Name: Tegveer Singh
ID: 100730432

What is EDA? What are its advantages and disadvantages?

In an event-driven architecture, events or updates/changes in component state are used trigger communication between microservices. It is really common in enterprise applications like Netflix and e-commerce websites

Advantages:

- Highly scalable due to decoupled services
- Highly responsive to applications
- No data loss due to backtracking through event logs
- Fault-tolerant and allows for root cause analysis in failures

Disadvantages:

- Over-engineering of services which is not required at times
- Error handling is difficult due to a large number of publishers and subscribers

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

Cluster:: A Kafka cluster is a group of brokers/servers that manage the Kafka pub/sub mechanism by handling its producers and Consumers
Broker: A Broker is a server inside of the Kafka cluster that manages offset assignment, committing of published messages to a disk as well handling the data fetch requests from consumers/subscribers of the data
Topic: Data in a broker is categorized into different topics. Topics can be scaled across multiple servers based on their categorization into partitions
Replica: A replica is a copy of a topic partition created to handle backups in case of broker failures. They increase data redundancy
Partition: Topics can be broken down into partitions. Partitions are separated through an offset(message key) number within the topic. Partitions act as ordered event logs and exist to increase scalability and throughput. To guarantee that all messages from a single user enter the same partition within a certain topic, we could simply use:
User_id % Number of partitions within the topic
Batches: A batch is a collection of messages produced for the same partition within a topic
Zookeeper: Zookeeper stores the most recent offset of message in a partition fetched by a consumer. This handles the order of consumption and makes sure consumer doesn't lose data when offline
Controller:: Clusters contain a broker that acts as a controller in the cluster. The controller is responsible for administrative tasks like assigning partitions to different brokers and monitoring broker failures.
Leader:: A broker acts as a leader for the partition it contains. In case of a broker failure, a new leader is assigned to the partition by the controller

Consumer: Analogous to subscribers and make fetch data requests for a topic from a broker
Producer: Analogous to publishers and send data under a particular topic to a broker
Consumer Group: Consumers are organized into consumer groups and all consumers in a group usually work towards consumption of a particular topic. This can be achieved in many ways and the most common way to do so is by assigning a single partition to a single consumer. This assists in creating a backup and scaling consumers horizontally.

Kafka with Node JS Video:
https://drive.google.com/file/d/1Ul8Eb-VGyBiGIOpvcMF_Iyi0t3JhJQu8/view?usp=sharing
Kafka with Python Video:
https://drive.google.com/file/d/1_SHXDcUARK9Kao2dHWqyEWOhltf5Cf_s/view?usp=sharing

Change to add persistence in the Kafka brokers and zookeeper
Under the broker configurations, add the following option

```
volumes:
      -
C:/users/tegve/Projects/Cloud-Computing-Milestone-2:/kafka/kafka-logs
:/var/lib/kafka/data
```

This specified file can act as logs for Kafka events