Technical Test Assignment for Full stack AI Engineer:

Objective

- To assess the candidate's ability to extract important information from a set of documents, create entities, establish relationships, determine anomaly points, and structure the output in JSON format for visualization on a canvas, with a focus on leveraging AI to facilitate these functions.
- To assess the candidate's ability to implement innovative AI solutions in overcoming the issues at hand.
- Evaluate the candidate's ability to understand the importance of User End State when implementing AI based enrichment modules.
- To assess the candidate's proficiency in optimizing model prompts and responses to meet specific criteria, aligning with the goal to develop cost-effective and scalable solutions.

Assignment Overview

You are tasked with building a backend system that processes a dataset of provided texts as input and uses AI (Not any NLP model) to generate the following:

- **Key Entities**: Extract Key Entities (e.g., Names, Locations, Organizations, DTGs, Account Nos, Tel Nos, Companies etc.) that are a focus in the data sets.
- **Relationship Extraction**: Identify relationships between these entities.
- **Anomaly Detection**: Extract various anomalies detected in the dataset and highlight on the canvas.

The solution must use any **locally hosted AI Model** instead of OpenAI or GPT-based APIs. The system should demonstrate effective model utilization, prompt engineering, and result formatting. You will be provided with a set of documents in various formats (PDF, excel, Word, etc.). It is very important that AI models are used to extract important information from these documents. **The not so high level of Accuracy of the untrained models is acceptable.**

Key Requirements

1. **Model Selection**:
   - Choose an open-source or locally hosted model suitable for text enrichment (e.g., models from Hugging Face or similar frameworks).
   - Ensure the selected model supports entity recognition, relationship extraction, and sentiment analysis.

2. **Backend Implementation**:
   - Identify and create entities based on the extracted information using AI techniques.
   - Entities could include people, organizations, dates, locations, Account Nos, IDs, Vehicle Nos, etc.
   - Relationship Creation:
     i. Establish relationships between the created entities using AI models.
     ii. Relationships could include associations, hierarchies, dependencies, etc.
   - Anomaly Detection:

i. Analyze the information to determine any anomaly points using AI algorithms.

ii. Anomalies could be inconsistencies, outliers, or unusual patterns.

- JSON Structuring:

  i. Structure the extracted information, entities, relationships, and anomalies in JSON format using AI tools.

  ii. Ensure the JSON is well-structured and follows best practices.

3. **Front End Visualization**.
   - Use a canvas (e.g., G6, Cytoscape.js) to visualize the key data and relationships.
   - Visualization should be interactive and provide insights into the data, with the help of AI for enhanced visualization techniques.
   - Design and implement an API endpoint that accepts list of texts.
   - Process the text to extract entities, relationships, and anomalies.
   - Extracted relationships should be shown on a canvas
   - Highlight anomaly points should be shown on a canvas.

4. **Performance and Optimization**:
   - Optimize model loading and inference time for scalability.
   - Minimize resource consumption while maintaining high accuracy.
   - Ensure responses are generated efficiently for a large set of text documents.

Evaluation Criteria

- **AI.** Use of AI in performing tasks.
- **Innovative Approaches**: Creativity in overcoming challenges and implementing additional useful features.
- **Performance**: Efficiency in handling large text inputs without sacrificing output quality and response times.
- **Prompt Optimization**: Skill in crafting prompts to generate **accurate**, **relevant** and **consistent** model responses.
- **Code Quality**: Clean, modular, and maintainable code with proper documentation and comments.
- **Model Utilization**: Effectiveness in selecting and using a local LLM that is small in size yet has high accuracy (a model that fits in 16gb VRAM at the maximum).
- **JSON Output Structure**: Adherence to the required output format.

Hints

- Consider using models newer and smaller models like gemma2, phi3, llama3.2:3b models
- May Use Ollama for model hosting and inference.
- For canvas, you can anything like keylines by Cambridge intelligence or any other similar library