## ****GEN AI PROJECT PHASE 3 SUBMISSION DOCUMENT****

### ****Phase 3: Final Report and Submission****

### ****1. Project Title:****

**Sentence generator using Generative AI**

### ****2. Summary of Work done****

#### ****Phase 1 – Proposal and Idea Submission (10 Marks):****

In this phase, we proposed developing a **Sentence Generator** using Generative AI techniques. The aim was to create a system that accepts a user-provided sentence and generates coherent textual continuations using a pre-trained transformer model. The objectives included:

* Understanding transformer-based generative models in NLP.
* Utilizing a pre-trained GPT-2 model for context-aware text generation.
* Implementing a simple, script-based interaction without relying on any web interface.

A detailed proposal was submitted, outlining the problem, objectives, methodology, tools, and expected output.

#### ****Phase 2 – Execution and Demonstration (15 Marks):****

We implemented the idea using **Python** and the **HuggingFace Transformers library**, focusing on direct model access rather than using the pipeline abstraction. The key steps included:

* Loading the gpt2-large model and tokenizer.
* Tokenizing the user input and passing it to the model’s generate() method with defined decoding strategies (top-k, top-p).
* Decoding and displaying multiple continuations (default 3 completions) for a given input.
* Testing was done by running the function generate\_text\_from\_model(prompt) with various prompts to evaluate the quality and coherence of the generated outputs.

Sample outputs and the complete code were documented and submitted.

### ****3. GitHub Repository Link****

You can access the complete codebase, README instructions, and any related resources at the following GitHub link:

🔗 <https://github.com/Komal-1308/IBM-Gen-AI-Project>

### ****4. Testing Phase****

#### ****4.1 Testing Strategy****

The system was evaluated using a range of sentence prompts to assess its performance in generating coherent and relevant continuations. The testing phase involved both **manual testing** and **automated testing** methods. The key areas assessed included:

* **Input Handling**: The system was tested with different forms of text input—short phrases, full sentences, and incomplete fragments—to ensure flexible handling.
* **Contextual Relevance**: Generated continuations were examined for contextual alignment with the input, aiming for logical and meaningful flow.
* **Edge Case Testing**: Tests included unconventional or ambiguous inputs to evaluate the model’s response to unfamiliar or malformed prompts.
  1. ***Types of Testing Conducted***

1. **Unit Testing:**

Core functions — including model loading, tokenization, and text generation — were tested independently to ensure correctness and stability.

1. **Integration Testing:**

The interaction between the tokenizer, model, and decoding strategy (generate() with top-k and top-p sampling) was verified in an end-to-end manner to ensure smooth data flow and output.

1. **User Testing**

Various input prompts were manually entered into the code to inspect the relevance and quality of generated next sentences. This helped assess the model's real-time behavior under different conditions.

1. **Performance Testing**

The model was tested with inputs of varying length and complexity to measure response time and check for any generation lags. Performance was acceptable for small to medium-length prompts on both Colab and local environments.

#### ****4.3 Results****

* **Accuracy**: The model effectively generated grammatically correct and contextually consistent sentence continuations when provided with complete input sentences. For example, an input like “I went to the park to play.” yielded continuations such as **“**Iplayed football with my friends and enjoyed the

evening.” These completions were coherent and in line with the given context.

* **Response Time**: The model produced outputs promptly, typically within 1–2 seconds per input, depending on sentence length and computational resources (e.g., Google Colab runtime).
* **Handling of Edge Cases**: For nonsensical or ambiguous inputs (e.g., “Bananas drive laptops”), the model still generated syntactically plausible outputs, although semantic relevance varied—highlighting the model’s ability to generalize even with unfamiliar contexts.

### ****5. Future Work****

While the current implementation successfully performs next sentence completion using a pre-trained GPT-2 model, there are several areas for potential enhancement:

1. **Model Fine-Tuning**  
   Fine-tune GPT-2 on a domain-specific dataset (e.g., conversational data, news articles, or academic writing) to produce more tailored and context-aware continuations.
2. **Discriminative Relevance Filtering**  
   Integrate a scoring mechanism to rank or filter generated completions based on semantic similarity or topic relevance, improving output quality for production-grade use.
3. **Evaluation Module**  
   Add a module to automatically assess the grammaticality and coherence of generated sentences using metrics like BLEU, ROUGE, or human feedback loops.
4. **Batch Inference Support**  
   Enable batch processing of multiple input sentences for larger datasets, useful in applications like dataset augmentation or predictive writing tools.
5. **Multilingual Support**  
   Extend the model’s capabilities to support sentence completion in other languages using multilingual models or language-specific GPT variants.

### ****6. Conclusion****

This project successfully demonstrates how Generative AI, specifically GPT-2, can be applied to the task of **Next Sentence Completion**. By taking a complete input sentence and generating plausible continuations, the model showcases its understanding of language flow and context. The results underline its potential for integration into applications such as AI writing assistants, dialogue systems, or educational tools. While current results are promising, there remains substantial scope for customization, fine-tuning, and enhancement for domain-specific and multilingual deployments.