Day16_class_Assignment:

Case Study: Order Processing System Using Kafka and Spring Boot

Scenario

A retail company wants to process orders in real-time. They decide to use Apache Kafka to handle asynchronous communication between a Producer Service (Order Service) and a Consumer Service (Order Processing Service).

Flow

1. Customer places an order through an API in the Order Service.

2. Order Service publishes the order details to a Kafka topic named order-topic.

3. Order Processing Service consumes messages from the order-topic.

4. Order Processing Service processes the order (e.g., confirming stock, charging payment). Entities Order Entity

This class represents the order details that will be sent from the Producer to the Consumer via Kafka. Attributes:

• orderId → Unique ID of the order

• customerName → Name of the customer placing the order

• productName → Name of the product ordered

• quantity → Quantity of the product

• price → Price per unit of the product

• orderDate → Date when the order is placed Example Order

JSON: {

"orderId": "ORD101", "customerName": "John Doe",

"productName": "Laptop",

"quantity": 2,

"price": 55000,

"orderDate": "2025-08-08"

}

Postman API Testing

1. API Endpoint to Send Order POST http://localhost:8080/api/orders

2. Request Body (JSON) {

"orderId": "ORD102",

"customerName": "Alice Johnson",

"productName": "Smartphone",

"quantity": 1,

"price": 30000,

"orderDate": "2025-08-08"

}

3. Expected Flow

• The API sends the order to order-topic in Kafka.

The Order Processing Service listens to order-topic and prints/logs:

Output: Received Order: Order(orderId=ORD102, customerName=Alice Johnson, productName=Smartphone, quantity=1, price=30000, orderDate=2025-08-08)

:Programs:

Pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>3.5.4</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
```

```
<artifactId>kafka-example</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>kafka-example</name>
<description>Demo project for Apache kafka</description>
<url/>
<licenses>
	<license/>
</licenses>
<developers>
	<developer/>
</developers>
<scm>
	<connection/>
	<developerConnection/>
	<tag/>
	<url/>
</scm>
<properties>
	<java.version>17</java.version>
</properties>
<dependencies>
	<dependency>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-web</artifactId>
	</dependency>
	<dependency>
		<groupId>org.springframework.kafka</groupId>
		<artifactId>spring-kafka</artifactId>
```

```
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.kafka</groupId>
            <artifactId>spring-kafka-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

Entirty:

Order.java:

```
package com.example.kafkaexample.entity;
```

```
import java.time.LocalDate;

public class Order {

    private String orderId;
    private String customerName;
    private String productName;
    private int quantity;
    private double price;
    private LocalDate orderDate;

    public Order() {
    }

    public Order(String orderId, String customerName, String productName, int quantity,
double price, LocalDate orderDate) {
        this.orderId = orderId;
        this.customerName = customerName;
        this.productName = productName;
        this.quantity = quantity;
        this.price = price;
        this.orderDate = orderDate;
    }

    public String getOrderId() {
        return orderId;
    }
```

```
public void setOrderId(String orderId) {
    this.orderId = orderId;
}

public String getCustomerName() {
    return customerName;
}

public void setCustomerName(String customerName) {
    this.customerName = customerName;
}

public String getProductName() {
    return productName;
}

public void setProductName(String productName) {
    this.productName = productName;
}

public int getQuantity() {
    return quantity;
}

public void setQuantity(int quantity) {
    this.quantity = quantity;
}
```

```
public double getPrice() {
    return price;
}

public void setPrice(double price) {
    this.price = price;
}

public LocalDate getOrderDate() {
    return orderDate;
}

public void setOrderDate(LocalDate orderDate) {
    this.orderDate = orderDate;
}

@Override
public String toString() {
    return "Order{" +
            "orderId='" + orderId + '\'' +
            ", customerName='" + customerName + '\'' +
            ", productName='" + productName + '\'' +
            ", quantity=" + quantity +
            ", price=" + price +
            ", orderDate=" + orderDate +
            '}';
}
```

```
}
```

Controller:

RestControllerForKafka:

```
package com.example.kafkaexample.controller;

import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.example.kafkaexample.entity.Order;
import com.example.kafkaexample.service.OrderProducer;

@RestController
@RequestMapping("/api/orders")
public class RestControllerForKafka {

    private final OrderProducer orderProducer;

    public RestControllerForKafka(OrderProducer orderProducer) {
        this.orderProducer = orderProducer;
    }

    @PostMapping
    public String sendOrder(@RequestBody Order order) {
        orderProducer.sendOrder(order);
        return "Order sent successfully";
```

```
    }
}
```

Service:

```
package com.example.kafkaexample.service;

import org.springframework.core.annotation.Order;
import org.springframework.kafka.annotation.KafkaListener;
import org.springframework.stereotype.Service;

@Service
public class OrderConsumer {

    @KafkaListener(topics = "order-topic", groupId = "order-group")
    public void consume(Order order) {
        System.out.println("Received Order: " + order);
    }
}
```

OrderProducer:

```
package com.example.kafkaexample.service;

import org.springframework.kafka.core.KafkaTemplate;
import org.springframework.stereotype.Service;

import com.example.kafkaexample.entity.Order;

@Service
public class OrderProducer {
```

```
    private static final String TOPIC = "order-topic";
    private final KafkaTemplate<String, Order> kafkaTemplate;

    public OrderProducer(KafkaTemplate<String, Order> kafkaTemplate) {
        this.kafkaTemplate = kafkaTemplate;
    }

    public void sendOrder(Order order) {
        kafkaTemplate.send(TOPIC, order);
        System.out.println("Order sent to Kafka: " + order);
    }
}
```

Application.properties:

```
server.port=8080
spring.kafka.bootstrap-servers=localhost:9092
spring.kafka.consumer.group-id=order_group
spring.kafka.consumer.auto-offset-reset=earliest
spring.kafka.consumer.key-deserializer=org.apache.kafka.common.serialization.StringDeserializer
spring.kafka.consumer.value-deserializer=org.apache.kafka.common.serialization.StringDeserializer
spring.kafka.producer.key-serializer=org.apache.kafka.common.serialization.StringSerializer
spring.kafka.producer.value-serializer=org.apache.kafka.common.serialization.StringSerializer
logging.level.org.springframework=INFO
logging.level.com.example=DEBUG
```

KafkaExampleApplication:

```
package com.example.kafkaexample;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class KafkaExampleApplication {

	public static void main(String[] args) {
		SpringApplication.run(KafkaExampleApplication.class, args);
	}

}
```