Project 99: System vs. User Prompt Split Test

Description:

This tool compares how a model responds when guidance is given as a **system message** vs as a **user prompt**. It helps you explore how the placement of instructions (as a role vs task) affects the final output.

---

system_vs_user_prompt_test.py

```python
import os
import openai
import gradio as gr

# Load OpenAI API key
openai.api_key = os.getenv("OPENAI_API_KEY")

# Function to compare system and user prompt styles
def split_test(instruction, task_input):
    try:
        # System prompt method
        response_sys = openai.ChatCompletion.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "system", "content": instruction},
                {"role": "user", "content": task_input}
            ]
        )

        # User prompt method
        combined_prompt = f"{instruction}\n\nNow complete this task:\n{task_ir
        response_usr = openai.ChatCompletion.create(
            model="gpt-3.5-turbo",
            messages=[
                {"role": "user", "content": combined_prompt}
            ]
        )

        sys_output = response_sys["choices"][0]["message"]["content"].strip()
        usr_output = response_usr["choices"][0]["message"]["content"].strip()

        return sys_output, usr_output

    except Exception as e:
        return f"Error: {str(e)}", ""

# Gradio interface
iface = gr.Interface(
    fn=split_test,
    inputs=[
        gr.Textbox(label="Instruction (e.g. Respond like a Shakespearean poet)
        gr.Textbox(label="Task Input (e.g. Describe the moon)", lines=2)
    ],
    outputs=[
        gr.Textbox(label="System Prompt Result", lines=6),
        gr.Textbox(label="User Prompt Result", lines=6)
    ],
    title="🔀 System vs User Prompt Split Test",
    description="Test if placing your instruction as a system message vs user
)

# Launch the experiment
iface.launch()
```