What is Kafka?

Kafka is a communication system between the sender and receiver. So when we want to send any data then we use Kafka.

- It works on the subscribing model.

- whatever we want to send the data we need to publish the data on kafka server and Receiver will subscribe to kafka and it listen to the kafka so receiver will receive the new data from kafka using subscription model.

- We can have the multiple receivers so it can be multiple receivers that subscribe to the kafka. So in this way kafka send the messages to different receivers whatever published by the sender on kafka.

What is the use of the Kafka?/ when to use kafka? / what is the actual advantage of using kafka? In which type of application we can use kafka?

Example:

Let's suppose we have a cab booking application and you will be a user and you 3want to book a cab. when once you try to book a cab that request will go to the server and based on your location or based on the cabs location one driver is assigned to you. once a driver is assigned you will get a notification that your driver/cab is assigned it will come to pick you in that time and we will get the constant update about the location of the driver after assigning cab.

How the location update of the driver get to the user?

Once the driver get the notification and driver start the ride to pick the user. AT particular time interval(consider every 1 second) there is an update of the location and that updated location is stored into the database of the application and from that database user get the updated location of the driver.for every time interval it is picking the updated data from the database.

But consider it is about only 1 ride but it might be Hundreds or Thousands of the the ride booked at the same time then for every user and every driver if we doing the same operation again and again into the DB in every time interval in that case it will not work, it's not going to scale very good because DB does not allow you to do those kind of opertaions we cannot have humongous operations on the database. So, for that we can have one intermediate application which can allow us to do each and everything. So instead of this DB we have a messaging system that is Apache Kafka.

Apache kafka is working on the publisher and subscriber model. So,from the drivers application the car driver is sending/publishing data of each and every moment while going to the user. Those data is stored in the one of the instances/topics of the kafka. On the same time user is also wants to know about his cab so the user application will subscribe the kafka or its particular instance in which cab drivers data is stored. By doing this whatever the updated location or messages is set by the driver in kafka instace those will be listen by the users application and user will updated by the drivers location.

Apache kafka is a distrivuted system it is not a centralised data system, it is distributed along the different servers, different replicas and different clusters so whatever number of load we have apache kafka will be able to handle that load or it will be able to serve the traffic very easily.

**Advantages of Apache kafka:**

- High Throughput

- Fault Tolerance(Kafka follows replication technique which created the multiple copies of data and stores in the multiple notes in the distributed)

- Durable(No data will be lost because it is stored in multible notes in the distributed system)

- Scalable (Using HighThroughput, fault tolerance or many replicas across the distributed system it will become scalable suppose in our e commerce app any sale is going on in that case extra clusters or extra resources or servers is required in that case kafka can manage it automatically according to our need so it is scalable)

-

#Architecture of Apache Kafka:

As we know Apache kafka is having a sender Application and receiver application and in between the sender and receiver we will having our Apache kafka.

Kafka is consist of two components

1.Kafka Cluster

2. Zookeper

Both kafka cluster and zookeper is the paret of our kafka ecosystem

----------------- | ----------------------------| --------------------

| Sender | | |---------------------- | | | |

| | | | | | |Receiver |

----------------- | | KafkaCluster | | --------------------

| | | |

| |-----------------------| |

| |

| |---------------------- | |

| | | |

| | Zookeeper | |

| | | |

| |-----------------------| |

|-----------------------------|

Kafka Ecosystem is consist of Kafka cluster and Zookeeper.

Kafka cluster is consist of Multiple Brokers.(Act as a database)

Brokers consists of Multiple Topics(Act as a database table to organize/categorize data)

Inside topics we are using partitions to store the data that data will be managed with the help of offset(offset means data will be managed and stored in the form of array indexing in the partitions)

What is Zookeeper?

Let's take an example of Zoo, In zoo there are many animals for those animal there is a zoo keeper who manages those animals he will keep all the details about animal like food cycle, which animal eat which thing, timings of its eating and Rest all thing same like that Zookeeper in apache kafka ecosystem manages all the things(metadata) like

How many clusters are available?, How many sender and receivers are there?, How many replicas are there?

All those thing are handled by our zookeeper. So, entire kafka cluster is managed by our zookeeper. so kafka's job is only to get the data and send the data but the metadata around it is managed by the zookeeper

Inside the kafka there are several brokers like broker 1, broker2 etc and inside those brokers topics/Instances are present or we create. In those topics our data will be stored.

Inside those topics multiple partitions are present in those partition the data is stored in the form of an array.

#How to install kafka in our system?

1. Goto browser and search for Kafka install

2. Apache kafka quickstart -> GetStarted -> Quickstart ->Download Kafka

3. Open Terminal and goto/locate download directory

4. Run this command to unzip the downloaded file: $ tar -xzf kafka\_2.13-3.6.1.tgz

5. Use this command to see the unzipped folder: $ cd kafka\_2.13-3.6.1

JDK 8 plus is rerquired

# How to run kafka with zookeeper?

In the downloaded directory within bin folder there is lots of sh files are present which we run directly if we are working on Mac or linux. But if we are working on windows then there is seprate folder for windows we need to go into windows directory and run bat files because windows will work with the bat files

-> Run the following command in order to start all services in correct order

To start Zookeeper service

-> $ bin/zookeeper-server-start.sh config/zookeeper.properties

For Windows

->C:\Users\AkashMaurya\Desktop\kafka\_2.13-3.6.1>bin\windows\zookeeper-server-start.bat config\zookeeper.properties

Zookeeper is running... on specified or default port.

Open another terminal session in the kafka folder and run:

Start the Kafka broker service

-> $ bin/kafka-server-start.sh config/server.properties

-> C:\Users\AkashMaurya\Desktop\kafka\_2.13-3.6.1>bin\windows\kafka-server-start.bat config\server.properties [windows(if we want to change port we change in the config file )]

bydefault kafka is running on the port: 9092

----------------------------------------------------------------------------------

Once both the server get started then we will perform all these operation using below tools

1. create topics with the help of kafka-topics tool

2. Produce example message with kafka-console-producer

3. Consume the message with kafka-console consumer

Open Another terminal in the kafka folder

1.Create a topic to store your events

-> bin\windows\kafka-topics.bat --create --topic quickstart-events --bootstrap-server localhost:9092

By running this command we will get all the commands related to kafka that we can use.

2. bin\windows\kafka-topics.bat --create --topic user-topic --bootstrap-server localhost:9092

Created topic user-topic.

3. bin\windows\kafka-acls.bat --help

for more commands

4. write come events into topics

Here we will use kafka-console-producer and we will tell the topic to create producer in the topic

bin\windows\kafka-console-producer.bat --topic user-topic --bootstrap-server localhost:9092

>This is my first event

>This is my second event

These messages are reach to whom who subscribed the kafka server that is consumer.

After creating producer we will open a new terminal in the kafka folder in which we will create a consumer who receives/ consumes those messages which are produced by the producer

5. Read the Events

bin\windows\kafka-console-consumer.bat --topic user-topic --from beginning --bootstrap-server localhost:9092

>This is my first event

>This is my second event

We can stop it Using CTRL+C

NOTE: We can create multiple consumers in different terminals which will receives the messages simultaneously from producer.

Usecase:

We will create 2 microservices one microservice will communicate with the another microservice with the help of Kafka.

We should create the same scenerio that we create using terminal except cluster(we need to run server mannually from the terminal)

- We can create topics in our project

- we can produce data in the the topic

Project:

Zomato Live Location update (using kafka server)

Consume Message Produce Message

END USER APP<---------------------APACHE KAFKA <-----------------------DELIVERY APP

| ^

| |

------------------------------------

Subscribe

We have 2 Microservices so we need to create 2 springboot projects.

1. Enduser App

2. Delivery App

3. Kafka server

So, In this application we will produce the message from Delivery App and consume the message from Enduser App and we will give the location update in the message. once the location of Delivery boy updated we will manually call the API and hit the url and the location will be updated and a message is produced in the topic and the enduser will listen that message from the topic because it subscribe that topic on kafka cluster.

After Completing Kafka Producer We Will create Consumer Application.



**ENDUSER APP**



* **Create a springboot maven application adding dependencies like:**



* Spring Web
* Kafka

A screenshot of a computer

Description automatically generated

* **Open application.properties file to add the configurations**

1. **Add server.port (because delivery app is already running on default port 8080 so we need to change the default port for User application)**

**server. port=8081**

1. **Add bootstrap (Kafka consumer server) which is running on localhost:9092**

**spring.kafka.consumer.bootstrap-servers=localhost:9092**

1. **Now we will add the group Id this group Id is a value of a String which identifies consumer group (In which group that consumer is available)**

**spring.kafka.consumer.group-id=group-1**

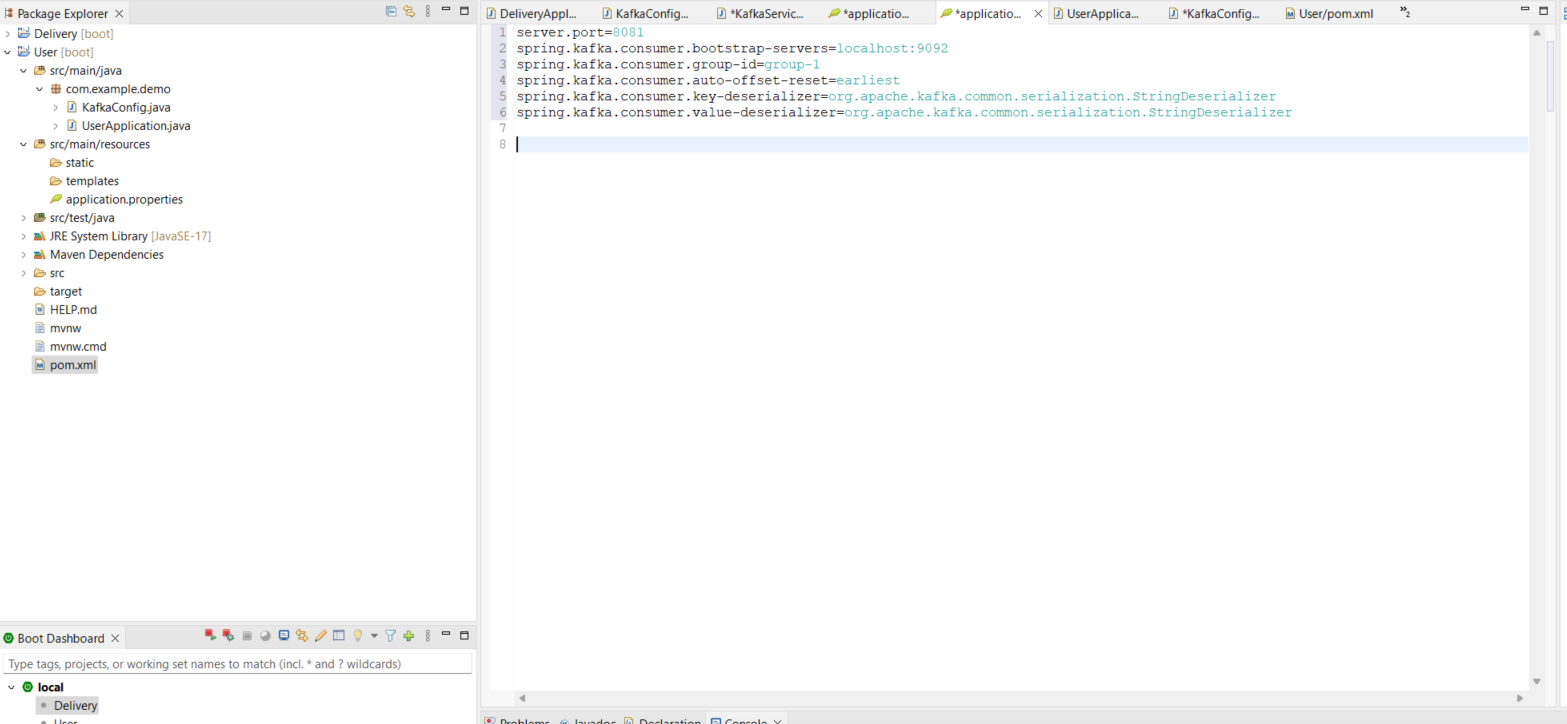
1. **Add the offset-reset which means when the offset is reaches to its last limit then by resetting itself where it will go? So will we keep it as “earliest” that means after reset itself offset comes to its initial state.**

**spring.kafka.consumer.auto-offset-reset=earliest**

1. **Now, we will add deserializer (for String) for Key and Value pair.**

**spring.kafka.consumer.key-deserializer=org.apache.kafka.common.serialization.StringDeserializer**

**spring.kafka.consumer.value-deserializer=org.apache.kafka.common.serialization.StringDeserializer**

****

* **Now, We can write the code for consumer in Main class or we can create a separate class**

**package** com.example.demo;

**import** org.springframework.context.annotation.Configuration;

**import** org.springframework.kafka.annotation.KafkaListener;

@Configuration

**public** **class** KafkaConfig {

@KafkaListener (topics = AppConstants.***LOCATION\_UPDATE\_TOPIC***, groupId=AppConstants.***GROUP\_ID***)

**public** **void** updatedLocation(String value) {

System.***out***.print(value);

}

}

Inside the class we will create a method which execute whenever messages are produced in that method, we take a parameter String value which contains the message which are coming from to producer and we are printing that message or saving into our DB

That method works with the help of kafka listener annotation. we will pass the topics and groupIds that we want to subscribe whenever any message produced in that subscribed topic from that groupId Listener annotation execute that menthod to save or print the message in DB or console.

But we required an annotation i.e, @KafkaListener Annotation that marks a method to be the target of a Kafka message listener on the specified topics.

* **Created an App constant class in which add Constant terms which we are using in above class method annotation**

**public** **class** AppConstants {

**public** **static** String *LOCATION\_UPDATE\_TOPIC*="location-update-topic";

**public** **static** String *GROUP\_ID*="group-1";

}

**RUN the EndUser and DeliveryBoy Application Simultaneously and check the messages on Console.**