!pip install gradio torch transformers accelerate sentencepiece

import gradio as gr

import torch

from transformers import AutoTokenizer, AutoModelForCausalLM

# ---------------------------

# 🔹 Load Model & Tokenizer

# ---------------------------

model\_name = "ibm-granite/granite-3.2-2b-instruct"

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

)

if tokenizer.pad\_token is None:

    tokenizer.pad\_token = tokenizer.eos\_token

# ---------------------------

# 🔹 Helper Function

# ---------------------------

def generate\_response(prompt, max\_length=512):

    inputs = tokenizer(prompt, return\_tensors="pt", truncation=True, max\_length=512)

    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,

            temperature=0.7,

            do\_sample=True,

            pad\_token\_id=tokenizer.eos\_token\_id

        )

        generated\_tokens = outputs[0][inputs["input\_ids"].shape[-1]:]

        response = tokenizer.decode(generated\_tokens, skip\_special\_tokens=True).strip()

        return response

# ---------------------------

# 🔹 Features

# ---------------------------

def concept\_explanation(concept):

    prompt = f"Explain the concept of {concept} in detail with example."

    response = generate\_response(prompt, max\_length=800)

    return "📘 \*\*Explanation:\*\*\n\n" + response + "\n\n---\n✍️ Created by EduTutor\_AI team"

def quiz\_generator(concept):

    prompt = f"Generate 5 quiz questions about {concept} with different question types (multiple choice, true/false, short answer). Provide answers at the end."

    response = generate\_response(prompt, max\_length=1200)

    return "📝 \*\*Quiz:\*\*\n\n" + response + "\n\n---\n✍️ Created by EduTutor\_AI team"

def quiz\_analysis(answers):

    prompt = f"Analyze these student answers and give feedback with strengths and improvements:\n\n{answers}"

    response = generate\_response(prompt, max\_length=800)

    return "📊 \*\*Performance Analysis:\*\*\n\n" + response + "\n\n---\n✍️ Created by EduTutor\_AI team"

# ---------------------------

# 🎨 Gradio UI

# ---------------------------

with gr.Blocks(theme=gr.themes.Soft(primary\_hue="indigo", secondary\_hue="cyan")) as app:

    # Header Banner

    gr.Markdown(

        """

        # 🎓 EduTutor AI Assistant

        🚀 \*Learn Smarter. Practice Better. Analyze Progress.\*

        """

    )

    with gr.Tabs():

        # Tab 1 - Concept Explanation

        with gr.TabItem("📘 Concept Explanation"):

            with gr.Row():

                with gr.Column(scale=1):

                    concept\_input = gr.Textbox(label="Enter a Concept", placeholder="e.g., Machine Learning")

                    explain\_btn = gr.Button("✨ Explain Concept", variant="primary")

                with gr.Column(scale=2):

                    explanation\_output = gr.Textbox(label="AI Explanation", lines=12, interactive=False, show\_copy\_button=True)

            explain\_btn.click(concept\_explanation, inputs=concept\_input, outputs=explanation\_output)

        # Tab 2 - Quiz Generator

        with gr.TabItem("📝 Quiz Generator"):

            with gr.Row():

                with gr.Column(scale=1):

                    quiz\_input = gr.Textbox(label="Enter a Topic", placeholder="e.g., Photosynthesis")

                    quiz\_btn = gr.Button("🎯 Generate Quiz", variant="primary")

                with gr.Column(scale=2):

                    quiz\_output = gr.Textbox(label="Quiz Questions & Answers", lines=15, interactive=False, show\_copy\_button=True)

            quiz\_btn.click(quiz\_generator, inputs=quiz\_input, outputs=quiz\_output)

        # Tab 3 - Student Performance

        with gr.TabItem("📊 Student Performance"):

            with gr.Row():

                with gr.Column(scale=1):

                    answers\_input = gr.Textbox(label="Paste Student Answers", placeholder="Enter quiz answers here...", lines=8)

                    analyze\_btn = gr.Button("🔍 Analyze Performance", variant="primary")

                with gr.Column(scale=2):

                    analysis\_output = gr.Textbox(label="Feedback & Suggestions", lines=12, interactive=False, show\_copy\_button=True)

            analyze\_btn.click(quiz\_analysis, inputs=answers\_input, outputs=analysis\_output)

    # Footer

    gr.Markdown("---")

    gr.Markdown("🚀 \*\*AI App created by EduTutor\_AI team\*\* | 🔗 Powered by Granite LLM & Gradio")

# Launch the app

app.launch(share=True)