

Ocean Poem Generator — Project Documentation

1. Project Overview

The **Ocean Poem Generator** is a simple web application built using Streamlit and the Hugging Face Transformers library. It generates poetic text based on user input prompts by leveraging a pretrained text generation model (bigscience/bloom-560m). This app provides a creative interface where users can enter prompts and receive generated poems, focusing here on ocean-themed poetry by default.

2. Features

- User-friendly web interface with Streamlit.
- Text generation powered by the bigscience/bloom-560m language model.
- Prompt input area with a default suggestion to write ocean-related poems.
- Caching of the model loading to optimize response times.
- Display of generated poem in a clear, readable format.

3. How It Works

1. The app loads the bigscience/bloom-560m model once and caches it for efficient reuse.
2. User enters a prompt in the text area.
3. When the "Generate Poem" button is clicked, the prompt is sent to the model to generate a poem.
4. The generated text is displayed under the prompt in a markdown block.

Install dependencies:

```
bash

pip install streamlit transformers
```

Run:

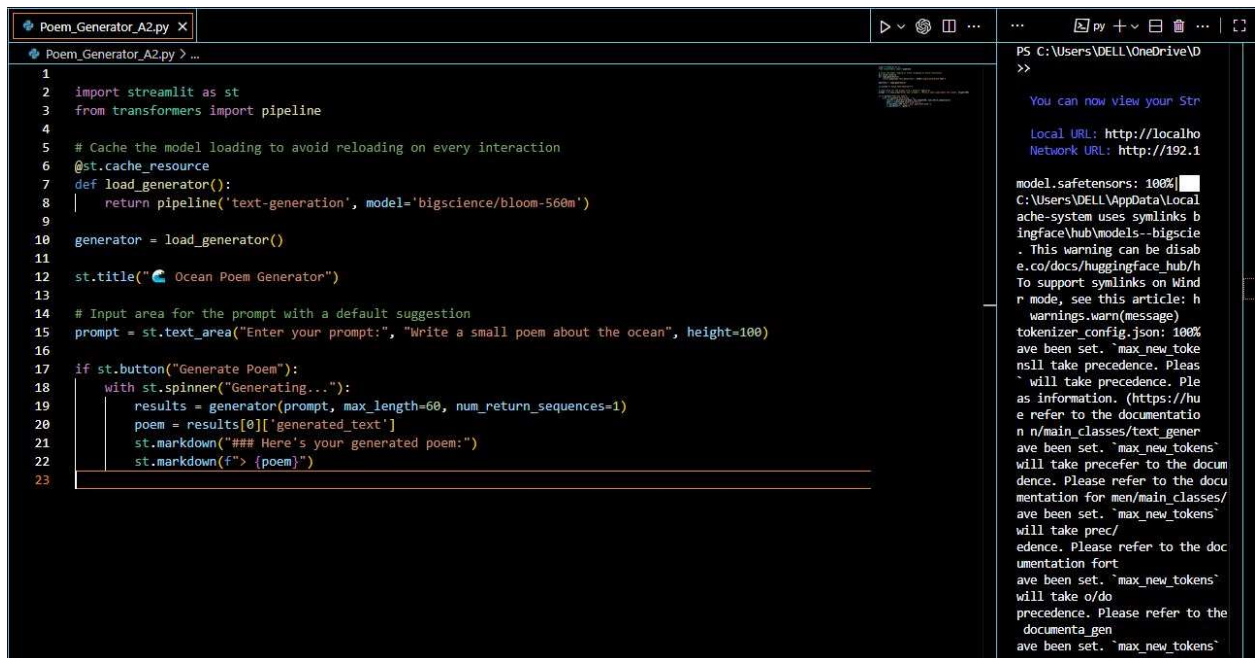
```
bash

streamlit run app.py
```

4. Usage Instructions

1. Open the web app in your browser (usually at `http://localhost:8501`).
2. Enter a text prompt or use the default prompt provided.
3. Click the "Generate Poem" button.
4. Wait a moment while the poem is generated.
5. View the generated poem below the button.

5. Code Explanation



```
1 import streamlit as st
2 from transformers import pipeline
3
4 # Cache the model loading to avoid reloading on every interaction
5 @st.cache_resource
6 def load_generator():
7     return pipeline('text-generation', model='bigscience/bloom-560m')
8
9 generator = load_generator()
10
11 st.title("🌊 Ocean Poem Generator")
12
13 # Input area for the prompt with a default suggestion
14 prompt = st.text_area("Enter your prompt:", "Write a small poem about the ocean", height=100)
15
16 if st.button("Generate Poem"):
17     with st.spinner("Generating..."):
18         results = generator(prompt, max_length=60, num_return_sequences=1)
19         poem = results[0]['generated_text']
20         st.markdown("### Here's your generated poem:")
21         st.markdown(f"> {poem}")
```

Fig1: Input Snapshot

- `load_generator()` loads the Bloom model and caches it using Streamlit's `@st.cache_resource` decorator.
- The app UI consists of a title, a text area for user input, and a button to generate the poem.
- When the button is clicked, the text generation pipeline is called with the user prompt.
- The generated poem is extracted and displayed using markdown.

6. Output Snapshots

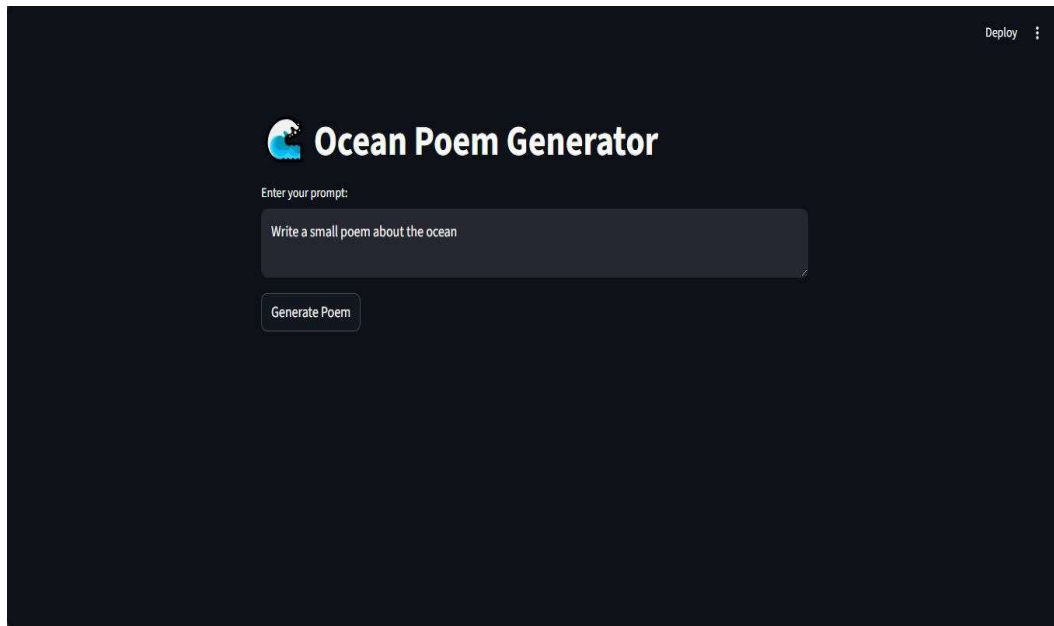


Fig2: Output Snapshot 1

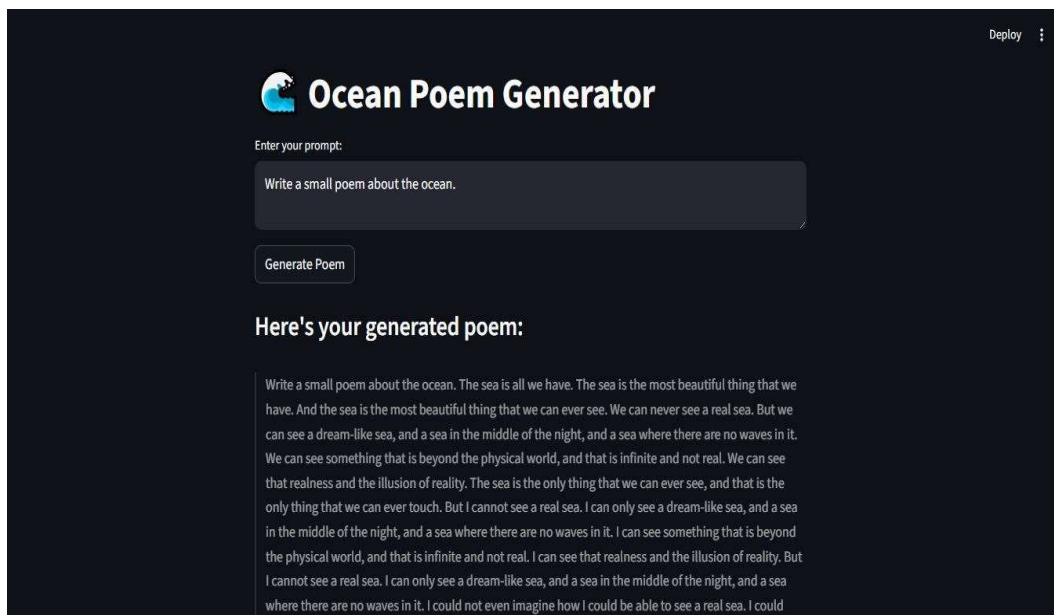


Fig3: Output Snapshot 2