1. Problem–Solution Fit The Problem
     1. Lack of accessible health guidance:
        + Millions of people have limited or no access to qualified doctors or timely health advice.
        + Minor symptoms are often ignored or misinterpreted, leading to serious complications.
     2. Health information is complex:
        + Online medical information is often confusing, misleading, or too technical.
        + Patients struggle to understand what actions to take based on symptoms or test results.
     3. Overloaded healthcare systems:
        + Doctors and clinics face high patient loads, reducing the time spent per individual.
        + Preventive care and early interventions are underutilized. The Solution – HealthAI

HealthAI is an intelligent, AI-powered virtual assistant that helps users:

* Understand their symptoms
* Get AI-generated diagnosis suggestions
* Access basic treatment and lifestyle recommendations
* Visualize personal health metrics for early intervention How It Solves the Problem

|  |  |  |
| --- | --- | --- |
| Problem | HealthAI Feature | Value |
| No access to doctors | Chat with AI trained on medical knowledge | 24/7 availability |
| Confusing online info | Simplified answers and explanations | Understandable by non-experts |
| Symptom confusion | Symptom-based disease prediction | Helps users take informed action |
| No preventive tracking | Health data visualization | Detects trends C risks early |

|  |  |  |
| --- | --- | --- |
| Problem | HealthAI Feature | Value |
| Overburdened clinics | Basic triage C self-care suggestions | Reduces unnecessary visits |

Fit Summary

|  |  |
| --- | --- |
| Aspect | Detail |
| Target User | Anyone needing basic health help, especially in remote or underserved areas |
| User Need | Simple, private, instant medical guidance |
| Solution Value | Empowers people to make early health decisions, reduces anxiety, encourages better habits |

## Proposed Solution

Overview

HealthAI is an AI-powered virtual healthcare assistant designed to bridge the gap between individuals and accessible medical

guidance. It leverages IBM Watson’s Granite-13B-instruct-v2 generative AI model to offer personalized, intelligent, and real-time support for basic healthcare needs.

Core Capabilities

* + 1. Conversational Health Assistant
       - Natural-language interaction with patients using an empathetic chatbot.
       - Responds to health-related questions with simplified explanations.
    2. Symptom-based Disease Prediction
       - Users input symptoms through a form.
       - AI analyzes and predicts possible conditions with confidence levels.
    3. Personalized Treatment Recommendations
       - Based on a diagnosed or user-entered condition.
       - Provides home remedies, medications, lifestyle changes, and when to consult a doctor.
    4. Health Data Analytics
       - Users can input personal health metrics (e.g., blood pressure, sugar levels).
       - System visualizes trends and highlights risks using interactive charts.

How It Works

1. Frontend Interface

Built using Streamlit, providing an intuitive, lightweight UI accessible from any device.

1. Backend Intelligence
   * Powered by Python and IBM’s Watson Machine Learning API.
   * Uses dynamic prompt engineering to guide AI outputs based on user input.
2. Data Handling
   * Temporary user data managed via session\_state in Streamlit.
   * Optional simulated patient history for testing analytics and trends.
3. Visualization
   * Plotly graphs for dynamic visualization of health indicators and AI feedback.

Why This Solution Works

|  |  |  |
| --- | --- | --- |
| Need | HealthAI Feature | Outcome |
| Accessible health info | AI chatbot + symptom checker | 24/7 guidance without clinic visits |
| Simplicity | Simplified medical explanations | Better user understanding |
| Preventive care | Health data analytics + alerts | Early intervention and tracking |
| Personalized response | Prompt-based AI logic | Tailored advice for each user |

Benefits

* Reduces dependence on in-person consultation for minor issues
* Helps users self-educate and track their own health
* Makes basic health advice accessible in rural/remote areas
* Scalable and can be integrated with real medical records in the future

## Solution Architecture

1. Layered Architecture Overview pgsql

CopyEdit

+ +

| Presentation UI | ← Streamlit (User Interface)

+ +

↓

+ +

| Application Logic | ← Python Modules (Chat, Diagnosis, Treatment, Analytics)

+ +

↓

+ +

| AI Processing Layer | ← IBM Granite 13B via Watson Machine Learning API

+ +

↓

+ +

| Data Management | ← Session State, CSV/JSON, Plotly, Optional Firebase

+ +

1. Component-wise Breakdown Frontend (Streamlit UI)
   * Patient-friendly interface
   * Sidebar for profile input
   * Navigation tabs:
     + Disease Prediction
     + Chatbot
     + Treatment Plan
     + Health Analytics

Application Logic (Python Backend)

* + chat.py – Generates conversational prompts, handles chatbot responses
  + diagnosis.py – Takes symptoms input, crafts prompts for AI diagnosis
  + treatment.py – Accepts disease name, returns treatment plans
  + analytics.py – Visualizes health metrics and trends
  + utils.py – API connection, prompt templates, session control AI Processing (Watsonx / IBM Granite 13B)
  + Prompts sent to IBM Granite model via Watson Machine Learning API
  + Receives structured responses (diagnosis, treatment, advice) Data Management Layer
  + Session State: Temporary user data during app use
  + CSV/JSON: Simulated patient health data (for analytics)
  + Plotly: Visualization for heart rate, BP, sugar, etc.
  + (Optional) Firebase or Cloud DB for persistent data in future versions

1. API C Workflow Flowchart plaintext

CopyEdit

[User Input (UI)]

↓

[Backend Python Script] → [Prompt Generation]

↓ ↓

[Call to IBM Watsonx API] ← [Prompt + User Context]

↓

[AI Response]

↓

[Formatted Output (Text or Chart)]

↓

[Display on Streamlit UI]

1. Security and Privacy
   * .env used for securing IBM API credentials
   * AI prompts avoid storing personally identifiable health data
   * Add data anonymization and consent mechanism for real deployment
2. Scalability Path (Future Ready)

|  |  |  |
| --- | --- | --- |
| Component | Current | Scalable To |
| AI Model | IBM Granite 13B via API | Fine-tuned on private healthcare data |
| Data Store | Session + CSV | Firebase / MongoDB / PostgreSQL |
| Hosting | Streamlit Cloud | Docker + AWS/GCP/IBM Cloud |
| Auth | No login | OAuth2 / Auth0 / JWT |

## PROJECT PLANNING s SCHEDULING

1. Project Timeline Overview

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Name** | **Duration** | **Key Deliverables** |
| 1 | Planning C Research | Week 1 | Problem statement, requirements analysis, empathy map |
| 2 | Design C Architecture | Week 2 | Solution design, UI  wireframes, data flow diagram |
| 3 | Development Phase 1 | Week 3 | Chatbot module, symptom checker |
| 4 | Development Phase 2 | Week 4 | Treatment plan, analytics module |
| 5 | Integration C Testing | Week 5 | Full system integration, testing, bug fixing |
| 6 | Deployment C Presentation | Week 6 | Streamlit deployment, final report, presentation slides |

## Task Breakdown by Week

**Week 1: Planning s Requirement Gathering**

* + Define the problem
  + Identify target users and pain points
  + Create empathy map
  + Perform brainstorming session
  + Finalize solution scope

## Week 2: Design

* + Draft UI flow (chat, prediction, analytics)
  + Prepare Data Flow Diagram (DFD)
  + Create Customer Journey Map
  + Finalize Tech Stack C Solution Architecture

## Week 3: Development – Phase 1

* + Set up Streamlit interface
  + Build AI Chatbot using IBM Granite model
  + Add prompt templates for QCA
  + Basic UI testing and prompt validation

## Week 4: Development – Phase 2

* + Implement Symptom Checker → Disease prediction
  + Develop Treatment Plan generator
  + Build Analytics Dashboard (Plotly)
  + Create reusable components (functions, API wrappers)

## Week 5: Integration s Testing

* + Integrate all modules
  + Validate output consistency
  + Conduct unit and functional testing
  + Collect user feedback (if applicable)

## Week 6: Deployment s Final Review

* + Host on Streamlit Cloud
  + Final bug fixes and polish
  + Prepare project report and documentation
  + Final presentation rehearsal

## Milestones

|  |  |
| --- | --- |
| **Milestone** | **Expected Completion** |
| Problem C Requirements Finalized | End of Week 1 |
| Design Documents Ready | End of Week 2 |
| Chatbot Working | Mid Week 3 |
| Full Module Integration | End of Week 5 |
| Final Report C Demo Ready | End of Week 6 |

**Deliverables**

* + Working AI Healthcare Assistant (Streamlit app)
  + Technical Documentation (SRS + Architecture + Planning)
  + Presentation Slides (PPT)
  + Final Report (PDF/DOC)
  + GitHub Repository (optional)

## Project Planning

1. **Project Title**

HealthAI – AI-Powered Virtual Healthcare Assistant

## Objectives

* + To provide users with instant, AI-based health guidance.
  + To help identify potential diseases based on symptoms.
  + To offer basic treatment suggestions and promote preventive care through data visualization.
  + To reduce burden on clinics and empower users, especially in underserved areas.

## Target Users

* + General public (especially in rural or remote areas).
  + Users with minor health symptoms seeking early advice.
  + Individuals managing chronic conditions who need regular tracking.

## Project Scope In Scope:

* + AI chatbot for general health queries.
  + Symptom-to-disease prediction.
  + Treatment suggestions for common conditions.
  + Visualization of personal health data.

## Out of Scope:

* + Real-time doctor consultations.
  + Emergency medical advice.
  + Integration with official medical records or hospital systems (in current version).

## Project Duration

1. **weeks**, structured in weekly development phases.

## Project Phases

|  |  |
| --- | --- |
| **Phase** | **Description** |
| 1. Initiation | Identify the problem, define the objective, analyze stakeholders. |
| 2. Planning | Define features, scope, timeline, and tools. Prepare empathy map, requirement analysis, and customer journey map. |
| 3. Design | Create UI wireframes, data flow diagrams, and solution architecture. |
| 4. Development | Build individual modules: chatbot, diagnosis, treatment, and analytics. |
| 5. Testing | Integrate modules, test functionality, and fix bugs. |
| 6. Deployment C Review | Deploy the application and finalize  documentation for submission/presentation. |

1. **Tools s Technologies**
   * **Frontend**: Streamlit (Python-based web UI)
   * **AI Model**: IBM Watsonx / Granite-13B-instruct-v2
   * **Backend**: Python
   * **Data Handling**: CSV/JSON, Session State
   * **Visualization**: Plotly
   * **Version Control**: GitHub (optional)

## Roles s Responsibilities

|  |  |
| --- | --- |
| **Role** | **Responsibility** |
| Project Lead | Overall planning, coordination |
| AI Developer | Prompt engineering, IBM API integration |
| UI/UX Developer | Streamlit interface |
| Tester | Functionality and UX testing |
| Documentation Lead | Preparing reports and presentation |

**G. Success Criteria**

* Fully functional AI healthcare assistant with:
  + Chatbot conversation
  + Symptom analysis
  + Treatment suggestion
  + Health metrics analytics
* Deployed on Streamlit or similar platform
* Report and presentation completed on time

**6**.**FUNCTIONAL AND PERFORMANCE TESTING:**

**Project Planning – HealthAI**

1. **Objective & Vision**
   * **Goal:** To create an AI-powered healthcare platform that delivers intelligent responses for patient queries, disease predictions, treatment recommendations, and health analytics.
   * **Vision:** Enhance healthcare accessibility using IBM Watson Machine Learning and Streamlit for personalized, reliable, and efficient virtual medical assistance.

## Scope of the Project

* + Develop a web-based platform integrating:
    - **Patient Chat**
    - **Disease Prediction**
    - **Treatment Plan Generation**
    - **Health Analytics**
  + Use **IBM Granite-13B Instruct v2** via **Watson ML API**
  + Focus on data visualization and interactive UI using **Streamlit + Plotly**
  + Ensure secure API management and ethical data handling practices

## Deliverables

* + Fully functional HealthAI Streamlit web app
  + Feature modules:
    - Chat interface
    - Prediction system
    - Treatment planner
    - Analytics dashboard
  + .env setup for secure key management
  + Complete documentation and deployment-ready code

## Development Methodology

* + **Agile approach** with iterative development and review cycles
  + Sprints divided per activity:
    - Sprint 1: Environment setup + Architecture
    - Sprint 2: Core features (Chat + Prediction)
    - Sprint 3: Treatment Plans + Analytics
    - Sprint 4: UI Enhancements + Testing
    - Sprint 5: Deployment + Final QA

## Tools & Technologies

* + **Languages:** Python
  + **Frameworks:** Streamlit, Flask
  + **Visualization:** Plotly, Matplotlib
  + **AI Model:** IBM Granite 13B Instruct v2
  + **APIs & Services:** IBM Watson Machine Learning
  + **Version Control:** Git + GitHub
  + **Environment Management:** Python venv, dotenv
  + **Hosting:** Streamlit Cloud

## Team Roles (If Applicable)

* + **AI/ML Developer:** Model integration, prompt engineering
  + **Frontend Developer:** Streamlit UI and chart design
  + **Backend Developer:** Data handling, API management
  + **QA Tester:** Feature validation and performance testing
  + **Project Manager:** Task coordination and progress tracking

## Risk Assessment

|  |  |  |
| --- | --- | --- |
| **Risk** | **Impact** | **Mitigation** |
| API Quota Limits | High | Use efficient prompting, monitor API usage |
| Inaccurate AI Outputs | Medium | Include disclaimers, validate results |
| UI Breaks on Deployment | Medium | Responsive testing and modular UI |
| Data Security | High | Use .env, never expose patient data |
| Delays in Integration | Medium | Agile sprint planning, milestone tracking |

1. **Communication Plan**
   * Use GitHub Issues for tracking bugs/features
   * Weekly progress meetings (or sprint retrospectives)
   * Documentation in README and Wiki for team onboarding

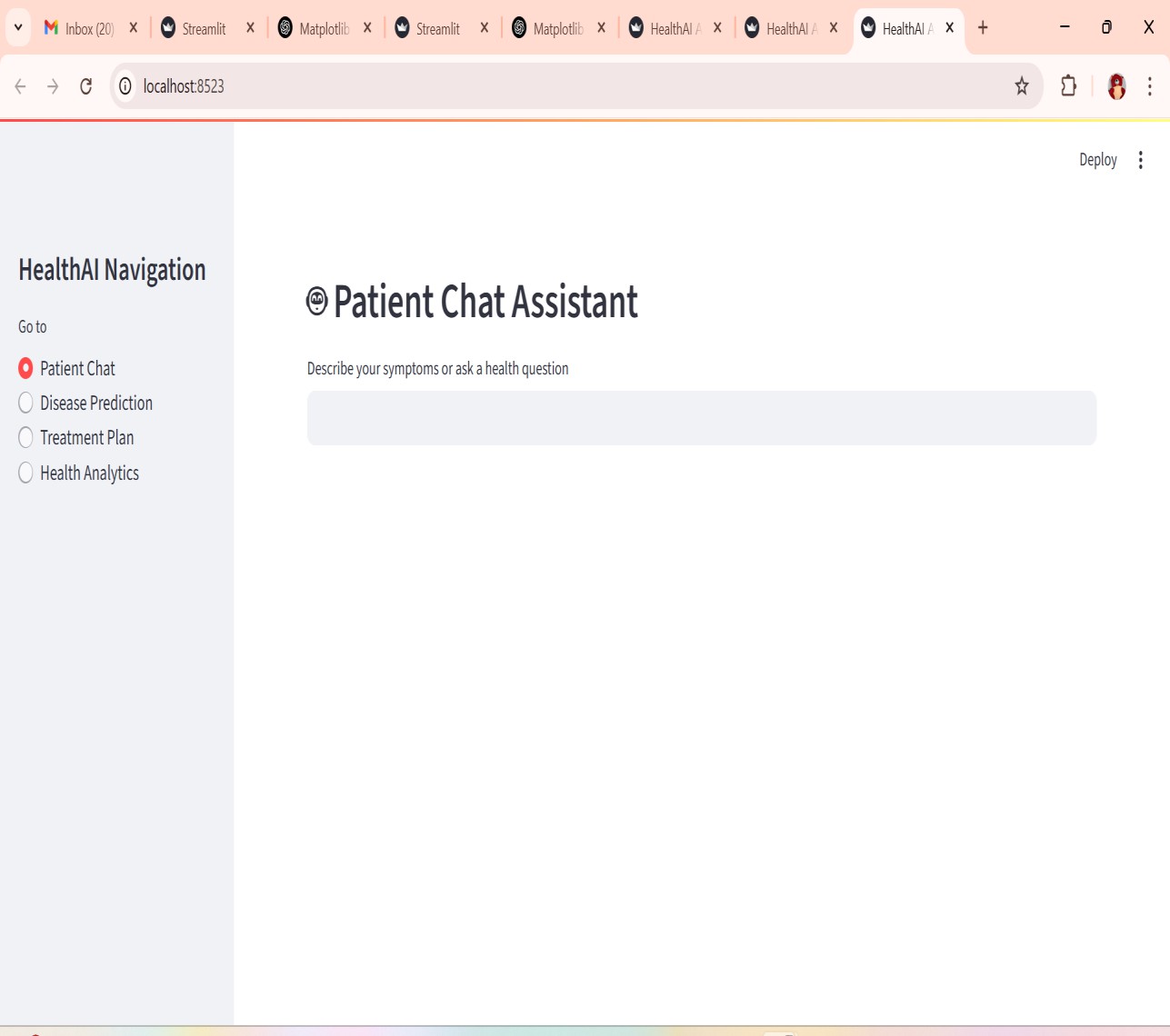
## Timeline Overview

* + Total duration: **5–6 weeks**
    - Week 1: Model setup + architecture
    - Week 2–3: Core feature development
    - Week 4: Visualization + profile utilities
    - Week 5: App.py integration + testing
    - Week 6: Deployment + final review

## Success Criteria

* + Smooth AI responses through IBM Watson API
  + Real-time interactivity with disease/treatment modules
  + Responsive UI across devices
  + Secured environment variable handling
  + All pages tested, deployed, and operational
* **7. RESULTS :**

**Output Screenshots :**

****

1. **ADVANTAGES & DISADVANTAGES:**

**Advantages**

* 1. **AI-Powered Medical Insights**

Utilizes IBM Granite 13B Instruct v2 for generating accurate and context-aware medical guidance, improving decision-making for users.

## 24/7 Availability

Provides users with round-the-clock healthcare assistance without the need for a live medical professional.

## Personalized Treatment Recommendations

Tailors treatment plans based on individual input and health metrics, increasing relevance and user trust.

## Early Detection & Prediction

The disease prediction module allows users to analyze symptoms early, enabling faster medical intervention.

## Interactive Data Visualization

Integrates dynamic visual tools (e.g., line charts, pie charts) to help users understand their health trends visually.

## User-Friendly Interface

Built with Streamlit for a clean, responsive, and intuitive layout, even for users with minimal technical knowledge**.**

## Scalable and Modular Architecture

Easily extendable to support new features, AI models, or third- party healthcare APIs.

## Secure API Key & Data Management

Uses environment variables and good development practices for protecting sensitive credentials and user data.

## Cost-Effective Healthcare Solution

Reduces the need for frequent in-person consultations, especially for common queries and routine monitoring**.**

## Cross-Platform Access

Deployable on the cloud (Streamlit Cloud), accessible from any device with a browser.

## Disadvantages :

1. **Limited Real-Time Diagnosis Capability**

Cannot replace professional medical diagnosis or physical examination—limited to informational support.

## Dependence on Prompt Quality

AI-generated outputs depend heavily on prompt structure; poorly phrased input may lead to irrelevant or incomplete results.

## No Integration with Live Health Devices

The current version doesn’t support real-time data ingestion from wearable or IoT medical devices.

## AI Output May Lack Clinical Validation

Recommendations are generated based on language modeling, not verified by licensed medical professionals.

## API Limitations

IBM Watson ML API usage may be restricted by usage limits, requiring monitoring or paid plans for scaling.

## No Offline Support

The platform requires an internet connection to access IBM Watson APIs—useless in ofline environments.

## Data Privacy Risk (If Deployed Improperly)

Without strict data handling protocols, user inputs might pose a privacy risk in production environments.

## Lack of Multilingual Support (Current Version)

The initial build is likely to support only English, limiting accessibility for non-English speakers.

# CONCLUSION :

* + HealthAI showcases how AI and machine learning can revolutionize digital healthcare by offering intelligent, personalized, and accessible medical support.
  + Powered by IBM’s Granite-13B Instruct v2 and built with

Streamlit, the platform enables users to engage with features like Patient Chat, Disease Prediction, Treatment Plans, and

Health Analytics through an intuitive interface.

* + While not a replacement for professional care, HealthAI

provides valuable preliminary insights and supports informed decision-making, especially in underserved areas.

* + Its scalable architecture allows for future upgrades like real- time device integration and telemedicine support, making it a strong foundation for next-generation AI healthcare solutions.

1. **FUTURE SCOPE :**

The HealthAI project lays a strong foundation for AI-assisted healthcare, and its architecture is designed to support a wide

range of future enhancements. Potential areas of development include:

## Real-Time Health Device Integration

* + - Connect wearable devices and IoT sensors to stream real-time health data (e.g., heart rate, glucose, blood pressure).
    - Enable continuous monitoring and alert systems for abnormal readings.

## Multilingual Support

* + - Expand AI responses to support regional and international languages.
    - Improve accessibility for non-English-speaking users in rural and diverse populations.

## Clinical Data Synchronization

* + - Integrate electronic health records (EHR) to personalize

predictions and treatment plans using historical patient data.

* + - Support integration with hospital databases for seamless information sharing.

## Telemedicine Integration

* + - Allow users to connect with certified doctors through in-app video/audio consultations.
    - Enable appointment scheduling, prescription management, and real-time doctor feedback.

## Enhanced AI Accuracy with Medical Datasets

* + - Train and fine-tune models on validated medical datasets to improve diagnosis accuracy and treatment recommendation reliability.
    - Incorporate AI feedback loops for continuous model learning and refinement.

## Advanced Data Privacy & Security

* + - Implement end-to-end encryption and HIPAA-compliant data management systems.
    - Use blockchain for immutable medical record tracking and access control.

## Predictive Health Risk Scoring

* + - Develop algorithms to forecast chronic disease risks based on user habits, family history, and lifestyle data.
    - Provide preventive care suggestions and reminders.

## Mobile App Deployment

* + - Launch a mobile version of HealthAI for Android/iOS to enhance accessibility.
    - Use push notifications for reminders, health alerts, and medication tracking.

# APPENDIX :

**Dataset Link:**

**GitHub & Project Demo Link :**