

HealthAI-Intelligent Healthcare

Assistant

GenerativeAIwithIBM



1.INTRODUCTION:

Health AI – Intelligent Healthcare Assistant

TeamMembers

- Balakrishnan D(834B69A172C3815AC174093758F80D80)
- Jeevanantham S(6749C2FA937A30B8A934E916D348F39F)
- Sanjay S(27AEE5BD0C8E9AF81E44BE3CE06BFDBE)
- Sivaguru U(C852305DCF319CB912F4AC5F4F6B5AC9)

2.PROJECTOVERVIEW :

HealthAI-intelligent healthcare Assistant is an AI-driven system designed to improvehealthcaredelivery through intelligent support. The solution integrates AI/ML modelswithLLMsandvector databases, user-friendly healthcare assistance.

• ConversationInterface

The conversational interface allows users to interact with the healthcare natural text or voice. It supports health queries, policy guidance and human experts.

- **Policy summarization**

The system automatically extracts and condenses healthcare policies into simple, easy-to-read summaries. It helps patients, citizens and staff quickly understand key update.

- **Eco-Tip Generator**

The eco-tip generator provides simple, actionable health and environment-friendly suggestions. It promotes sustainable practices like waste reduction, energy, saving methods for community health.

- **Citizen feedback loop**

The citizen feedback loop collects patient and public opinion through survey, chat, or forms. It analyses feedback using sentiment and topic detection to improve system enhancement.

KPI • Forecasting

KPI forecasting predicts key healthcare performance indicators such as bed occupancy, patients' inflow, staff availability, and resource usage. It helps in planning, decision-making, improving overall Healthcare efficiency.

- **Anomaly Detection**

Anomaly detection identifies unusual pattern in healthcare data, such as sudden diseases spikes or abnormal resources usage. It enables early alerts and quick action to improve patient safety.

- **Multimodal input support**

Multimodal input support allows a healthcare assistant to receive and process text, voice, image and sensor data. This enables more natural, accurate, and efficient interaction for patients and medical staff.

- **Stream lit to Gradio UI**

Stream lit to Gradio UI allows converting python apps into interactive web interface with easy deployment. Gradio provide simple drag-and-drop components for inputs and outputs, user interaction.

3.ARCHITECTURE:

- Front-end Architecture of Health AI-Intelligent Healthcare assistant interaction such as web, mobile or voice app, allowing patients or citizens to accessible way.

- Back-end Architecture of Health AI-Intelligent Healthcare assistant handles business logic, connect LLM for prediction and recommendations provide APLs for secure, real-time communication with the front end

LLM Integration:

-

The LLM Processes user queries, interpret intent, and generate response while interacting with for knowledge retrieval and personalized recommendations.

- **Vector sector:**

The vector sector in health care AI stores embedding of medical data, policies, records to enable fast semantic search, allowing the AI to Retrieve contextually relevant information guidance.

- **ML Modules:**

The ML Modules provide analytics, anomaly detection and personalized health recommendations by analysing patient data, trends medical knowledge to support.

4.SETUP INTRODUCTION:

- **Prerequisites:**

- Install python, node and code editor
- Access open AI API and vector database
- Basic ML knowledge and libraries
- Front end skills (react, angular or flutter)
- Medical database or pre-trained models

- **Installation process:**

- Install python, node and set up a virtual environment.
- Install backend and AI libraries
- Setup vector database and create index.
- Run back end and front end test queries and AI responses.

5.FOLDER STRUCTURE:

- App: application code
- DeotcaaT sts:: :t uSmUDeetoteonaittctraaagtdifoilnes
- s
-
-

- Ven: Virtual environment
- Uxtil: The Uxtil utility function
-

6. RUNNING THE APPLICATION:

To start the project:

- Open terminal and clone the project from GitHub.
- Navigate to the project folder
- Health AI-Intelligent healthcare assistant-AI.
- Create and activate a virtual environment.
- Install dependencies using `pip install -r requirements.txt`.
- Set up database and vector store (initialize configs).
- Configure API keys (OpenAI/LLM, database, etc.).
- Start backend server using `python backend/API/main.py`.
- Run frontend UI using `streamlit run frontend/app.py` Or `gradio app.py`.

Frontend (Streamlit):

Enter frontend folder - run `streamlit` – open browser- use Health AI-Intelligent healthcare assistant UI.

Backend (fast API):

Enter backend folder – run fast API server – open API docs – backend ready for health AI-Intelligent healthcare assistant UI integration.

7. API DOCUMENTATION:

API Documentation provides endpoint for patient interaction, health record management, and AI – drive recommendations, enabling seamless healthcare support through conversational and analytical modules.

8. AUTHENTICATION:

Register user/client – user signs up or application register to receive API credentials.

Generate API key/token – system issues a unique API key or JWT token.

Send authentication request – client sends login request with credentials.

Receive access token – server validates and responds with an access token.

Use token in API calls – includes authorization: bearer <token>in header for every request.

9.USER INTERFACE:

- Provides a simple login for secure access.
- CDAIha gyseas oih atbpwreindsdopr zw ehd ses tsilopaotduhops ar rewdcnlosinhimcod nrctvd
-
-

10.TESTING:

- Testing in the health AI – intelligent healthcare assistant ensures the system works reliably and safety for patients and doctors. First, unit testing is performed to verify individual modules like authentication, chat, and health records access.
User acceptance testing (UAT) then validate that real user finds the system
- useful, accurate, easy to use. This step-by-step testing process guarantees a robust and trustworthy healthcare assistant.

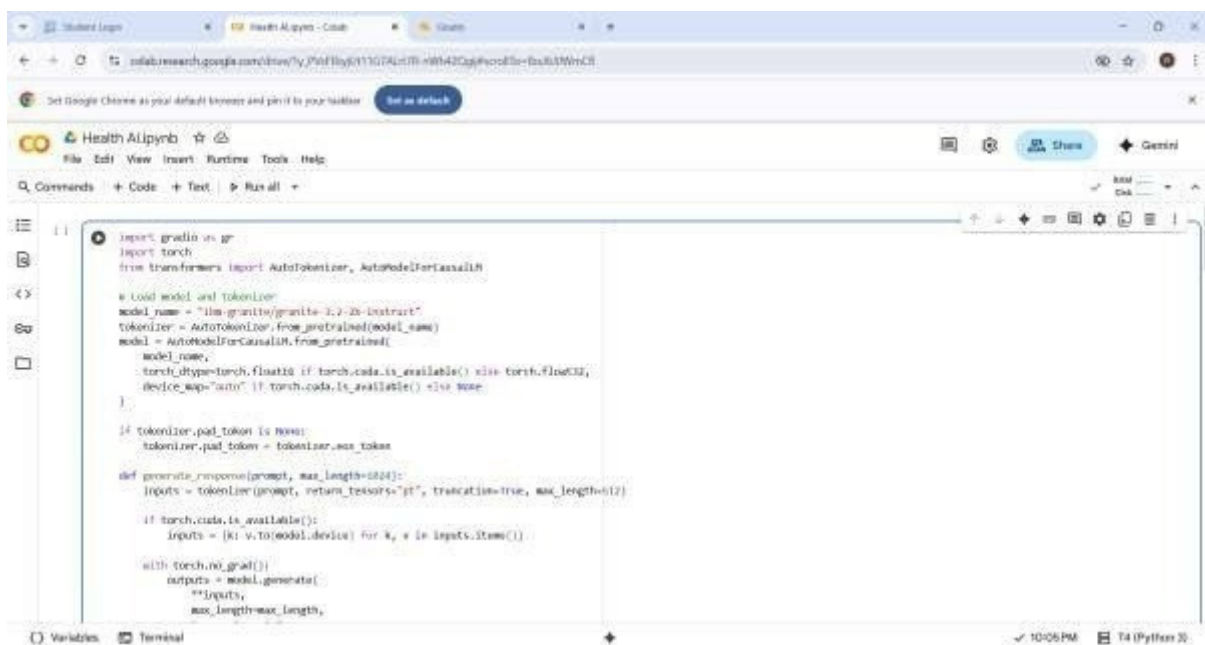
11.KNOWN ISSUES:

The Health AI-Intelligent healthcare assistant, while highly efficient, currently faces certain limitations. Occasional inaccuracies in symptom analysis may occur due to limited training data for rare conditions. Multimodal input processing, such as interpreting images and text together, may experience delay or misinterpretation under poor-quality inputs. Some users may encounter minor UI inconsistencies across different device these issues, improve reliability and enhance overall user experience. testing and initial deployment, several issues were identified that may affected the system performance or user experience. Some feature may response slowly under heavy load and occasionally UI observed on different device. Certain data inputs may trigger unexpected errors if they do not match the expected format additionally, integration with external APIs can server downtime will be provided in future releases to stability and functionality. 12.FUTURE ENHANCEMENTS:

In future as Health AI-Intelligent healthcare assistant update the system improvement for the health AI assistant will focus on expanding its capabilities and improving patient care. Planned enhancement include incorporating more advanced diagnosis tools, supporting multimodal input such as voice and image, and integrating with additional health data sources for personalized recommendations. The system will also implement predictive analytics for early disease detection, real- time health

monitoring and enhance natural language understanding for more accurate and empathetic interactions. The user interface will be refined for greater accessibility and responsibility while automated error handling and notification system will be implemented to ensure smoother operations. These enhancements aim to provide a more robust, efficient and user- friendly solution.

.13PROJECT SCREENSHOT:



```
import gradio as gr
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

# Load model and tokenizer
model_name = "gpt2-medium"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    torch_dtype=torch.float16 if torch.cuda.is_available() else torch.float32,
    device_map="auto" if torch.cuda.is_available() else None
)

# Tokenizer padding
tokenizer.pad_token = tokenizer.eos_token

def generate_response(prompt, max_length=256):
    inputs = tokenizer(prompt, return_tensors="pt", truncation=True, max_length=64)
    if torch.cuda.is_available():
        inputs = {k: v.to(model.device) for k, v in inputs.items()}
    with torch.no_grad():
        outputs = model.generate(
            *inputs,
            max_length=max_length,
```

```
Health Alpyrb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text + Run all +
1.1
max_length=max_length,
temperature=0.7,
do_sample=True,
pad_token_id=tokenizer.eos_token_id
}

response = tokenizer.decode(outputs[0], skip_special_tokens=True)
response = response.replace(prompt, "").strip()
return response

def disease_prediction(symptoms):
    prompt = f"Based on the following symptoms, provide possible medical conditions and general medication suggestions. Always emphasize the importance of consulting a doctor for advice."
    return generate_response(prompt, max_length=200)

def treatment_plan(condition, age, gender, medical_history):
    prompt = f"Generate personalized treatment suggestions for the following patient information. Include home remedies and general medication guidelines, if applicable. Condition: {condition}, Age: {age}, Gender: {gender}, Medical History: {medical_history}"
    return generate_response(prompt, max_length=200)

# Create a simple interface
with gr.Blocks() as app:
    gr.Markdown("A Medical AI Assistant")
    gr.Markdown("Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.")

    with gr.Tab():
        with gr.TabItem("Disease Prediction"):
            with gr.Column():
                with gr.Column():
                    symptoms_input = gr.Textbox(
```

```
Health Alpyrb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text + Run all +
1.1
symptoms_input = gr.Textbox(
    label="Enter symptoms",
    placeholder="e.g., fever, headache, cough, fatigue...",
    lines=4
)

predict_btn = gr.Button("Analyze Symptoms")

with gr.Column():
    prediction_output = gr.Textbox(label="Possible conditions & recommendations", lines=10)

predict_btn.click(disease_prediction, inputs=symptoms_input, outputs=prediction_output)

with gr.TabItem("Treatment Plans"):
    with gr.Column():
        with gr.Column():
            condition_input = gr.Textbox(
                label="Medical condition",
                placeholder="e.g., diabetes, hypertension, migraine...",
                lines=2
            )

            age_input = gr.Number(label="Age", value=30)
            gender_input = gr.Dropdown(
                choices=["Male", "Female", "Other"],
                label="Gender",
                value="Male"
            )

            history_input = gr.Textbox(
                label="Medical history",
```

```
Health Alpyrb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text + Run all +
1.1
    label="Medical history",
    placeholder="Previous conditions, allergies, medications or None",
    lines=1
)
plan_btn = gr.Button("Generate Treatment Plan")

with gr.Column():
    plan_output = gr.Textbox(label="Personalized treatment plan", lines=20)

plan_btn.click(treatment_plan, inputs=[condition_input, age_input, gender_input, history_input], outputs=[plan_output])

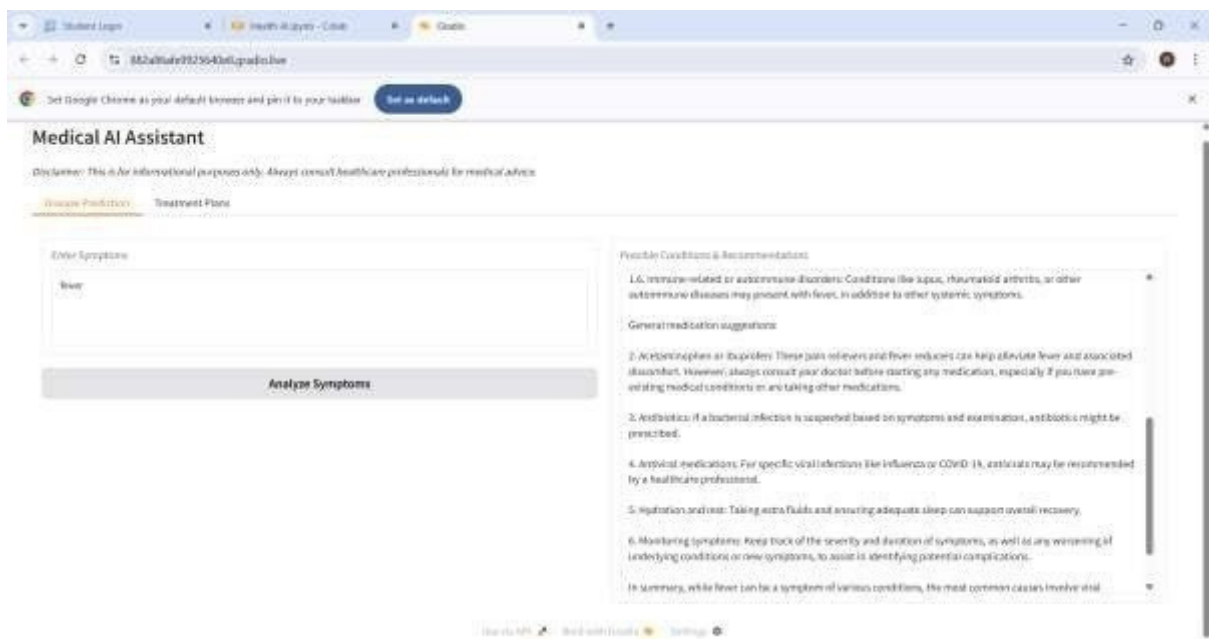
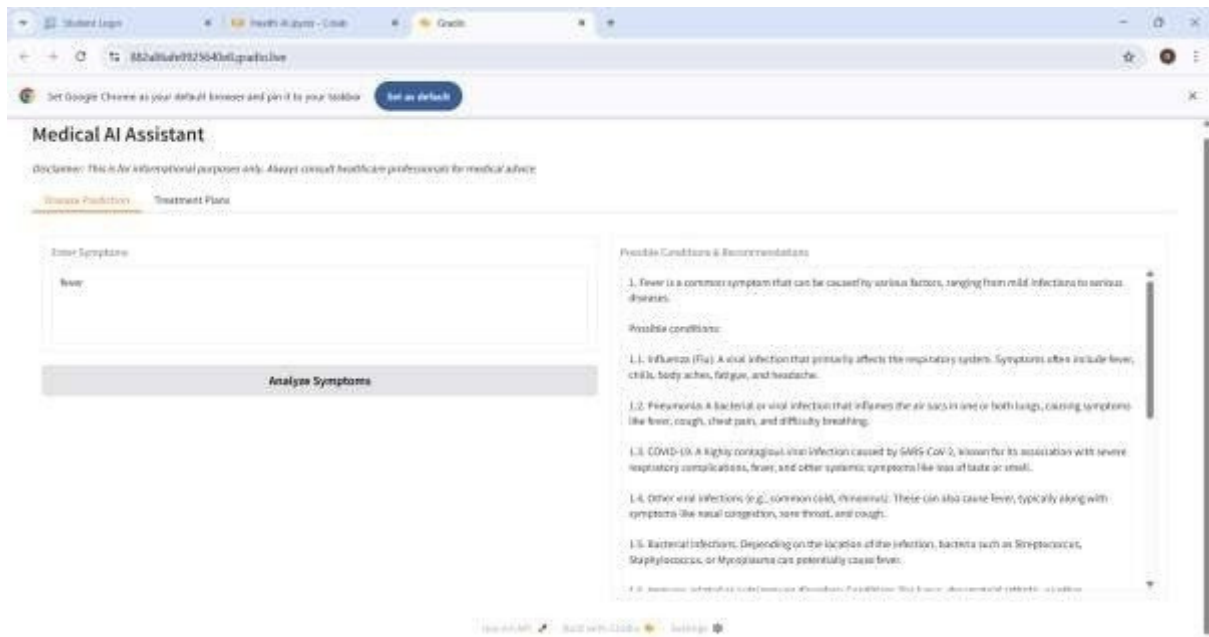
app.launch(share=True)

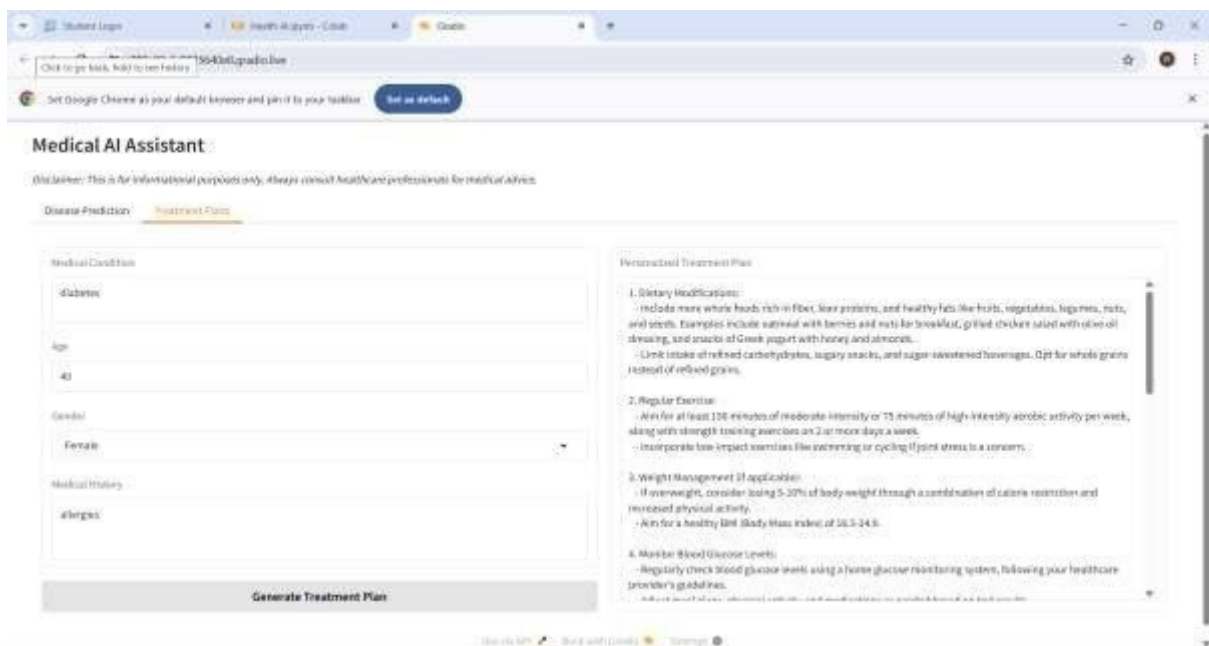
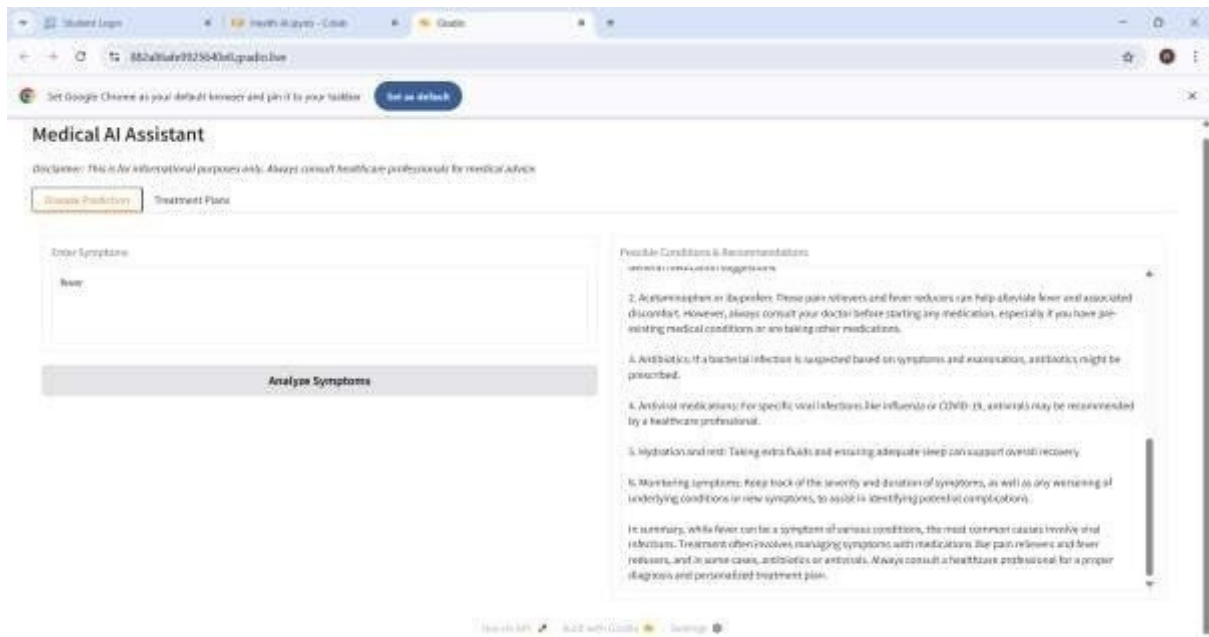
/usr/local/lib/python3.12/dist-packages/huggingface_hub/utils/_auth.py:84: UserWarning:
The secret 'HF_TOKEN' does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your Colab.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
tokenizer_config.json 8.88k/7 [00:00<00:00, 55.7MB/s]
vocab.json 777k/7 [00:00<00:00, 16.7MB/s]
merges.txt 442k/7 [00:00<00:00, 21.5MB/s]
tokenizer.json 3.48M/7 [00:00<00:00, 55.8MB/s]
added_tokens.json 100% 87.0k/0 [00:00<00:00, 6.01MB/s]
special_tokens_map.json 100% 701/701 [00:00<00:00, 84.1MB/s]
```

```
Health Alpyrb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text + Run all +
added_tokens.json 100% 87.0k/0 [00:00<00:00, 6.01MB/s]
special_tokens_map.json 100% 701/701 [00:00<00:00, 84.1MB/s]
config.json 100% 708/708 [00:00<00:00, 81.8MB/s]
"torch_dtype" is deprecated! use "dtype" instead!
model.safetensors.index.json 28.5k/1 [00:00<00:00, 3.16MB/s]
Fetching 2 files 100% 2/2 [01:29<00:00, 50.05s/s]
model-00002-of-00002.safetensors 100% 67.9M/67.1M [00:01<00:00, 73.1MB/s]
model-00001-of-00002.safetensors 100% 5.39G/5.00G [01:28<00:00, 10.7MB/s]
Loading checkpoint shards 100% 2/2 [00:19<00:00, 8.89s/s]
generation_config.json 100% 137/137 [00:00<00:00, 15.7MB/s]
Colab notebook detached. To show errors in Colab notebook, set debug=True in launch()
* running on public URL: https://huggingface.co/spaces/00000000000000000000000000000000

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run "gradio deploy" from the terminal in the working directory to deploy to Hugging Face Spaces.

Medical AI Assistant
Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.
Generate Prediction Treatment Plans
```



Medical AI Assistant

Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.

Disease Prediction **Treatment Plan**

Medical Condition

Diabetes

Age

40

Gender

Female

Medical History

allergies

Generate Treatment Plan

Personalized Treatment Plan

- Regularly check blood glucose levels using a home glucose monitoring system, following your healthcare provider's guidelines.
- Adjust meal plans, physical activity, and medications as needed based on test results.

II. Medication Management:

- Oral Anti-Diabetic Medications:
 - Metformin (e.g., Fortamet, Glucophage): Start with 500mg twice daily, increasing if necessary. It improves insulin sensitivity and reduces hepatic glucose production.
 - Sulfonylureas (e.g., Glipizide, Glyburide): Initiate at 2.5-5mg once daily, but may not be recommended due to potential side effects like hypoglycemia.
 - DPP-4 Inhibitors (e.g., Sitagliptin, Linagliptin): Begin with 100mg daily, usually as a single pill. They increase insulin hormone levels, improving beta-cell function.
 - GLP-1 Receptor Agonists (e.g., Exenatide, Liraglutide): Start with low doses (e.g., 0.2-0.6mg once daily) and can be gradually increased. They mimic insulin hormones and stimulate insulin secretion while suppressing glucagon.
 - SGLT Inhibitors (e.g., Canagliflozin, Empagliflozin): Initiate at 30-100mg once daily, especially for those with kidney issues or obesity. They promote urinary glucose excretion and can lead to weight loss.
- Insulin Therapy (if lifestyle measures aren't sufficient):
 - Begin with basal insulin (e.g., Lantus, Toujeo) at low doses (e.g., 20-30 units/day) and consider adding prandial insulin (e.g., Novolog, Apidra) as needed for mealtime control.
 - Consult a diabetes educator or endocrinologist for individualized insulin dosing and injection techniques.

Medical AI Assistant

Disclaimer: This is for informational purposes only. Always consult healthcare professionals for medical advice.

Disease Prediction **Treatment Plan**

Medical Condition

Diabetes

Age

40

Gender

Female

Medical History

allergies

Generate Treatment Plan

Personalized Treatment Plan

- Insulin or medication evaluation for insulin management, incorporating lifestyle changes and regular monitoring.

6. Insulin Management:

- Penicillin Insulin: May help lower blood glucose levels. Add 1-2 teaspoons of fenugreek powder to meats or drinks.
- Chromium: Might improve insulin sensitivity. Add 1/2 to 1 teaspoon of chromium to food or beverages daily.
- Bitter Melon: Could potentially aid in glucose metabolism. Consume fresh or cooked bitter melon as a vegetable.
- Turmeric: Possesses anti-inflammatory properties and may support glycemic control. Add consistency rich foods (e.g., curry) or consume curcumin supplements, following healthcare provider recommendations.
- Aloe Vera Juice: Some studies suggest it may normalize blood sugar levels. Drink 1-2 ounces daily, but consult a doctor first.

7. Allergy Management:

- Maintain an up-to-date allergy action plan with your healthcare provider, including emergency medications and allergen avoidance strategies.
- Carry epinephrine auto-injector (e.g., EpiPen) for severe allergic reactions and ensure easy access in case of an emergency.

8. Regular Follow-ups:

- Schedule appointments with your

14.VIDEO LINK:

https://drive.google.com/file/d/19IDLZoK_Yyeonl2skCNnR3VFXBfgrU7J/view?usp=drivesdk

THANK YOU

