

AI Creative Writing Assistant

Project Overview

The AI Creative Writing Assistant is an interactive tool designed to help users generate creative content, refine their writing,

and brainstorm story ideas using advanced language models. The application is deployed on Hugging Face Spaces and provides

real-time text generation, rewriting, and enhancement features.

Project Link: <https://huggingface.co/spaces/Fatma5shawqy/creative-writing-assistant>

Design Overview

The system is designed with a clean, minimal, and user-friendly interface using Gradio or Streamlit.

The UI allows users to input text, select writing options, and receive real-time creative responses.

The backend integrates AI models using Hugging Face APIs.

Design Goals:

- Simplicity and accessibility
- Fast interaction
- Clear and readable layout
- Mobile-friendly structure

Wireframe

The wireframe includes the following components:

Creative Writing UI

Input Text Box

[User enters prompt]

Generate Button

Output Display Area

[AI-generated text appears here]

Additional buttons may include:

- Rewrite text
- Expand text
- Adjust creativity level

```
current working directory: /Users/omar/

[4]: train_df, test_df = train_test_split(final_df[['clean_text', 'processed_text']], test_size=0.2, random_state=42)
train_df.to_csv("train_clean.csv", index=False)
test_df.to_csv("test_clean.csv", index=False)

print(" Done! Saved:")
print(" train_clean.csv")
print(" test_clean.csv")
print(" full_clean.csv")

Done! Saved:
train_clean.csv
test_clean.csv
full_clean.csv

[5]: # from datasets import load_dataset
# import pandas as pd

# train_df = pd.read_csv("train_clean.csv")
# test_df = pd.read_csv("test_clean.csv")

# from datasets import Dataset
# train_dataset = Dataset.from_pandas(train_df)
# test_dataset = Dataset.from_pandas(test_df)

[6]: from transformers import AutoTokenizer
from datasets import load_dataset
import pandas as pd
train_df = pd.read_csv("train_clean.csv")
test_df = pd.read_csv("test_clean.csv")

train_dataset = Dataset.from_pandas(train_df)
test_dataset = Dataset.from_pandas(test_df)

model_name = "gpt2-small"
tokenizer = AutoTokenizer.from_pretrained(model_name)

def tokenize(batch):
    return tokenizer(
        batch["clean_text"],
        batch["processed_text"],
        truncation=True,
        padding="max_length",
        max_length=128
    )

train_dataset = train_dataset.map(tokenize, batched=True, batch_size=32)
test_dataset = test_dataset.map(tokenize, batched=True, batch_size=32)

train_dataset = train_dataset.remove_columns(["clean_text", "processed_text"])
test_dataset = test_dataset.remove_columns(["clean_text", "processed_text"])

[7]: import pandas as pd

train_df = pd.read_csv(r"D:\depi_final_pro\depi_final_pro\train_clean.csv")
test_df = pd.read_csv(r"D:\depi_final_pro\depi_final_pro\test_clean.csv")

[8]: # max_length=128,
# truncation=True,
# padding="max_length"
# )

model_inputs["labels"] = labels["input_ids"]
return model_inputs

[26]: train_dataset.column_names

[26]: ["clean_text", "processed_text"]

[27]: bad_rows = train_df[
    (~train_df["clean_text"].apply(lambda x: isinstance(x, str))) |
    (~train_df["processed_text"].apply(lambda x: isinstance(x, str)))
]

bad_rows.head(), len(bad_rows)

[27]: (Empty DataFrame
Columns: [clean_text, processed_text]
Index: [],
0)

[28]: train_df = train_df.dropna(subset=["clean_text", "processed_text"]).reset_index(drop=True)
test_df = test_df.dropna(subset=["clean_text", "processed_text"]).reset_index(drop=True)

[29]: train_df["clean_text"] = train_df["clean_text"].astype(str)
train_df["processed_text"] = train_df["processed_text"].astype(str)

test_df["clean_text"] = test_df["clean_text"].astype(str)
test_df["processed_text"] = test_df["processed_text"].astype(str)
```

AI Creative Writing Assistant

اكتب جملة وسيقوم النموذج بإكمال النص الإبداعي تلقائياً

prompt

output

Clear

Submit

Flag

Model Description

The system relies on Hugging Face Large Language Models (LLMs) such as:

- Llama 3
- Qwen
- GPT-2

These models are accessed using the Hugging Face Inference API.

The Python backend uses transformers, langchain, and custom prompt templates to structure inputs and produce high-quality outputs.

Model Features:

- Text generation
- Creative rewriting
- Brainstorming
- Style and tone adjustment
- Grammar assistance

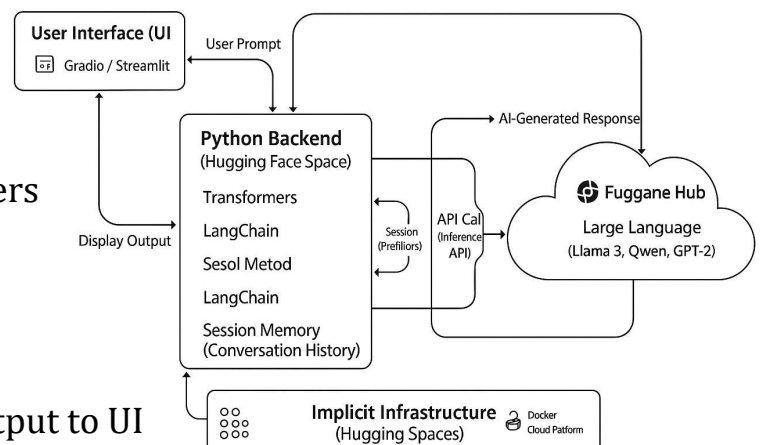
Visuals (Architecture Diagram Description)

The architecture includes:

- User Interface (Gradio/Streamlit)
- Python Backend hosted on Hugging Face Spaces
- LLM Model Hub (Hugging Face API)
- Session memory handling conversation flow
- Cloud infrastructure running Docker containers

Flow:

User → UI → Backend → LLM → Backend → Output to UI



Tutorials (How to Use)

Step 1: Visit the application:

<https://huggingface.co/spaces/Fatma5shawqy/creative-writing-assistant>

Step 2: Enter any writing prompt in the text box.

Examples:

- "Write a story about a time-traveling archaeologist."
- "Rewrite this paragraph to sound more dramatic."
- "Give me character ideas for a fantasy novel."

Step 3: Click the Generate button.

Step 4: Review the output and use:

- Rewrite
- Expand
- Adjust creativity mode

as needed.

Step 5: Repeat the process to continue writing with AI assistance.

Full Report

Problem:

Many writers struggle with idea generation, structure, and creativity. Traditional tools offer either grammar checking

or simple text generation without creativity enhancement.

Solution:

An AI-powered assistant capable of brainstorming, rewriting, expanding, and guiding users through the writing process.

User Personas:

- Aspiring writers
- Content creators
- Educators
- Hobbyists

Key Features:

- Idea & prompt generation
- Story structure generation
- Character & world-building tools
- Tone/style adjustment
- Real-time AI rewriting
- Grammar enhancement

Testing:

- Unit tests for backend functions
- Integration tests for UI → model → response
- User testing for clarity and creativity quality

Deployment:

- Hosted on Hugging Face Spaces
- Python backend
- Hugging Face Inference API for model calls

Project Team:

- Hosam Eldein Mohamed Elsheshtawy (Team Leader)
- Alaa Taha EL Maria (MI Engineer)
- Fatma Shawky Mohamed Elhely (Developer)
- Mohamed Mohamed Abdelkawy (MLOPS)
- Yasser Ahmed Shawky (Development & Training)
- Eslam Ragab Kamal (Tester)