Project Milestone-- Data Ingestion Software-- Kafka Clusters
Faazil Shaikh - 100707829

Answer the following questions:

1. **What is EDA? What are its advantages and disadvantages?**
   - EDA Stands for Event Driven Architecture
   - Utilizes events(changes in state) to communicate across services

   Advantages:

   - High scalability due to being triggered based on states and therefore does not require any sort of polling which helps lower CPU costs
   - Lower latency also because it is an event triggered system and there isn't any sort of unnecessary polling happening reducing the amount of network transactions
   - Can simplify logic as you can simply trigger any other event

   Disadvantages:

   - Slightly more difficult to debug when having multiple events trigger each other as you have to track them all
   - Can have inconsistencies depending on the nature of the event trigger

2. **In Kafka, what's meant by cluster, broker, topic, replica, partition, zookeeper, controller, leader, consumer, producer, and consumer group?**

Cluster - consisting of one possibly multiple kafka brokers

Broker - The system that holds the pub data and also has one or more partitions per topic.

Topic - Stream of messages belonging to a specific category, they are also divided into partitions

Replica - a copy used as a backup or a failsafe

Partition - Subset(s) of data from a topic so it can work with any amount of data

Zookeeper - Maintains a list of topics and messages while monitoring the state of cluster nodes

Controller - One broker in a cluster would be in charge of keeping track of the many replica and partition states.

Leader - A broker who is the owner of a partition in a particular cluster.

Consumer - receives messages in that subject from various partition subsets

Producer - Creator of messages to many topics

Consumer Group - Consumers who consume data from topics as a collective group

3. **Follow the following video to install Kafka into your local machine and create topics, consumers, and producers using Kafka's built-in tools.**

https://drive.google.com/file/d/13xUMqloJ1chQJOOUONfC3OUP6QsqZah-/view?usp=sharing

4. **Follow the following video to generate NodeJS scripts for creating topics, consumers, and producers**

https://drive.google.com/file/d/13xUMqloJ1chQJOOUONfC3OUP6QsqZah-/view?usp=sharing

5. **Prepare a video showing the codes that generated topics, produce messages, and consume them in both NodeJS and python. Your video should display the possible producer and consumer scenarios.**

https://drive.google.com/file/d/13xUMqloJ1chQJOOUONfC3OUP6QsqZah-/view?usp=sharing

6. **A problem in the used YAML file to create the docker images is that the data inside Kafka clusters are not persistent which means if the docker images are down, all its messages are lost. Update the YAML file for persistent data (hint: it's related to the volume options in Kafka brokers and zookeeper). Describe how this update solves the problem.**

volumeMounts:

- mountPath: "./"

  name: p-storage

```
  volumes:
    - name: p-storage
      persistentVolumeClaim:
        claimName: p-storage-claim
```

We create a persistent volume and a claim for it which allows access to the storage

7. **Follow the following video about Kafka in Confluent Cloud and use the shown CLI to create a topic, consumer, and producer. Also update your python code to create a consumer, and producer using Kafka in Confluent Cloud (hint: only the connection information of Kafka Cluster has to be updated). Record a video illustrating those tools and showing them in action**

https://drive.google.com/file/d/13xUMqloJ1chQJOOUONfC3OUP6QsqZah-/view?usp=sharing