Project 82: Chat with Website Content (Scraped Once)

Description:
This project lets users enter a website URL, scrapes the visible content once, and enables Q&A over that content. It's like turning a static webpage into an interactive chatbot. Perfect for product pages, blogs, or documentation.

---

chat_with_website.py

```
import os
import openai
import gradio as gr
import requests
from bs4 import BeautifulSoup
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
import numpy as np

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

# Load embedding model
embedding_model = SentenceTransformer("all-MiniLM-L6-v2")

# Function to scrape website content
def scrape_website(url):
    try:
        response = requests.get(url)
        soup = BeautifulSoup(response.text, "html.parser")
        paragraphs = soup.find_all(["p", "li"])
        text = " ".join([para.get_text() for para in paragraphs])
        return text[:5000]  # Limit for practical embedding
    except Exception as e:
        return f"Error scraping website: {str(e)}"

# Split text into chunks
def chunk_text(text, chunk_size=500):
    return [text[i:i+chunk_size] for i in range(0, len(text), chunk_size)]
```

```python
# Embed chunks and search
def search_website_chunks(query, chunks):
    query_vec = embedding_model.encode([query])[0]
    chunk_vecs = embedding_model.encode(chunks)
    similarities = cosine_similarity([query_vec], chunk_vecs)[0]
    top_indices = np.argsort(similarities)[-3:][::-1]
    return [chunks[i] for i in top_indices]

# Main Q&A function
def chat_with_site(url, query):
    text = scrape_website(url)
    if text.startswith("Error"):
        return text

    chunks = chunk_text(text)
    top_chunks = search_website_chunks(query, chunks)
    context = "\n\n".join(top_chunks)

    prompt = (
        f"You are an assistant answering questions based on the content of thi
        f"Question: {query}\nAnswer:"
    )

    try:
        response = openai.ChatCompletion.create(
            model="gpt-3.5-turbo",
            messages=[{"role": "user", "content": prompt}]
        )
        return response["choices"][0]["message"]["content"].strip()
    except Exception as e:
        return f"OpenAI Error: {str(e)}"

# Gradio UI
iface = gr.Interface(
    fn=chat_with_site,
    inputs=[
        gr.Textbox(label="Website URL (scraped once)"),
        gr.Textbox(label="Ask a question about the website")
    ],
    outputs="text",
    title="🌐 Chat with Website Content",
    description="Enter a website URL and ask questions about its visible conte
```

```
)

# Launch
iface.launch()
```