**Report:** **Real-Time E-commerce Order Processing System Using Kafka**

Preyali Dave (202318028)

To develop a Kafka-based system for managing e-commerce orders in real-time, you'll need to set up producers, consumers, and implement message filtering logic. Below are the steps you can follow to achieve this:

**Step 1: Set Up Kafka**

1. **Install Kafka:** Ensure Kafka is installed and running on your system or a server.
2. **Create Kafka Topics:** Create Kafka topics named **inventory\_orders** and **delivery\_orders** for each producer to send messages to.

**Step 2: Implement Kafka Producers**

* + **Inventory Orders Producer (inventory\_orders\_producer):** This producer should filter messages where the **type** field is **inventory**.
  + Implement a Kafka producer that reads inventory-related events from a data source (like a database or event stream) and sends messages with **type** set to **inventory** to the **inventory\_orders** topic.
  + **Delivery Orders Producer (delivery\_orders\_producer):** This producer should filter messages where the **type** field is **delivery**.
  + Develop a Kafka producer that reads delivery-related events and sends messages with **type** set to **delivery** to the **delivery\_orders** topic.

**Step 3: Implement Kafka Consumers**

* 1. **Inventory Data Consumer (inventory\_data\_consumer):** Configure a Kafka consumer that subscribes to the **inventory\_orders** topic.
  2. Implement logic to process inventory messages received by updating inventory databases or systems accordingly.
  3. **Delivery Data Consumer (delivery\_data\_consumer):**
  4. Set up a Kafka consumer for the **delivery\_orders** topic.
  5. Develop logic to handle delivery-related messages such as scheduling deliveries, updating delivery status, and notifying customers.

**Step 4: Develop Message Filtering Logic**

* 1. Producer Message Filtering: Implement logic within each producer (**inventory\_orders\_producer** and **delivery\_orders\_producer**) to filter messages based on the **type** field from the incoming data source.
  2. Only send messages to Kafka if they match the desired **type** (i.e., **inventory** or **delivery**).

**Additional Considerations**

* **Error Handling:** Implement error handling within producers and consumers to manage exceptions or failed operations gracefully.
* **Scalability:** Design your system to handle increasing loads by considering Kafka partitioning, consumer groups, and scaling strategies.
* **Monitoring and Logging:** Utilize Kafka monitoring tools and logging frameworks to monitor system performance and troubleshoot issues effectively.

By following these steps and best practices, you'll be able to develop a robust Kafka-based e-commerce order management system capable of real-time inventory management and delivery processing.