**Text Generation Tool for Creative Writing Assistance**

**Problem Statement:**

In the realm of creative writing, writers often face challenges such as writer's block or the need for inspiration to continue their stories. The Creative Writers Guild (CWG) is conducting a study to develop tools that can aid writers in generating content ideas and overcoming creative blocks. They require a tool capable of generating coherent and contextually relevant text based on a given prompt. By implementing an AI-driven text generation system using GPT-2, CWG aims to provide writers with a source of inspiration and assistance, ultimately helping them to enhance their creative writing process and produce compelling narratives.

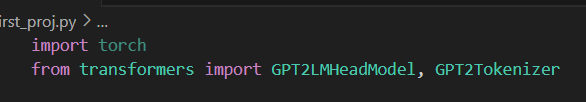


**Pre-Requisites:**

To complete this project,

* Anaconda Navigator: <https://www.anaconda.com/anaconda-navigator> (Jupyter notebook, Spyder)
* PyCharm: <https://www.jetbrains.com/pycharm/download/?section=windows> (Download PyCharm community edition)
* Or you can use any Python IDE

Libraries to be installed,



**Objective:**

The objective of this project is to create a user-friendly application for generating creative text based on user prompts. By leveraging GPT-2, users can input a prompt, and the application will generate and display a continuation of the text. This tool aims to assist writers, students, and content creators in generating ideas and overcoming writer's block.

**Tasks:**

1. Import necessary modules
2. Load pre-trained GPT-2 model and tokenizer
3. Define functions to generate text
4. Take user input (prompt for text generation)
5. Generate text based on the input and display the generated text

**Solution Development Procedure:**

**Task 1: Import necessary module**

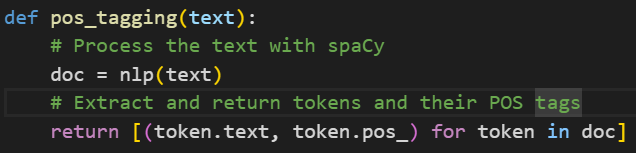
* Go to any Python IDE.
* This task involves importing the required libraries and modules. In this case, we will import the transformers and torch libraries.

**Task 2: Load pre-trained NLP Model**

****

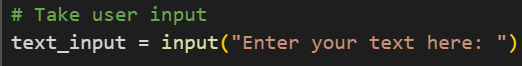
* Load a pre-trained NLP model from spaCy. The model en\_core\_web\_sm is a small English model that includes vocabulary, syntax, and named entities.

**Task 3: Define functions to perform POS tagging**

****

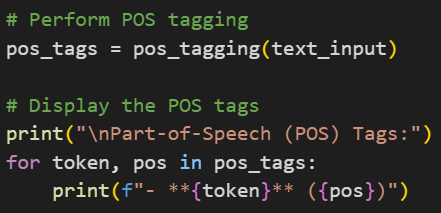
* Define a function named pos\_tagging that processes the input text with the spaCy model, extracts tokens, and returns a list of tuples containing each token and its corresponding POS tag.

**Task 4: Take User Input**



* Prompt the user to input a text string that they want to analyze for POS tagging. This allows the application to accept dynamic input for analysis.

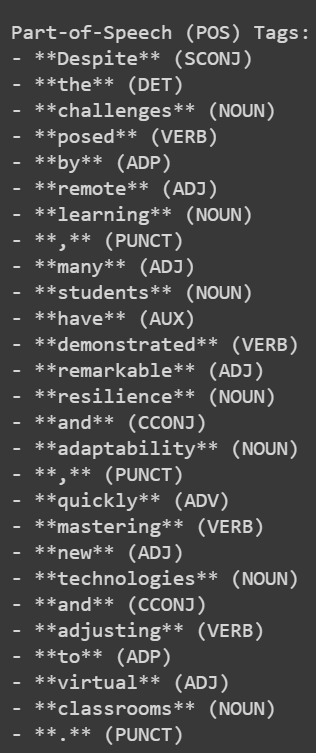
**Task 5: Analyze the input and display identified POS tags**

****

* Use the pos\_tagging function to process the user input and print out the tokens along with their POS tags. This helps the user to see the grammatical structure of their input text.

**Input text** : “Despite the challenges posed by remote learning, many students have demonstrated remarkable resilience and adaptability, quickly mastering new technologies and adjusting to virtual classrooms.”

**Output :**



**Conclusion:**  
In conclusion, this project provides a straightforward yet powerful solution for POS tagging in student essays. With its intuitive interface and robust POS tagging capabilities, users can quickly analyze grammatical structures and gain insights from students' writing. This tool is invaluable for educators and researchers, aiding in the extraction and analysis of syntactic patterns to support the development of personalized learning tools and targeted educational interventions. By efficiently processing large volumes of textual data, the application enhances the understanding of language use across different grade levels, contributing to improved educational outcomes.