Project Milestone-- Data Ingestion Software-- Kafka Clusters

- EDA
  - An event is something that happens
  - Can be communicated in the form of an event notification
  - Can contain data associated with the even or it can just be a notification that something happened
  - An event is immutable
  - Event is the base of what is called event driven architecture (EDA)
  - EDA has three components: producer, broker, consumer
  - A producer creates the events that are going to be redirected by a broker into the right consumers
  - The consumers will react to this event and execute the things that they need to execute
  - Also referred to as publish/subscribe model
- Advantages
  - Allows you to decouple different components
  - Allows you to invert dependencies
  - Allows you to scale better
  - Events can be persisted and in this way the information can be retrieved even at later stages and not only during the call
  - The system is more robust because there is no single point of failure
- Disadvantages
  - Utilizing a broker adds some performance hit
  - Broker can be weak at keeping data synced between producers and consumers which causes eventual consistency
  - More complexity because it's hard to track what happens and when it happens and how the communication lines are happening throughout the system
- Cluster
  - Kafka runs in a cluster consisting of one or more servers
- Broker
  - A single Kafka server within the cluster
- Topic
  - When messages are sent to a broker, they’re sent to a specific topic
  - Topics provide a way of categorizing data that is being sent
  - They can be further weakened into a number of partitions
- Partition
  - Partitions allow for parallelizing a topic by dividing the data in a specific topic across multiple brokers
  - Each partition is placed on a separate node to allow for multiple consumers to read from a topic in parallel
- Leader
  - When communication is taking place with a kafka cluster, the messages are sent to the partition’s leader

        - The leader is the one responsible for writing the message to its own in sync replica and when that message is committed, the leader propagates the message to additional replicas on different brokers
- Replica
    - In the case that a leader fails, a replica would take over as the new leader
- Controller
    - The kafka controller is responsible for detecting the loss of the leader and will elect a new leader from the pool of in sync replicas
- Zookeeper
    - Maintains list of active brokers
    - Elects controller
    - Manages configuration of the topics and partitions
- Consumer
    - Pulls messages off of a Kafka topic
- Producer
    - Pushes messages into a Kafka topic
- Consumer group
    - For a given topic, consumers can be organized into consumer groups
    - Each consumer in the consumer group reads from a unique partition and the group as a whole consumes all messages from the entire topic