Project 87: One-Page Handbook Q&A

Description:
This app allows users to upload a short one-page handbook (in `.txt`, `.md`, or `.pdf` format) and ask questions about it. Whether it's a cheat sheet, internal guide, or FAQ, the assistant reads and responds based on the content.

`one_page_handbook_qa.py`

```python
import os
import openai
import gradio as gr
import PyPDF2

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

# Extract text from file (.txt, .md, .pdf supported)
def extract_text(file):
    filename = file.name
    if filename.endswith(".txt") or filename.endswith(".md"):
        return file.read().decode("utf-8")
    elif filename.endswith(".pdf"):
        pdf_reader = PyPDF2.PdfReader(file)
        return "\n".join(page.extract_text() for page in pdf_reader.pages)
    else:
        return "Unsupported file type."

# Generate answer using handbook content
def chat_with_handbook(file, question):
    try:
        handbook_text = extract_text(file)

        prompt = (
            f"You are an expert assistant who answers questions based only on
            f"Handbook:\n{handbook_text[:4000]}\n\n"  # limit input size
            f"Question: {question}\nAnswer:"
        )
        # Generate response
        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"Error: {str(e)}"

# Gradio interface
iface = gr.Interface(
    fn=chat_with_handbook,
    inputs=[
        gr.File(label="Upload Handbook (.txt, .md, or .pdf)", file_types=[".tx
        gr.Textbox(label="Ask a Question")
    ],
    outputs="text",
    title="📘 One-Page Handbook Q&A",
    description="Upload a short guide or handbook and chat with it. Great for
)

# Launch app
iface.launch()
```