## Code

Click here to download the source code - [Source code](https://drive.google.com/file/d/1nGjWLAJPbbc3QXkG4ehqaxzlw2sm2Kfl/view?usp=sharing)

## Report

* 1. Click [here](https://drive.google.com/file/d/1niaRRDQRqeXLVuHJOgaVt7T86417l0YF/view?usp=sharing) to view detailed document

Team Name - Gradient Gurus

Shrinath Suresh - [suresh.shrinath@nielseniq.com](mailto:suresh.shrinath@nielseniq.com)

Gokulnath T - [gokulnath.t@nielseniq.com](mailto:gokulnath.t@nielseniq.com)

* 1. Approach
     1. We tried three models with different configurations.We experimented with three models, each with varying architectures.
     2. The architecture details are outlined in the documentation. Notably, we observed that models with a larger number of parameters yield more accurate results. Initially, we tested smaller models and gradually increased the dataset size from 1 billion to 3 billion and then to 8 billion tokens, ultimately achieving 48% accuracy with the LLaMA 8 billion model.
     3. Our findings indicated that increasing model size correlates with improved accuracy. For instance, in tests with OpenAI's models, using only 5% of the dataset led to high accuracy. Full dataset utilization could further enhance results. We also explored GPT-4 and GPT-4 Mini for potential improvements. To reduce infrastructure costs, we implemented a local FAISS vector database for vector indexing and searching using the Sentence Transformer model to convert text into embeddings.
     4. Our innovative approach involved combining web search results with similarity searches, which significantly improved accuracy. We also refined our preprocessing and postprocessing steps to enhance the model's output quality.
     5. We learned that large language models (LLMs) can exhibit inconsistency. To address this, we incorporated both preprocessing and postprocessing techniques to boost accuracy.While some configurations underperformed, our adjustments in preprocessing and the integration of diverse data sources helped overcome these challenges.
     6. Key techniques included using a batch API for inference to manage costs effectively, postprocessing LLaMA's output for improved accuracy, and leveraging a complete dataset of 20 million tokens for comprehensive training.