03 LLMs-Code - KirkYagami 👨‍💻🕵️

Store the APIs in `.env` file:

```
ANTHROPIC_API_KEY=sk-ant-api03-15UuxL2CNHBsqwOCBF0Q5QhGIsymax8RzirKQm
```

Anthropic - Claude

```
pip install anthropic
pip install python-dotenv
```

`claude_main.py:`

```python
import os
from anthropic import Anthropic
from dotenv import load_dotenv


load_dotenv()
client = Anthropic(api_key=os.getenv('ANTHROPIC_API_KEY'))

def estimate_token(prompt):
    count = client.count_tokens(prompt)
    return count


def submit_prompt(prompt, system_prompt):
    try:
        full_text = []
        with client.messages.stream(
            model='claude-2.1',
            system=system_prompt,
            max_tokens=1024,
            messages=[
                {"role": "user", "content": prompt}
            ]
        ) as stream:
            for text in stream.text_stream:
                full_text.append(text)
        print(''.join(full_text))
    except Exception as e:
        print(f"Error in submit_prompt: {e}")
        raise e
```

```python
if __name__=="__main__":
    prompt = input("Enter your prompt: ")
    print(f"Tokens for this request: {estimate_token(prompt=prompt)}")

    system_prompt = """
    Your task is to take the code snippet provided and explain it in simple, easy-to-under
    """
    submit_prompt(prompt, system_prompt)
```

Llama - Deep Infra

https://deepinfra.com/ : Go to dashboard and create your API keys.
Store the APIs in `.env` file:

```
DEEP_INFRA_API_KEY=3Cp39e34BUzTmbeRML6P6k
```

CLI-based :

`llama11b.py`

```python
from openai import OpenAI
import os
from dotenv import load_dotenv


# Load environment variables from the .env file
load_dotenv()


# Create an OpenAI client with your Deepinfra token and endpoint
openai = OpenAI(
    api_key=os.getenv('DEEP_INFRA_API_KEY'),
    base_url="https://api.deepinfra.com/v1/openai",
)

def generate_code(prompt):
    # Create a chat completion request to generate code based on the prompt
    chat_completion = openai.chat.completions.create(
        model="meta-llama/Llama-3.2-11B-Vision-Instruct",
```

```
        messages=[
            {
                "role": "system",
                "content": "You are a code generator. Write Python code based on the user'
            },
            {"role": "user", "content": prompt},
        ],
    )

    return chat_completion.choices[0].message.content, chat_completion.usage.prompt_tokens

if __name__ == "__main__":
    prompt = input("Enter your coding request: ")
    generated_code, prompt_tokens, completion_tokens = generate_code(prompt)

    print("Generated Code:\n")
    print(generated_code)
    print(f"Prompt tokens used: {prompt_tokens}, Completion tokens used: {completion_token
```

Web-based

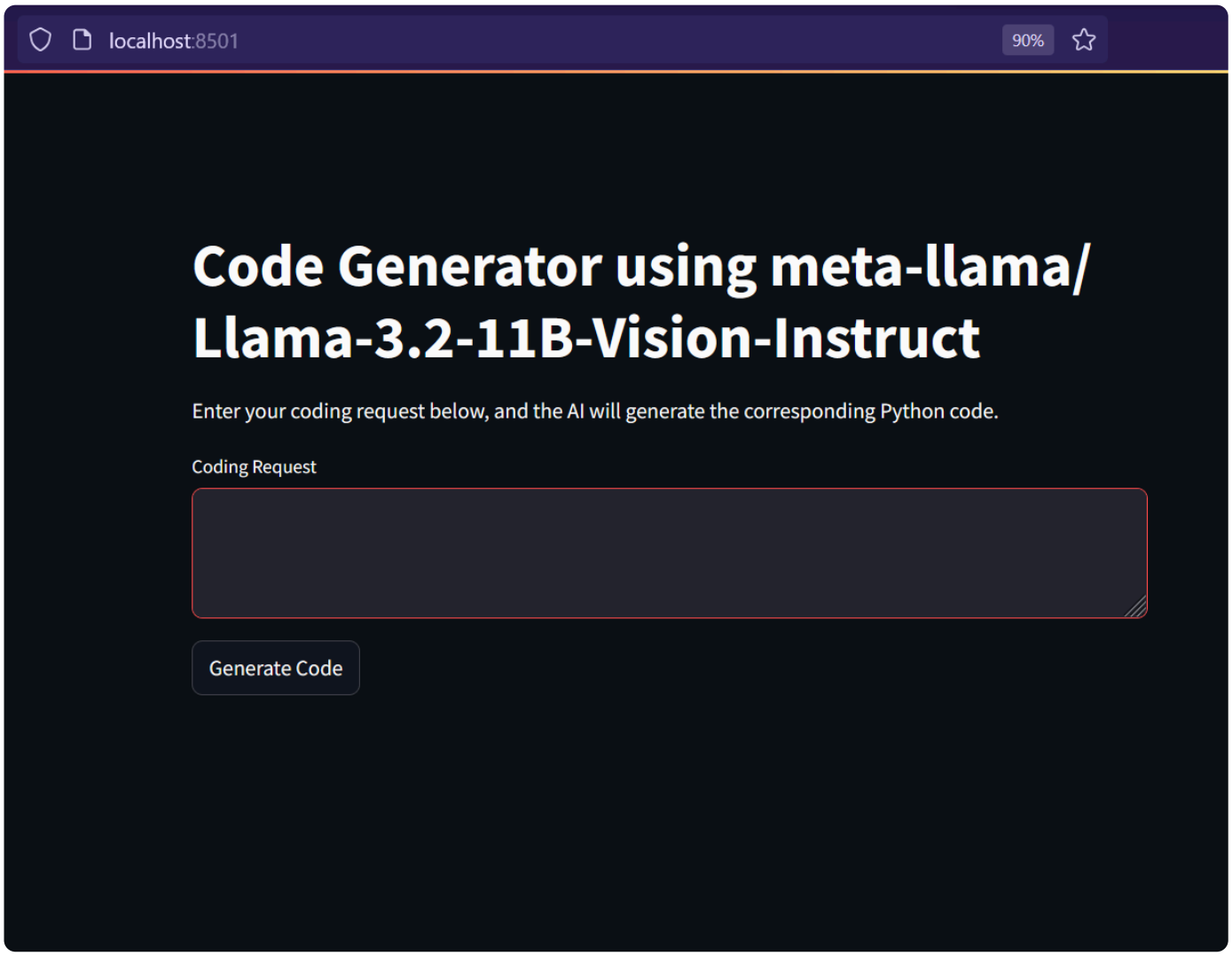
`code-gen.py:`

```python
import os
import streamlit as st
from openai import OpenAI
from dotenv import load_dotenv


load_dotenv()

openai = OpenAI(
    api_key=os.getenv('DEEP_INFRA_API_KEY'),
    base_url="https://api.deepinfra.com/v1/openai",
)

def generate_code(prompt):
    # Create a chat completion request to generate code based on the prompt
    chat_completion = openai.chat.completions.create(
        model="meta-llama/Llama-3.2-11B-Vision-Instruct",
        messages=[
            {
                "role": "system",
                "content": "You are a code generator. Write Python code based on the user'
            },
```

```
            {"role": "user", "content": prompt},
        ],
    )

    return chat_completion.choices[0].message.content, chat_completion.usage.prompt_tokens

# Streamlit app
def main():
    st.title("Code Generator using meta-llama/Llama-3.2-11B-Vision-Instruct")
    st.write("Enter your coding request below, and the AI will generate the corresponding

    prompt = st.text_area("Coding Request", height=50)

    if st.button("Generate Code"):
        if prompt:
            with st.spinner("Generating code..."):
                generated_code, prompt_tokens, completion_tokens = generate_code(prompt)
                st.subheader("Generated Code:")
                # Format the generated code using Markdown
                st.markdown(f"python\n{generated_code}\n")
                st.write(f"**Prompt tokens used:** {prompt_tokens}, **Completion tokens us
        else:
            st.warning("Please enter a coding request.")

if __name__ == "__main__":
    main()
```

Execute the above file as:

```
streamlit run code-gen.py
```


localhost:8501
90%
Code Generator using meta-llama/ Llama-3.2-11B-Vision-Instruct
Enter your coding request below, and the AI will generate the corresponding Python code.
Coding Request
Generate Code