# Week 7 Assignment: Diving into Apache Kafka

## Objective:

To gain a foundational understanding of Kafka, focusing on topic creation, producing and consuming messages, and conducting performance tests.

### *1. Environment Initialization*

- Navigate to the Kafka directory and start the Docker containers:

```
cd kafka
```

- Start Kafka

```
docker-compose up -d
```

- Open **two** terminal sessions and in each, access the Kafka container:

```
docker exec -it kafka_kafka_1 bash
```

- `If you can’t access the Kafka container, it could be due to a container name change. In this cause use:`

```
docker exec -it kafka-kafka-1 bash
```

### *2. Topic Creation and Verification in Kafka (On One Terminal Only)*

**Exercise 1:** Create a Kafka topic named ‘my-topic’.

```
/opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-topic --bootstr
ap-server localhost:9092
```

**Exercise 2:** List the topics to verify that ‘my-topic’ has been successfully created.

```
/opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --list --bootstrap-server localhost
:9092
```

**Deliverable:** Screenshot showing the ‘my-topic’ listed amongst the topics.

### *3. Producing and Consuming Messages in Kafka*

**Exercise 3:** In the first terminal, start a Kafka consumer:

```
/opt/kafka_2.13-2.8.1/bin/kafka-console-consumer.sh --topic my-topic --from-b
eginning --bootstrap-server localhost:9092
```

**Exercise 4:** In the second terminal, start a Kafka producer:

```
/opt/kafka_2.13-2.8.1/bin/kafka-console-producer.sh --topic my-topic --bootst
rap-server localhost:9092
```

- Type some text into the producer and press ‘Enter’. Note that the text appears on the consumer terminal.

**Deliverable:** Screenshots of the producer terminal with your entered text and the consumer terminal showing the received message.

To exit the console and producer shells type `CTRL + C`.

Close your second terminal.

### *4. Kafka Performance Tests*

**Exercise 5:** Run a performance test on the producer using the Kafka producer performance test script with provided arguments:

```
/opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --topic my-topic --num-
records 50000 --record-size 100 --throughput 1000 --producer-props bootstrap.
servers=localhost:9092 key.serializer=org.apache.kafka.common.serialization.S
tringSerializer value.serializer=org.apache.kafka.common.serialization.String
Serializer
```

**Exercise 6:** Following the producer test, run a consumer performance test on ‘my-topic’:

```
/opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost
:9092 --topic my-topic --messages 50000
```

**Deliverable:**

- Screenshots of both the producer and consumer performance test results.
- Discuss the meaning of the results.

### *5. Expanding Kafka and Running Additional Performance Tests*

- Exit the Kafka container and scale Kafka instances to 3:
  ```
  exit
  docker-compose scale kafka=3
  ```
- Re-enter the `kafka_kafka_1` container:
  ```
  docker exec -it kafka_kafka_1 bash
  ```
- `If you can’t access the Kafka container, it could be due to a container name change. In this cause use:`
  ```
  docker exec -it kafka-kafka-1 bash
  ```

**Exercise 7:** Create a topic, this time partitioned and replicated across all three Kafka instances:

```
/opt/kafka_2.13-2.8.1/bin/kafka-topics.sh --create --topic my-partitioned-top
ic --replication-factor 3 --partitions 3 --bootstrap-server localhost:9092
```

**Exercise 8:** Conduct the producer and consumer performance tests on the new topic, observing differences:

```
/opt/kafka_2.13-2.8.1/bin/kafka-producer-perf-test.sh --topic my-partitioned-
topic --num-records 50000 --record-size 100 --throughput 1000 --producer-prop
s bootstrap.servers=localhost:9092 key.serializer=org.apache.kafka.common.ser
ialization.StringSerializer value.serializer=org.apache.kafka.common.serializ
ation.StringSerializer
```

Followed by:

```
/opt/kafka_2.13-2.8.1/bin/kafka-consumer-perf-test.sh --broker-list localhost
:9092 --topic my-partitioned-topic --messages 50000
```

**Deliverable:**

- Screenshots of the performance tests on the partitioned topic.
- Include your observations on performance variations between a single Kafka instance and the scaled setup.

## Shutting Down

Ensure all Docker containers are turned off with `docker-compose down` for each directory. If you're using google cloud, please shut down your virtual machine to preserve cloud costs.