Lab 12

Part 1

First start **Zookeeper** by double clicking the file …\kafka_2.11-1.1.0\startzookeeper
Then start **Kafka** by double clicking the file …\kafka_2.11-1.1.0\startkafka

Given are 2 applications: KafkaSender and KafkaReceiver.

In IntelliJ open the projects **KafkaReceiver and KafkaSender**

Run KafkaReceiver

```
Receiver is running and waiting for messages
```

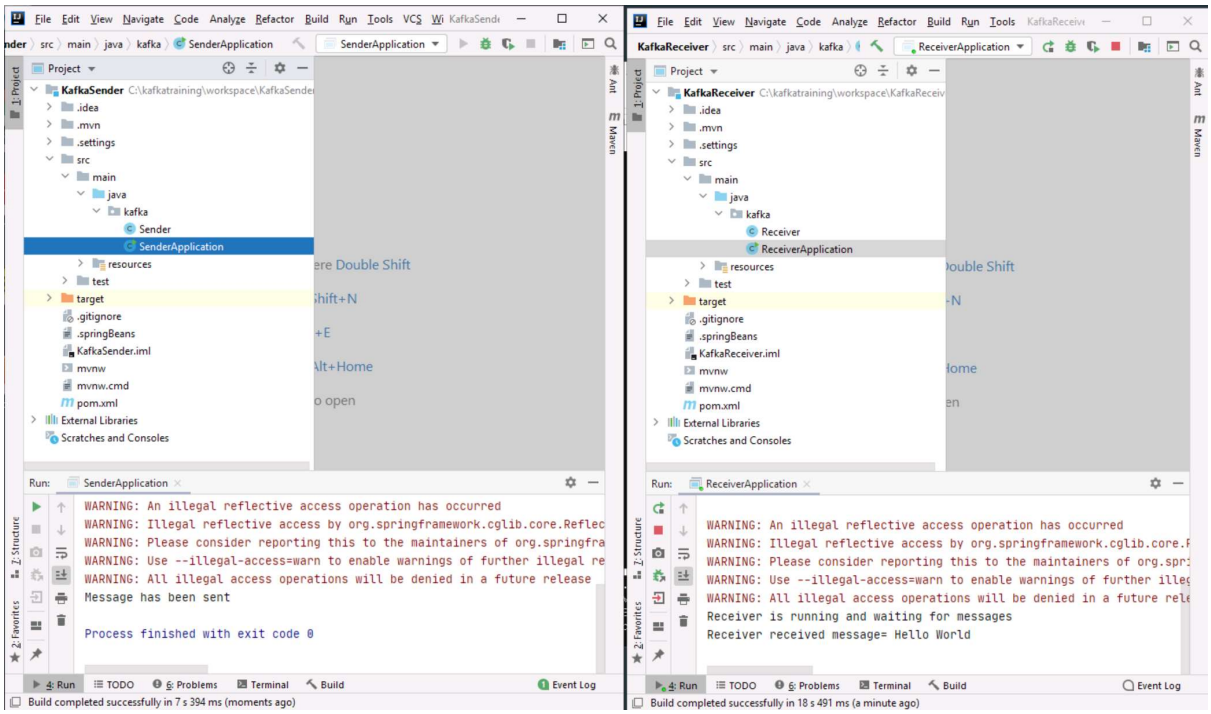
Run KafkaSender

In the console you will see the message:

```
Message has been sent
```

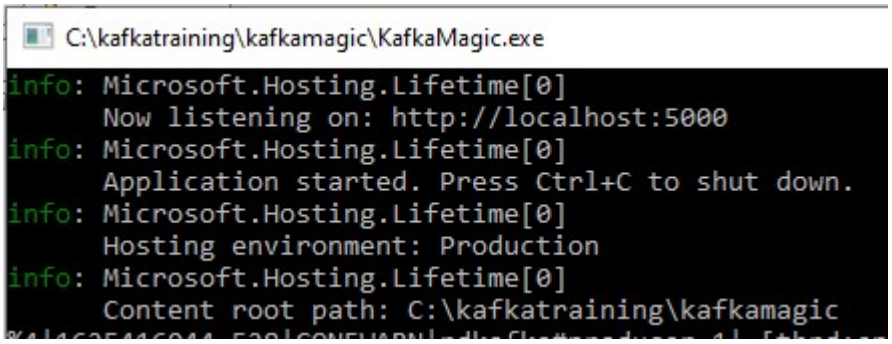
In the receiver you see the following output:

```
Receiver is running and waiting for messages
Receiver received message= Hello World
```

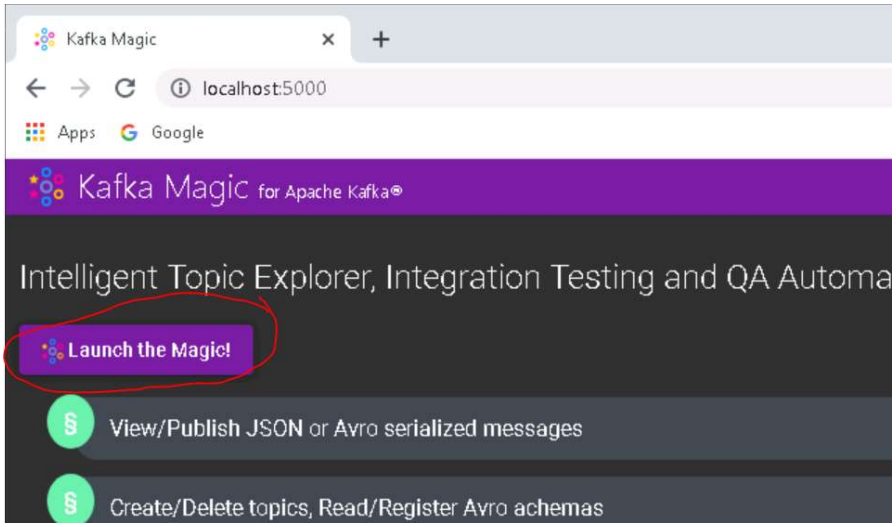


Run the sender a few times more.

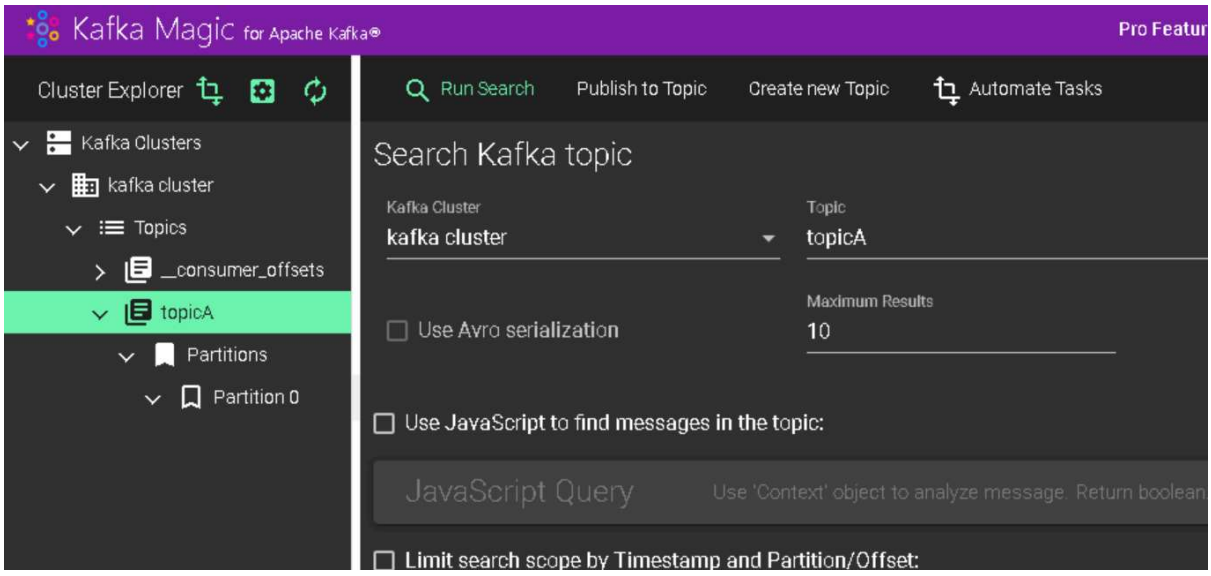
Now start KafkaMagic by double-clicking the file **…\ kafkamagic\KafkaMagic.exe**



Open in a browser the following URL: **http://localhost:5000/**



Click the **Launch the Magic!** button



Select **TopicA** and click **Run Search**

Scroll to the bottom and you can now inspect the messages in the topicA
Run the sender again and see the message appear in the topic and in the receiver application.

In the ReceiverApplication write a new Receiver class:

```
@Service
public class Receiver2 {

    @KafkaListener(topics = {"topicA"})
    public void receive(@Payload String message) { System.out.println("Receiver2 received message= "+ message); }

}
```

Restart the Receiver and see that only one receiver received the message send by the sender.

Modify Receiver2 as follows

```
@Service
public class Receiver2 {

    @KafkaListener(topics = {"topicA"}, groupId = "gid2")
    public void receive(@Payload String message) { System.out.println("Receiver2

}
```

Restart the Receiver and notice that both receivers received the message send by the sender.

In the KafkaSender project, create a new Sender that sends a message to **TopicA2**. In the KafkaReceiver project add a new Receiver that listens to messages in **TopicA2**

Part2

Somewhere on the interstate are 2 cameras installed. The distance between camera1 and camera2 is ½ mile. On this interstate the maximum speed is 70 miles/hour.

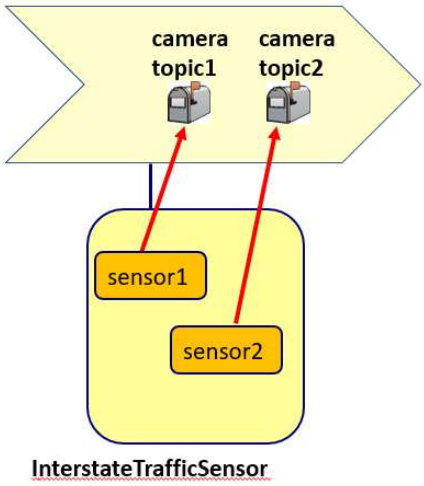
Both cameras make a picture of every car that passes, and the cameras have build-in image recognition software that detects and extracts the license plate. The cameras create the following record for every car that passes the camera:

```
public class SensorRecord {
      public String licencePlate;
      public int minute;
      public int second;
      public int cameraId;
```

The cameras are connected to a kafka messaging system.

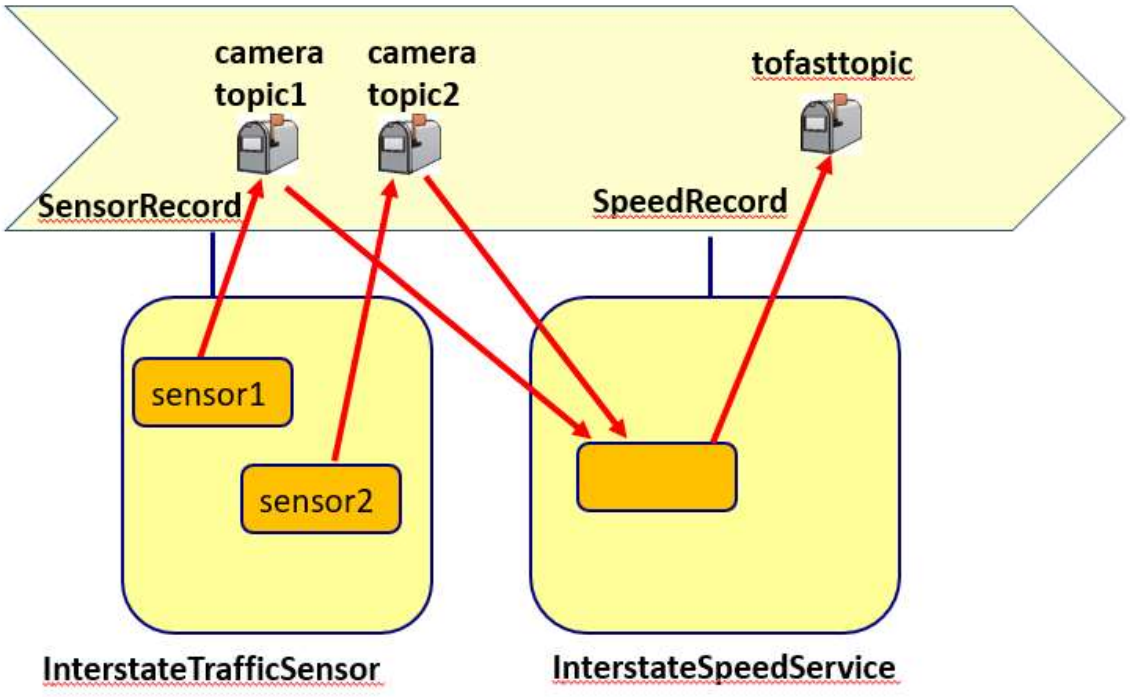
Camera1 publishes its SensorRecords into the topic **cameratopic1**
Camera2 publishes its SensorRecords into the topic **cameratopic2**

Given is the project **InterstateTrafficSensor** that simulates both cameras. If you run it, it will publish many SensorRecords into kafka.



Our first job is to calculate the speed of every car.
(speed in miles per hour = 0.5 / time in seconds *3600.
Then we only need to handle the cars that drive more than 72 miles/hour. Because our detection systems are not 100% accurate, we allow a maximum speed of 72 miles/hour.

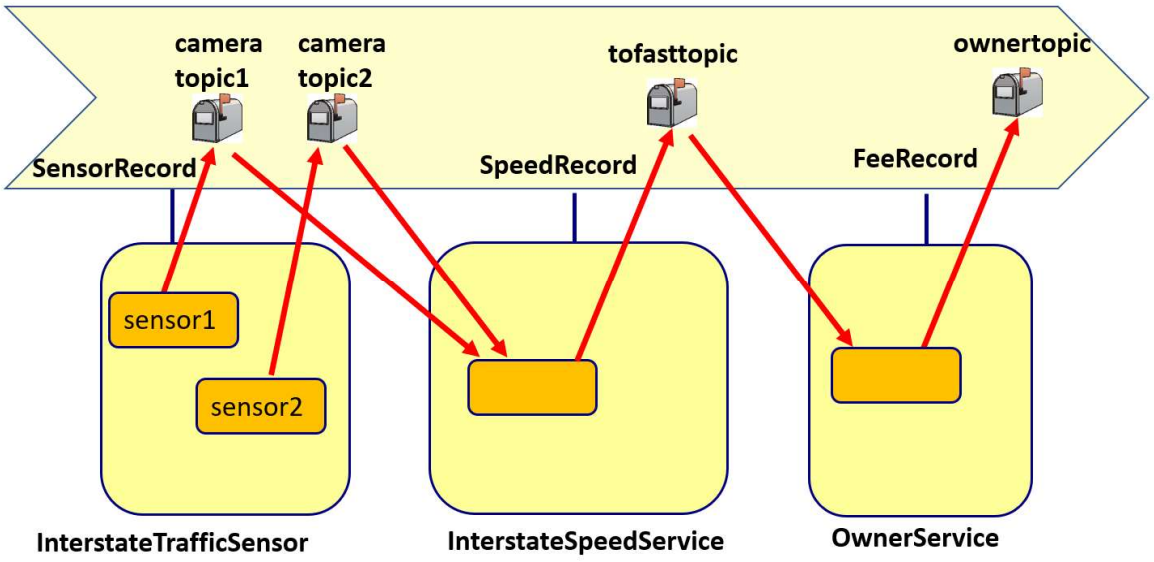
a. Write a SpeedService that writes to the console all license plates and their corresponding speed



The algorithm to compute the speed is attached to this lab

Once we know which cars drive more than 72 miles/hour, we need to know the owner information that is registered with the particular license plate.

b. Write an OwnerService that writes to the console the owners info of all cars that drive more than 72 m/h all



If we know the owner information, then we have to calculate the fee that this owner has to pay. The formula is as follows:

72 – 77 miles/hour = $ 25
77 - 82 miles/hour = $ 45
82 – 90 miles/hour = $ 80
> 90 miles/hour = $ 125

c. Write a FeeCalculatorService that writes to the console the following information of all cars that drive too fast: license plate, owner info, speed, amount of the fee

Part 3

We discussed the disadvantages of a monolith. We also looked at component based architecture (modular monolith) and microservice architecture. Explain clearly when would you choose a modular monolith and when would you choose microservice architecture.

What to hand in?

1. A zip file containing all services for part 1
2. A PDF for part 2