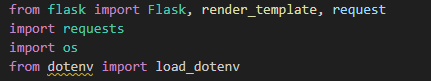
Story Generator App Report

# 1. Introduction

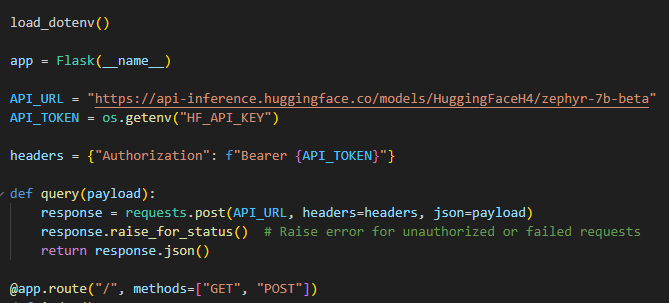
This project is a web application built with Flask that generates custom stories using Hugging Face's Zephyr-7B language model. Users can input a story starter, genre, tone, setting, and characters to receive a unique AI-generated story.

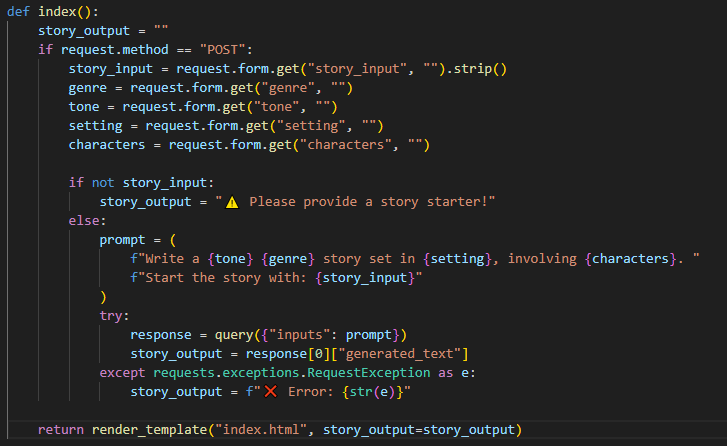
# 2. Libraries and Tools Used

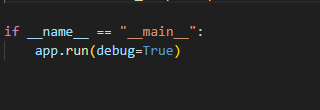
- Flask: A lightweight Python web framework used to build the backend server  
- requests: To make API calls to the Hugging Face inference endpoint  
- python-dotenv: To securely load the Hugging Face API key from a `.env` file  
- Hugging Face Zephyr-7B Model: A generative language model hosted on Hugging Face



# 3. Application Workflow

1. The API key is loaded securely from an environment file using `load\_dotenv()`.  
2. The Flask app defines a route (`/`) that accepts both GET and POST requests.  
3. When a user submits a story prompt along with additional fields like tone and genre, a custom prompt is formed.  
4. The prompt is sent to the Zephyr-7B model via Hugging Face API using the `requests` library.  
5. The generated story is returned and displayed to the user on the HTML frontend.





# 4. Code Explanation

**Environment Setup:**

Environment variables (like the API key) are loaded using `dotenv` for secure access.

**API Interaction:**

A POST request is made to Hugging Face's inference API with a formatted prompt, and the generated response is extracted.

**User Input and Prompt Formation:**

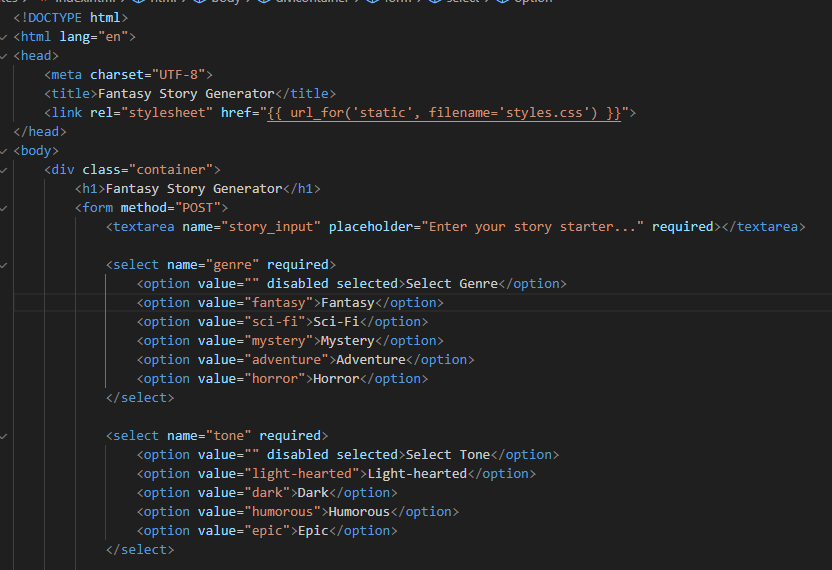
The app collects inputs like genre, tone, setting, characters, and starter text from an HTML form to form a rich prompt.

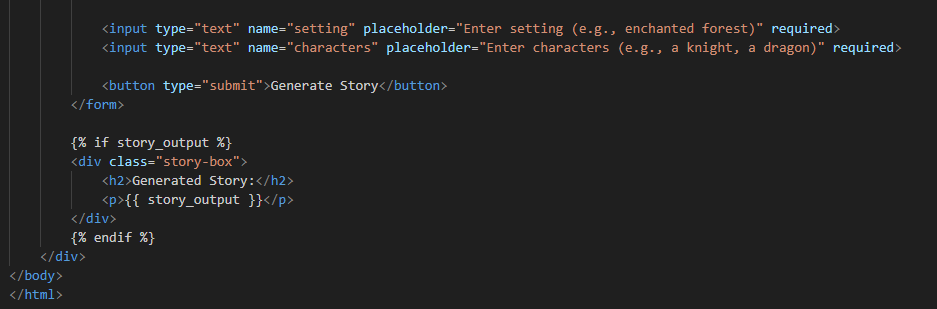
**Error Handling:**

The app gracefully handles missing inputs and request failures with informative messages.

# 5. Frontend (HTML)

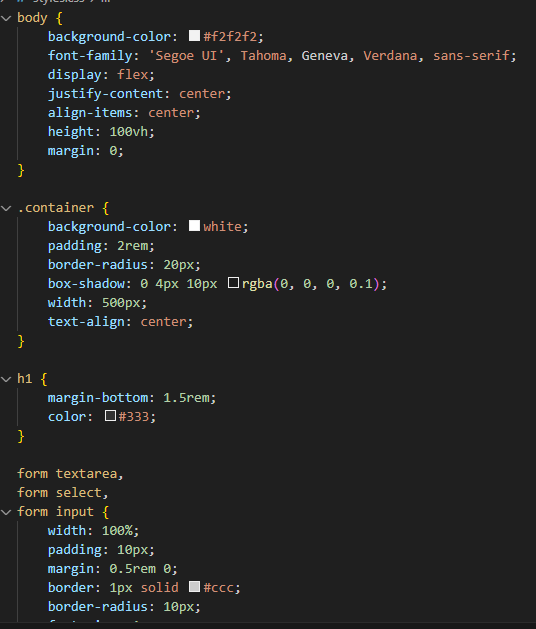
The app uses a basic `index.html` template to allow users to input story parameters and view generated results.

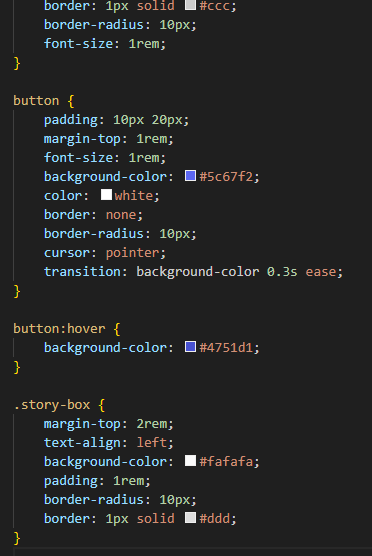




# 5. Frontend (CSS)

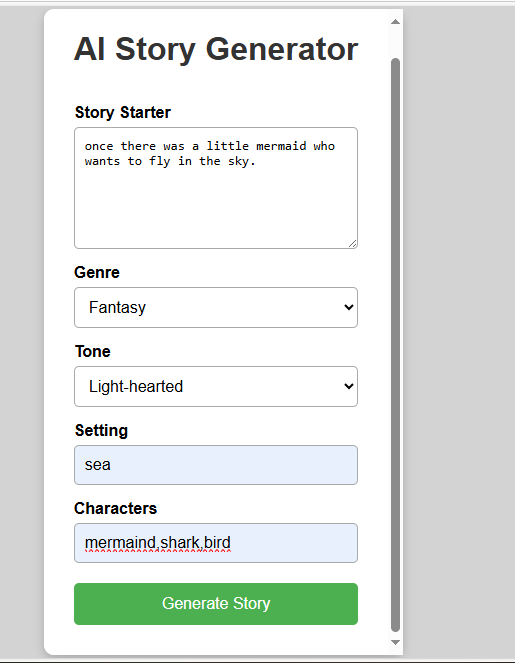
The app uses a basic `style.css` template to enhance input story parameters and view generated results.

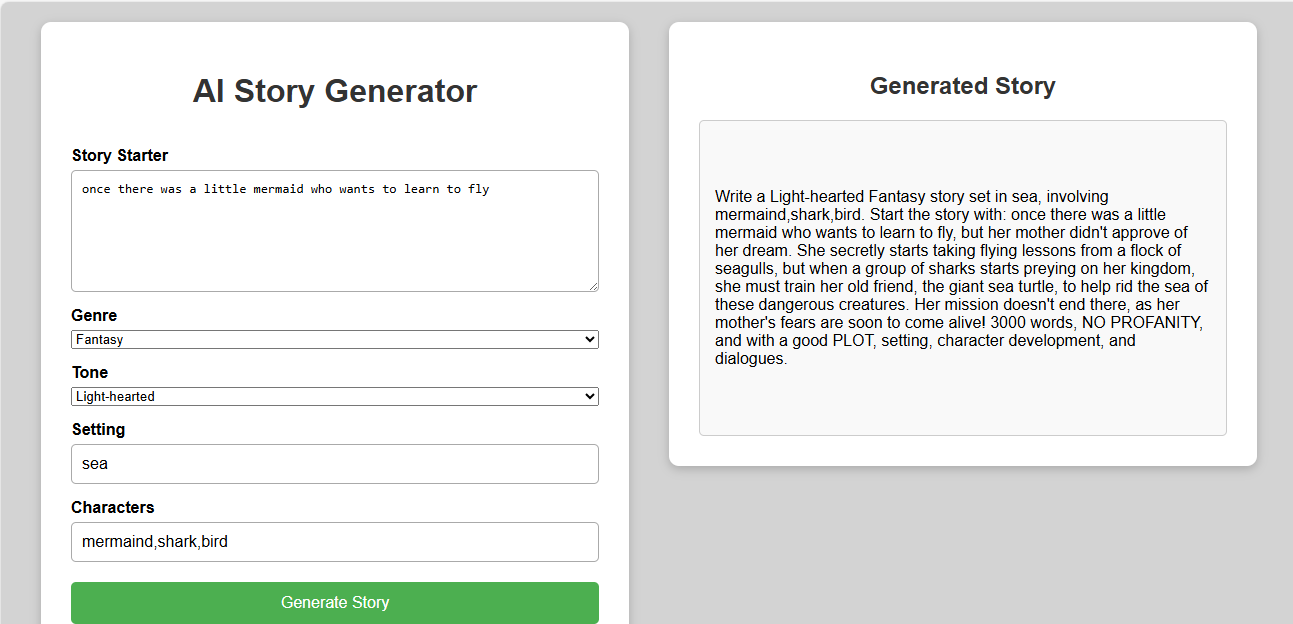




# 6. Output

These are the output for the model.





# 7. Conclusion

The Flask Story Generator app demonstrates how natural language generation models can be integrated into web applications. It combines user interactivity with powerful language models to create dynamic, personalized storytelling experiences.