Name: Tay Hui Chun

Matric No: A0170109N

Task D

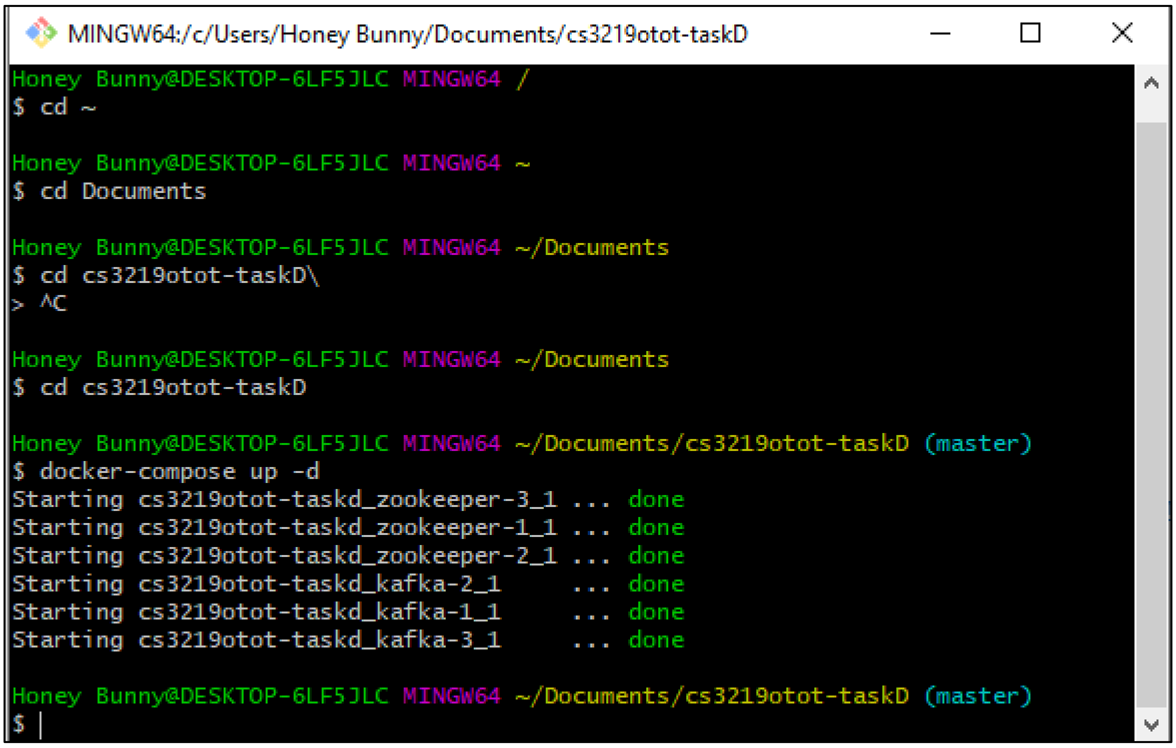
GitHub Repository Link: https://github.com/huichun66/cs3219otot-taskD

Referenced Tutorial Link: https://medium.com/better-programming/kafka-docker-run-multiple-kafka-brokers-and-zookeeper-services-in-docker-3ab287056fd5

Instructions:

1) `docker-compose up -d`

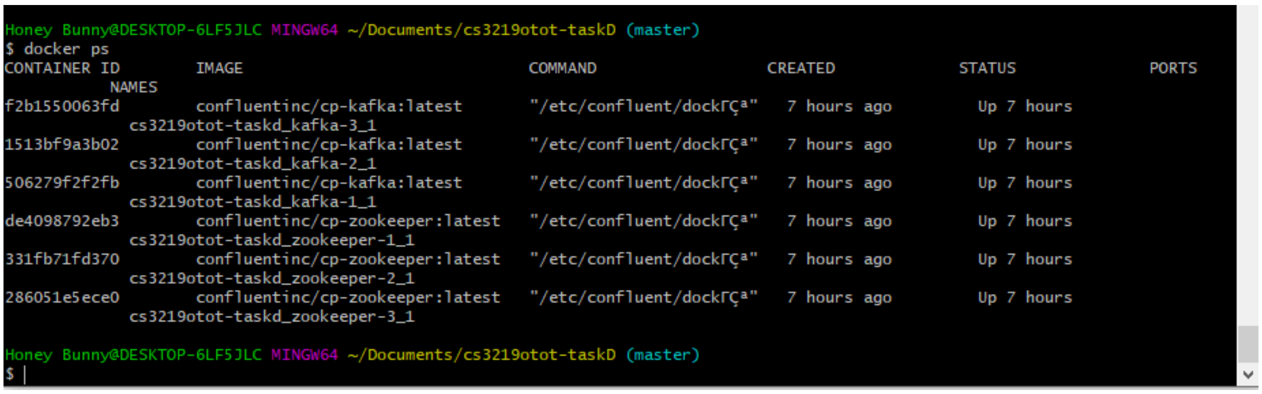
Execute the above command to start the docker services for the 3-node Kafka cluster.



```
Honey Bunny@DESKTOP-6LF5JLC MINGW64 /
$ cd ~

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~
$ cd Documents

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents
$ cd cs3219otot-taskD\
> ^C

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents
$ cd cs3219otot-taskD

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker-compose up -d
Starting cs3219otot-taskd_zookeeper-3_1 ... done
Starting cs3219otot-taskd_zookeeper-1_1 ... done
Starting cs3219otot-taskd_zookeeper-2_1 ... done
Starting cs3219otot-taskd_kafka-2_1     ... done
Starting cs3219otot-taskd_kafka-1_1     ... done
Starting cs3219otot-taskd_kafka-3_1     ... done

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ 
```

2) `docker ps`

The following screenshot shows the 3-node Apache Kafka cluster that is managed by a zookeeper ensemble after executing `docker ps` command.



```
Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker ps
CONTAINER ID        IMAGE                              COMMAND                  CREATED             STATUS              PORTS
           NAMES
f2b1550063fd        confluentinc/cp-kafka:latest       "/etc/confluent/dockΓÇª"   7 hours ago         Up 7 hours
         cs3219otot-taskd_kafka-3_1
1513bf9a3b02        confluentinc/cp-kafka:latest       "/etc/confluent/dockΓÇª"   7 hours ago         Up 7 hours
         cs3219otot-taskd_kafka-2_1
506279f2f2fb        confluentinc/cp-kafka:latest       "/etc/confluent/dockΓÇª"   7 hours ago         Up 7 hours
         cs3219otot-taskd_kafka-1_1
de4098792eb3        confluentinc/cp-zookeeper:latest   "/etc/confluent/dockΓÇª"   7 hours ago         Up 7 hours
         cs3219otot-taskd_zookeeper-1_1
331fb71fd370        confluentinc/cp-zookeeper:latest   "/etc/confluent/dockΓÇª"   7 hours ago         Up 7 hours
         cs3219otot-taskd_zookeeper-2_1
286051e5ece0        confluentinc/cp-zookeeper:latest   "/etc/confluent/dockΓÇª"   7 hours ago         Up 7 hours
         cs3219otot-taskd_zookeeper-3_1

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ |
```

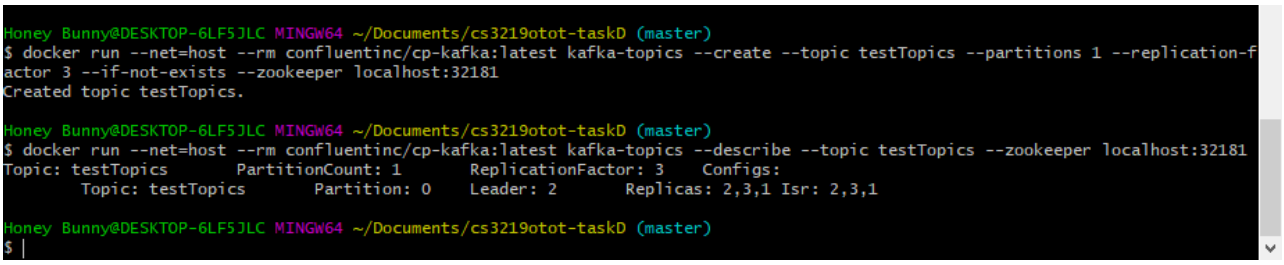
3) `docker logs <zookeeper_container_id>`

`docker logs <kafka_container_id>`

Execute the above commands to check the logs to see if the Zookeeper ensemble is ready/Kafka brokers have booted up successfully.

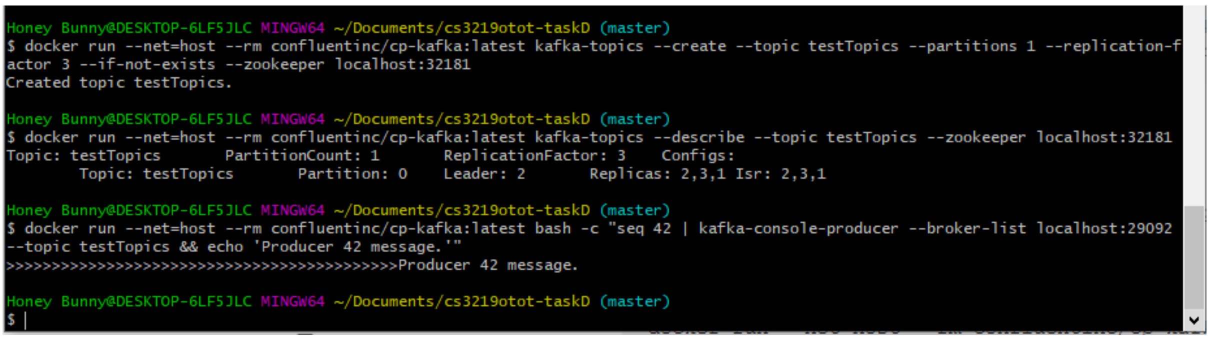
4) `docker run --net=host --rm confluentinc/cp-kafka:latest kafka-topics --create --topic <topic_name> --partitions <Number_of_partitions> --replication-factor <number_of_replication_factor> --if-not-exists --zookeeper localhost:32181`

The command above is to test if the broker is working as expected. We test if they are working by creating a topic. Then we verify if the topic is created successfully by describing the topic.



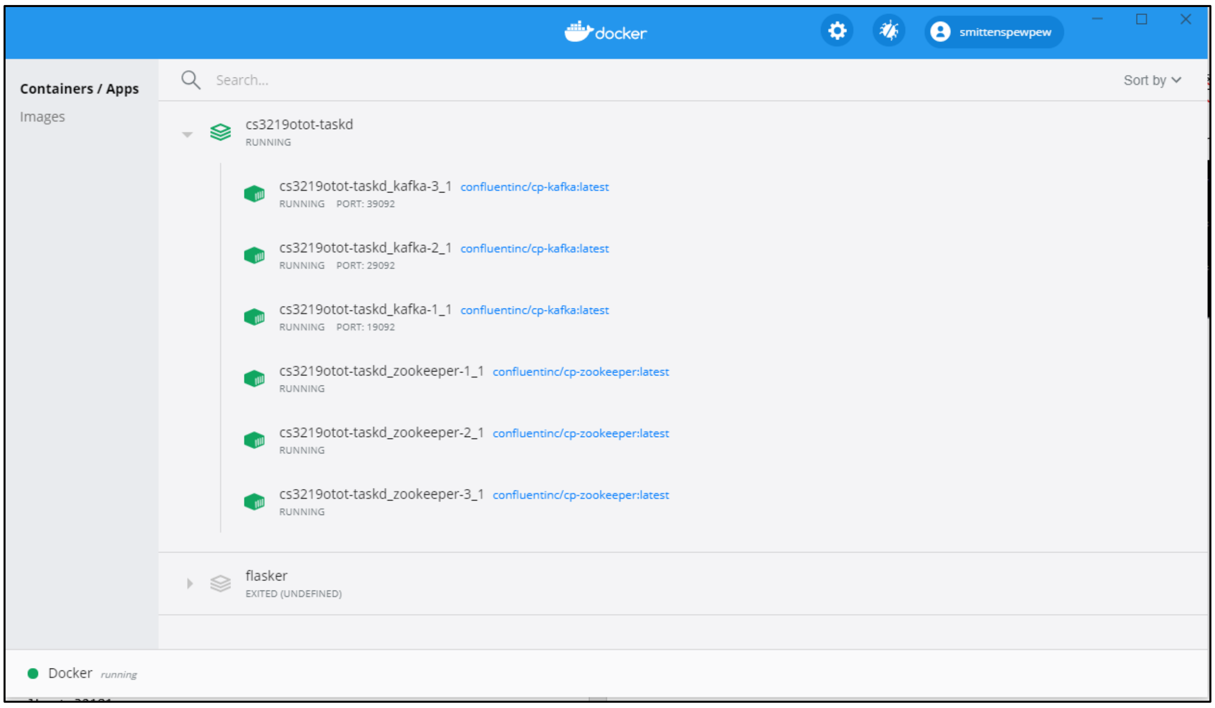
```
Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker run --net=host --rm confluentinc/cp-kafka:latest kafka-topics --create --topic testTopics --partitions 1 --replication-f
actor 3 --if-not-exists --zookeeper localhost:32181
Created topic testTopics.

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker run --net=host --rm confluentinc/cp-kafka:latest kafka-topics --describe --topic testTopics --zookeeper localhost:32181
Topic: testTopics       PartitionCount: 1       ReplicationFactor: 3    Configs:
        Topic: testTopics       Partition: 0    Leader: 2       Replicas: 2,3,1 Isr: 2,3,1

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ |
5) docker run --net=host --rm confluentinc/cp-kafka:latest bash -c “seq 42
| kafka-console-producer --broker-list localhost:29092 --topic testTopic
&& echo ‘Producer 42 message.’”
```

Afterwards, we try to generate some data to the topic that we have created. The command above will pass 42 integers using the console producer that is shipped with Kafka.



```
Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker run --net=host --rm confluentinc/cp-kafka:latest kafka-topics --create --topic testTopics --partitions 1 --replication-f
actor 3 --if-not-exists --zookeeper localhost:32181
Created topic testTopics.

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker run --net=host --rm confluentinc/cp-kafka:latest kafka-topics --describe --topic testTopics --zookeeper localhost:32181
Topic: testTopics       PartitionCount: 1       ReplicationFactor: 3    Configs:
        Topic: testTopics       Partition: 0    Leader: 2       Replicas: 2,3,1 Isr: 2,3,1

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ docker run --net=host --rm confluentinc/cp-kafka:latest bash -c "seq 42 | kafka-console-producer --broker-list localhost:29092
--topic testTopics && echo 'Producer 42 message.'"
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>Producer 42 message.

Honey Bunny@DESKTOP-6LF5JLC MINGW64 ~/Documents/cs3219otot-taskD (master)
$ |
```

<u>Docker Desktop Screenshot</u>



cs3219otot-taskd
RUNNING

cs3219otot-taskd_kafka-3_1 confluentinc/cp-kafka:latest
RUNNING PORT: 39092

cs3219otot-taskd_kafka-2_1 confluentinc/cp-kafka:latest
RUNNING PORT: 29092

cs3219otot-taskd_kafka-1_1 confluentinc/cp-kafka:latest
RUNNING PORT: 19092

cs3219otot-taskd_zookeeper-1_1 confluentinc/cp-zookeeper:latest
RUNNING

cs3219otot-taskd_zookeeper-2_1 confluentinc/cp-zookeeper:latest
RUNNING

cs3219otot-taskd_zookeeper-3_1 confluentinc/cp-zookeeper:latest
RUNNING

flasker
EXITED (UNDEFINED)

Docker *running*