Data Engineer Hiring Case Study

Scenario:

You're working for a company that analyzes user interaction data from multiple platforms and generates real-time insights. Your task is to build a scalable data pipeline that

1. Generates interaction data,
2. Processes it in real-time,
3. Ingests the data into a NoSQL database, while performing aggregations.
4. Finally, you will pull this data into a dashboard for visualization.

The challenge is divided into three sections: data generation and streaming, NoSQL storage with real-time aggregation, and dashboarding. Each task builds upon the previous one, forming a complete data pipeline.

Problem 1: Random Data Generator and Kafka Producer

Task:

Create a **random data generator** that simulates user interaction logs at scale. The generated data should include:

- user_id: Unique identifier for each user
- item_id: Identifier for the item interacted with
- interaction_type: Type of interaction (e.g., click, view, purchase)
- timestamp: When the interaction occurred

Requirements:

1. The data generator and Kafka producers should be scalable.
2. Implement a Kafka producer that publishes this generated data to a Kafka topic. Able to handle large amounts of data.
3. Granular controls over **rate** of Data generation / Kafka production for high-throughput simulations.

Problem 2: Kafka Consumer and Real-time Aggregations

Task:

Create a **Kafka consumer** that consumes the interaction data in real-time. The consumer should perform basic aggregations (e.g., calculating averages, minimum or maximum values for interactions). This processed data should then be ingested into a NoSQL database of your choice (e.g., MongoDB, Elasticsearch, Cassandra).

Requirements:

1. Write a Kafka consumer that:
   - Consumes messages from the Kafka topic created in Problem 1.
   - Performs **real-time aggregations**, such as:
   - Average number of interactions per user.
   - Maximum and minimum interactions per item.
   - Updates the NoSQL database with these aggregated values.
2. Choose a NoSQL database for storage and justify your choice based on performance, scalability, and ease of integration.

Expectations:

- Efficient handling of high-throughput real-time streams.
- Aggregations must be performed continuously as new data arrives.
- Clear schema design for storing aggregated results in the NoSQL database.

Problem 3: Data Visualization and Dashboarding

Task:

The company wants to visualize the real-time aggregation results for monitoring purposes. The data stored in the NoSQL database must be pulled and displayed on a simple dashboard.

Requirements:

1. Build a pipeline that:
   - Pulls the aggregated data from the NoSQL database in real-time or near real-time.

- Populates a dashboard using a visualization tool like **Kibana**, **Excel**, or any other basic dashboarding tool of your choice.

2. The dashboard should display the following metrics:

- Average interactions per user.
- Maximum and minimum interactions per item.
- Any other aggregation metric you find relevant.

Expectations:

- The dashboard should be user-friendly and provide real-time or near real-time updates.
- Data retrieval from the NoSQL database should be efficient and optimized.

Bonus Task (Optional):

- Implement a mechanism for alerting or notifications when certain thresholds are exceeded (e.g., when interaction counts for a particular item surpass a limit).

Submission Guidelines:
Usage of ChatGPT or other services **are Allowed**. However, disclosure if Chat GPT was **used** or **not used** is mandatory. The interview will be suited accordingly.

- Code must be well-documented.
- Provide the data generator, Kafka producer, Kafka consumer, and all relevant scripts for NoSQL integration and dashboarding.
- Prepared to execute the pipeline on local / online IDEs.
- Explanation of design choices, including parameter configurations for the data generator and consumer.

This flow is meant to demonstrates the candidate's comprehensive understanding of data engineering concepts at scale.