Dream Text Forge – AI based Story Generator

1. Project Overview

DreamText Forge is an interactive web app built with **Streamlit** that lets you turn your ideas into AI-generated text magic. You can choose from curated story prompts or create your own, then watch the AI weave them into creative narratives.

2. Features

- **Prompt Suggestions** — Start with one of the built-in story starters.
- **Custom Prompt Input** — Write your own ideas from scratch.
- **Adjustable Output Length** — Fine-tune the maximum length of generated text.
- **Real-Time Generation** — Get AI-generated text instantly.
- **Lightweight UI** — Clean and easy-to-use Streamlit interface.
- **Cached Model Loading** — Faster performance thanks to caching.

3. How It Works

i. Model Loading

a. When you first start the app, it loads the GPT-2 model from Hugging Face's transformers library using the pipeline("text-generation") API.

b. This is wrapped in @st.cache_resource so the model stays in memory between runs, making subsequent generations much faster.

ii. User Prompt Selection

a. You can either:

i. Pick from one of the built-in prompt suggestions (story starters), or

ii. Write your own from scratch.

b. The selected or written prompt is editable before submission.

iii. Parameter Configuration

a. You can control how long the generated text will be using the Max Length slider.

b. In the code, other parameters like temperature (creativity) and repetition_penalty (avoid repeated phrases) are preset but can be changed in the source.

iv. Text Generation

a. When you click Generate, the app sends your prompt to the GPT-2 model.

b. GPT-2 predicts the next words in the sequence until it reaches the max_length or the end-of-sequence token.

v. Result Display

a. The generated text is displayed immediately below in a clean, scrollable format.

b. If no prompt is entered, a warning is shown.

vi. Caching for Speed

a. Because load_model() is cached, the heavy GPT-2 model is only loaded once, greatly reducing response time after the first generation.

4. Code Explanation

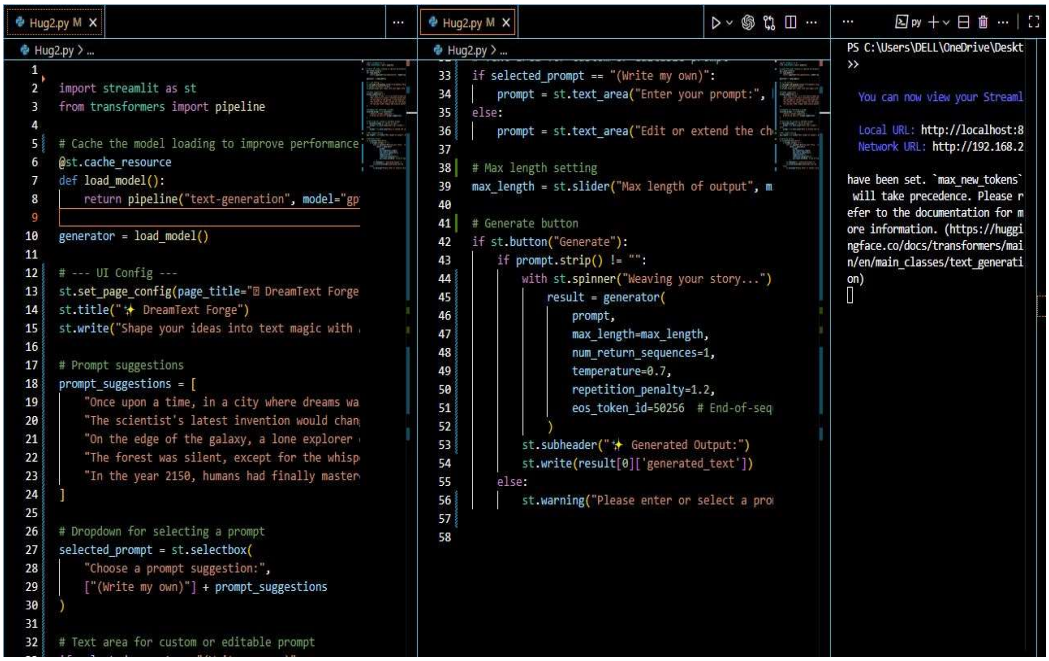

Fig1: Input Snapshot

5. Output Snapshots

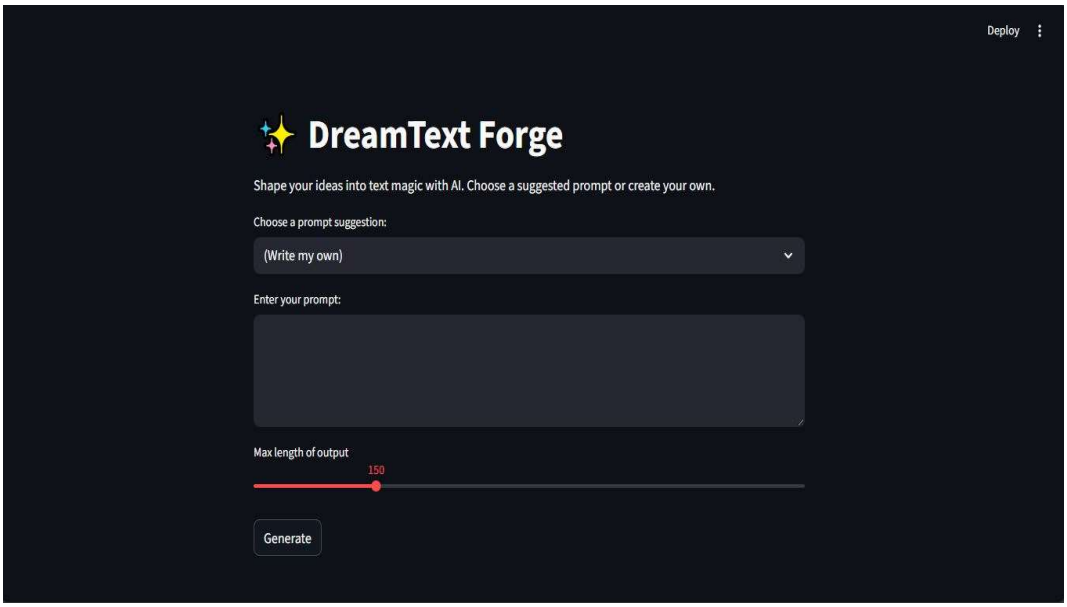

Fig2: Output Snapshot 1

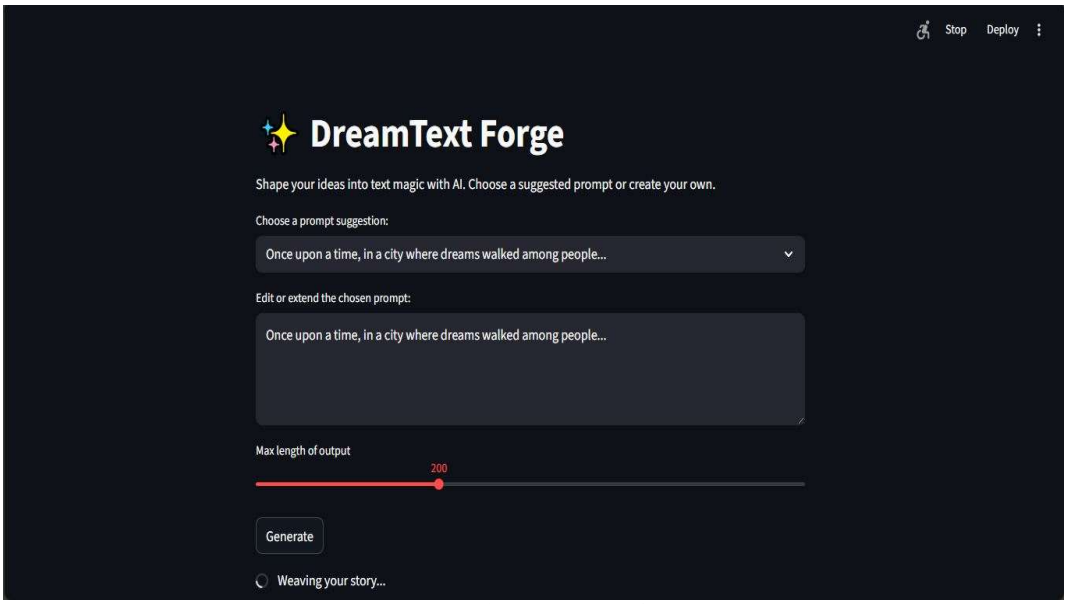

Fig3: Output Snapshot 2

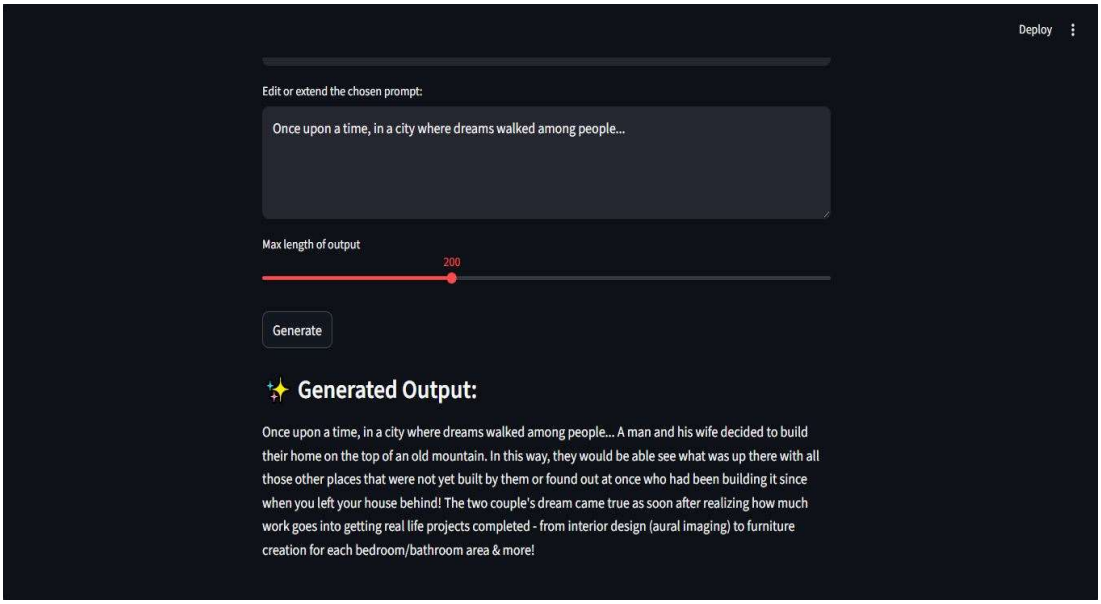

Fig4: Output Snapshot 3