Local LLM Interface App

1. Objective

The goal of this project is to **install a local LLM** (Large Language Model) and interact with it via a **Streamlit web interface**. Users can input prompts and observe AI-generated responses in real time, while also comparing performance metrics like response time.

2. Instructions & Implementation

Step 1: Install and Setup Local LLM

1. Install a local LLM such as **GPT-2** or **Ollama** (e.g., LLaMA 3.2).
2. Ensure **Python**, **PyTorch**, and **Transformers** library are installed.
3. Verify GPU availability, if possible (torch.cuda.is_available()), otherwise use CPU.

Step 2: Run Streamlit App

- Save the provided script as streamlit_local_llm_interactive.py.
- Launch the app using:

```bash
streamlit run streamlit_local_llm_interactive.py
```

In the sidebar, select your model (e.g., GPT-2) and verify that the device is detected correctly.

Step 3: Test Prompt Generation

- Enter a prompt.
- Click **Generate** 🤖.
- The app will display:
  - Generated text
  - Response time
- Example of token generation:

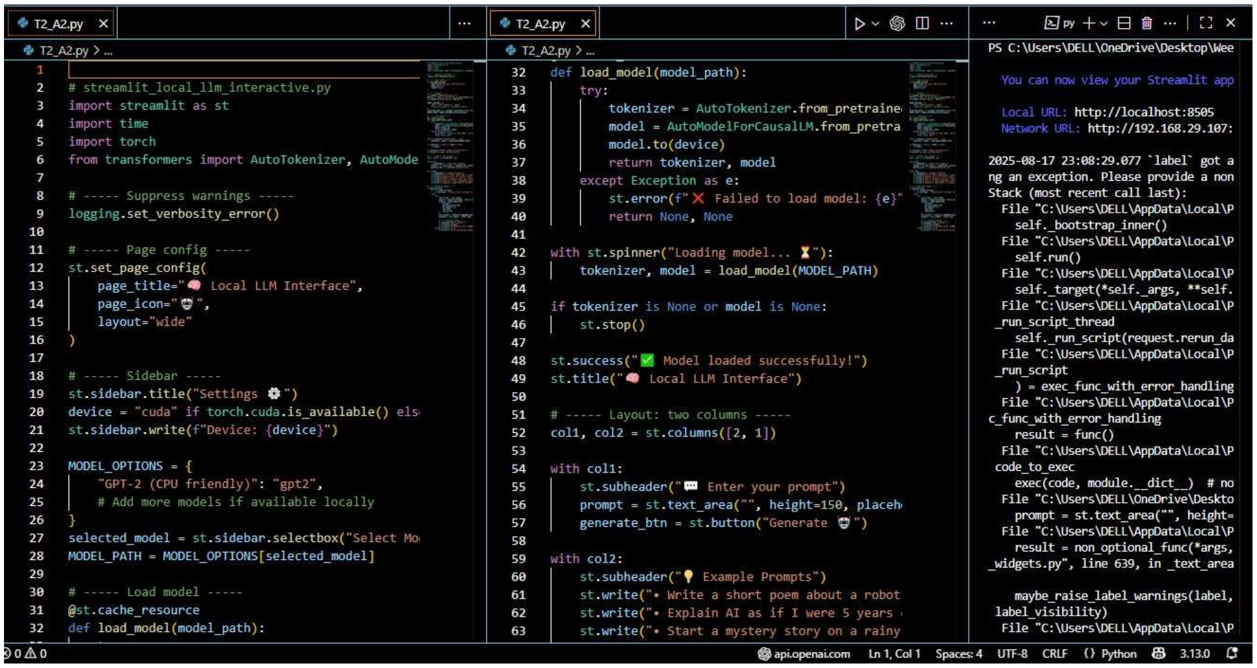
```python
inputs = tokenizer(prompt, return_tensors="pt").to(device)
outputs = model.generate(**inputs, max_new_tokens=150, do_sample=True, top_k=50, top_p=0.95)
text_output = tokenizer.decode(outputs[0], skip_special_tokens=True)
```

Step 4: Troubleshooting

- If the model fails to load, check:
    - Correct model path
    - PyTorch and Transformers versions
    - GPU availability
- If generation fails:
    - Reduce max_new_tokens
    - Disable sampling parameters (top_k, top_p) for smaller models

3. Deliverables

- Streamlit App: Interactive interface for real-time.

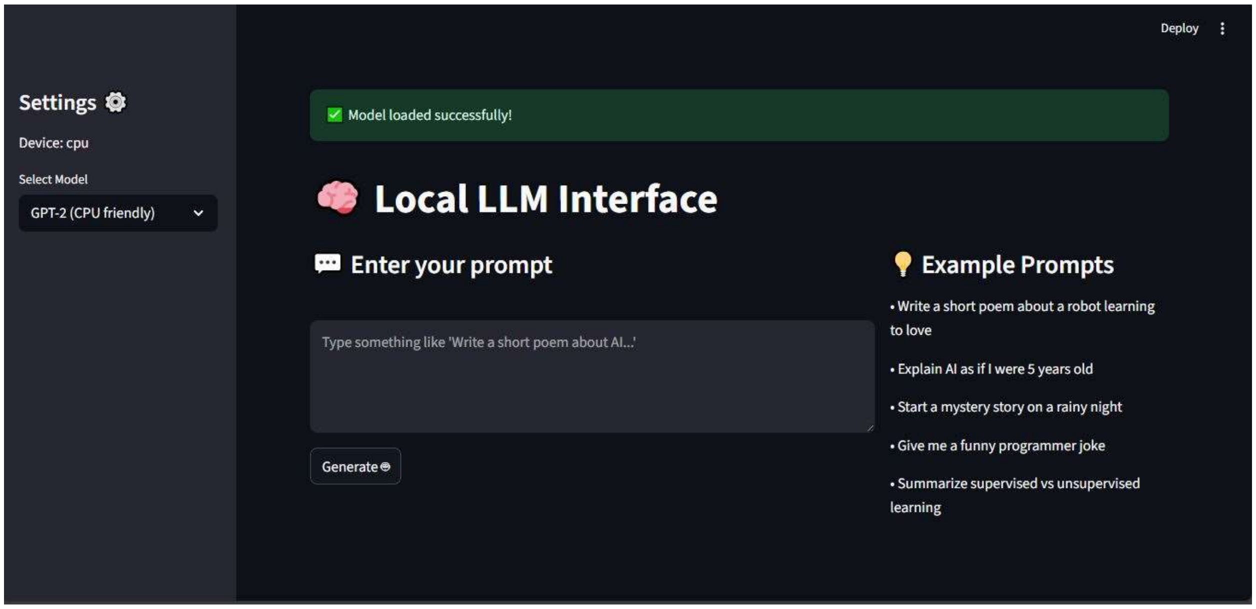


```python

# streamlit_local_llm_interactive.py
import streamlit as st
import time
import torch
from transformers import AutoTokenizer, AutoMode

# ----- Suppress warnings -----
logging.set_verbosity_error()

# ----- Page config -----
st.set_page_config(
    page_title="🧠 Local LLM Interface",
    page_icon="🤖",
    layout="wide"
)

# ----- Sidebar -----
st.sidebar.title("Settings ⚙️")
device = "cuda" if torch.cuda.is_available() els
st.sidebar.write(f"Device: {device}")

MODEL_OPTIONS = {
    "GPT-2 (CPU friendly)": "gpt2",
    # Add more models if available locally
}
selected_model = st.sidebar.selectbox("Select Mo
MODEL_PATH = MODEL_OPTIONS[selected_model]

# ----- Load model -----
@st.cache_resource
def load_model(model_path):
    try:
        tokenizer = AutoTokenizer.from_pretraine
        model = AutoModelForCausalLM.from_pretra
        model.to(device)
        return tokenizer, model
    except Exception as e:
        st.error(f"❌ Failed to load model: {e}"
        return None, None

with st.spinner("Loading model... ⏳"):
    tokenizer, model = load_model(MODEL_PATH)

if tokenizer is None or model is None:
    st.stop()

st.success("✅ Model loaded successfully!")
st.title("🧠 Local LLM Interface")

# ----- Layout: two columns -----
col1, col2 = st.columns([2, 1])

with col1:
    st.subheader("💬 Enter your prompt")
    prompt = st.text_area("", height=150, placeh
    generate_btn = st.button("Generate 🤖")

with col2:
    st.subheader("💡 Example Prompts")
    st.write("• Write a short poem about a robot
    st.write("• Explain AI as if I were 5 years
    st.write("• Start a mystery story on a rainy
```

```
PS C:\Users\DELL\OneDrive\Desktop\Wee

  You can now view your Streamlit app

  Local URL: http://localhost:8505
  Network URL: http://192.168.29.107:

2025-08-17 23:08:29.077 `label` got a
ng an exception. Please provide a non
Stack (most recent call last):
  File "C:\Users\DELL\AppData\Local\P
    self._bootstrap_inner()
  File "C:\Users\DELL\AppData\Local\P
    self.run()
  File "C:\Users\DELL\AppData\Local\P
    self._target(*self._args, **self.
  File "C:\Users\DELL\AppData\Local\P
 _run_script_thread
    self._run_script(request.rerun_da
  File "C:\Users\DELL\AppData\Local\P
 _run_script
    ) = exec_func_with_error_handling
  File "C:\Users\DELL\AppData\Local\P
c_func_with_error_handling
    result = func()
  File "C:\Users\DELL\AppData\Local\P
 code_to_exec
    exec(code, module.__dict__)  # no
  File "C:\Users\DELL\OneDrive\Deskto
    prompt = st.text_area("", height=
  File "C:\Users\DELL\AppData\Local\P
    result = non_optional_func(*args,
_widgets.py", line 639, in _text_area

    maybe_raise_label_warnings(label,
 label_visibility)
  File "C:\Users\DELL\AppData\Local\P
```

Fig1: Input Snapshot

4. Output


Settings

Device: cpu

Select Model: GPT-2 (CPU friendly)

Model loaded successfully!

Local LLM Interface

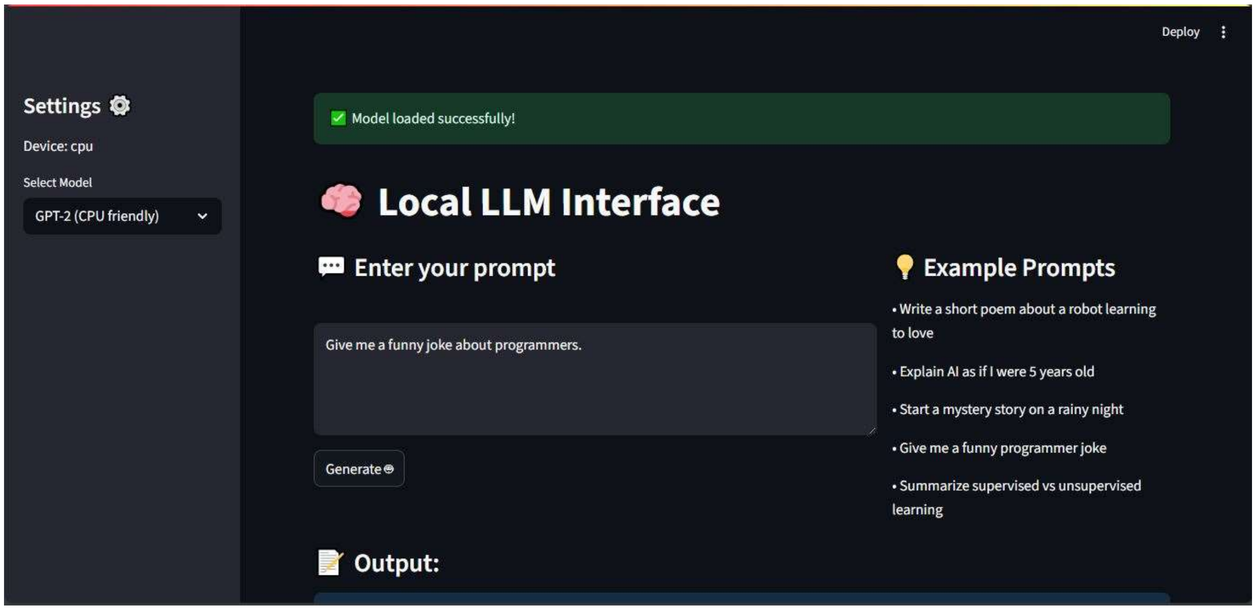
Enter your prompt

Type something like 'Write a short poem about AI...'

Example Prompts

- Write a short poem about a robot learning to love
- Explain AI as if I were 5 years old
- Start a mystery story on a rainy night
- Give me a funny programmer joke
- Summarize supervised vs unsupervised learning

Fig2: Output Snapshot1


Settings

Device: cpu

Select Model: GPT-2 (CPU friendly)

Model loaded successfully!

Local LLM Interface

Enter your prompt

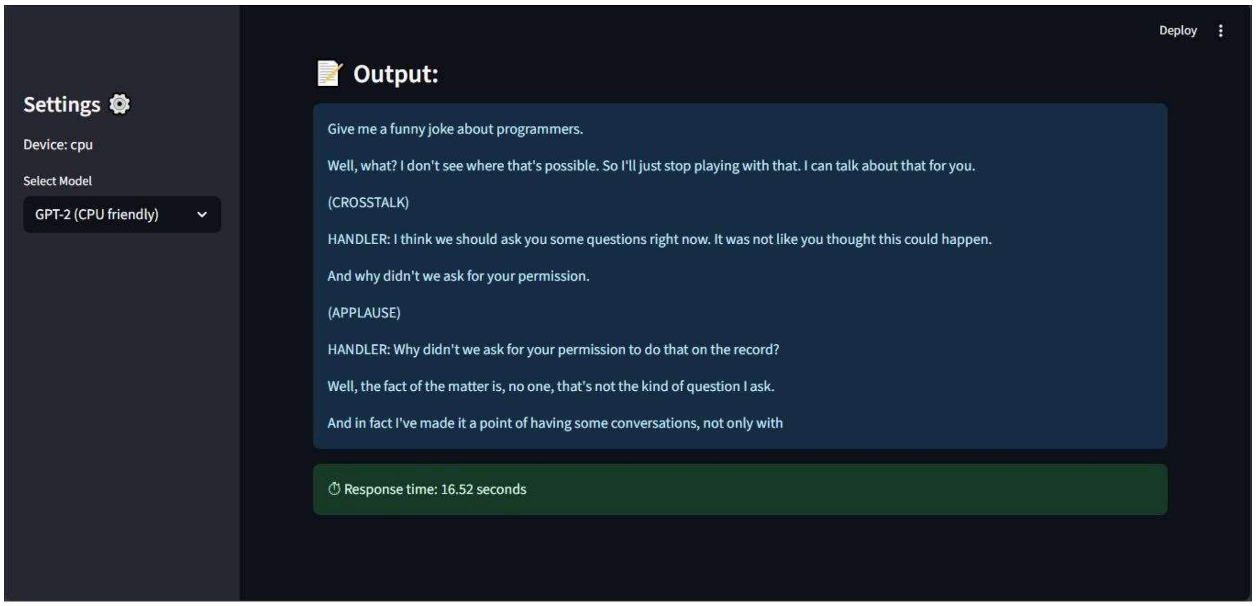
Give me a funny joke about programmers.

Example Prompts

- Write a short poem about a robot learning to love
- Explain AI as if I were 5 years old
- Start a mystery story on a rainy night
- Give me a funny programmer joke
- Summarize supervised vs unsupervised learning

Output:

Fig3: Output Snapshot2



Device: cpu

GPT-2 (CPU friendly)

📝 Output:

Give me a funny joke about programmers.

Well, what? I don't see where that's possible. So I'll just stop playing with that. I can talk about that for you.

(CROSSTALK)

HANDLER: I think we should ask you some questions right now. It was not like you thought this could happen.

And why didn't we ask for your permission.

(APPLAUSE)

HANDLER: Why didn't we ask for your permission to do that on the record?

Well, the fact of the matter is, no one, that's not the kind of question I ask.

And in fact I've made it a point of having some conversations, not only with

⏱ Response time: 16.52 seconds

Fig4: Output Snapshot3