Microservices with Kafka, Zookeeper, Spring Boot

**Prerequisite**

* JDK 1.7+
* Maven 3+
* Kafka 2.10-0.10.1.1
* Zookeeper 3.4.9

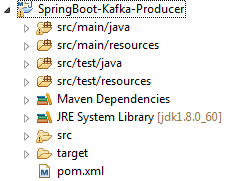
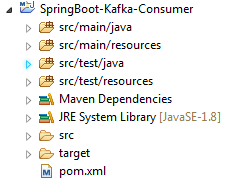
**Stack**

* Java
* Spring Boot

**Projects**

Create the below services (projects) in maven

1. SpringBoot-Kafka-Producer
2. SpringBoot-Kafka-Consumer

1. **SpringBoot-Kafka-Producer**

**Project Structure**

├── src

│   └── main

│   ├── java

│   │   └── com

│   │   └── springboot

│   │   └── kafka

│ │   └── Producer

│ │ ├── SpringBootKafkaProducerApplication.java │

│   │ │

│ │ ├── models

│ │ │ ├── Sender.java

│ │ │ ├── SenderConfig.java

│ │ │ ├── ConfigProperties.java

│ │ │ ├── JsonConfiguration.java

│ │ │ └── FundTransferRequest.java

│ │ │

│ │ └── ProducerController

│ │ └── ProducerController.java

│ │

│ │

│ └──resources

│ ├── application.properties

│ ├── application.yml

│ └── logback.xml

│

└── pom.xml

**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 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <groupId>com.springboot.kafka</groupId>  <artifactId>kafkaProducer</artifactId>  <version>0.0.1-SNAPSHOT</version>    <parent>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-parent</artifactId>  <version>1.4.0.RELEASE</version>  <relativePath/> <!-- lookup parent from repository -->  </parent>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  <java.version>1.8</java.version>  </properties>  <dependencies>  <dependency>  <groupId>org.springframework.kafka</groupId>  <artifactId>spring-kafka</artifactId>  <version>1.1.2.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-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> |

|  |
| --- |
| **SpringBootKafkaProducerApplication.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Producer/ SpringBootKafkaProducerApplication.java** |
| **package** com.springboot.kafka.Producer;  **import** org.springframework.boot.SpringApplication;  **import** org.springframework.boot.autoconfigure.SpringBootApplication;  @SpringBootApplication  **public** **class** SpringBootKafkaProducerApplication {  **public** **static** **void** main(String[] args){  SpringApplication.*run*(SpringBootKafkaProducerApplication.**class**, args);    }  } |

|  |
| --- |
| **Sender.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/** **Producer /models/ Sender.java** |
| package com.springboot.kafka.Producer.models;  import org.apache.kafka.clients.producer.Producer;  import org.apache.kafka.clients.producer.ProducerRecord;  import org.springframework.beans.factory.annotation.Autowired;  public class Sender {    @Autowired  private Producer<String,FundTransferRequest> producer;    public void sendMessage(String topic,FundTransferRequest object) throws InterruptedException {    producer.send(new ProducerRecord<String, FundTransferRequest>(topic, object));  System.out.println("Message sent successfully");  producer.flush();    }    } |

|  |
| --- |
| **SenderConfig.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/** **Producer /models/ SenderConfig.java** |
| **package** com.springboot.kafka.Producer.models;  **import** java.util.Properties;  **import** org.apache.kafka.clients.producer.KafkaProducer;  **import** org.apache.kafka.clients.producer.Producer;  **import** org.springframework.context.annotation.Bean;  **import** org.springframework.context.annotation.Configuration;  **import** org.springframework.kafka.support.serializer.JsonSerializer;  @Configuration  **public** **class** SenderConfig {    @Bean  **public** Properties producerConfigs() {    Properties props = **new** Properties();  props.put("bootstrap.servers", "localhost:9092");  props.put("acks", "all");  props.put("retries", 0);  props.put("batch.size", 16384);  props.put("linger.ms",0);  props.put("buffer.memory", 33554432);  props.put("key.serializer",  "org.apache.kafka.common.serialization.StringSerializer");  props.put("value.serializer", JsonSerializer.**class**);  **return** props;  }  @Bean  **public** Producer<String,FundTransferRequest> producerFactory() {  **return** **new** KafkaProducer<String, FundTransferRequest>(producerConfigs());  }  @Bean  **public** Sender sender() {  **return** **new** Sender();  }  } |

|  |
| --- |
| **JsonConfiguration.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/** **Producer /models/ JsonConfiguration.java** |
| package com.springboot.kafka.Producer.models;  import java.util.HashMap;  import java.util.Map;  import org.apache.kafka.clients.consumer.ConsumerConfig;  import org.apache.kafka.common.serialization.StringDeserializer;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.annotation.Bean;  import org.springframework.kafka.config.ConcurrentKafkaListenerContainerFactory;  import org.springframework.kafka.core.ConsumerFactory;  import org.springframework.kafka.core.DefaultKafkaConsumerFactory;  import org.springframework.kafka.support.serializer.JsonDeserializer;  public class JsonConfiguration {  @Autowired  private ConfigProperties configProperties;  @Bean  public ConsumerFactory<String, Object> consumerFactory() {  Map<String, Object> props = new HashMap<>();  props.put(ConsumerConfig.BOOTSTRAP\_SERVERS\_CONFIG, this.configProperties.getBrokerAddress());  props.put(ConsumerConfig.GROUP\_ID\_CONFIG, "fundresponsegroup");  props.put(ConsumerConfig.ENABLE\_AUTO\_COMMIT\_CONFIG, true);  props.put(ConsumerConfig.AUTO\_COMMIT\_INTERVAL\_MS\_CONFIG, 100);  props.put(ConsumerConfig.SESSION\_TIMEOUT\_MS\_CONFIG, 15000);  JsonDeserializer<Object> jsonDeserializer = new JsonDeserializer<>(Object.class);  return new DefaultKafkaConsumerFactory<>(props, new StringDeserializer(), jsonDeserializer);  }  @Bean  public ConcurrentKafkaListenerContainerFactory<String, Object> kafkaListenerContainerFactory() {  ConcurrentKafkaListenerContainerFactory<String, Object> factory =  new ConcurrentKafkaListenerContainerFactory<>();  factory.setConsumerFactory(consumerFactory());  return factory;  }  } |

|  |
| --- |
| **ConfigProperties.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/** **Producer /models/ ConfigProperties.java** |
| **package** com.springboot.kafka.Producer.models;  **import** org.springframework.boot.context.properties.ConfigurationProperties;  @ConfigurationProperties(prefix = "kafka")  **public** **class** ConfigProperties {  **private** String brokerAddress;  **private** String topic;    **public** String getBrokerAddress() {  **return** **this**.brokerAddress;  }  **public** **void** setBrokerAddress(String brokerAddress) {  **this**.brokerAddress = brokerAddress;  }  **public** String getTopic() {  **return** **this**.topic;  }  **public** **void** setTopic(String topic) {  **this**.topic = topic;  }  } |

|  |
| --- |
| **FundTransferRequest.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/** **Producer /models/FundTransferRequest.java** |
| **package** com.springboot.kafka.Producer.models;  **public** **class** FundTransferRequest {    **public** FundTransferRequest(){    }      **public** FundTransferRequest(String fromAccNo, String toAccNo, String amount,  String ifscCode, String desc, String refId) {    **this**.fromAccNo = fromAccNo;  **this**.toAccNo = toAccNo;  **this**.amount = amount;  **this**.ifscCode = ifscCode;  **this**.desc = desc;  **this**.refId = refId;  }  **private** String fromAccNo;  **public** String getFromAccNo() {  **return** fromAccNo;  }  **public** **void** setFromAccNo(String fromAccNo) {  **this**.fromAccNo = fromAccNo;  }  **public** String getToAccNo() {  **return** toAccNo;  }  **public** **void** setToAccNo(String toAccNo) {  **this**.toAccNo = toAccNo;  }  **public** String getAmount() {  **return** amount;  }  **public** **void** setAmount(String amount) {  **this**.amount = amount;  }  **public** String getIfscCode() {  **return** ifscCode;  }  **public** **void** setIfscCode(String ifscCode) {  **this**.ifscCode = ifscCode;  }  **public** String getDesc() {  **return** desc;  }  **public** **void** setDesc(String desc) {  **this**.desc = desc;  }  **public** String getRefId() {  **return** refId;  }  **public** **void** setRefId(String refId) {  **this**.refId = refId;  }  **private** String toAccNo;  **private** String amount;  **private** String ifscCode;  **private** String desc;  **private** String refId;  @Override  **public** String toString() {  **return** "FundTransferRequest {fromAccNo=" + fromAccNo + ", toAccNo="  + toAccNo + ", amount=" + amount + ", ifscCode=" + ifscCode  + ", desc=" + desc + ", refId=" + refId + "}";  }    } |

|  |
| --- |
| **ProducerController.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/** **Producer /ProducerController/ ProducerController.java** |
| package com.springboot.kafka.Producer.ProducerController;  import javax.validation.Valid;  import org.apache.kafka.clients.consumer.ConsumerRecord;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.annotation.Import;  import org.springframework.kafka.annotation.EnableKafka;  import org.springframework.kafka.annotation.KafkaListener;  import org.springframework.kafka.config.ConcurrentKafkaListenerContainerFactory;  import org.springframework.web.bind.annotation.RequestBody;  import org.springframework.web.bind.annotation.RequestMapping;  import org.springframework.web.bind.annotation.RestController;  import com.springboot.kafka.Producer.models.ConfigProperties;  import com.springboot.kafka.Producer.models.FundTransferRequest;  import com.springboot.kafka.Producer.models.JsonConfiguration;  import com.springboot.kafka.Producer.models.Sender;  @EnableKafka  @Import({ JsonConfiguration.class, ConfigProperties.class })  @RestController  @RequestMapping("/kafka")  public class ProducerController{  @Autowired  private Sender sender;    @Autowired  private ConfigProperties configProperties;    @Autowired  private ConcurrentKafkaListenerContainerFactory<String, Object> con;    ConsumerRecord<String,Object> objects = null;  @RequestMapping("/kafkaProducer")  public String kafkaProducer(@Valid @RequestBody FundTransferRequest object) throws InterruptedException {  sender.sendMessage(this.configProperties.getTopic(),object);  return "Successfully Sent";  }      @KafkaListener(topics = "fundTransferResponseTopic")  public void listen(ConsumerRecord<String,Object> object){  objects=object;  System.out.println("Received: Value: " + object.value()+" Key : " + object.key()+" Offset : " + object.offset());  }    @RequestMapping("/kafkaProducerSuccess")  public String kafkaSuccess(){    listen(objects);  return "Message Received Successfully";      }    } |

|  |
| --- |
| **application.properties** |
| **Path 🡪 src/main/resources/application.properties** |
| server.port: 8077 |

|  |
| --- |
| **application.yml** |
| **Path 🡪 src/main/resources/application.yml** |
| kafka:  brokerAddress: localhost:9092  topic: fundTransferRequestTopic    spring:  jmx:  enabled: false |

|  |
| --- |
| **logback.xml** |
| **Path 🡪 src/main/resources/logback.xml** |
| <configuration>  <appender name=*"STDOUT"* class=*"ch.qos.logback.core.ConsoleAppender"*>  <!-- encoders are assigned the type ch.qos.logback.classic.encoder.PatternLayoutEncoder  by default -->  <encoder>  <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n  </pattern>  </encoder>  </appender>  <root level=*"warn"*>  <appender-ref ref=*"STDOUT"* />  </root>  </configuration> |

1. **SpringBootKafka-Consumer**

**Project Structure**

├── src

│   └── main

│   ├── java

│   │   └── com

│   │   └── springboot

│   │   └── kafka

│ │   └── Consumer

│ │ ├── SpringBootKafkaConsumerApplication.java

│   │ │

│ │ └── models

│ │ ├── JsonConfiguration.java

│ │ ├── ConfigProperties.java

│ │ ├── Sender.java

│ │ ├── FundTransferRequest.java

│ │ └── SenderConfig.java

│ │

│ │

│ └──resources

│ ├── application.properties

│ ├── application.yml

│ └── logback.xml

│

└── pom.xml

**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 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <groupId>com.springboot.kafka</groupId>  <artifactId>kafksConsumer</artifactId>  <version>0.0.1-SNAPSHOT</version>  <name>SpringBootKafka-Consumer</name>  <parent>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-parent</artifactId>  <version>1.4.0.RELEASE</version>  <relativePath /> <!-- lookup parent from repository -->  </parent>  <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  <java.version>1.8</java.version>  </properties>  <dependencies>  <dependency>  <groupId>org.springframework.kafka</groupId>  <artifactId>spring-kafka</artifactId>  <version>1.1.2.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-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> |

|  |
| --- |
| **SpringBootKafkaConsumerApplication.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Consumer/ SpringBootKafkaConsumerApplication.java** |
| package com.springboot.kafka.Consumer;  import java.util.HashMap;  import java.util.Map;  import org.springframework.beans.BeansException;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.boot.SpringApplication;  import org.springframework.boot.autoconfigure.SpringBootApplication;  import org.springframework.context.annotation.Bean;  import org.springframework.context.annotation.Import;  import org.springframework.kafka.annotation.EnableKafka;  import org.springframework.kafka.annotation.KafkaListener;  import org.springframework.web.client.RestTemplate;  import com.springboot.kafka.Consumer.models.ConfigProperties;  import com.springboot.kafka.Consumer.models.FundTransferRequest;  import com.springboot.kafka.Consumer.models.JsonConfiguration;  import com.springboot.kafka.Consumer.models.Sender;  @SpringBootApplication  @EnableKafka  @Import({ JsonConfiguration.class, ConfigProperties.class })  public class SpringBootKafkaConsumerApplication {    public static void main(String[] args) throws BeansException, InterruptedException {    SpringApplication.run(SpringBootKafkaConsumerApplication.class, args);    }    @Bean  public Listener listener() {  return new Listener();  }    public static class Listener {    @Autowired  private Sender sender;    @Autowired  private ConfigProperties configProperties;    @Autowired  private RestTemplate restTemplate;    @KafkaListener(topics = "fundTransferRequestTopic")  public void listen(FundTransferRequest object) {    System.out.println("Received: Value: " + object);    try {    Map<String, Object> jsonValues = new HashMap<String, Object>();  jsonValues.put("ReferenceId", object.getRefId());  jsonValues.put("ErrorCode", "1245");  jsonValues.put("Error\_Desc" , "Success");  Object objectreturn=jsonValues;  sender.sendMessage(this.configProperties.getResponseTopic(),objectreturn);  String msg= restTemplate.getForObject(this.configProperties.getRestURL(),String.class);  System.out.println("In Consumer second msg : "+msg);  } catch (InterruptedException e) {  e.printStackTrace();  }  }  }    } |
|  |

|  |
| --- |
| **JsonConfiguration.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Consumer/ models/JsonConfiguration.java** |
| package com.springboot.kafka.Consumer.models;  import java.util.HashMap;  import java.util.Map;  import org.apache.kafka.clients.consumer.ConsumerConfig;  import org.apache.kafka.common.serialization.StringDeserializer;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.context.annotation.Bean;  import org.springframework.kafka.config.ConcurrentKafkaListenerContainerFactory;  import org.springframework.kafka.core.ConsumerFactory;  import org.springframework.kafka.core.DefaultKafkaConsumerFactory;  import org.springframework.kafka.support.serializer.JsonDeserializer;  public class JsonConfiguration {  @Autowired  private ConfigProperties configProperties;    @Bean  public ConsumerFactory<String, FundTransferRequest> consumerFactory() {  Map<String, Object> props = new HashMap<>();  props.put(ConsumerConfig.BOOTSTRAP\_SERVERS\_CONFIG, this.configProperties.getBrokerAddress());  props.put(ConsumerConfig.GROUP\_ID\_CONFIG, "fundrequestgroup");  props.put(ConsumerConfig.ENABLE\_AUTO\_COMMIT\_CONFIG, true);  props.put(ConsumerConfig.AUTO\_COMMIT\_INTERVAL\_MS\_CONFIG, 100);  props.put(ConsumerConfig.SESSION\_TIMEOUT\_MS\_CONFIG, 15000);  JsonDeserializer<FundTransferRequest> jsonDeserializer = new JsonDeserializer<>(FundTransferRequest.class);  return new DefaultKafkaConsumerFactory<>(props, new StringDeserializer(), jsonDeserializer);  }  @Bean  public ConcurrentKafkaListenerContainerFactory<String, FundTransferRequest> kafkaListenerContainerFactory() {  ConcurrentKafkaListenerContainerFactory<String, FundTransferRequest> factory =  new ConcurrentKafkaListenerContainerFactory<>();  factory.setConsumerFactory(consumerFactory());  return factory;  }  } |

|  |
| --- |
| **ConfigProperties.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Consumer/models/ ConfigProperties.java** |
| **package** com.springboot.kafka.Consumer.models;  **import** org.springframework.boot.context.properties.ConfigurationProperties;  @ConfigurationProperties(prefix = "kafka")  **public** **class** ConfigProperties {  **private** String brokerAddress;  **private** String responseTopic;    **public** String getResponseTopic() {  **return** responseTopic;  }  **public** **void** setResponseTopic(String responseTopic) {  **this**.responseTopic = responseTopic;  }  **private** String restURL;    **public** String getRestURL() {  **return** restURL;  }  **public** **void** setRestURL(String restURL) {  **this**.restURL = restURL;  }  **public** String getBrokerAddress() {  **return** **this**.brokerAddress;  }  **public** **void** setBrokerAddress(String brokerAddress) {  **this**.brokerAddress = brokerAddress;  }    } |

|  |
| --- |
| **Sender.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Consumer/models/Sender.java** |
| package com.springboot.kafka.Consumer.models;  import org.apache.kafka.clients.producer.Producer;  import org.apache.kafka.clients.producer.ProducerRecord;  import org.springframework.beans.factory.annotation.Autowired;  public class Sender {    @Autowired  private Producer<String,Object> producer;    public void sendMessage(String topic,Object object) throws InterruptedException {    producer.send(new ProducerRecord<String, Object>(topic, object));  System.out.println("Message sent successfully");  producer.flush();    }    } |

|  |
| --- |
| **SenderConfig.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Consumer/models/SenderConfig.java** |
| package com.springboot.kafka.Consumer.models;  import java.util.Properties;  import org.apache.kafka.clients.producer.KafkaProducer;  import org.apache.kafka.clients.producer.Producer;  import org.springframework.context.annotation.Bean;  import org.springframework.context.annotation.Configuration;  import org.springframework.kafka.support.serializer.JsonSerializer;  import org.springframework.web.client.RestTemplate;  @Configuration  public class SenderConfig {  @Bean  public Properties producerConfigs() {    Properties props = new Properties();  props.put("bootstrap.servers", "localhost:9092");  props.put("acks", "all");  props.put("retries", 0);  props.put("batch.size", 16384);  props.put("linger.ms",0);  props.put("buffer.memory", 33554432);  props.put("key.serializer",  "org.apache.kafka.common.serialization.StringSerializer");  props.put("value.serializer", JsonSerializer.class);  return props;  }    @Bean  public RestTemplate restTemplate() {  RestTemplate restTemplate = new RestTemplate();  return restTemplate;  }    @Bean  public Producer<String,Object> producerFactory() {  return new KafkaProducer<String, Object>(producerConfigs());  }    @Bean  public Sender sender() {  return new Sender();  }  } |

|  |
| --- |
| **FundTransferRequest.java** |
| **Path 🡪 src/main/java/com/springboot/kafka/Consumer/models/FundTransferRequest.java** |
| **package** com.springboot.kafka.Consumer.models;  **public** **class** FundTransferRequest {  @Override  **public** String toString() {  **return** "FundTransferRequest {fromAccNo=" + fromAccNo + ", toAccNo="  + toAccNo + ", amount=" + amount + ", ifscCode=" + ifscCode  + ", desc=" + desc + ", refId=" + refId + "}";  }      **public** FundTransferRequest(){    }    **public** FundTransferRequest(String fromAccNo, String toAccNo, String amount,  String ifscCode, String desc, String refId) {  **super**();  **this**.fromAccNo = fromAccNo;  **this**.toAccNo = toAccNo;  **this**.amount = amount;  **this**.ifscCode = ifscCode;  **this**.desc = desc;  **this**.refId = refId;  }  **private** String fromAccNo;  **public** String getFromAccNo() {  **return** fromAccNo;  }  **public** **void** setFromAccNo(String fromAccNo) {  **this**.fromAccNo = fromAccNo;  }  **public** String getToAccNo() {  **return** toAccNo;  }  **public** **void** setToAccNo(String toAccNo) {  **this**.toAccNo = toAccNo;  }  **public** String getAmount() {  **return** amount;  }  **public** **void** setAmount(String amount) {  **this**.amount = amount;  }  **public** String getIfscCode() {  **return** ifscCode;  }  **public** **void** setIfscCode(String ifscCode) {  **this**.ifscCode = ifscCode;  }  **public** String getDesc() {  **return** desc;  }  **public** **void** setDesc(String desc) {  **this**.desc = desc;  }  **public** String getRefId() {  **return** refId;  }  **public** **void** setRefId(String refId) {  **this**.refId = refId;  }  **private** String toAccNo;  **private** String amount;  **private** String ifscCode;  **private** String desc;  **private** String refId;    } |

|  |
| --- |
| **application.properties** |
| **Path 🡪 src/main/resources/application.properties** |
| **server.port: 8088** |

Mention your topic name in application.yml on **responseTopic:** <Your topicName>

|  |
| --- |
| **application.yml** |
| **Path 🡪 src/main/resources/application.yml** |
| kafka:  brokerAddress: localhost:9092  restURL: http://localhost:8077/kafka/kafkaProducerSuccess  responseTopic: fundTransferResponseTopic  spring:  jmx:  enabled: false |

|  |
| --- |
| **logback.xml** |
| **Path 🡪 src/main/resources/ logback.xml** |
| <configuration>  <appender name=*"STDOUT"* class=*"ch.qos.logback.core.ConsoleAppender"*>  <encoder>  <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n  </pattern>  </encoder>  </appender>  <root level=*"warn"*>  <appender-ref ref=*"STDOUT"* />  </root>  </configuration> |

**Build and run the application**

Navigate to the project where pom file is located and use the below command

mvn clean spring-boot:run

D:\Microservices\_Kafka\SpringBoot-Kafka-Producer>mvn clean spring-boot:run

D:\Microservices\_Kafka\SpringBoot-Kafka-Consumer>mvn clean spring-boot:run

**Test and run the application**

|  |  |  |
| --- | --- | --- |
| Service Name | Port Number | Main Class |
| SpringBoot-Kafka-Producer | 8077 | SpringBootKafkaProducerApplication.java |
| SpringBoot-Kafka-Consumer | 8088 | SpringBootKafkaConsumerApplication.java |

After Kafka Installation and Topic Creation

1. Start the SpringBoot-Kafka-Producer and SpringBoot-Kafka-Consumer application



1. Run the Kafka Producer in SoapUI or in any rest API client:

<http://localhost:8077/kafka/kafkaProducer>

input:

{

" fromAccNo ":"109238237298372712343",

" toAccNo ":"1342343413121q234",

" amount ":"10000",

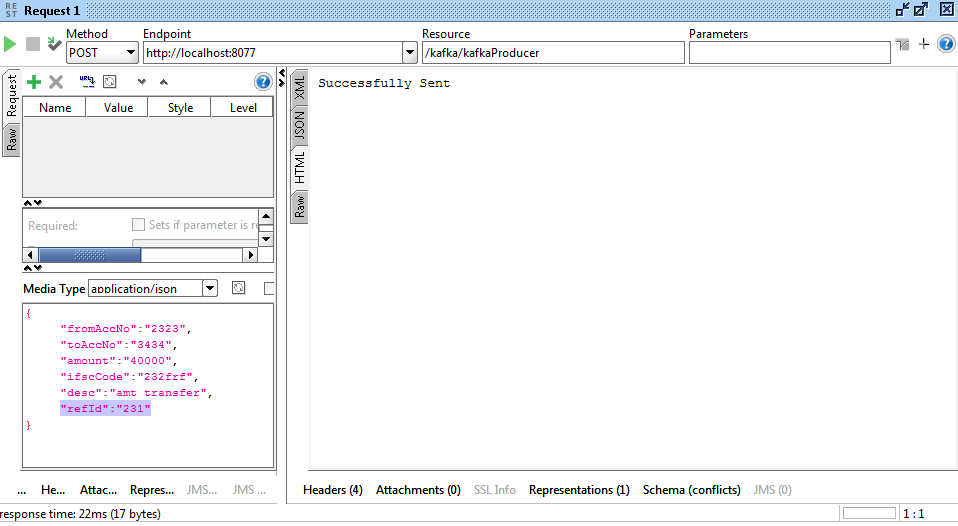
" ifscCode ":"HDFC0003213",

" desc ":"Transfer money",

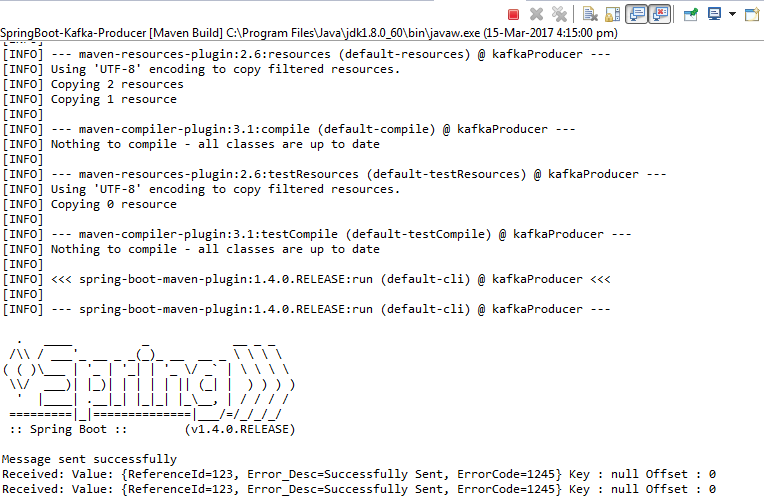
"refId":"231"

}

**In POST method**



**In Producers Console:**



**In Consumer Console:**

