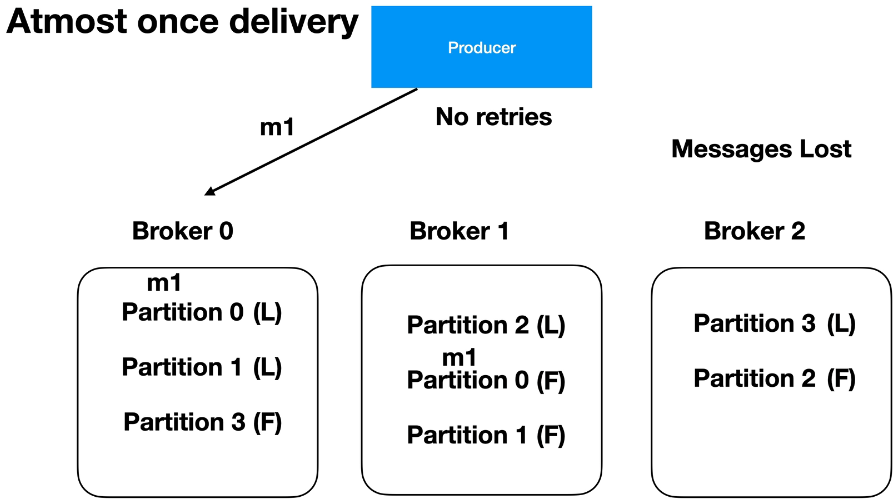
1. Agenda:
   1. We will cover Message Delivery Idempotency that is no duplicate msgs.
2. Kafka Producer API, along with Kafka Broker, supports the following 3 Message Delivery Semantics.
   1. At least once: **Default**
   2. At most once.
   3. Only once/idempotency : we want to achieve this one.  
      Graphical user interface, text, application

      Description automatically generated
3. **At Least Once Delivery**:
   1. Producer sends a msg to Broker.
   2. Broker writes the msg to leader partition and all the follower partitioners (if partition is replicated).  
      Broker treats it committed and at this point, it sends an acknowledgement back to the producer and everyone (partitions) is happy.
   3. But the problem with this is that if broker tries to send an acknowledgement to the producer and the acknowledgement gets failed due to some reason, the producer API tries to send the same back again.  
      If you’re okay with the message duplication and reprocessing of the same message in the use case you’re working with, you can go with the **default** one which is at least once delivery.  
      Diagram

      Description automatically generated
4. **At Most Once Delivery**:
   1. Producer sends msg only once.
   2. In case Broker is not able to write to all the partitions, then msg is lost.
   3. Configuration: By setting retries to 0.  
      
5. **Exactly Once Semantics**: **Skipped**
   1. This is what we desire most of the time.
   2. We enable it idempotency by using a producer configuration called **enbable.idempotenc**=true.
   3. Producer.send() will generate a unique sequence number for every message.  
      So, it is unique within a given partition inside a broker.
   4. Broker will do also some magic.
      1. It maintains the msg seq number.
      2. It also generates a unique producer ID for each producer instance which will be used by the producer in the msg every time it sends a msg.
      3. So broker ensures that