# HealthAI – Generative AI Healthcare Assistant

**Team ID :**
LTVIP2025TMID32218

**Team Member(s) :**

**Name :** Grandhi Neha Nikitha

**Mail :** gnehanikitha@gmail.com

**Student ID:** 24B05A0507

**College :**

Shri Vishnu Engineering College for Women

**Department :**
Computer Science & Engineering

## 📌 Project Description

**HealthAI** is a Generative AI-powered virtual healthcare assistant that helps users identify diseases based on symptoms and recommends natural home remedies. It integrates **IBM Granite's large language model** with an intuitive **Gradio interface**, allowing users to interact with the model in real time via a simple web UI.

The goal of the project is to **assist users in primary health assessment** by providing instant and intelligent suggestions using modern natural language processing (NLP) techniques.

Here are detailed **Scenarios** (use cases) for your **HealthAI – Generative AI Healthcare Assistant** project, which you can include in your documentation:

## 🧪 Use Case Scenarios

### Scenario 1: Symptom-based Self-Diagnosis

**User:** A college student experiencing fatigue, sore throat, and headache.
**Steps:**

1. Opens the HealthAI web interface.
2. Enters: `fatigue, sore throat, headache` in the **"Symptoms Identifier"** tab.
3. Clicks on "**Predict Disease**."
4. The AI model suggests: **"You might be experiencing symptoms of viral infection or mild flu."**

**Outcome:** The user gets a probable diagnosis and considers seeking medical attention or rest.

## Scenario 2: Natural Remedy Suggestion

**User:** A homemaker diagnosed with **gastritis**, looking for natural treatments.
**Steps:**

1. Switches to the **"Home Remedies"** tab.
2. Enters: `gastritis` in the input box.
3. Clicks "**Get Remedy**."
4. The system responds:
   *"Try drinking warm water with ginger and honey, or aloe vera juice. Avoid spicy foods."*

**Outcome:** The user gets useful natural suggestions that can be tried at home.

### Scenario 3: General Health Query Answering

**User:** A working professional feeling fatigued frequently and curious about natural ways to boost energy.

**Steps:**

1. Switches to the **"Ask a Health Question"** tab.
2. Enters the question: *"How to boost energy levels naturally?"*
3. Clicks **"Get Answer."**
4. The system responds:
   *"To naturally boost energy, maintain a balanced diet, stay hydrated, exercise regularly, reduce stress, and ensure you get 7–9 hours of sleep each night."*

**Outcome:** The user receives AI-powered lifestyle advice to improve energy and wellness.

### Scenario 3: Quick Health Advice in Remote Areas

**User:** A farmer in a rural area with limited access to doctors, experiencing cough and chest pain.
**Steps:**

1. Connects to the AI assistant via mobile or shared system.
2. Enters symptoms: `cough, chest pain` in the interface.
3. Gets a likely disease (e.g., **bronchitis**) and advice to consult a professional urgently.

**Outcome:** Offers **first-level triage advice** to someone in an underserved area.

### Scenario 4: Learning Tool for Medical Students

**User:** A medical student preparing for diagnostic case studies.
**Steps:**

1. Enters various combinations of symptoms into HealthAI.
2. Observes how the model interprets them and what diseases it predicts.

**Outcome:** Assists in **learning and testing** differential diagnosis skills using an AI model.

### Scenario 5: Parents Checking Child's Symptoms

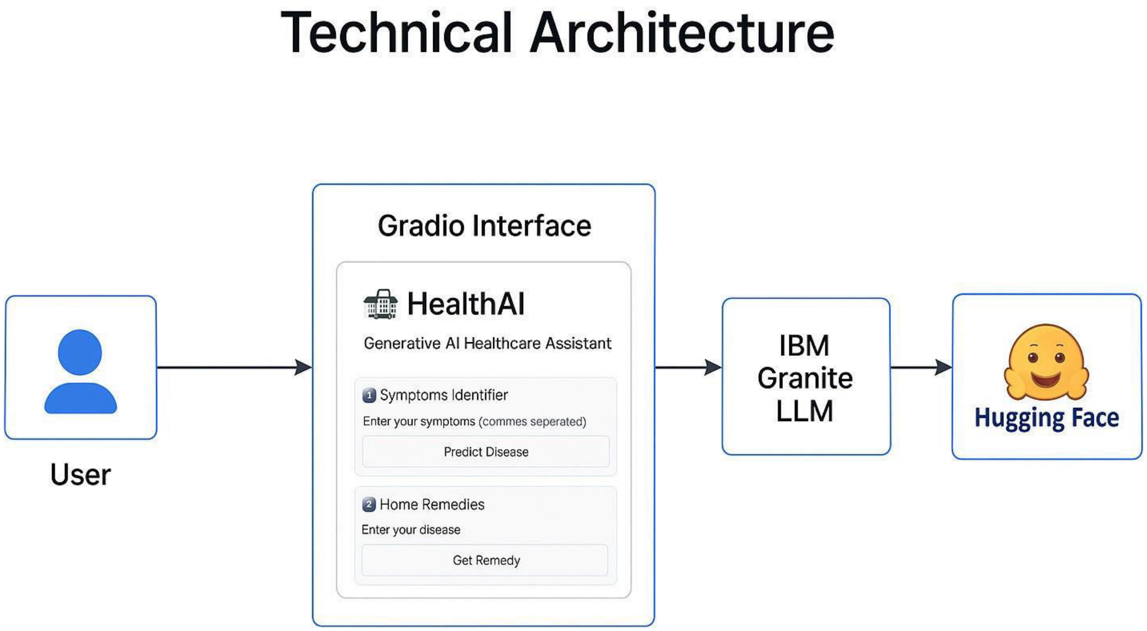
**User:** A parent concerned about their child’s symptoms: `fever, rash, and vomiting`.

**Steps:**

1. Opens HealthAI on their device.
2. Inputs the symptoms.
3. Receives output suggesting a possible condition like **measles** or **food poisoning**, with advice to consult a doctor.

**Outcome:** Immediate AI support gives the parent peace of mind while deciding the next step.

## Technical Architecture:



# 🛠️Project Development Activities

## Activity 1: Model Selection and Architecture

- Researched and selected the IBM Granite 3.3-2B Instruct model from Hugging Face for its advanced generative capabilities.
- Integrated the model using the `transformers` library for seamless NLP functionality.
- Designed the technical architecture outlining the interaction between User → Gradio UI → Model → Hugging Face API.

## Activity 2: Core Functionalities Development

- Developed three primary functions:
  1. `identify_disease()` – identifies the possible disease from user-input symptoms.
  2. `home_remedies()` – recommends natural home remedies for a given disease.
  3. `health_query()` – provides AI-generated answers to general health-related questions.
- Implemented prompt engineering to guide the model's responses effectively.
- Validated responses for multiple test cases to ensure accuracy and relevance.

## Activity 3: App.py Development

- Created a Python script (`app.py`) combining:
  - Model loading
  - Pipeline setup
  - Function definitions
  - Gradio interface structure
- Ensured token authentication and optimized device mapping (`device_map="auto"`) for performance

## Activity 4: Frontend Development

- **Built a user-friendly Gradio-based interface with two tabs:**

    1. **Symptoms Identifier: Input symptoms, view predicted disease.**

    2. **Home Remedies: Input disease, view remedy.**

    3. **Ask a Health Question : Input general health-related questions, receive AI-generated answers.**

- **Used markdown and labels for clarity and enhanced UX.**

- **Added interactive buttons that trigger backend functions on click.**

## Activity 5: Deployment

- **Hosted the app on Google Colab and generated a Gradio public shareable link for users.**

- **Enabled GPU support and handled model checkpoint loading.**

- **Verified app functionality across devices (desktop and mobile) and ensured accessibility.**

- **Addressed runtime issues related to token loading and system memory.**

# Milestone 1: Model Selection and Architecture

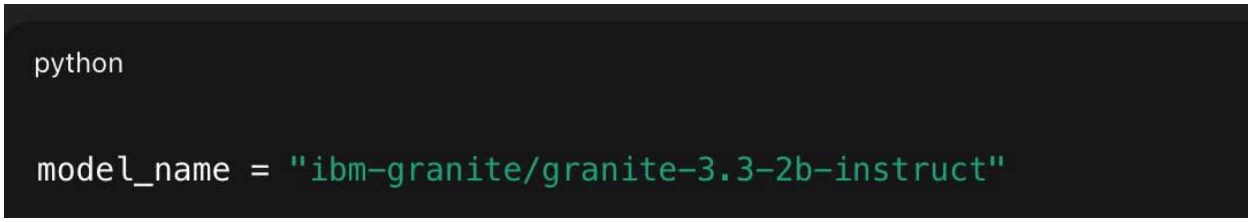
## Objective:

**To select and integrate a capable pre-trained language model for the HealthAI system and define the overall technical flow for interaction between components.**

### Tasks & Implementation:

**1️⃣ Model Selection**

**After analyzing various LLMs, we selected:**

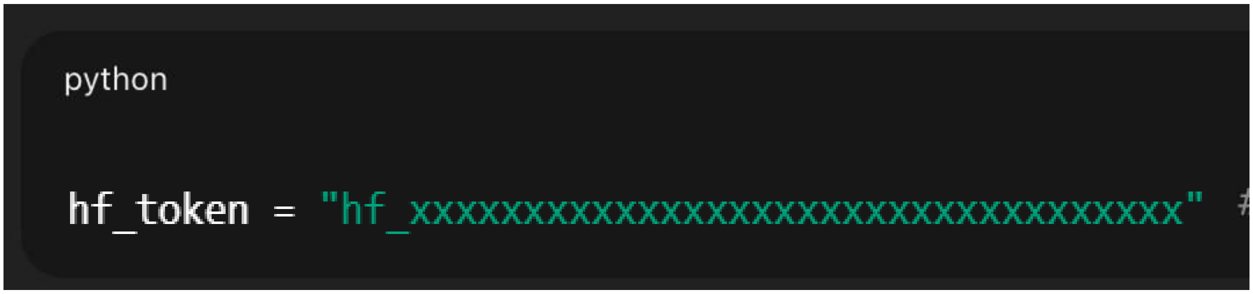


```python
model_name = "ibm-granite/granite-3.3-2b-instruct"
```

- **Chosen for its:**
  - **Instruction-tuned capabilities.**
  - **Good performance on generative healthcare tasks.**
  - **Easy integration using Hugging Face.**

**2️⃣ Token Authentication Setup**

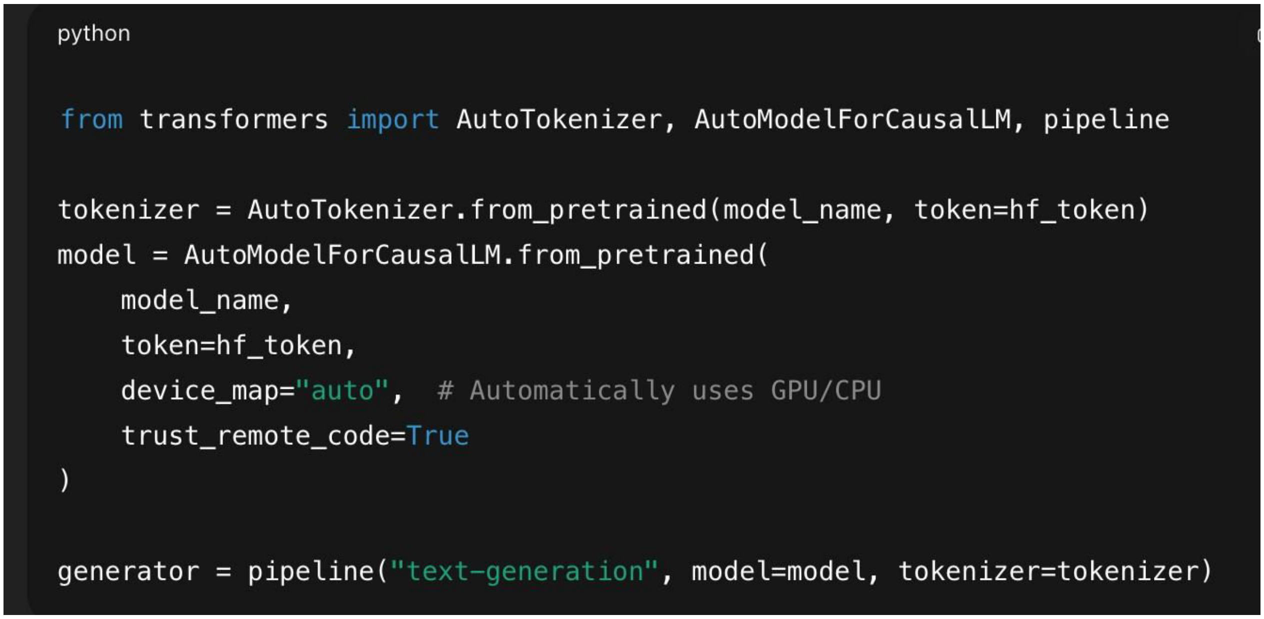
**To access private or gated models, we generated a Hugging Face token:**



```python
hf_token = "hf_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" #
```

## 3️⃣ Model Loading & Integration

**We used the `transformers` library to load both the tokenizer and model with GPU support (if available):**

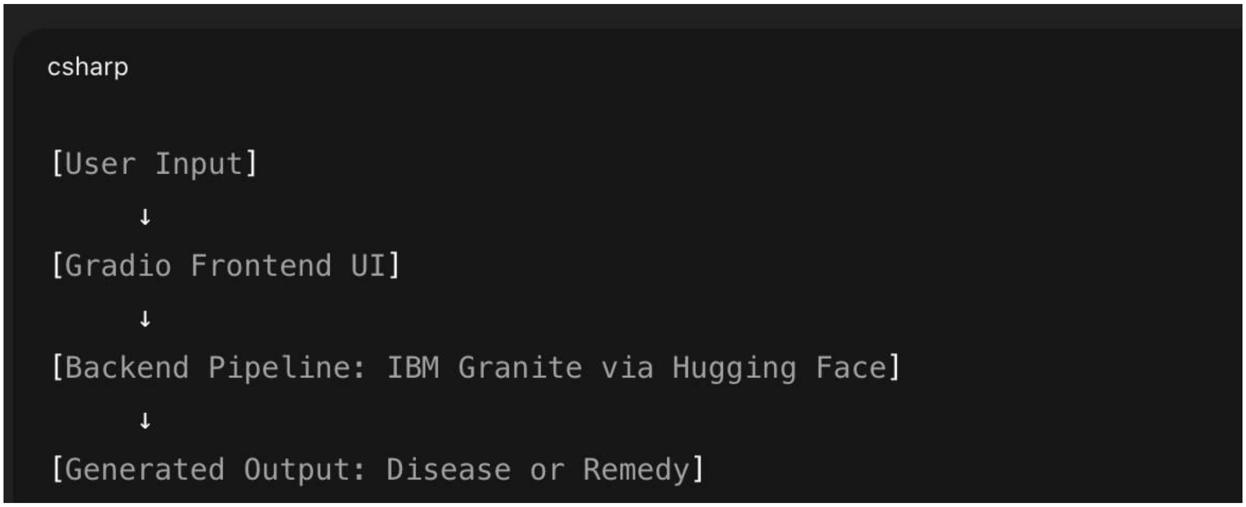


python

```python
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline

tokenizer = AutoTokenizer.from_pretrained(model_name, token=hf_token)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    token=hf_token,
    device_map="auto",  # Automatically uses GPU/CPU
    trust_remote_code=True
)

generator = pipeline("text-generation", model=model, tokenizer=tokenizer)
```

## 4️⃣ Technical Architecture Design

The architecture was planned as:



csharp

```csharp
[User Input]
      ↓
[Gradio Frontend UI]
      ↓
[Backend Pipeline: IBM Granite via Hugging Face]
      ↓
[Generated Output: Disease or Remedy]
```

# Milestone 2: Core Functionalities Development

## Objective:

**To implement the core AI features that enable the model to:**

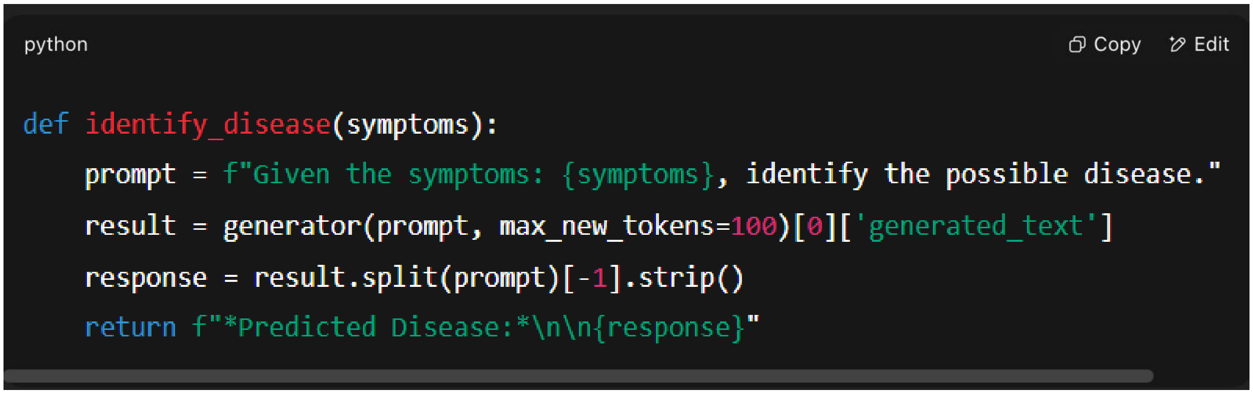
1. **Identify possible diseases from user-provided symptoms.**
2. **Recommend natural home remedies for given diseases.**
3. **Answer health-related questions with accurate and helpful information using AI.**

## Tasks & Implementation:

### 1️⃣ Function 1: Disease Prediction from Symptoms

**Goal: Accept symptoms as input and return the most likely disease using the LLM.**

**def identify_disease(symptoms):**



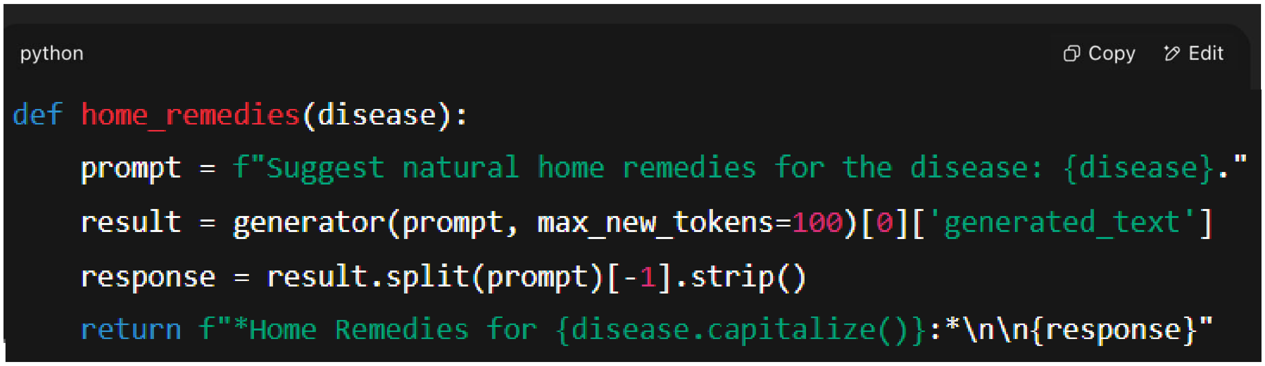
```python
def identify_disease(symptoms):
    prompt = f"Given the symptoms: {symptoms}, identify the possible disease."
    result = generator(prompt, max_new_tokens=100)[0]['generated_text']
    response = result.split(prompt)[-1].strip()
    return f"*Predicted Disease:*\n\n{response}"
```

**Explanation:**

- `prompt` **is dynamically generated based on user input.**
- **The AI model is instructed to behave like a "medical assistant."**
- **We split the output to remove the original prompt and return only the answer.**

### 2 Function 2: Natural Remedy Suggestion

**Goal: Provide a natural home remedy for a specific disease.**



```python
def home_remedies(disease):
    prompt = f"Suggest natural home remedies for the disease: {disease}."
    result = generator(prompt, max_new_tokens=100)[0]['generated_text']
    response = result.split(prompt)[-1].strip()
    return f"*Home Remedies for {disease.capitalize()}:*\n\n{response}"
```

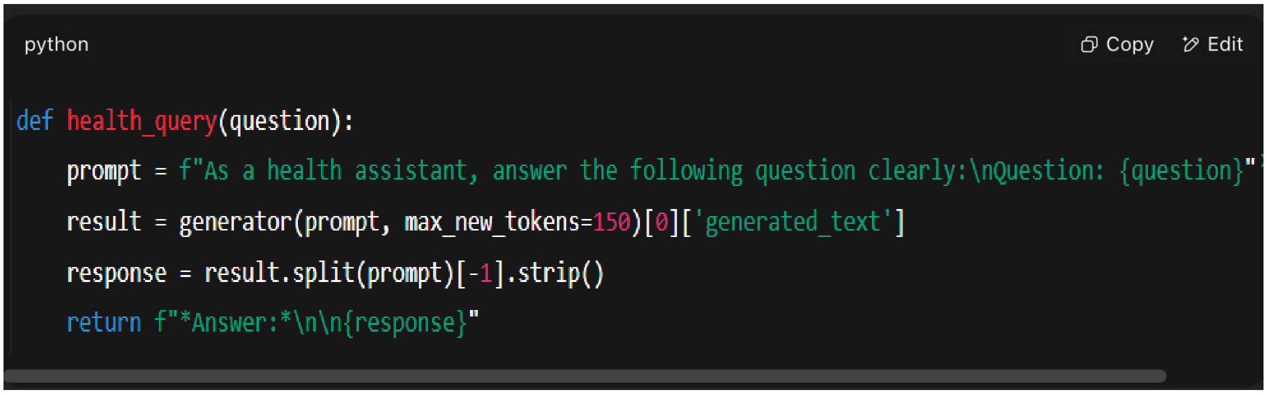
**Explanation:**

- **Designed to act like a wellness coach.**
- **Provides non-medicated, home-based remedies (e.g., herbal, food-based).**
- **Ensures outputs are safe and accessible.**

### 3 Function 3: Health Query Response

**Goal: Provide clear, AI-generated answers to general health-related questions.**

**def health_query(question):**

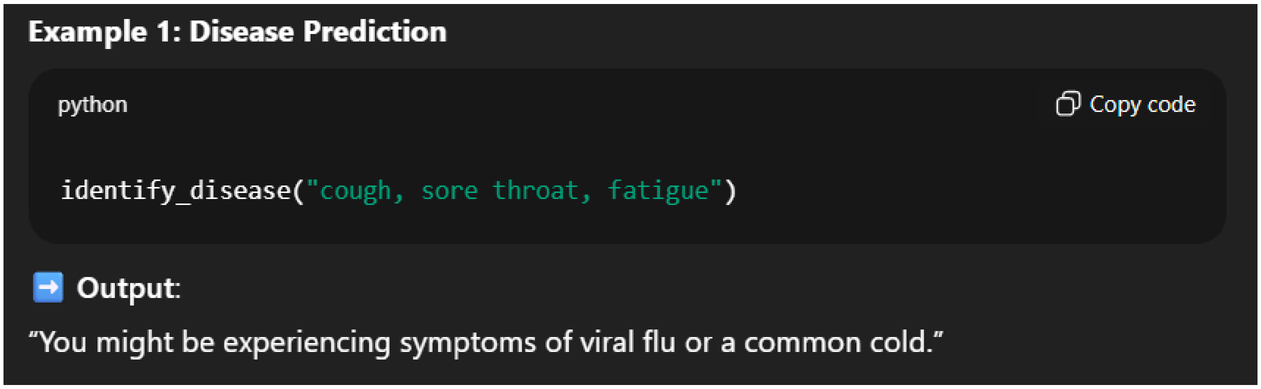


```python
def health_query(question):
    prompt = f"As a health assistant, answer the following question clearly:\nQuestion: {question}"
    result = generator(prompt, max_new_tokens=150)[0]['generated_text']
    response = result.split(prompt)[-1].strip()
    return f"*Answer:*\n\n{response}"
```

**Explanation:**

- **Designed to behave like a digital health assistant.**
- **Offers answers to a wide range of health topics .**
- **Helps users get quick guidance without searching multiple sources.**

**Function Testing**

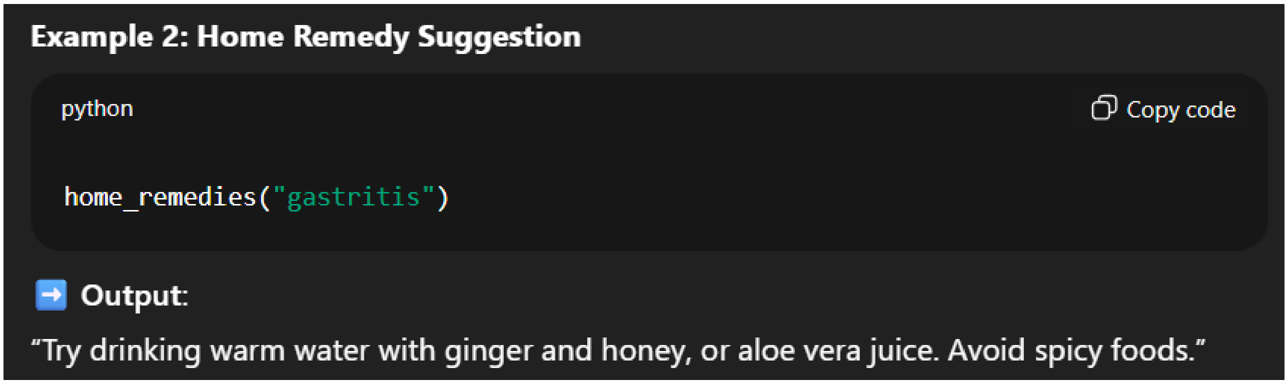


**Example 1: Disease Prediction**

```python
identify_disease("cough, sore throat, fatigue")
```

➡️ **Output**:

"You might be experiencing symptoms of viral flu or a common cold."

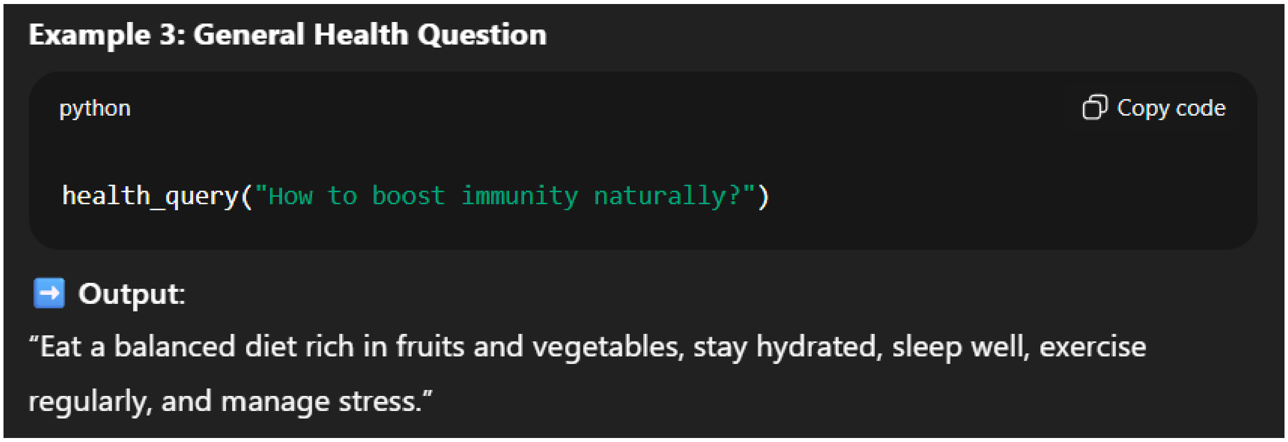


**Example 2: Home Remedy Suggestion**

```python
home_remedies("gastritis")
```

➡️ **Output**:

"Try drinking warm water with ginger and honey, or aloe vera juice. Avoid spicy foods."



**Example 3: General Health Question**

```python
health_query("How to boost immunity naturally?")
```

➡️ **Output**:

"Eat a balanced diet rich in fruits and vegetables, stay hydrated, sleep well, exercise regularly, and manage stress."

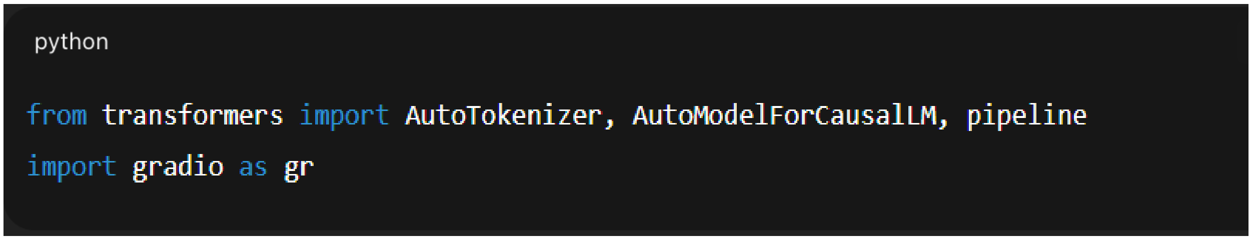
# Milestone 3: App.py Development

## Objective:

**To develop a single executable script (`app.py`) that combines model loading, core functionalities, and the Gradio-based frontend into a complete working application.**

## Tasks & Implementation:

### 1️⃣ Importing Required Libraries

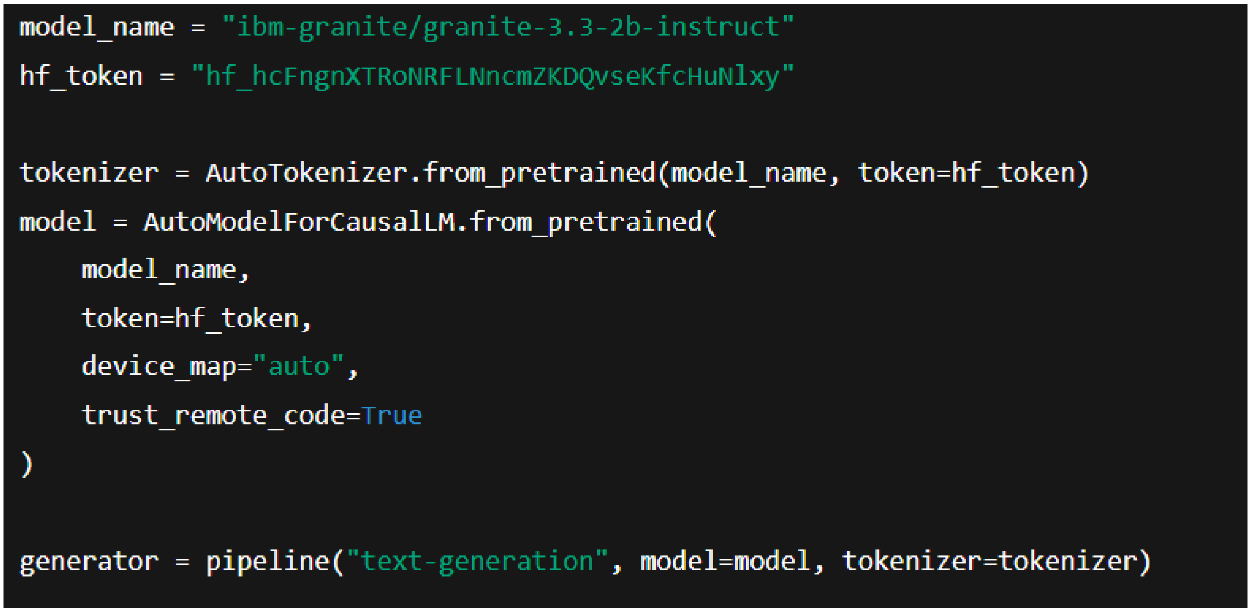


python

```python
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline
import gradio as gr
```

- `transformers` **for model access and inference.**
- `gradio` **for building the interactive web UI.**

### 2️⃣ Model Initialization

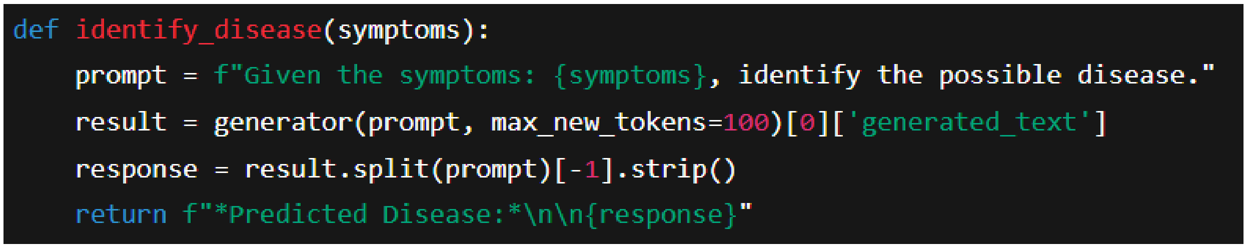


```python
model_name = "ibm-granite/granite-3.3-2b-instruct"
hf_token = "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"

tokenizer = AutoTokenizer.from_pretrained(model_name, token=hf_token)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    token=hf_token,
    device_map="auto",
    trust_remote_code=True
)

generator = pipeline("text-generation", model=model, tokenizer=tokenizer)
```

- This initializes the tokenizer and model with Hugging Face token authentication.
- `device_map="auto"` allows it to choose between GPU/CPU automatically.

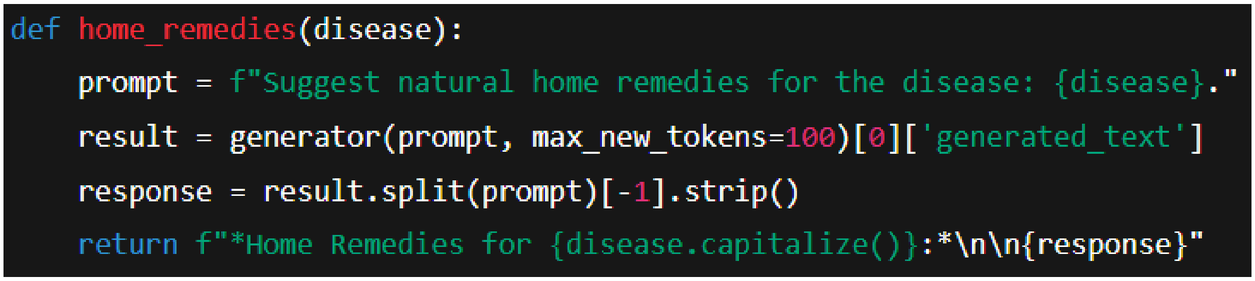
### 3️⃣ Defining Core Functions

**Identify Disease Function:**



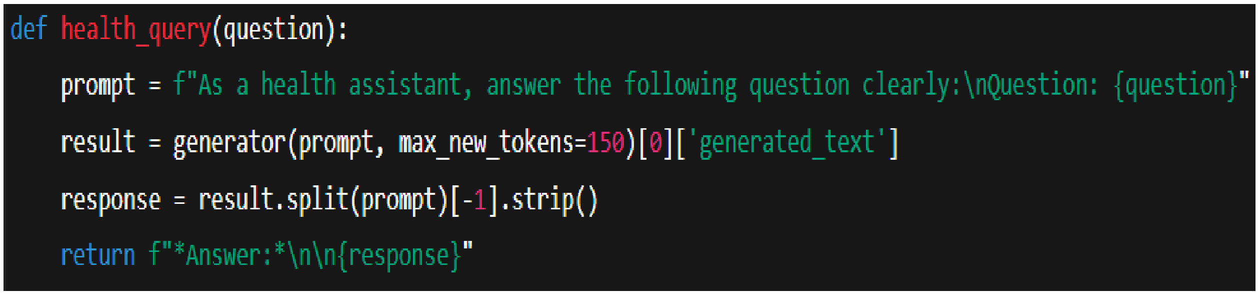
```
def identify_disease(symptoms):
    prompt = f"Given the symptoms: {symptoms}, identify the possible disease."
    result = generator(prompt, max_new_tokens=100)[0]['generated_text']
    response = result.split(prompt)[-1].strip()
    return f"*Predicted Disease:*\n\n{response}"
```

**Suggest Remedy Function:**



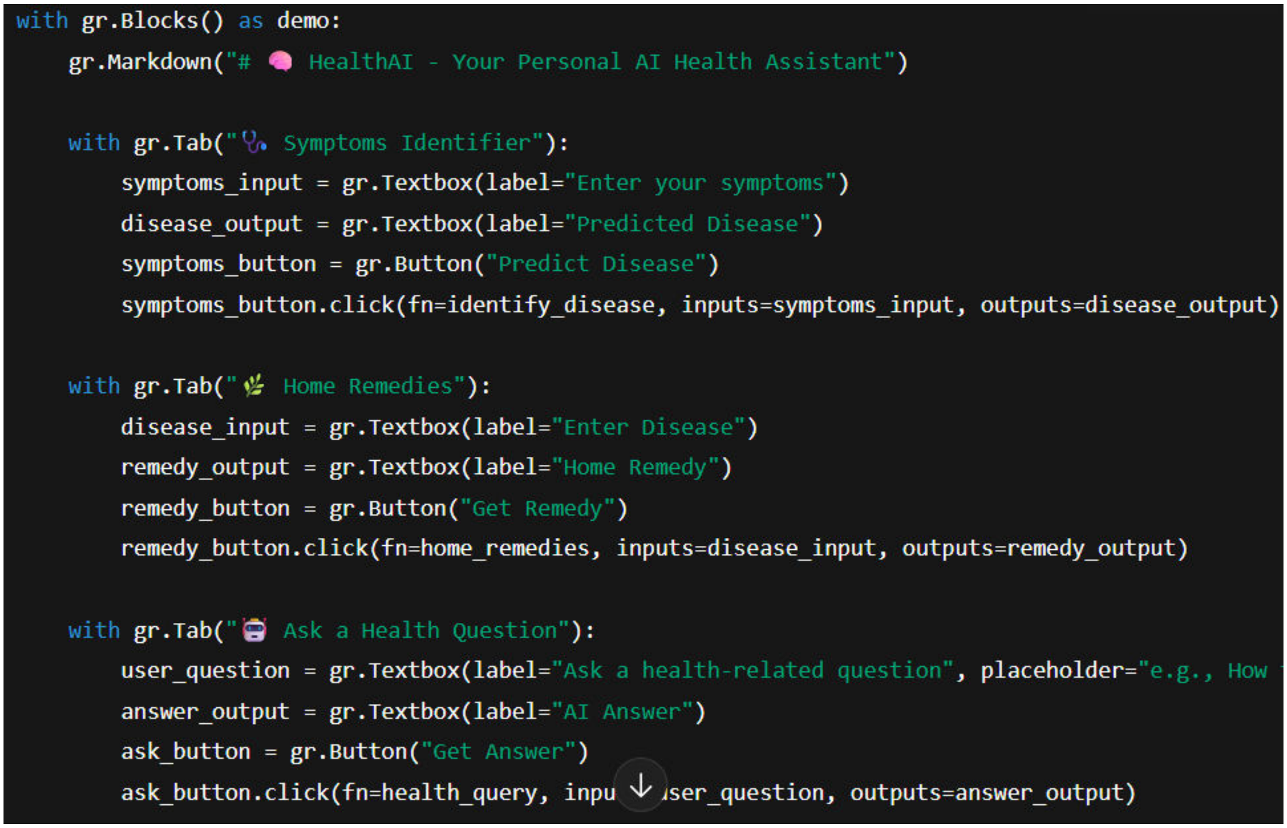
```
def home_remedies(disease):
    prompt = f"Suggest natural home remedies for the disease: {disease}."
    result = generator(prompt, max_new_tokens=100)[0]['generated_text']
    response = result.split(prompt)[-1].strip()
    return f"*Home Remedies for {disease.capitalize()}:*\n\n{response}"
```

**Ask a Health Question Function:**



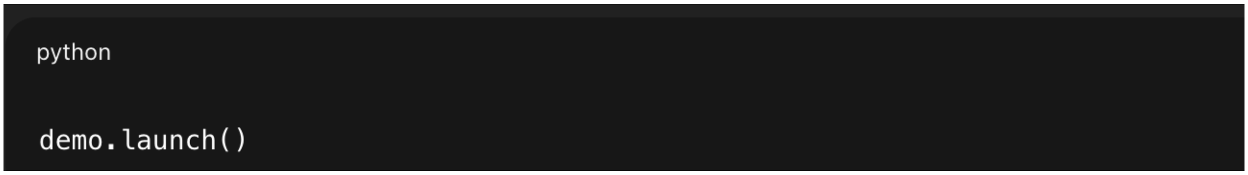
```
def health_query(question):
    prompt = f"As a health assistant, answer the following question clearly:\nQuestion: {question}"
    result = generator(prompt, max_new_tokens=150)[0]['generated_text']
    response = result.split(prompt)[-1].strip()
    return f"*Answer:*\n\n{response}"
```

## 4️⃣ Gradio Frontend Setup



```python
with gr.Blocks() as demo:
    gr.Markdown("# 🧠 HealthAI - Your Personal AI Health Assistant")

    with gr.Tab("🩺 Symptoms Identifier"):
        symptoms_input = gr.Textbox(label="Enter your symptoms")
        disease_output = gr.Textbox(label="Predicted Disease")
        symptoms_button = gr.Button("Predict Disease")
        symptoms_button.click(fn=identify_disease, inputs=symptoms_input, outputs=disease_output)

    with gr.Tab("🌿 Home Remedies"):
        disease_input = gr.Textbox(label="Enter Disease")
        remedy_output = gr.Textbox(label="Home Remedy")
        remedy_button = gr.Button("Get Remedy")
        remedy_button.click(fn=home_remedies, inputs=disease_input, outputs=remedy_output)

    with gr.Tab("🤖 Ask a Health Question"):
        user_question = gr.Textbox(label="Ask a health-related question", placeholder="e.g., How
        answer_output = gr.Textbox(label="AI Answer")
        ask_button = gr.Button("Get Answer")
        ask_button.click(fn=health_query, inpu ↓ ser_question, outputs=answer_output)
```

## 5️⃣ App Launch



python

```python
demo.launch()
```

# Milestone 4: Frontend Development

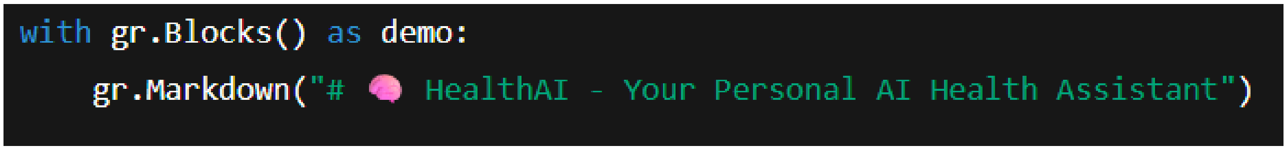
## Objective:

To create a clean, user-friendly web interface using Gradio that allows users to:

- Input symptoms or disease names.
- View AI-generated outputs in a simple, interactive layout.

## Tasks & Implementation:

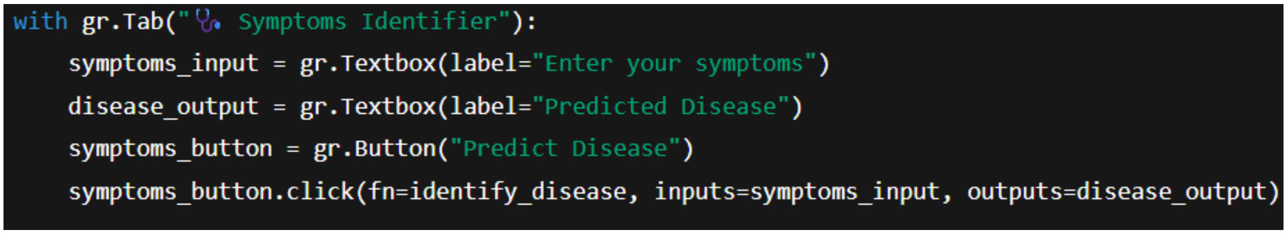
### 1️⃣ Gradio Interface Setup



```
with gr.Blocks() as demo:
    gr.Markdown("# 🧠 HealthAI - Your Personal AI Health Assistant")
```

- Creates a Gradio Blocks layout.
- Adds a Markdown title for branding and clarity.

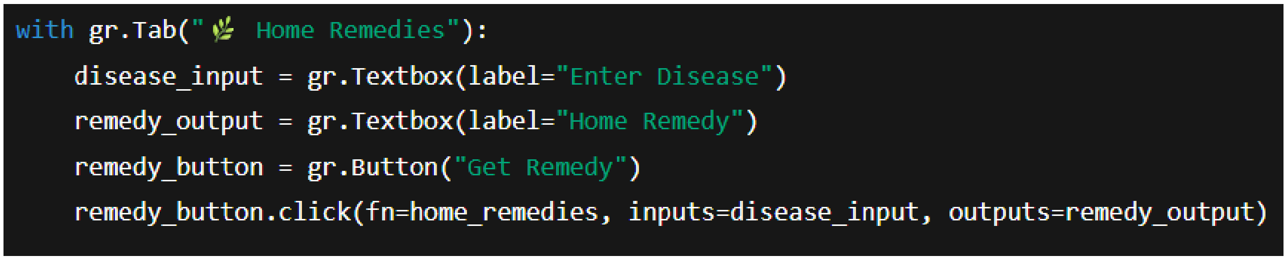
### 2️⃣ Tab 1: Symptoms Identifier



```
with gr.Tab("🩺 Symptoms Identifier"):
    symptoms_input = gr.Textbox(label="Enter your symptoms")
    disease_output = gr.Textbox(label="Predicted Disease")
    symptoms_button = gr.Button("Predict Disease")
    symptoms_button.click(fn=identify_disease, inputs=symptoms_input, outputs=disease_output)
```

- Provides a Textbox for users to enter symptoms.
- Displays the predicted disease in a readonly textbox.
- Clicking the “Predict Disease” button triggers the backend function.

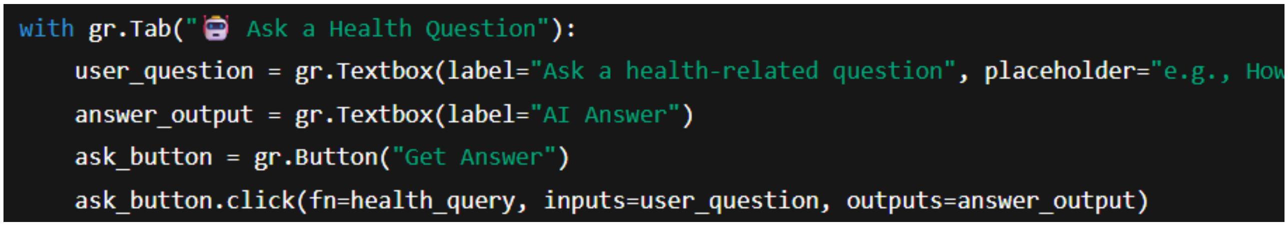
### 3️⃣ Tab 2: Home Remedies



```
with gr.Tab("🌿 Home Remedies"):
    disease_input = gr.Textbox(label="Enter Disease")
    remedy_output = gr.Textbox(label="Home Remedy")
    remedy_button = gr.Button("Get Remedy")
    remedy_button.click(fn=home_remedies, inputs=disease_input, outputs=remedy_output)
```

- **Accepts a disease name as input.**
- **Displays AI-generated natural remedy in output textbox.**
- **Fully interactive with real-time backend integration.**

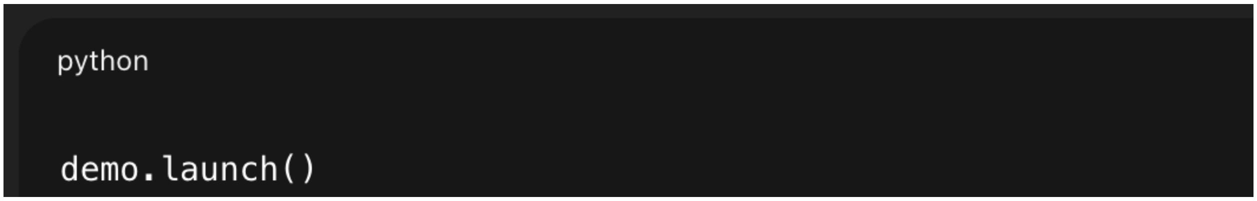
### 4️⃣ Tab 3 : Ask a Health Question



```
with gr.Tab("🤖 Ask a Health Question"):
    user_question = gr.Textbox(label="Ask a health-related question", placeholder="e.g., How
    answer_output = gr.Textbox(label="AI Answer")
    ask_button = gr.Button("Get Answer")
    ask_button.click(fn=health_query, inputs=user_question, outputs=answer_output)
```

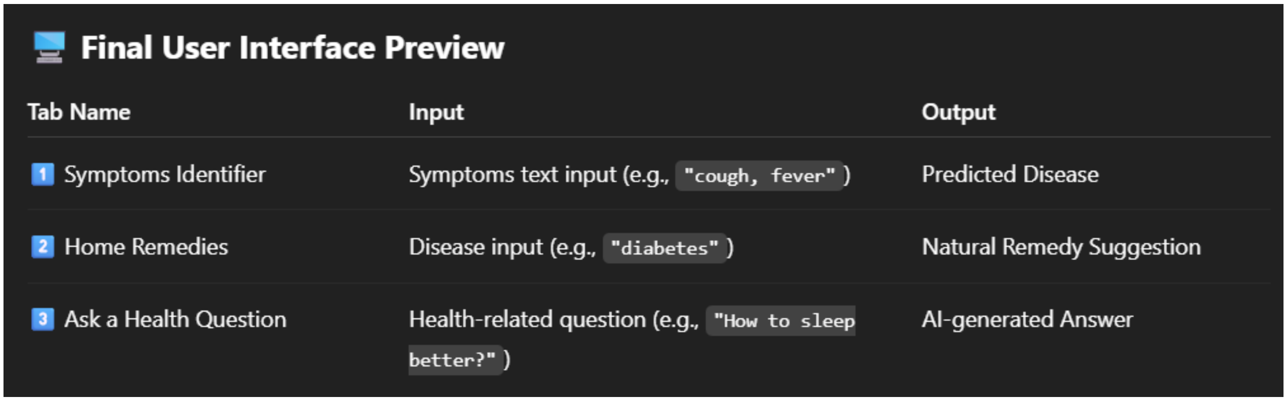
- **Allows users to type in any general or specific health-related question.**
- **Returns clear and concise answers tailored to the user's query.**
- **Useful for common health doubts like immunity tips, diet advice, or prevention strategies.**

### 5️⃣ App Launch



```
python

demo.launch()
```

- **Launches the Gradio app either locally or via a public link (if on Colab/Jupyter).**
- **Displays an easy-to-use 2-tab UI with live interaction.**



## 🖥️ Final User Interface Preview

| Tab Name | Input | Output |
|---|---|---|
| 1️⃣ Symptoms Identifier | Symptoms text input (e.g., `"cough, fever"`) | Predicted Disease |
| 2️⃣ Home Remedies | Disease input (e.g., `"diabetes"`) | Natural Remedy Suggestion |
| 3️⃣ Ask a Health Question | Health-related question (e.g., `"How to sleep better?"`) | AI-generated Answer |

# Milestone 5: Deployment

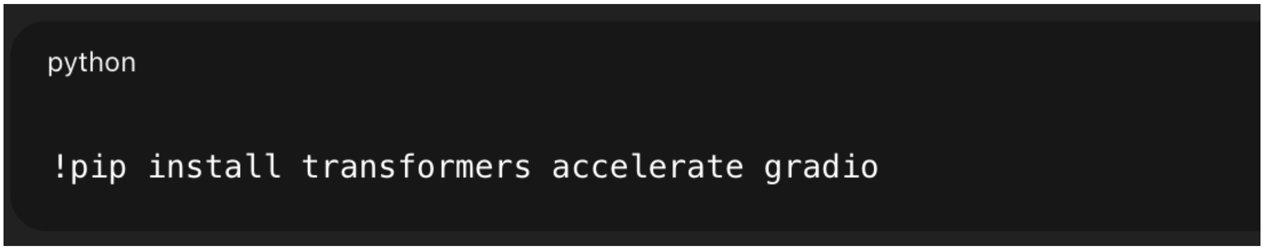
## Objective:

**To deploy the HealthAI application in an accessible environment so users can interact with the model through a public web interface.**

## Tasks & Implementation:

### 1️⃣ Environment Setup:

- **Platform: Google Colab (notebook-based deployment)**
- **Dependencies installed:**

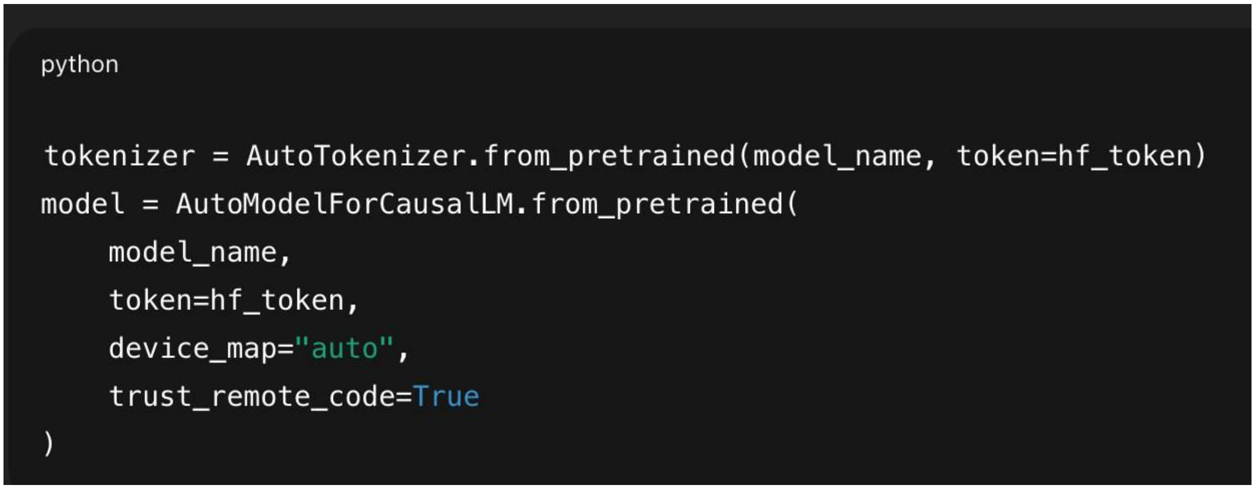


```
!pip install transformers accelerate gradio
```

- **Verified installation of required packages:** `transformers`, `gradio`, `huggingface_hub`, **etc.**

### 2️⃣ Model Loading in Colab

- **Used Hugging Face token to authenticate and download the IBM Granite model:**



```
tokenizer = AutoTokenizer.from_pretrained(model_name, token=hf_token)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    token=hf_token,
    device_map="auto",
    trust_remote_code=True
)
```

**Handled model checkpoint downloading and device allocation (CPU/GPU).**

### 3️⃣ Gradio App Launch in Colab

- **Final line in the script:**

```python
demo.launch()
```

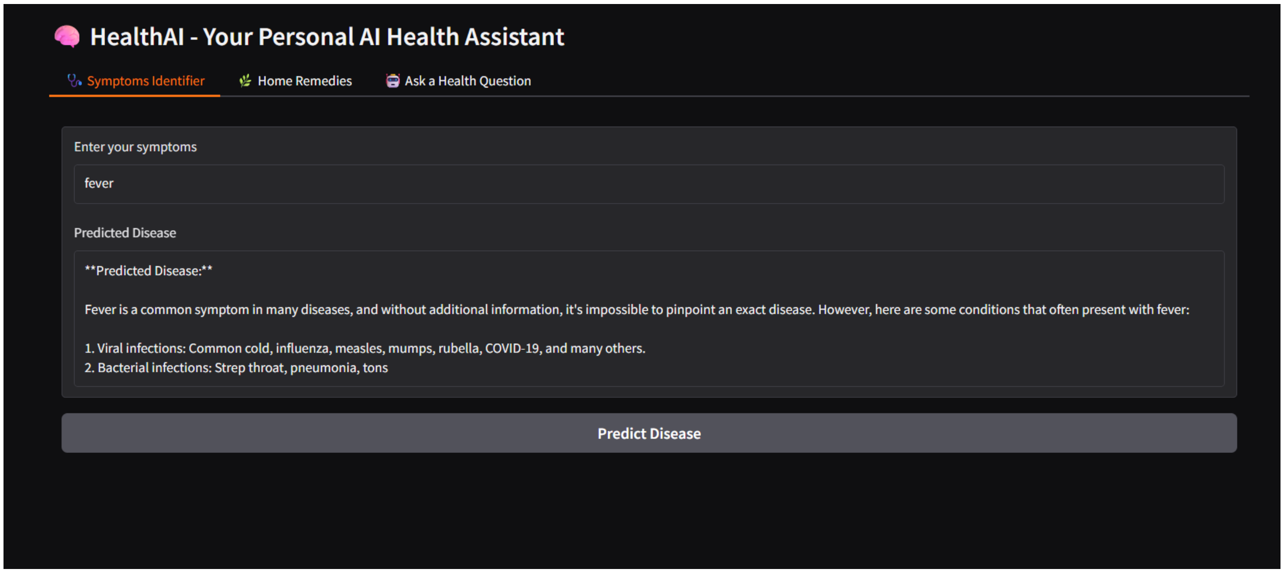
**Colab generates a temporary public URL, for example:**

```csharp
Running on public URL: https://7b0f3e754733638450.gradio.live
```

- **This URL allows anyone to access the AI assistant directly via browser.**

**Exploring Application Features: Real-Time Corrections Page:**

1️⃣ Symptoms Identifier



🧠 HealthAI - Your Personal AI Health Assistant

Enter your symptoms

fever

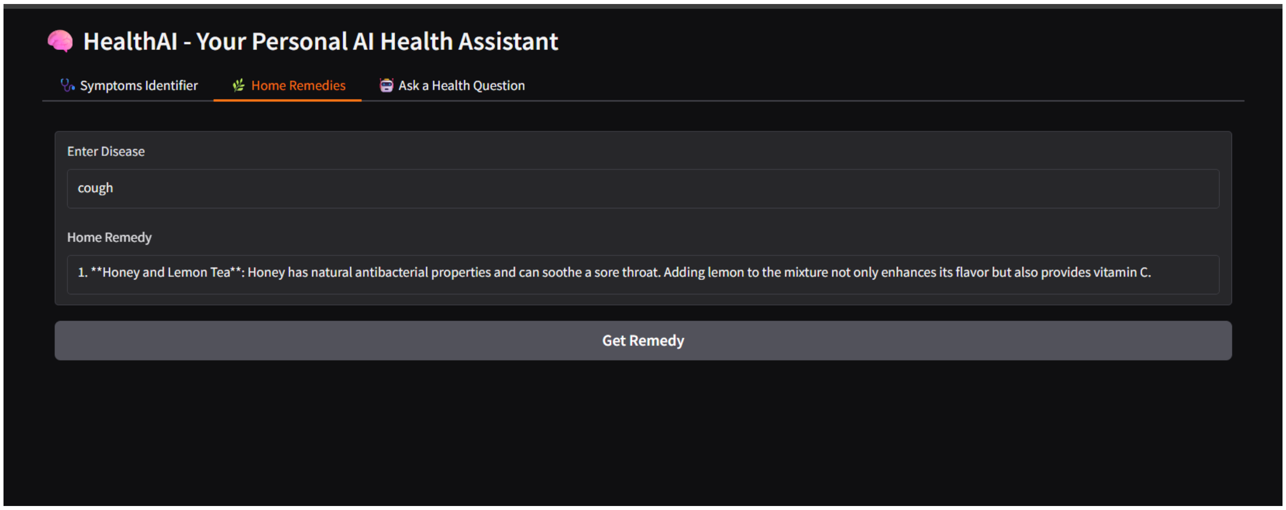
Predicted Disease

**Predicted Disease:**

Fever is a common symptom in many diseases, and without additional information, it's impossible to pinpoint an exact disease. However, here are some conditions that often present with fever:

1. Viral infections: Common cold, influenza, measles, mumps, rubella, COVID-19, and many others.
2. Bacterial infections: Strep throat, pneumonia, tons

## 2️⃣ Home Remedies



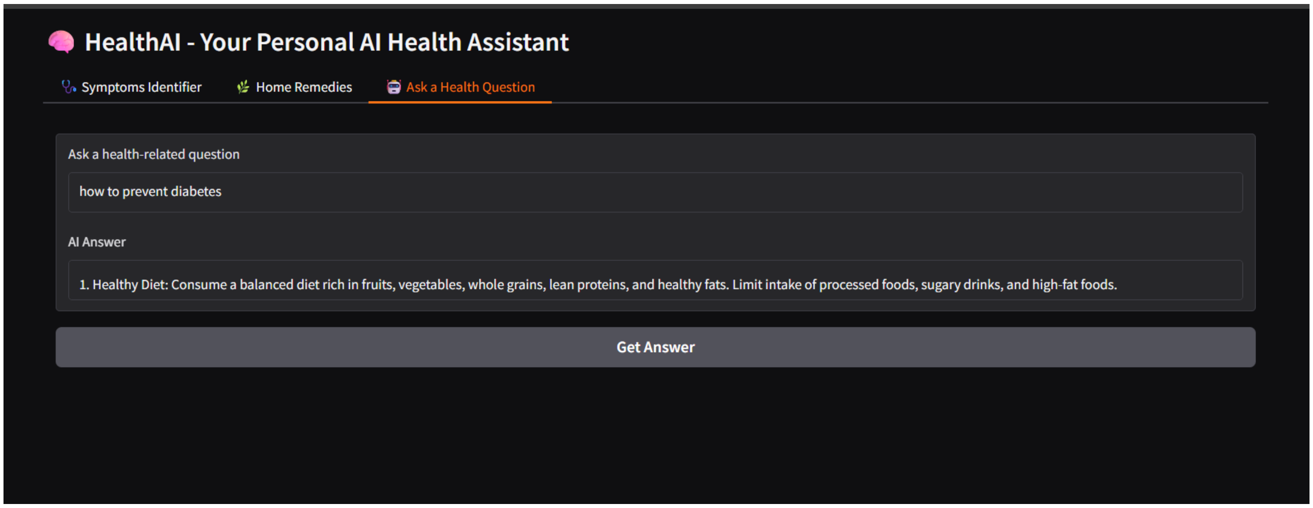
Enter Disease

cough

Home Remedy

1. **Honey and Lemon Tea**: Honey has natural antibacterial properties and can soothe a sore throat. Adding lemon to the mixture not only enhances its flavor but also provides vitamin C.

## 3️⃣ Gradio App Launch in Colab



Ask a health-related question

how to prevent diabetes

AI Answer

1. Healthy Diet: Consume a balanced diet rich in fruits, vegetables, whole grains, lean proteins, and healthy fats. Limit intake of processed foods, sugary drinks, and high-fat foods.

# Conclusion

**The HealthAI project demonstrates the powerful synergy between modern Generative AI models and user-friendly web interfaces to address real-world healthcare needs. By integrating the IBM Granite LLM through Hugging Face and deploying a fully functional app using Gradio, this solution offers intelligent, responsive, and accessible healthcare support.**

**By leveraging this combination, HealthAI offers a seamless platform that:**

- 🩺 **Predicts potential diseases based on user-input symptoms.**
- 🌿 **Suggests natural home remedies for common conditions.**
- 🤖 **Answers general health-related queries in real time using AI.**

**This project not only empowers users with AI-assisted self-assessment, but also sets a foundation for AI-driven health tools that can scale to remote areas, educational contexts, and future integrations with wearables and voice assistants.**

**Overall, HealthAI reflects the future of personalized, accessible, and smart digital healthcare—delivered by AI.**