**Assignment Report on Using Kafka to Process Orders in Real-Time E-Commerce**

**Name:- Rishi Pawar**

**ID :- (202318037)**

Set up producers, consumers, and message filtering logic in order to create a Kafka-based system for real-time order management in e-commerce. The actions you can take to do this are listed below:

**Step 1: Install Kafka & Create Kafka Topics:** Ensure Kafka is installed and running on your system or a server. Then create 2 Kafka topics ‘kafkademo1’ and ‘kafkademo2’ each for inventory and delivery respectively.

Using the following instructions (either Terminal or Command Center), we may first configure Zookeeper and Kafka, then create relevant topics, and then initialize producer and consumer for each topic.

**zookeeper** - ./bin/zookeeper-server-start.sh ./config/zookeeper.properties

**server** - ./bin/kafka-server-start.sh ./config/server.properties

**Create topic** - ./bin/kafka-topics.sh --create --topic topicBDPdemo --bootstrap-server localhost:9092

**producer** - ./bin/kafka-console-producer.sh --topic topicBDPdemo1 --bootstrap-server localhost:9092

**consumer** - ./bin/kafka-console-consumer.sh --topic topicBDPdemo1 --bootstrap-server localhost:9092 --from-beginning

**Step 2: Creating Kafka Producers**

Data is sent to the appropriate producers after being filtered by category (such as "Inventory" or "Delivery").

1. **Inventory Orders Producer (inventory\_orders\_producer):** 
   * This producer takes filtered type “inventory” that has to be sent to consumer of 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 **kafkademo1** topic.
2. **Delivery Orders Producer (delivery\_orders\_producer):** 
   * This producer takes filtered type “delivery” that has to be sent to consumer of delivery.
   * Implement a Kafka producer that reads delivery-related events from a data source (like a database or event stream) and sends messages with **type** set to **delivery** to the **kafkademo2** topic.

**Step 3: Creating Kafka Consumers**

1. **Inventory Data Consumer (inventory\_data\_consumer):** 
   * Configure a Kafka consumer that subscribes to the **kafkademo1 (Inventory)** topic.
   * Implement any logic to process inventory messages received by updating inventory databases or systems accordingly.
2. **Delivery Data Consumer (delivery\_data\_consumer):**

* Set up a Kafka consumer for the **kafkademo2 (delivery)** topic.
* 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:**

* Message filtering has been implemented in producer where if the message “type” is inventory it is sent to **kafkademo1** topic and received by consumer of inventory.
* Similar strategy has been implemented for delivery.

By following these steps and best practices, a robust Kafka-based e- commerce order management system capable of real-time inventory management and delivery processing has been developed.