Project Milestone-- Data Ingestion Software-- Kafka Clusters

EDA

- Event-driven architecture is a software design model for applications with data processing.
- In such a system, information between components is communicated using events.
- Event producers emit events that act as a message consumers can detect.
- Event consumers listening can detect events that are emitted by producers and react accordingly.

Advantages

- Loose coupling is implemented by default in such a system because event producers have no idea who the consumers are and what they are listening to.
- Because of loose coupling between the producer and consumer, faults may happen in the system and it won't affect productivity. That is because if a fault happens in the consumer, the producer does not know and does not get affected.
- The system is a lot more extensible because new consumers can be added without affecting the whole system. Updates to consumers or producers can be done without affecting the rest of the system.

Disadvantages

- With such a system, the responsibility falls completely on the producers to emit the correct events. A lot of effort is required to ensure the integrity of the events emitted by producers which is not a simple task.
- In large scale systems, it gets difficult to keep track of messages between hundreds of producers and consumers. Common issues that may arise like duplicates names of consumers and producers and duplicate events.
- Because of the complexity of such a system, a lot of effort is required to monitor event flow and handle errors as they arise.

Kafka

- **Cluster**: group of servers running Kafka. It provides data replication.
- **Broker**: represents a single server running Kafka. They are managed by a Zookeeper.
- **Topic**: a subject name that is used to categories records in Kafka.
- **Replica**: a copy of original data used for fault tolerance.
- **Partition**: splitting of event logs into partitions to allow them to work independently.
- **ZooKeeper**: manages Kafka brokers, topics and messages in a cluster.
- **Controller**: in a cluster, one of the brokers is assigned as a controller. It assumes the administrator role as it manages partitions and replicas.
- **Leader**: one server is assigned a leader in the partition. It is responsible for processing read and write requests for the partition.
- **Consumer**: subscribes to topics to consume events that are produced.
- **Producer**: produces events to a topic in Kafka.
- **Consumer group**: consumers working together to consume events from several topics.