==============

Apache Kafka

==============

=> Apache Kafka is a distributed streaming platform

=> Apache Kafka is called as Message Broker

=> Apache Kafka is used to process real time data feeds with high throughput and low latency

Ex : flights data, sensors data, stocks data, news data, social media etc....

=> Kafka works based on Publisher and Subscriber model

===================

Kafka Terminology

===================

Zookeeper

Kafka Server

Kafka Topic

Message

Publisher

Subscriber

===========

Kafka APIs

===========

Connector API

Publisher API

Subscriber API

Streams API

===

Spring Boot + Apache Kafka Application

==

Step-1 : Download Zookeeper from below URL

URL : http://mirrors.estointernet.in/apache/zookeeper/stable/

Step-2 : Download Apache Kafka from below URL

URL : http://mirrors.estointernet.in/apache/kafka/

Step-3 : Set Path to ZOOKEEPER in Environment variables upto bin folder

### Note: zookeeper.properties file will be available in kafka/config folder. You can copy zookeeper.properties and server.properties files from kafka/config folder to kafka/bin/windows folder. ###

Step-4 : Start Zookeeper server using below command from kafka/bin/windows folder

Command : zookeeper-server-start.bat zookeeper.properties

Step-5: Start Kafka Server using below command from Kakfa folder

Command : kafka-server-start.bat server.properties

Step-6 : Create Kakfa Topic using below command from kafka/bin/windows folder

```
Command : kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic amazon_orders_topic
```

```
                kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic kafka-testing-topic
```

Step-7 : View created Topics using below command

```
   Command : kafka-topics.bat --list --bootstrap-server localhost:9092
```

```
##############################
Kafka Producer App Development
##############################
```

```
=============================
1) Add below dependencies
================================
```

```
<dependencies>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-web</artifactId>
```

```xml
        </dependency>

        <dependency>
            <groupId>org.apache.kafka</groupId>
            <artifactId>kafka-streams</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.kafka</groupId>
            <artifactId>spring-kafka</artifactId>
        </dependency>

        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</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>
```

==

2) Create Kafka Constants class

==

```
public class AppConstants {


	public static final String TOPIC = "ashokit_order_topic";

	public static final String HOST = "localhost:9092";


}
```

=======================================

3) Create Model class to represent data

=======================================

```
@Data
public class Order {


	private String id;

	private Double price;

	private String email;


}
```

=======================================

4) Create Kafka Producer Config class

=======================================

```
@Configuration
public class KafkaProduceConfig {
    @Bean
    public ProducerFactory<String, Order> producerFactory() {

        Map<String, Object> configProps = new HashMap<>();

        configProps.put(ProducerConfig.BOOTSTRAP_SERVERS_CONFIG,
AppConstants.HOST);
        configProps.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
StringSerializer.class);
        configProps.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
JsonSerializer.class);

        return new DefaultKafkaProducerFactory<>(configProps);
    }

    @Bean
    public KafkaTemplate<String, Order> kafkaTemplate() {
        return new KafkaTemplate<>(producerFactory());
    }

}
```

==============================

4) Create Service Class

============================

```
@Service
public class OrderService {
    @Autowired
    private KafkaTemplate<String, Order> kafkaTemplate;

    public String addMsg(Order order) {

        // publish msg to kafka topic
        kafkaTemplate.send(AppConstants.TOPIC, order);

        return "Msg Published To Kafka Topic";
    }
}
```

=================================

5) Create RestController classs

==============================

```
@RestController
public class OrderRestController {

    @Autowired
    private OrderService service;

    @PostMapping("/order")
    public String createOrder(@RequestBody Order order) {
        String msg = service.addMsg(order);
        return msg;
    }
}
```

=====================================

6) Run the application and test it

=====================================

```
{
   "id" : "OD101",
   "price" : 200.00,
   "email" : "smith@gmail.com"
}
```

#################################

Kafka Subscriber App Dvelopment

#################################

===========================

1) Add below dependencies

===========================

```
<dependencies>
                        <dependency>
                                    <groupId>org.springframework.boot</groupId>
                                    <artifactId>spring-boot-starter-web</artifactId>
                        </dependency>
```

```
        <dependency>

            <groupId>org.apache.kafka</groupId>

            <artifactId>kafka-streams</artifactId>

        </dependency>

        <dependency>

            <groupId>org.springframework.kafka</groupId>

            <artifactId>spring-kafka</artifactId>

        </dependency>


        <dependency>

            <groupId>com.fasterxml.jackson.core</groupId>

            <artifactId>jackson-databind</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>
```

==========================

2) Create Constants class

=========================

```
public class KafkaConstants {


    public static final String TOPIC = "ashokit_order_topic";

    public static final String HOST = "localhost:9092";


}
```

================================

3) Create Model class

================================

```
@Data
public class Order {


    private String id;

    private Double price;

    private String email;


}
```

====================================

4) Create Consumer Config

====================================

```
@Configuration
public class KafkaConsumerConfig {
    }


    @Bean
    public ConsumerFactory<String, Order> consumerFactory() {

        Map<String, Object> configProps = new HashMap<String, Object>();

        configProps.put(ConsumerConfig.BOOTSTRAP_SERVERS_CONFIG, AppConstants.HOST);
        configProps.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG, StringDeserializer.class);
        configProps.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG, JsonDeserializer.class);

        return new DefaultKafkaConsumerFactory<>(configProps, new StringDeserializer(), new JsonDeserializer<>());

    }

    @Bean
    public ConcurrentKafkaListenerContainerFactory<String, Order> kafkaListnerFactory() {

        ConcurrentKafkaListenerContainerFactory<String, Order> factory =
                new ConcurrentKafkaListenerContainerFactory<>();

        factory.setConsumerFactory(consumerFactory());

        return factory;
    }

}
```

==

5) Add below method in boot app start class

==

```
@KafkaListener(topics = AppConstants.TOPIC, groupId="group_ashokit_order")
public void subscribeMsg(String order) {
                System.out.print("*** Msg Recieved From Kafka *** :: ");
                System.out.println(order);
        //logic
}
```

=======================================

6) Run the application

=======================================

#######   7) Send Request to Producer app and observer Subscriber app console ############