**Spring Boot Kafka Demo**

Kafka:

Apache Kafka is an open-source distributed event streaming platform used by thousands of companies for high-performance data pipelines, streaming analytics, data integration, and mission-critical applications.

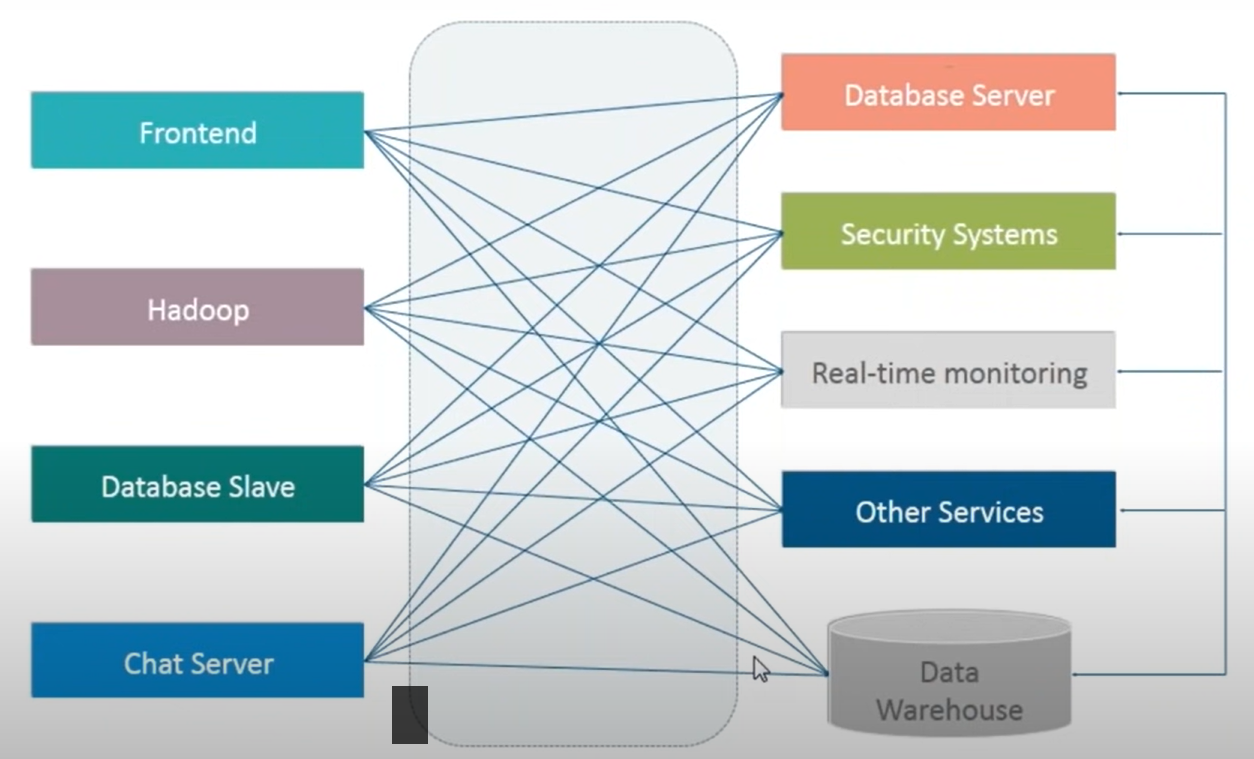
Advantages of Kafka:

1. Apache Kafka is capable of handling millions of data or messages per second.
2. Apache Kafka works as a mediator between the source system and the target system. Thus, the source system (producer) data is sent to the Apache Kafka, where it decouples the data, and the target system (consumer) consumes the data from Kafka.
3. Apache Kafka is having extremely high performance, i.e., it has really low latency value less than 10ms which proves it as a well-versed software.
4. Organizations such as NETFLIX, UBER, Walmart, etc. and over thousands of such firms make use of Apache Kafka.
5. Apache Kafka is able to maintain the fault-tolerance. Fault-tolerance means that sometimes a consumer successfully consumes the message that was delivered by the producer. But, the consumer fails to process the message back due to backend database failure, or due to presence of a bug in the consumer code. In such a situation, the consumer is unable to consume the message again. Consequently, Apache Kafka has resolved the problem by reprocessing the data.
6. Learning Kafka is a good source of income. So, those who wish to raise their income in future in IT sector can learn.

Problems :

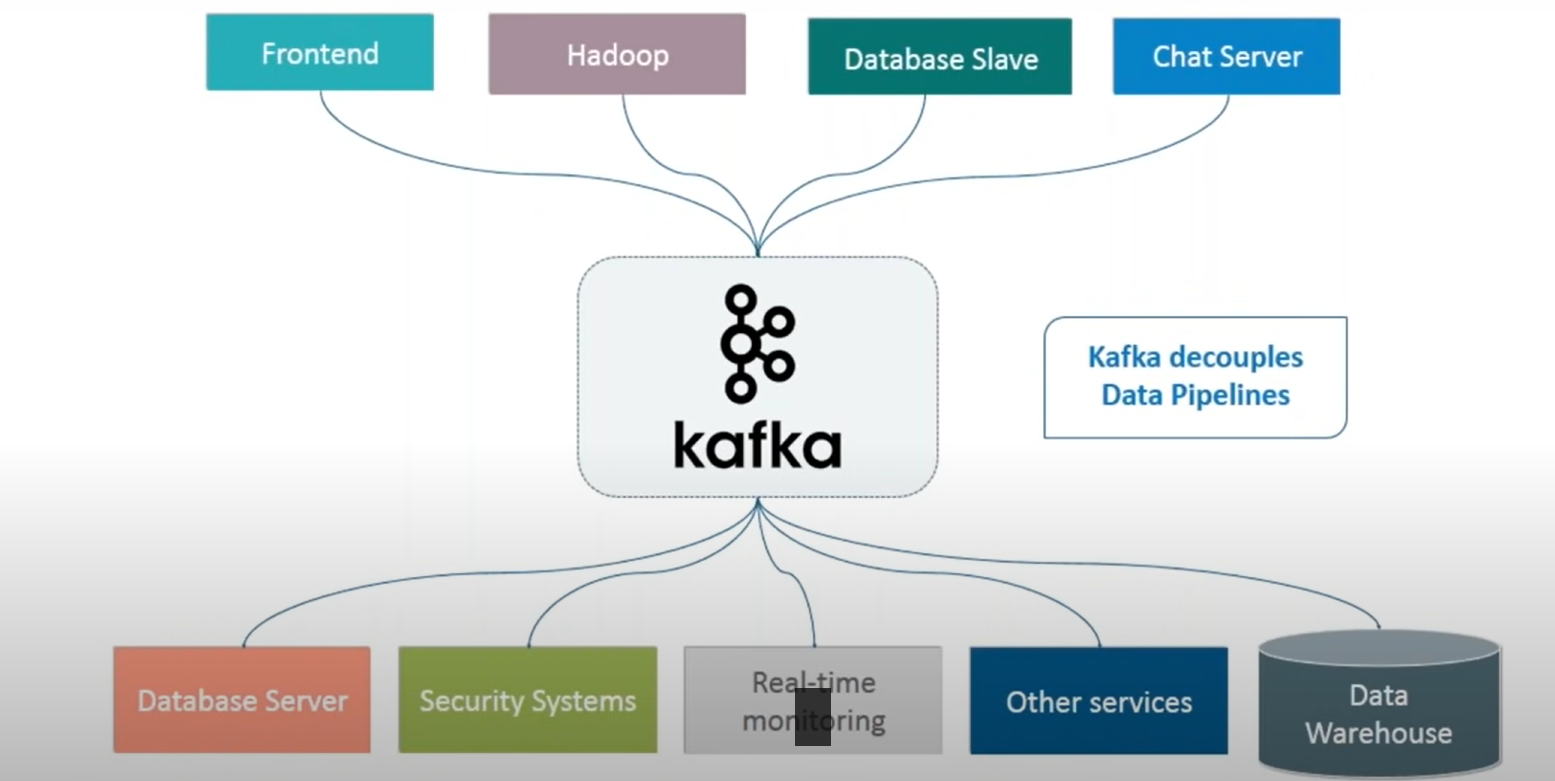
Previously to work with applications 2 or 3 servers enough. But present days number of applications

Increased and numbers of servers also increased. To transfer the data between multiple system with pipe lines very complex to handle.

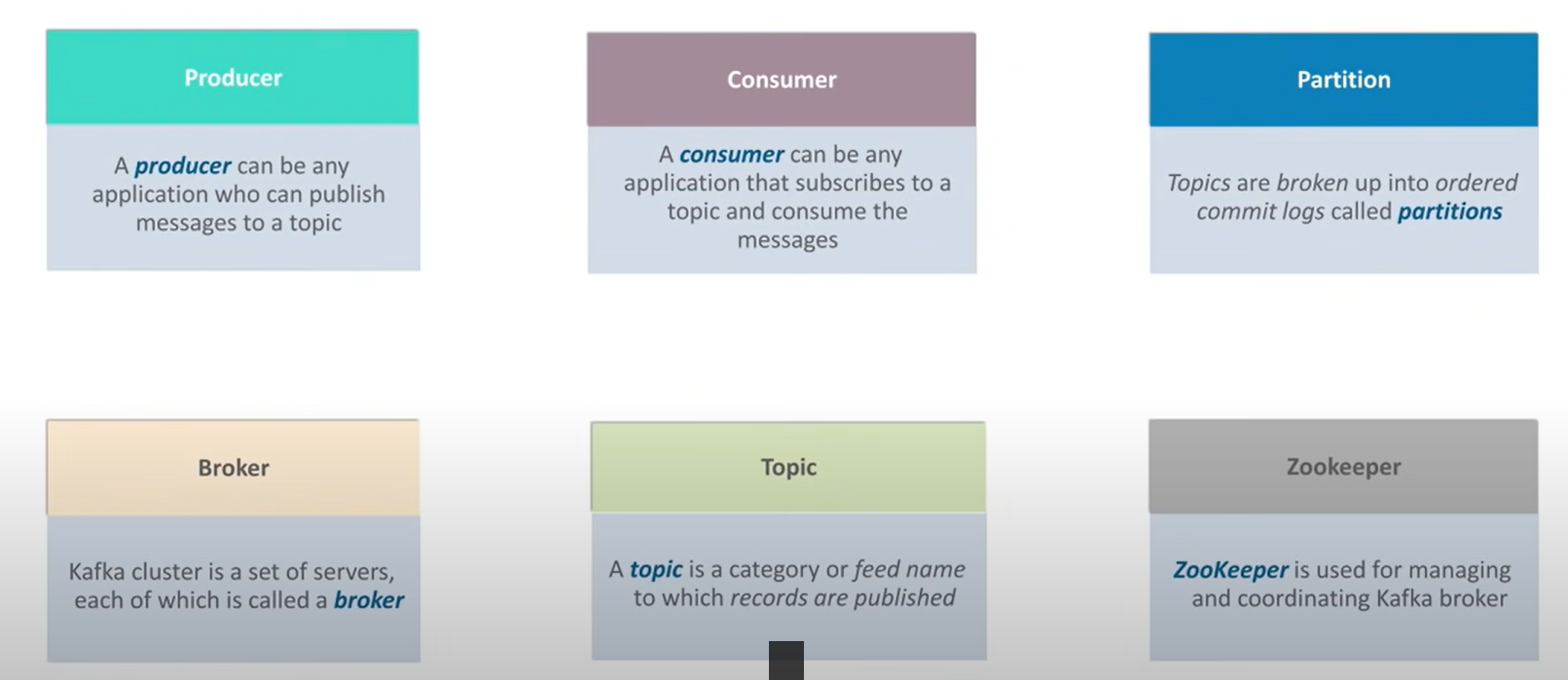


Solution:

To resolve above problem Kafka is introduced. Kafka reduces the complexity between the application to transfer the data between multiple systems. Producer produces the data to the particular topic and then Consumer consumes that related data from the particular topic.



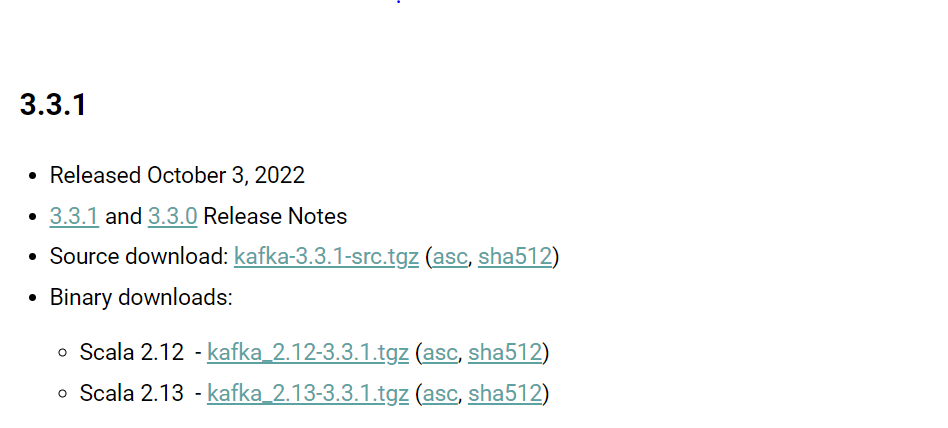
Kafka Terminologies :



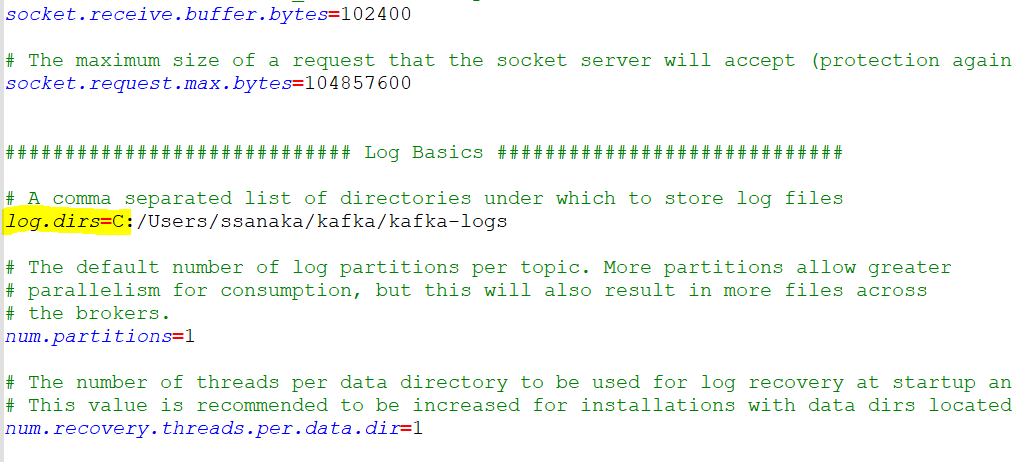
Sample Kafka Demo using Console :

1. Install Kafka server using below URL.

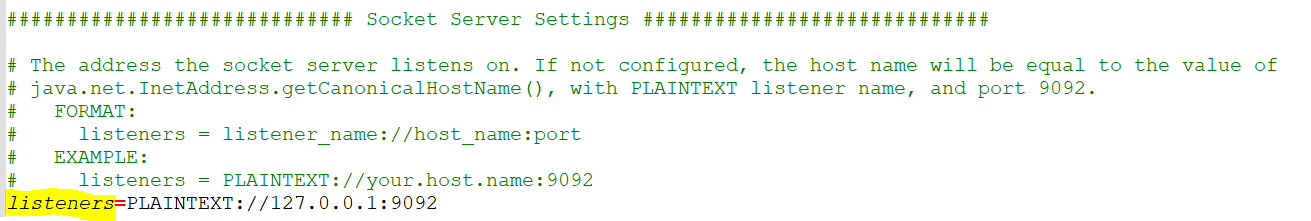
<https://kafka.apache.org/download>



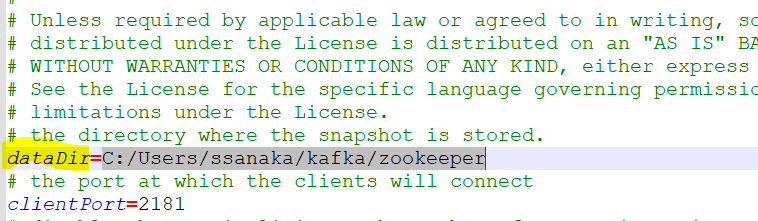
1. Extract Kafka directory.
2. Change Log Directory path in server.properties for maintaining the logs.



1. Un Comment the below highlight line in server.properties

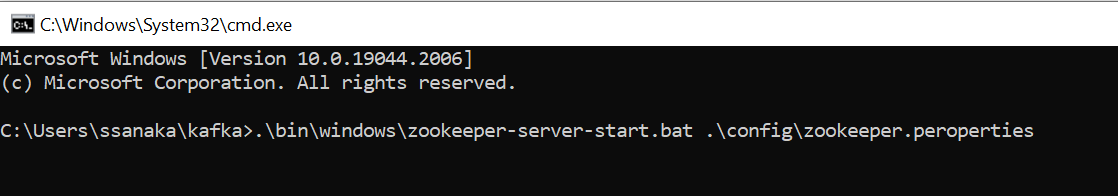


1. Change data directory path zookeeper.properties

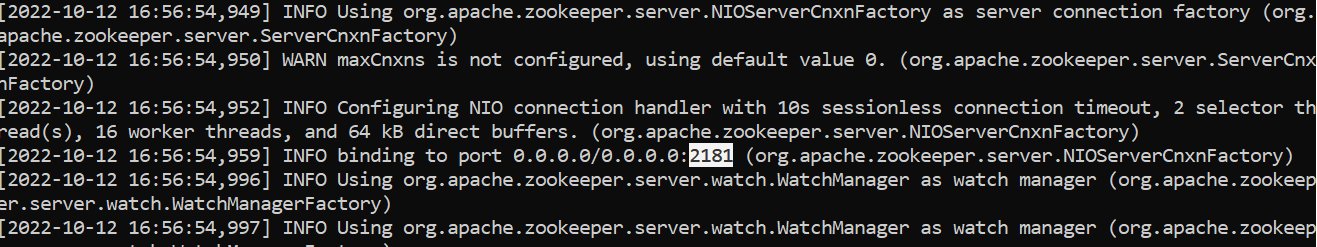


1. Navigate to Kafka Directory and run start zookeeper using below command.

* .\bin\windows\zookeeper-server-start.bat .\config\zookeeper.peroperties

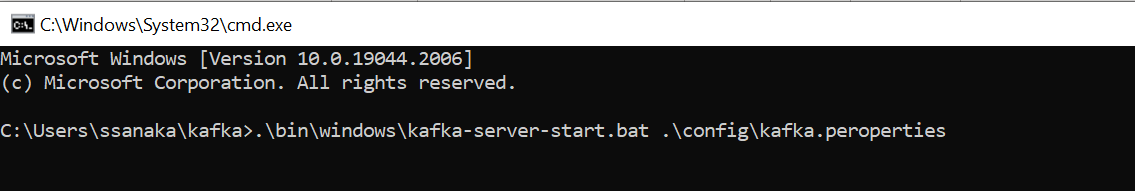


* Now Zookeeper Started and running on port no: 2181

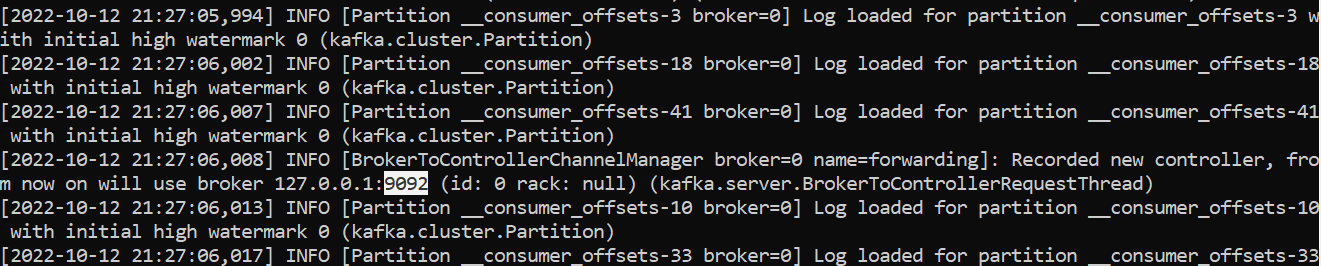


1. Start Kafka server using below command.

* .\bin\windows\kafka-server-start.bat .\config\kafka.peroperties

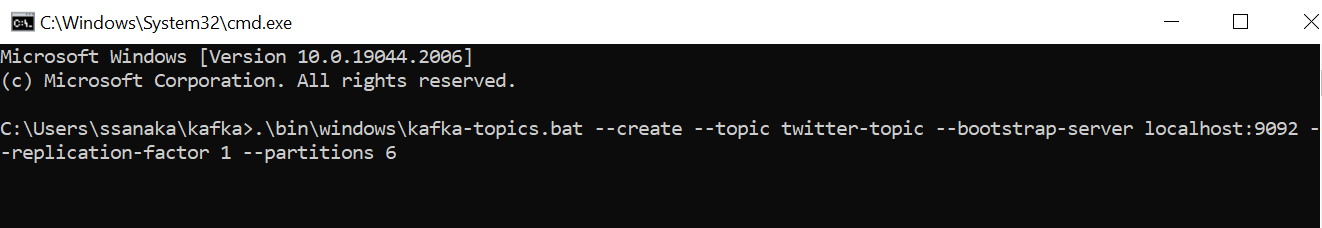


* Now Kafka server started in port no: 9092



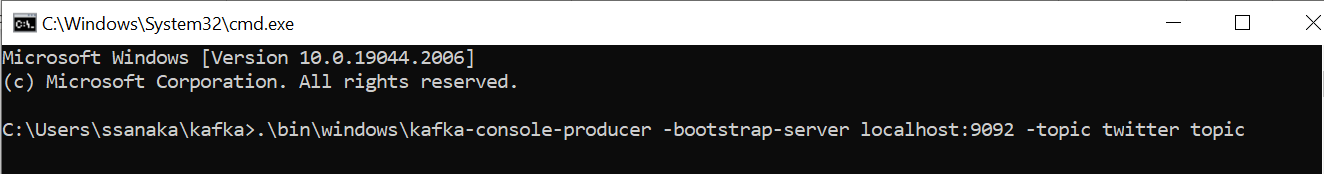
1. Create topic using below command.

* .\bin\windows\kafka-topics.bat --create --topic twitter-topic --bootstrap-server localhost:9092 --replication-factor 1 --partitions 2



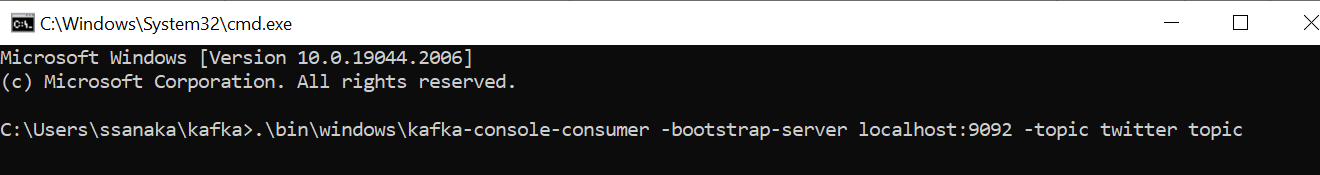
1. Create producer using below command to producer data to twitter topic.

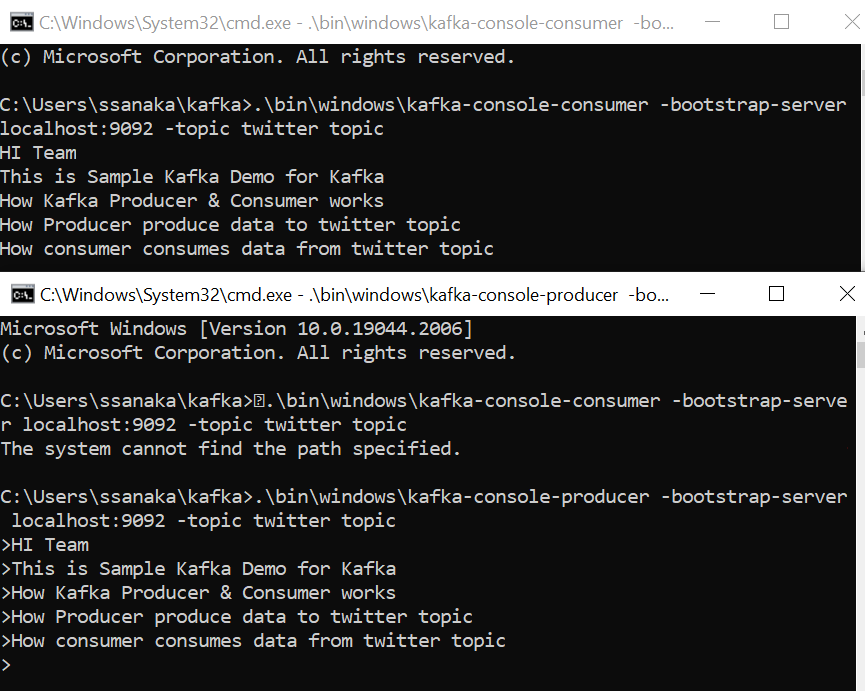
* . \bin\windows\kafka-console-producer -bootstrap-server localhost:9092 -topic twitter topic



1. Create Consumer using below command to consume data from twitter topic.

* . \bin\windows\kafka-console-consumer -bootstrap-server localhost:9092 -topic twitter topic



1. Producer producing data to twitter topic & Consumer consuming data from twitter topic.

**Spring Boot Kafka Demo**

1. Application Workflow:

###### Graphical user interface, application Description automatically generated

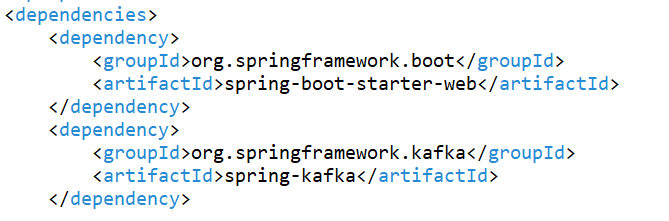


1. **Application Implementation** :

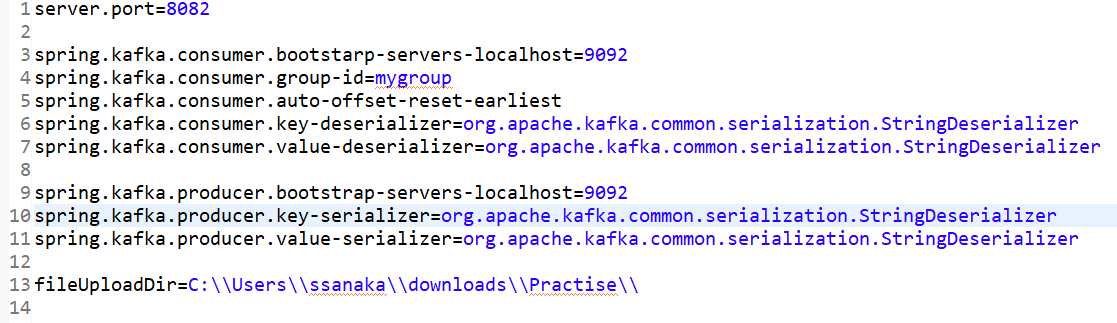
1) Implemented Spring Boot Example & Added Dependency's

a) Kafka

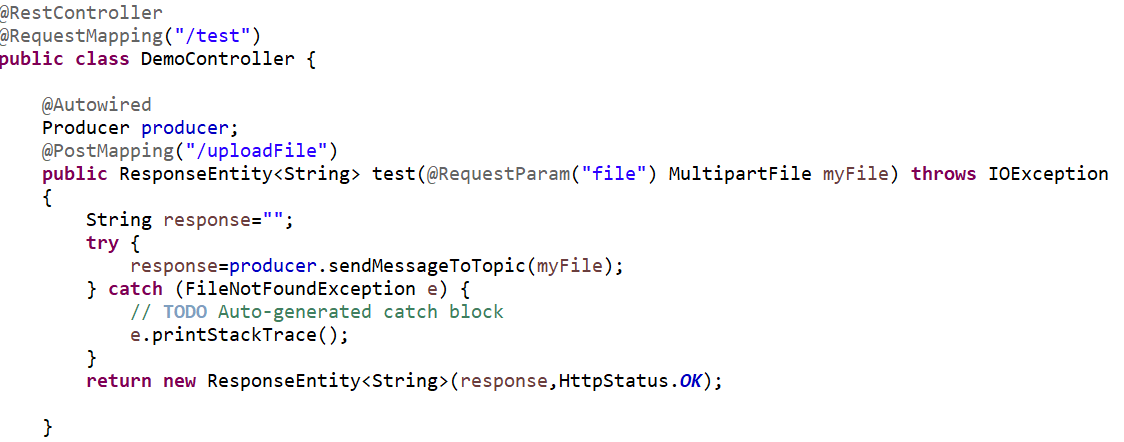
b) Spring Starter Web



2) Add configuration data in application. Properties



3) Implemented Rest Controller for accepting client request.



4) Create Producer Service and create **Kafka Producer**.

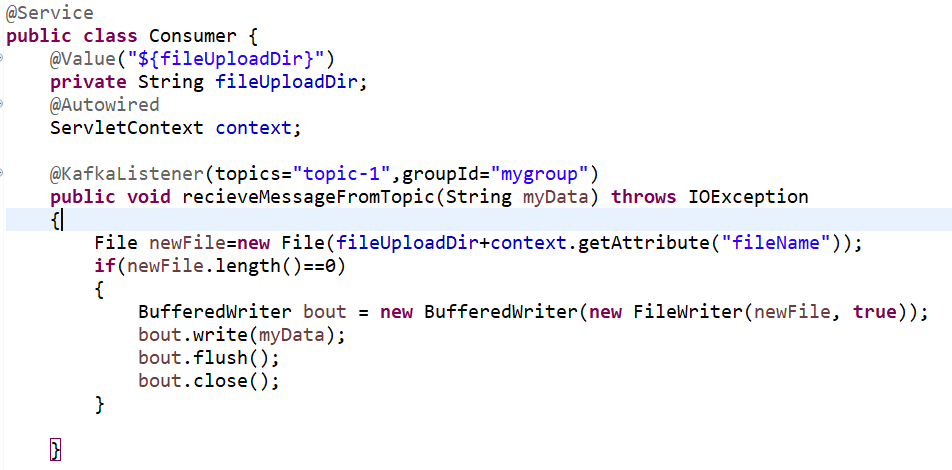
A) Use **Kafka Template** in producer to send data to topic.



5) Create Consumer Service and create **Kafka Consumer**.

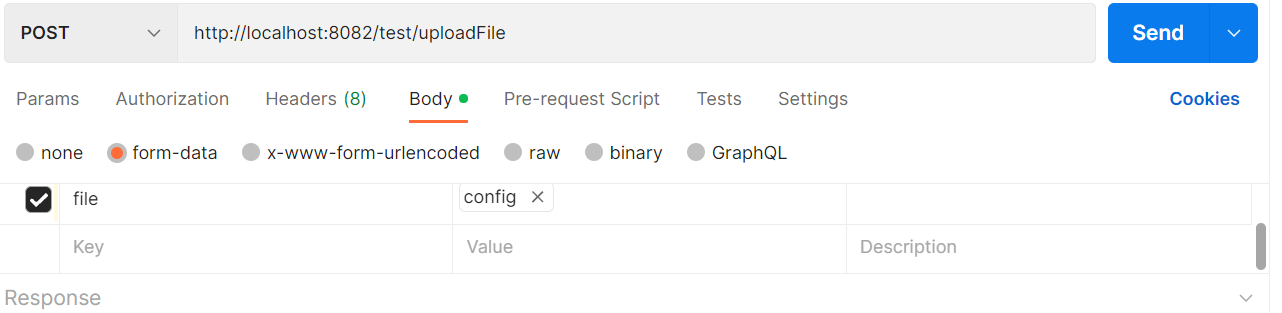
A) Consume data from **Kafka** **Topic** and store it in local folder.

B) Use **Kafka Listener** in consumer. It will automatically consume data from topic.



6) Run Spring Boot Application.

7) Open Postman and test the application.



8) Get the response.

