# **Practical Assessment 6: Foundational Generative AI with LLMs**

### **Objective:**

Gain hands-on experience with foundational Generative AI concepts by working with a pre-trained LLM using the Hugging Face Transformers library. Learn to perform text generation, summarization, translation, and fine-tuning for a specific use case.

NOTE: Try to stick with one model only because these models are quite large to download. Hugging face models are still basic so they won't perform that good.

## **Task 1: Setup and Model Exploration**

### **Instructions:**

* Install and import the Hugging Face transformers and datasets libraries.
* Choose a **pretrained model** from the Hugging Face model hub (e.g., gpt2, facebook/bart-large, t5-small).
* Explore the tokenizer and model architecture.

### **Deliverables:**

* Code to load and test a model (e.g., generate a sample output).
* Brief explanation of tokenizer, model inputs/outputs.

## **Task 2: Basic Text Generation**

### **Instructions:**

* Use the model for **basic text generation** (e.g., complete a prompt or story).
* Try different decoding strategies: **greedy**, **top-k**, **top-p (nucleus)**, and **beam search**.

### **Deliverables:**

* Code for generating text using various strategies.
* Comparison of results and brief observations.

## **Task 3: Zero-Shot Tasks with LLMs**

### **Instructions:**

Use models like T5 or BART to perform:

* **Summarization** of a news article or paragraph.
* **Translation** (e.g., English to French or German).
* **Question answering** using a text context and query.

### **Deliverables:**

* Code for each task.
* Input/output examples for each task.

## **Task 4: Fine-Tuning on a Small Custom Dataset**

### **Instructions:**

* Select a small text dataset (e.g., CSV with prompt-response pairs, like Q&A, chatbot data).
* Tokenize the dataset.
* Fine-tune a smaller model (t5-small, distilgpt2) using Hugging Face's Trainer API.

### **Deliverables:**

* Dataset format and preprocessing steps.
* Fine-tuning script and training logs.
* Evaluate fine-tuned model on a few test prompts.

## **Task 5: Model Saving and Inference Pipeline**

### **Instructions:**

* Save the fine-tuned model and tokenizer.
* Write an inference function or script to load and run predictions on new inputs.

### **Deliverables:**

* Code to save/load model.
* Sample predictions on new text inputs.

## **Task 6: Deploy a Simple Text Generation API**

### **Instructions:**

* Build a Flask or FastAPI app to serve your fine-tuned model.
* Accept input prompt via POST request and return generated response.

### **Deliverables:**

* app.py file.
* Screenshot or curl output showing API prediction.

## **Task 7: GitHub Repository & Documentation**

### **Instructions:**

* Upload all code and documentation to GitHub.
* Include:
  + README.md with project summary, instructions to run the model and API, sample outputs.
  + Optional: .ipynb demo notebook.

### **Deliverables:**

* GitHub link.
* Clear and detailed README.md.