**Intro**

Understanding the basic web application is a great start for anyone who wants to go into development.

In the last post we talked about how Kafka can be used as a message subscribe system for handling data. Today we will be setting up a web service that acts as a producer. A consumer program will listen for data and publish to a database (MySQL will be used) Some HTML will be used to create a basic form to call our web service.

The skills I recommend are Java OOP concepts and some basic SQL.

Here is a diagram for the project we will be creating.

The data we will be receiving will be an entity to our database for a guest.

We will be using the Spring framework for our java code. One benefit of spring is that we will have package management and injection and it will auto wire our project which helps simplify things. It also will help is directly connect to our SQL database. Taking an online course on Spring would be a good idea.

We will again create 2 Java programs, one for the producer and one for the consumer. In our producer program we created a REST controller to send data over HTTP. This program also creates our Kafka producer and topic.

@RestController  
@RequestMapping("/api/v1/guest")  
public class GuestController {  
 @Autowired  
 KafkaTemplate<String, Guest> kafkaTemplate;  
  
 @RequestMapping(consumes = { MediaType.*APPLICATION\_FORM\_URLENCODED\_VALUE*})  
 public void postMessage(Guest guest) {  
 kafkaTemplate.send("quickstart-events", guest);  
 }  
}

The consumer program connects to our Kafka topic and listens for incoming data, it will then publish this data to our database which is on MySQL.

The application.properties file will contain code like this, and this can also be used for other databases services such as PostgreSQL.

spring.datasource.url=jdbc:MySQL://localhost:3306/sys  
spring.datasource.username=root  
spring.datasource.password=\*\*\*\*\*  
spring.jpa.database-platform=org.hibernate.dialect.MySQLDialect  
spring.jpa.hibernate.dll.auto = none  
spring.jpa.hibernate.show-sql=true

I was testing the HTTP request’s using Postman which helps to create and test API’s. This is done by allowing users to create and save HTTP requests and see the results.

Full code can be found here: https://github.com/creatosix/spring-boot