# LLM-Powered Quiz Question Generator

## Overview of the Selected LLM

### Rationale for Choosing GPT-4-turbo and GPT-3.5-turbo

In this project, I chose OpenAI's GPT-4-turbo for generating quiz questions and GPT-3.5-turbo for answering them, based on their performance and my experience with these tools. Initially, I used GPT-3.5-turbo for both generating and answering questions. While it was effective at answering, it struggled to produce varied and challenging questions. Switching to GPT-4-turbo for question generation marked a significant improvement. The questions became more detailed and better suited for different difficulty levels, making the quizzes more engaging and educational.

One of the key reasons for selecting GPT-4-turbo is its training on a vast dataset, which includes extensive information across various subjects, including world history. This extensive training makes GPT-4-turbo particularly adept at handling topics in world history, enabling it to generate nuanced and contextually rich questions that are ideal for educational purposes.

Additionally, my familiarity with OpenAI's API influenced the decision to use these models. Having previously worked with OpenAI's tools, I was able to quickly integrate them into the project, streamlining the development process.

For potential scaling or to reduce costs in the future, alternatives like the open-source LLaMa-3 8B or the more affordable proprietary model Claude 3 Sonnet might be considered. These models could offer similar functionalities at a lower cost.

By choosing these specific models and considering future alternatives, I aimed to balance quality, cost, and ease of use in developing the quiz question generator.

## Prompt Engineering Techniques

In developing the quiz question generator, I employed several prompt engineering techniques to optimize the model's performance in creating engaging and educationally relevant content. Here's a straightforward breakdown of these techniques:

- **Few-Shot Learning:**
    - **Usage:** This approach involves presenting the model with examples of high-quality quiz questions. It's similar to showing someone a template and asking them to create something similar.
    - **Relevance for the assignment:** Few-shot learning is effective because it helps the model understand the format and level of detail needed for the quiz questions. It ensures that the generated questions are not only accurate but also appropriately challenging for the target audience.

- **Context-setting:**
    - **Usage:** I specified the audience and question types in the prompts to ensure the content is suitable for the intended learners.
    - **Relevance for the assignment:** By setting the context, the model tailors its responses to fit the cognitive levels and learning styles of different groups. This makes the questions more relevant and engaging, which is crucial for educational tools.

- **Role-playing:**
    - **Usage:** In some prompts, the model is directed to act as an educational content creator or an Ai designed to assist educators or a master quiz creator, which sets specific expectations for its output.
    - **Relevance for the assignment:** Role-playing guides the model to think and respond as the specified role might think,  enhancing the output due to thinking as the role would. In our case, role playing will make the model focus on educational value and engagement, which enhances the quality of the quiz questions.

- **Instructional Prompts:**
    - **Usage:** These prompts give the model clear instructions on how to think and structure the questions, guiding it through the process of question creation.
    - **Relevance for the assignment:** Instructional prompts lead to higher-quality outputs because they direct the model's attention to key details, ensuring that the questions are both informative and thought-provoking.

- **Soft Prompting:**
    - **Usage:** Instead of rigid instructions, soft prompting suggests approaches the model can take, allowing some creative freedom.
    - **Relevance for the assignment:** This technique encourages the model to use its capabilities to generate diverse and interesting questions, which can make the quiz more engaging and enjoyable for users.

- **Zero-shot Prompting:**
    - **Usage:** This technique involves asking the model to perform a task without prior examples, relying entirely on its pre-trained knowledge.
    - **Relevance for the Assignment**: Zero-shot prompting tests the model's ability to generate diverse quiz questions independently, showcasing its versatility and comprehension. This approach streamlines prompt design, enhances content variety, and maintains user engagement by producing unique and unpredictable questions.

Each of these techniques helps to create a quiz question generator that not only meets educational standards but also keeps users interested and engaged, leveraging the model’s capabilities to their fullest. Just like using different teaching methods in a classroom, these techniques ensure that the generated questions address various learning needs and preferences.

## Instructions for Running the Tool

Here's a clear outline of every command line instruction needed to setup and run the QuizGenerator.py script:

Install Python:
Make sure Python is installed on your computer. You can verify this by running:

```
python --version
```

If Python is not installed, download and install it from the official Python website.

Install the OpenAI library:
Open your terminal and install the openai library using pip:

```
pip install openai
```

Set Up the OpenAI API Key:
You need an API key from OpenAI to use their services. Once you have your API key, set it as an environment variable:
On Windows:

```
set OPENAI_API_KEY=your_api_key_here
```

On macOS and Linux:

```
export OPENAI_API_KEY=your_api_key_here
```

Run the Quiz Generator Script:
Navigate to the directory where QuizGenerator.py is located.
To run the quiz generator, first open your terminal and change to the directory containing the QuizGenerator.py script. You can use the cd command followed by the directory path to do this. For example:

```
cd path_to_directory
```

Run the script:
Once in the correct directory, execute the script by running the following command in your terminal and follow the instructions that will be presented:

```
python QuizGenerator.py
```

Enjoy your high level quizzes :)

**Ideas to enhance the prompts or the tool:**

1. **Topic Customization:** Allow users to select their desired topics, enabling the tool to generate quizzes tailored to specific interests or study needs.
2. **Enhanced Output Options:** Integrate functionality to export quizzes to PDF or Word formats, and include options to directly email the quiz to users.
3. **Advanced Prompt Engineering:** Implement techniques such as prompt chaining and chain-of-thought prompting to enrich the complexity and depth of the generated quiz questions.
4. **Adaptive Questioning:** Introduce a feature that allows users to answer questions interactively, with the tool then adjusting the difficulty of subsequent questions based on user responses using Conditional Prompting.
5. **Multimodal Question Generation:** Utilize multimedia elements to craft questions, such as generating images that depict historical events or concepts, to enhance engagement and understanding.
6. **Interactive Feedback:** Provide immediate, detailed feedback on each answer, including explanations for why an answer is correct or incorrect, supplemented with links to additional learning resources.
7. **Model Fine-Tuning:** Customize the underlying model specifically for quiz creation, optimizing its ability to generate accurate and relevant educational content.