

M1 SA PART 2

I. Running Kafka Zookeeper

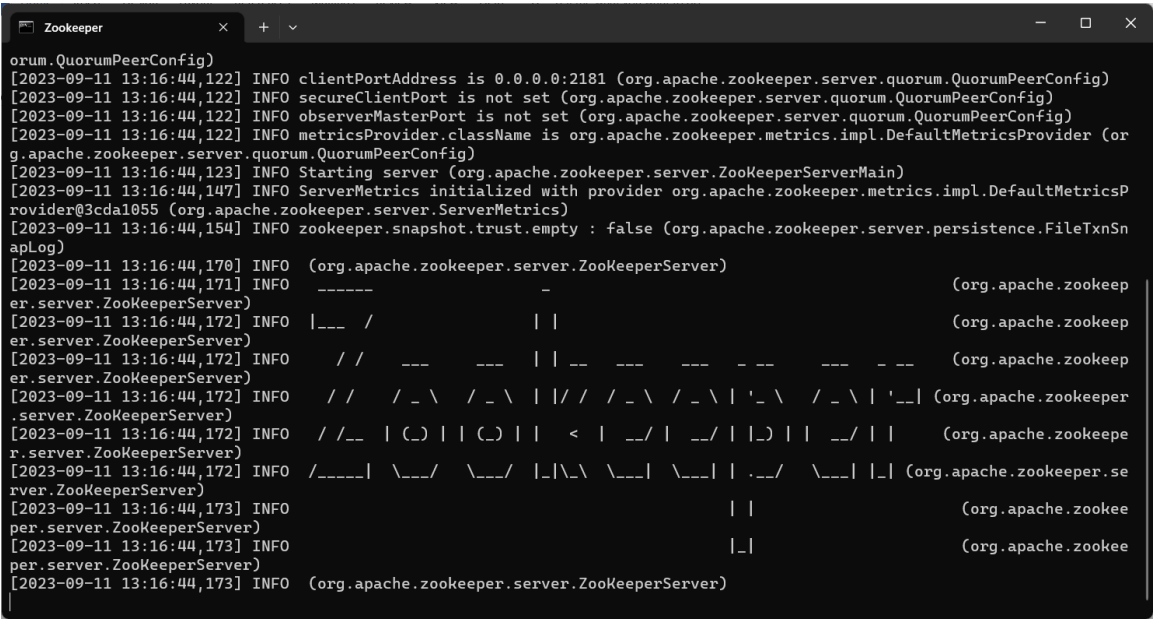
Open command line (CLI). Access the kafka folder using cd command. In this case, kafka is accessed through the command **cd C:\kafka**

To start the zookeeper, access the zookeeper-server-start.bat file along with the zookeeper.properties file. The following command is run on the CLI to start the zookeeper:

```
.\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties
```

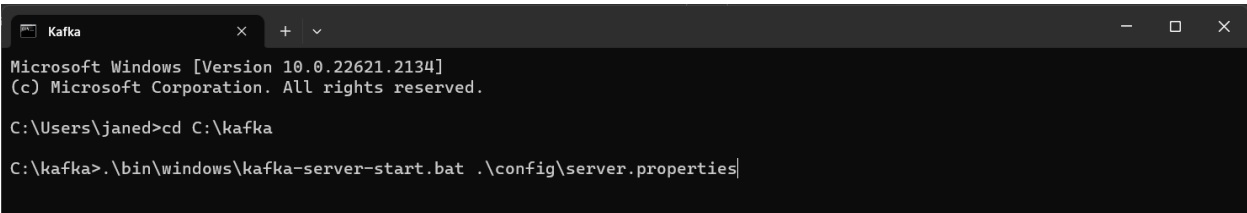


The zookeeper should now start. The output is shown below.

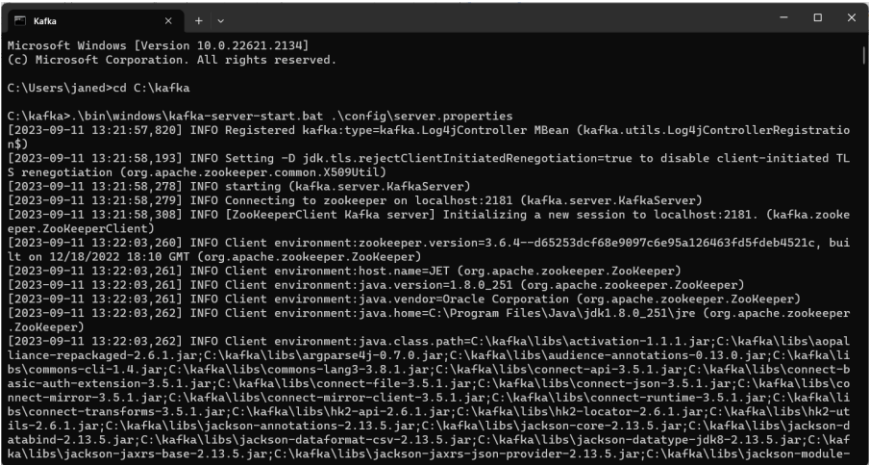


II. Running Kafka Server

Open a new CLI. Access the kafka folder with the cd command. Access the kafka-server-start.bat file along with server.properties file to run the kafka server. The command **.\bin\windows\kafka-server-start.bat .\config\server.properties** is entered in the CLI.



The kafka server now starts after executing the command. The output is shown below.



### III. Create Topic

A topic must first be created before data can be produced by the producer and consumed by the consumer. A kafka topic is a virtual group where messages are contained in proper order.

Open a new CLI and access the windows folder within the bin folder within the kafka folder. In this case, the command `cd C:\kafka\bin\windows` is executed.

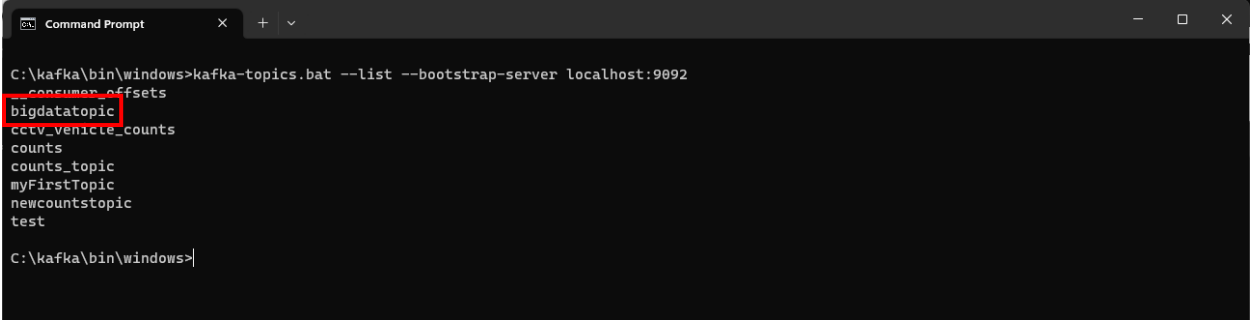
Execute the command below to create a new topic. `--topic bigdatatopic` specifies that the topic name is bigdatatopic.

**kafka-topics.bat --create --bootstrap-server localhost:9092 --replication-factor 1 --partitions 1 --topic bigdatatopic**



Verify that the topic **bigdatatopic** is created by executing the command below. The topic created is shown in the list.

**kafka-topics.bat --list --bootstrap-server localhost:9092**



### IV. Run Producers

A python script is used to run a producer instance for producing messages in the kafka server. The python script of producer1 instance with sensor\_01 is shown below. The script encompasses the following

#### Libraries and Modules:

**KafkaProducer:** class from kafka used to create the producer instance

**datetime:** provides date and time utilities

**time:** used to access sleep method in the script to incorporate delay in sending messages

**random:** used for generating random numbers

**uuid:** used for creating unique IDs for each message sent

The producer instance is create using `KafkaProducer()` and assigned to `kafka_producer_obj`. The producer instance is assigned to send messages to the bigdatatopic within the loopback 127.0.0.1:9092. A message\_list is created and a message variable is initialized to create and send the messages. A loop iterates 3600 times. In each iteration, a message is created containing the following: **timeuuid\_id, lgu\_code, sensor\_id, date\_saved, time\_saved, total, car, bus, truck, jeepney, bike, tryke, others**. Thus, 3600 messages will be produced by the producer. The message is printed in the console. The message is then sent to the topic **bigdatatopic** using the `kafka_producer_obj.send()` where it can be consumed by the consumer. The `time.sleep(1)` code ensures that there is a 1 second delay between creating and sending messages. Below is an instance of a producer, **Kafka Producer1** with **sensor\_01**.

```

1  from kafka import KafkaProducer
2  from datetime import datetime
3  import time
4  from json import dumps
5  import random
6  import uuid
7
8  KAFKA_TOPIC_NAME_CONS = "bigdatatopic" #topic
9  KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
10
11  if __name__ == "__main__":
12      print("Kafka Producer1 Application Started ... ")
13      kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
14                                         value_serializer=lambda x: dumps(x).encode('utf-8'))
15
16      message_list = []
17      message = None
18      for i in range(3600):
19          i = i + 1
20          date_today = datetime.now()
21          message = {}
22          print("Preparing message: " + str(i))
23
24          car = random.randint(a: 0, b: 4)
25          bus = random.randint(a: 0, b: 2)
26          truck = random.randint(a: 0, b: 2)
27          jeepney = random.randint(a: 0, b: 2)
28          bike = random.randint(a: 0, b: 5)
29          tryke = random.randint(a: 0, b: 3)
30          others = random.randint(a: 0, b: 2)

```

```

32      total = car + bus + truck + jeepney + bike + tryke + others
33
34      message["timeuuid_id"] = str(uuid.uuid1())
35      message["lgu_code"] = '1200'
36      message["sensor_id"] = 'sensor_01'
37      message["date_saved"] = str(date_today.strftime('%m/%d/%Y'))
38      message["time_saved"] = str(date_today.strftime("%X"))
39      message["total"] = total
40      message["car"] = car
41      message["bus"] = bus
42      message["truck"] = truck
43      message["jeepney"] = jeepney
44      message["bike"] = bike
45      message["tryke"] = tryke
46      message["others"] = others
47
48
49      print("Message: ", message)
50
51      kafka_producer_obj.send(KAFKA_TOPIC_NAME_CONS, message)
52      time.sleep(1)
53
54      print("Kafka Producer Application Completed. ")

```

## Producer2 Instance with sensor\_02

```

11  KAFKA_TOPIC_NAME_CONS = "bigdatatopic" #topic
12  KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14  if __name__ == "__main__":
15      print("Kafka Producer2 Application Started ... ")
16      kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                         value_serializer=lambda x: dumps(x).encode('utf-8'))

```

```

41      #send data
42      message["timeuuid_id"] = str(uuid.uuid1())
43      message["lgu_code"] = '1210'
44      message["sensor_id"] = 'sensor_02'

```

Producer3 Instance with sensor\_03

```
11     KAFKA_TOPIC_NAME_CONS = "bigdatatopic"    #topic
12     KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14     if __name__ == "__main__":
15         print("Kafka Producer3 Application Started ... ")
16         kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                             value_serializer=lambda x: dumps(x).encode('utf-8'))
18
19         #send data
20         message["timeuuid_id"] = str(uuid.uuid1())
21         message["lgv_code"] = '1220'
22         message["sensor_id"] = 'sensor_03'
```

Producer4 Instance with sensor\_04

```
11     KAFKA_TOPIC_NAME_CONS = "bigdatatopic"    #topic
12     KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14     if __name__ == "__main__":
15         print("Kafka Producer4 Application Started ... ")
16         kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                             value_serializer=lambda x: dumps(x).encode('utf-8'))
18
19         #send data
20         message["timeuuid_id"] = str(uuid.uuid1())
21         message["lgv_code"] = '1230'
22         message["sensor_id"] = 'sensor_04'
```

Producer5 Instance with sensor\_05

```
11     KAFKA_TOPIC_NAME_CONS = "bigdatatopic"    #topic
12     KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14     if __name__ == "__main__":
15         print("Kafka Producer5 Application Started ... ")
16         kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                             value_serializer=lambda x: dumps(x).encode('utf-8'))
18
19         #send data
20         message["timeuuid_id"] = str(uuid.uuid1())
21         message["lgv_code"] = '1240'
22         message["sensor_id"] = 'sensor_05'
```

Producer6 Instance with sensor\_06

```
11     KAFKA_TOPIC_NAME_CONS = "bigdatatopic"    #topic
12     KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14     if __name__ == "__main__":
15         print("Kafka Producer6 Application Started ... ")
16         kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                             value_serializer=lambda x: dumps(x).encode('utf-8'))
18
19         #send data
20         message["timeuuid_id"] = str(uuid.uuid1())
21         message["lgv_code"] = '1250'
22         message["sensor_id"] = 'sensor_06'
```

Producer7 Instance with sensor\_07

```
11 KAFKA_TOPIC_NAME_CONS = "bigdatatopic" #topic
12 KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14 if __name__ == "__main__":
15     print("Kafka Producer7 Application Started ... ")
16     kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                       value_serializer=lambda x: dumps(x).encode('utf-8'))

41 #send data
42 message["timeuuid_id"] = str(uuid.uuid1())
43 message["lgv_code"] = '1260'
44 message["sensor_id"] = 'sensor_07'
```

Producer8 Instance with sensor\_08

```
11 KAFKA_TOPIC_NAME_CONS = "bigdatatopic" #topic
12 KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14 if __name__ == "__main__":
15     print("Kafka Producer8 Application Started ... ")
16     kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                       value_serializer=lambda x: dumps(x).encode('utf-8'))

41 #send data
42 message["timeuuid_id"] = str(uuid.uuid1())
43 message["lgv_code"] = '1270'
44 message["sensor_id"] = 'sensor_08'
```

Producer9 Instance with sensor\_09

```
11 KAFKA_TOPIC_NAME_CONS = "bigdatatopic" #topic
12 KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14 if __name__ == "__main__":
15     print("Kafka Producer9 Application Started ... ")
16     kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                       value_serializer=lambda x: dumps(x).encode('utf-8'))

41 #send data
42 message["timeuuid_id"] = str(uuid.uuid1())
43 message["lgv_code"] = '1280'
44 message["sensor_id"] = 'sensor_09'
```

Producer10 Instance with sensor\_10

```
11 KAFKA_TOPIC_NAME_CONS = "bigdatatopic" #topic
12 KAFKA_BOOTSTRAP_SERVERS_CONS = '127.0.0.1:9092'
13
14 if __name__ == "__main__":
15     print("Kafka Producer10 Application Started ... ")
16     kafka_producer_obj = KafkaProducer(bootstrap_servers=KAFKA_BOOTSTRAP_SERVERS_CONS,
17                                       value_serializer=lambda x: dumps(x).encode('utf-8'))

41 #send data
42 message["timeuuid_id"] = str(uuid.uuid1())
43 message["lgv_code"] = '1290'
44 message["sensor_id"] = 'sensor_10'
```

To run a producer, access the python virtual environment where the python producer script is executed using the cd command. In this case, **C:\DS\_BigDataPipeline\BDvenv** is the command executed. Activate the virtual environment. Use the command **Scripts\activate**



```

Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\janed>cd C:\DS_BigDataPipeline\BDvenv

C:\DS_BigDataPipeline\BDvenv>Scripts\activate

(BDvenv) C:\DS_BigDataPipeline\BDvenv>|
```

Access the file folder where the producer python script is located. In this case, the python script is in the DS\_BigDataPipeline folder. The command **cd C:\DS\_BigDataPipeline** is executed.

```

Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\janed>cd C:\DS_BigDataPipeline\BDvenv

C:\DS_BigDataPipeline\BDvenv>Scripts\activate

(BDvenv) C:\DS_BigDataPipeline\BDvenv>cd C:\DS_BigDataPipeline

(BDvenv) C:\DS_BigDataPipeline>|
```

Run the producer python script to start producing and sending messages to the specified kafka topic. The command executed is **python virtual\_producer1.py** where virtual\_producer1.py is the python script of the first producer. The producer should start producing messages as shown.

```

(BDvenv) C:\DS_BigDataPipeline>python virtual_producer1.py
Kafka Producer1 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:26', 'car': 1, 'others': 0, 'jeepney': 1, 'lgu_code': '1200', 'timeuuid_id': '99e799ae-50bc-11ee-86b7-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:27', 'car': 4, 'others': 0, 'jeepney': 0, 'lgu_code': '1200', 'timeuuid_id': '9a827a21-50bc-11ee-af7b-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 2, 'bus': 1, 'date_saved': '09/12/2023', 'total': 12})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:28', 'car': 2, 'others': 1, 'jeepney': 1, 'lgu_code': '1200', 'timeuuid_id': '9b1c9740-50bc-11ee-b047-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 12})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:29', 'car': 3, 'others': 0, 'jeepney': 2, 'lgu_code': '1200', 'timeuuid_id': '9bb7509e-50bc-11ee-bb48-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 14})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:30', 'car': 2, 'others': 1, 'jeepney': 1, 'lgu_code': '1200', 'timeuuid_id': '9c52310f-50bc-11ee-9b64-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:31', 'car': 1, 'others': 0, 'jeepney': 1, 'lgu_code': '1200', 'timeuuid_id': '9cec7540-50bc-11ee-8ca4-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:02:32', 'car': 4, 'others': 2, 'jeepney': 0, 'lgu_code': '1200', 'timeuuid_id': '9d881900-50bc-11ee-a7d3-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10})
```

Simultaneously Running Producers

Producer1

```

(BDvenv) C:\DS_BigDataPipeline>python virtual_producer1.py
Kafka Producer1 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:06:59', 'car': 0, 'others': 1, 'jeepney': 2, 'lgu_code': '1200', 'timeuuid_id': '3c9d0aee-50bd-11ee-8c26-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:00', 'car': 2, 'others': 1, 'jeepney': 2, 'lgu_code': '1200', 'timeuuid_id': '3d3923de-50bd-11ee-9ea2-089798cc0a5d', 'bike': 0, 'truck': 2, 'tryke': 3, 'bus': 2, 'date_saved': '09/12/2023', 'total': 12})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:01', 'car': 0, 'others': 0, 'jeepney': 2, 'lgu_code': '1200', 'timeuuid_id': '3dd36811-50bd-11ee-8471-089798cc0a5d', 'bike': 0, 'truck': 2, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 4})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:02', 'car': 3, 'others': 1, 'jeepney': 0, 'lgu_code': '1200', 'timeuuid_id': '3e6dfa61-50bd-11ee-9a6c-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 3, 'bus': 1, 'date_saved': '09/12/2023', 'total': 13})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:03', 'car': 1, 'others': 0, 'jeepney': 2, 'lgu_code': '1200', 'timeuuid_id': '3f081780-50bd-11ee-b5d3-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 2, 'bus': 1, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:04', 'car': 3, 'others': 1, 'jeepney': 0, 'lgu_code': '1200', 'timeuuid_id': '3fa2d0e1-50bd-11ee-8513-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:05', 'car': 0, 'others': 2, 'jeepney': 1, 'lgu_code': '1200', 'timeuuid_id': '403d150f-50bd-11ee-ad76-089798cc0a5d', 'bike': 5, 'truck': 2, 'tryke': 0, 'bus': 1, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 8
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:06', 'car': 2, 'others': 0, 'jeepney': 1, 'lgu_code': '1200', 'timeuuid_id': '40d8b8cf-50bd-11ee-b08f-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 3, 'bus': 1, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 9
('Message: ', {'sensor_id': 'sensor_01', 'time_saved': '00:07:07', 'car': 4, 'others': 1, 'jeepney': 2, 'lgu_code': '1200', 'timeuuid_id': '4172fd00-50bd-11ee-90c2-089798cc0a5d', 'bike': 3, 'truck': 0, 'tryke': 2, 'bus': 1, 'date_saved': '09/12/2023', 'total': 13})
```

## Producer2

```

C:\DS_BigDataPipeline>python virtual_producer2.py
Kafka Producer2 Application Started ...
Preparing message: 1
{'Message': '1', 'sensor_id': 'sensor_02', 'time_saved': '00:07:00', 'car': 1, 'others': 1, 'jeepney': 1, 'lgu_code': '1210', 'timeuuid_id': '3d294561-50bd-11ee-9728-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 10}}
Preparing message: 2
{'Message': '2', 'sensor_id': 'sensor_02', 'time_saved': '00:07:01', 'car': 4, 'others': 2, 'jeepney': 2, 'lgu_code': '1210', 'timeuuid_id': '3dc8561-50bd-11ee-86a9-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 3, 'bus': 1, 'date_saved': '09/12/2023', 'total': 17}}
Preparing message: 3
{'Message': '3', 'sensor_id': 'sensor_02', 'time_saved': '00:07:02', 'car': 3, 'others': 0, 'jeepney': 0, 'lgu_code': '1210', 'timeuuid_id': '3e5fa280-50bd-11ee-a798-089798cc0a5d', 'bike': 0, 'truck': 0, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 4}}
Preparing message: 4
{'Message': '4', 'sensor_id': 'sensor_02', 'time_saved': '00:07:03', 'car': 1, 'others': 1, 'jeepney': 0, 'lgu_code': '1210', 'timeuuid_id': '3ef9bf9e-50bd-11ee-8082-089798cc0a5d', 'bike': 1, 'truck': 2, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 5}}
Preparing message: 5
{'Message': '5', 'sensor_id': 'sensor_02', 'time_saved': '00:07:04', 'car': 2, 'others': 1, 'jeepney': 1, 'lgu_code': '1210', 'timeuuid_id': '3f94a011-50bd-11ee-a754-089798cc0a5d', 'bike': 1, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 9}}
Preparing message: 6
{'Message': '6', 'sensor_id': 'sensor_02', 'time_saved': '00:07:05', 'car': 2, 'others': 0, 'jeepney': 1, 'lgu_code': '1210', 'timeuuid_id': '402ee440-50bd-11ee-aeb0-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 9}}
Preparing message: 7
{'Message': '7', 'sensor_id': 'sensor_02', 'time_saved': '00:07:06', 'car': 0, 'others': 1, 'jeepney': 1, 'lgu_code': '1210', 'timeuuid_id': '40ccaaf8-50bd-11ee-b9d4-089798cc0a5d', 'bike': 5, 'truck': 2, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 13}}
Preparing message: 8
{'Message': '8', 'sensor_id': 'sensor_02', 'time_saved': '00:07:07', 'car': 4, 'others': 2, 'jeepney': 0, 'lgu_code': '1210', 'timeuuid_id': '4164a51e-50bd-11ee-880d-089798cc0a5d', 'bike': 0, 'truck': 0, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 6}}
Preparing message: 9
{'Message': '9', 'sensor_id': 'sensor_02', 'time_saved': '00:07:08', 'car': 4, 'others': 2, 'jeepney': 2, 'lgu_code': '1210', 'timeuuid_id': '420e9700-50bd-11ee-b8f5-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 1, 'bus': 2, 'date_saved': '09/12/2023', 'total': 13}}

```

### Producer3

```

C:\DS_BigDataPipeline>python virtual_producer3.py
Kafka Producer3 Application Started ...
Preparing message: 1
{'Message': '1', 'sensor_id': 'sensor_03', 'time_saved': '00:07:00', 'car': 1, 'others': 0, 'jeepney': 0, 'lgu_code': '1220', 'timeuuid_id': '3d9fad1-50bd-11ee-a4df-089798cc0a5d', 'bike': 0, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 5}}
Preparing message: 2
{'Message': '2', 'sensor_id': 'sensor_03', 'time_saved': '00:07:01', 'car': 0, 'others': 2, 'jeepney': 0, 'lgu_code': '1220', 'timeuuid_id': '3e39cb00-50bd-11ee-a215-089798cc0a5d', 'bike': 3, 'truck': 0, 'tryke': 3, 'bus': 1, 'date_saved': '09/12/2023', 'total': 9}}
Preparing message: 3
{'Message': '3', 'sensor_id': 'sensor_03', 'time_saved': '00:07:02', 'car': 2, 'others': 1, 'jeepney': 2, 'lgu_code': '1220', 'timeuuid_id': '3ed3e821-50bd-11ee-9db8-089798cc0a5d', 'bike': 2, 'truck': 1, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10}}
Preparing message: 4
{'Message': '4', 'sensor_id': 'sensor_03', 'time_saved': '00:07:03', 'car': 0, 'others': 2, 'jeepney': 1, 'lgu_code': '1220', 'timeuuid_id': '3f6e7a70-50bd-11ee-838e-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 8}}
Preparing message: 5
{'Message': '5', 'sensor_id': 'sensor_03', 'time_saved': '00:07:04', 'car': 2, 'others': 1, 'jeepney': 2, 'lgu_code': '1220', 'timeuuid_id': '40090cc0-50bd-11ee-b980-089798cc0a5d', 'bike': 5, 'truck': 1, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 13}}
Preparing message: 6
{'Message': '6', 'sensor_id': 'sensor_03', 'time_saved': '00:07:05', 'car': 0, 'others': 2, 'jeepney': 0, 'lgu_code': '1220', 'timeuuid_id': '40a2b400-50bd-11ee-a0e6-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 6}}
Preparing message: 7
{'Message': '7', 'sensor_id': 'sensor_03', 'time_saved': '00:07:06', 'car': 1, 'others': 1, 'jeepney': 0, 'lgu_code': '1220', 'timeuuid_id': '413ef400-50bd-11ee-af0b-089798cc0a5d', 'bike': 3, 'truck': 0, 'tryke': 0, 'bus': 1, 'date_saved': '09/12/2023', 'total': 6}}
Preparing message: 8
{'Message': '8', 'sensor_id': 'sensor_03', 'time_saved': '00:07:07', 'car': 3, 'others': 2, 'jeepney': 2, 'lgu_code': '1220', 'timeuuid_id': '41db0da1-50bd-11ee-a034-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 15}}
Preparing message: 9
{'Message': '9', 'sensor_id': 'sensor_03', 'time_saved': '00:07:08', 'car': 2, 'others': 2, 'jeepney': 0, 'lgu_code': '1220', 'timeuuid_id': '42752ac0-50bd-11ee-8ea4-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 11}}

```

## Producer4

```
(BDEVENV) C:\DS_BigDataPipeline>python virtual_producer4.py
Kafka Producer4 Application Started ...
Preparing message: 1
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:01', 'car': 3, 'others': 2, 'jeepney': 2, 'lg_u_code': '1230', 'timeuuid_id': '3e085ac0-50bd-11ee-bf24-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 11}}
Preparing message: 2
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:02', 'car': 2, 'others': 2, 'jeepney': 1, 'lg_u_code': '1230', 'timeuuid_id': '3ea42591-50bd-11ee-9fb7-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 13}}
Preparing message: 3
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:03', 'car': 0, 'others': 1, 'jeepney': 2, 'lg_u_code': '1230', 'timeuuid_id': '3f3edef0-50bd-11ee-815a-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10}}
Preparing message: 4
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:04', 'car': 1, 'others': 2, 'jeepney': 1, 'lg_u_code': '1230', 'timeuuid_id': '3fd9984f-50bd-11ee-9d19-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 11}}
Preparing message: 5
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:05', 'car': 1, 'others': 0, 'jeepney': 0, 'lg_u_code': '1230', 'timeuuid_id': '4075b140-50bd-11ee-99d7-089798cc0a5d', 'bike': 5, 'truck': 2, 'tryke': 3, 'bus': 2, 'date_saved': '09/12/2023', 'total': 13}}
Preparing message: 6
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:06', 'car': 0, 'others': 1, 'jeepney': 0, 'lg_u_code': '1230', 'timeuuid_id': '410f5930-50bd-11ee-b777-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 3, 'bus': 2, 'date_saved': '09/12/2023', 'total': 10}}
Preparing message: 7
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:07', 'car': 2, 'others': 1, 'jeepney': 1, 'lg_u_code': '1230', 'timeuuid_id': '41a8da0f-50bd-11ee-90d6-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 5}}
Preparing message: 8
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:08', 'car': 1, 'others': 0, 'jeepney': 2, 'lg_u_code': '1230', 'timeuuid_id': '42439370-50bd-11ee-b428-089798cc0a5d', 'bike': 2, 'truck': 1, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 8}}
Preparing message: 9
{'Message': 1, 'sensor_id': 'sensor_04', 'time_saved': '00:07:09', 'car': 1, 'others': 2, 'jeepney': 0, 'lg_u_code': '1230', 'timeuuid_id': '42ddd7a1-50bd-11ee-a7e1-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 8}}
```

## Producer5

```

C:\DS_BigDataPipeline>python virtual_producer5.py
Kafka Producer5 Application Started ...
Preparing message: 1
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:02', 'car': 2, 'others': 0, 'jeepney': 0, 'lg_u_code': '1240', 'timeuuid_id': '3e75746e-50bd-11ee-93a9-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 3, 'bus': 2, 'date_saved': '09/12/2023', 'total': 13})
Preparing message: 2
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:03', 'car': 3, 'others': 0, 'jeepney': 0, 'lg_u_code': '1240', 'timeuuid_id': '3f0fc161-50bd-11ee-a2fa-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 8})
Preparing message: 3
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:04', 'car': 3, 'others': 1, 'jeepney': 1, 'lg_u_code': '1240', 'timeuuid_id': '3fa9d5c0-50bd-11ee-945e-089798cc0a5d', 'bike': 2, 'truck': 0, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 9})
Preparing message: 4
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:05', 'car': 4, 'others': 1, 'jeepney': 1, 'lg_u_code': '1240', 'timeuuid_id': '4043f2e1-50bd-11ee-b30d-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 13})
Preparing message: 5
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:06', 'car': 3, 'others': 2, 'jeepney': 0, 'lg_u_code': '1240', 'timeuuid_id': '40dfc4c0-50bd-11ee-ad6c-089798cc0a5d', 'bike': 2, 'truck': 0, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 9})
Preparing message: 6
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:07', 'car': 0, 'others': 0, 'jeepney': 1, 'lg_u_code': '1240', 'timeuuid_id': '4179dacf-50bd-11ee-8c6e-089798cc0a5d', 'bike': 2, 'truck': 1, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 7
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:08', 'car': 3, 'others': 1, 'jeepney': 2, 'lg_u_code': '1240', 'timeuuid_id': '4213d0de-50bd-11ee-9794-089798cc0a5d', 'bike': 3, 'truck': 2, 'tryke': 1, 'bus': 2, 'date_saved': '09/12/2023', 'total': 14})
Preparing message: 8
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:09', 'car': 1, 'others': 2, 'jeepney': 2, 'lg_u_code': '1240', 'timeuuid_id': '42ae6330-50bd-11ee-bd0a-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 9
('Message:', {'sensor_id': 'sensor_05', 'time_saved': '00:07:10', 'car': 0, 'others': 1, 'jeepney': 2, 'lg_u_code': '1240', 'timeuuid_id': '4349b8d1-50bd-11ee-9e4a-089798cc0a5d', 'bike': 0, 'truck': 1, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 6})

```

## Producer6

```
(BDenv) C:\DS_BigDataPipeline>python virtual_producer6.py
Kafka Producer6 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:38', 'car': 2, 'others': 2, 'jeepney': 1, 'lgu_code': '1250', 'timeuuid_id': '53e2d604-50bd-11ee-bae1-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 13})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:39', 'car': 4, 'others': 1, 'jeepney': 1, 'lgu_code': '1250', 'timeuuid_id': '547ca540-50bd-11ee-b394-089798cc0a5d', 'bike': 0, 'truck': 0, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:40', 'car': 0, 'others': 1, 'jeepney': 1, 'lgu_code': '1250', 'timeuuid_id': '55171080-50bd-11ee-a17c-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:41', 'car': 2, 'others': 1, 'jeepney': 0, 'lgu_code': '1250', 'timeuuid_id': '55b1c9de-50bd-11ee-818c-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:42', 'car': 1, 'others': 2, 'jeepney': 2, 'lgu_code': '1250', 'timeuuid_id': '5645c52e-50bd-11ee-b2df-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 9})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:43', 'car': 0, 'others': 2, 'jeepney': 1, 'lgu_code': '1250', 'timeuuid_id': '566679df-50bd-11ee-994b-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 1, 'bus': 2, 'date_saved': '09/12/2023', 'total': 8})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:44', 'car': 1, 'others': 2, 'jeepney': 0, 'lgu_code': '1250', 'timeuuid_id': '57810b9e-50bd-11ee-b23d-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 5})
Preparing message: 8
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:45', 'car': 3, 'others': 2, 'jeepney': 2, 'lgu_code': '1250', 'timeuuid_id': '581b28c0-50bd-11ee-b8a1-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 1, 'bus': 2, 'date_saved': '09/12/2023', 'total': 14})
Preparing message: 9
('Message: ', {'sensor_id': 'sensor_06', 'time_saved': '00:07:46', 'car': 0, 'others': 0, 'jeepney': 0, 'lgu_code': '1250', 'timeuuid_id': '58b59400-50bd-11ee-bdda-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 7})
```

## Producer7

```
(BDEVENV) C:\DS_BigDataPipeline>python virtual_producer7.py
Kafka Producer7 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:04', 'car': 1, 'others': 0, 'jeepney': 2, 'lg_u_code': '1260', 'timeuuid_id': '3fb67df0-50bd-11ee-ae90-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:05', 'car': 1, 'others': 2, 'jeepney': 0, 'lg_u_code': '1260', 'timeuuid_id': '40524ac0-50bd-11ee-bc3b-089798cc0a5d', 'bike': 1, 'truck': 1, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 8})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:06', 'car': 0, 'others': 2, 'jeepney': 0, 'lg_u_code': '1260', 'timeuuid_id': '40ee158f-50bd-11ee-8d42-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:07', 'car': 4, 'others': 0, 'jeepney': 1, 'lg_u_code': '1260', 'timeuuid_id': '4187e48f-50bd-11ee-a309-089798cc0a5d', 'bike': 0, 'truck': 0, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:08', 'car': 1, 'others': 2, 'jeepney': 0, 'lg_u_code': '1260', 'timeuuid_id': '422228c0-50bd-11ee-8a55-089798cc0a5d', 'bike': 0, 'truck': 0, 'tryke': 3, 'bus': 2, 'date_saved': '09/12/2023', 'total': 8})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:09', 'car': 1, 'others': 0, 'jeepney': 0, 'lg_u_code': '1260', 'timeuuid_id': '42bcb11-50bd-11ee-a16d-089798cc0a5d', 'bike': 3, 'truck': 0, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 6})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:10', 'car': 3, 'others': 1, 'jeepney': 2, 'lg_u_code': '1260', 'timeuuid_id': '435810b0-50bd-11ee-83a7-089798cc0a5d', 'bike': 0, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 8
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:11', 'car': 4, 'others': 0, 'jeepney': 1, 'lg_u_code': '1260', 'timeuuid_id': '43f22dcf-50bd-11ee-90a6-089798cc0a5d', 'bike': 1, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 9
('Message: ', {'sensor_id': 'sensor_07', 'time_saved': '00:07:12', 'car': 3, 'others': 0, 'jeepney': 2, 'lg_u_code': '1260', 'timeuuid_id': '448cc01e-50bd-11ee-922c-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 0, 'bus': 2, 'date_saved': '09/12/2023', 'total': 11})
```



Producer8

```

(BDvenv) C:\DS_BigDataPipeline>python virtual_producer8.py
Kafka Producer8 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:04', 'car': 3, 'others': 1, 'jeepney': 1, 'lgu_code': '1270', 'timeuuid_id': '4017d9cf-50bd-11ee-b1bd-089798cc0a5d', 'bike': 3, 'truck': 0, 'tryke': 3, 'bus': 1, 'date_saved': '09/12/2023', 'total': 12})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:05', 'car': 1, 'others': 0, 'jeepney': 0, 'lgu_code': '1270', 'timeuuid_id': '40b35680-50bd-11ee-a6d5-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 9})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:06', 'car': 4, 'others': 0, 'jeepney': 1, 'lgu_code': '1270', 'timeuuid_id': '414d73a1-50bd-11ee-bbb1-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 3, 'bus': 1, 'date_saved': '09/12/2023', 'total': 14})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:07', 'car': 1, 'others': 0, 'jeepney': 1, 'lgu_code': '1270', 'timeuuid_id': '41e96580-50bd-11ee-b621-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 2, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:09', 'car': 4, 'others': 0, 'jeepney': 1, 'lgu_code': '1270', 'timeuuid_id': '42835b8f-50bd-11ee-aab9-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 13})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:10', 'car': 2, 'others': 0, 'jeepney': 2, 'lgu_code': '1270', 'timeuuid_id': '431e8a1e-50bd-11ee-9675-089798cc0a5d', 'bike': 0, 'truck': 2, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 9})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:11', 'car': 2, 'others': 0, 'jeepney': 2, 'lgu_code': '1270', 'timeuuid_id': '43b94380-50bd-11ee-883a-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 8
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:12', 'car': 0, 'others': 0, 'jeepney': 1, 'lgu_code': '1270', 'timeuuid_id': '4453aec0-50bd-11ee-9199-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 8})
Preparing message: 9
('Message: ', {'sensor_id': 'sensor_08', 'time_saved': '00:07:13', 'car': 0, 'others': 0, 'jeepney': 0, 'lgu_code': '1270', 'timeuuid_id': '44ee1a00-50bd-11ee-9ae2-089798cc0a5d', 'bike': 1, 'truck': 2, 'tryke': 2, 'bus': 0, 'date_saved': '09/12/2023', 'total': 5})

```

Producer9

```

(BDvenv) C:\DS_BigDataPipeline>python virtual_producer9.py
Kafka Producer9 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:05', 'car': 2, 'others': 2, 'jeepney': 0, 'lgu_code': '1280', 'timeuuid_id': '4071b99e-50bd-11ee-8181-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 3, 'bus': 2, 'date_saved': '09/12/2023', 'total': 15})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:06', 'car': 1, 'others': 0, 'jeepney': 1, 'lgu_code': '1280', 'timeuuid_id': '410ce82e-50bd-11ee-9735-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 2, 'bus': 1, 'date_saved': '09/12/2023', 'total': 6})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:07', 'car': 4, 'others': 2, 'jeepney': 0, 'lgu_code': '1280', 'timeuuid_id': '41a8da0f-50bd-11ee-992a-089798cc0a5d', 'bike': 2, 'truck': 0, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 12})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:08', 'car': 3, 'others': 2, 'jeepney': 0, 'lgu_code': '1280', 'timeuuid_id': '42439370-50bd-11ee-a37b-089798cc0a5d', 'bike': 0, 'truck': 0, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 5})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:09', 'car': 0, 'others': 0, 'jeepney': 1, 'lgu_code': '1280', 'timeuuid_id': '42ddd7a1-50bd-11ee-9779-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 0, 'bus': 1, 'date_saved': '09/12/2023', 'total': 3})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:10', 'car': 4, 'others': 0, 'jeepney': 2, 'lgu_code': '1280', 'timeuuid_id': '4379544f-50bd-11ee-a27f-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:11', 'car': 1, 'others': 2, 'jeepney': 1, 'lgu_code': '1280', 'timeuuid_id': '44159451-50bd-11ee-8bb8-089798cc0a5d', 'bike': 5, 'truck': 1, 'tryke': 2, 'bus': 2, 'date_saved': '09/12/2023', 'total': 14})
Preparing message: 8
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:12', 'car': 4, 'others': 1, 'jeepney': 0, 'lgu_code': '1280', 'timeuuid_id': '44b09bd1-50bd-11ee-a098-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 0, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 9
('Message: ', {'sensor_id': 'sensor_09', 'time_saved': '00:07:13', 'car': 3, 'others': 2, 'jeepney': 1, 'lgu_code': '1280', 'timeuuid_id': '454b2e1e-50bd-11ee-abf1-089798cc0a5d', 'bike': 1, 'truck': 0, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 10})

```

Producer10

```

(BDvenv) C:\DS_BigDataPipeline>python virtual_producer10.py
Kafka Producer10 Application Started ...
Preparing message: 1
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:03', 'car': 2, 'others': 0, 'jeepney': 1, 'lgu_code': '1290', 'timeuuid_id': '3f528e00-50bd-11ee-bd91-089798cc0a5d', 'bike': 4, 'truck': 0, 'tryke': 0, 'bus': 0, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 2
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:04', 'car': 2, 'others': 0, 'jeepney': 2, 'lgu_code': '1290', 'timeuuid_id': '3fecab1e-50bd-11ee-930a-089798cc0a5d', 'bike': 1, 'truck': 2, 'tryke': 2, 'bus': 1, 'date_saved': '09/12/2023', 'total': 10})
Preparing message: 3
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:05', 'car': 1, 'others': 2, 'jeepney': 1, 'lgu_code': '1290', 'timeuuid_id': '40867a1e-50bd-11ee-9a85-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 1, 'bus': 1, 'date_saved': '09/12/2023', 'total': 12})
Preparing message: 4
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:06', 'car': 3, 'others': 0, 'jeepney': 0, 'lgu_code': '1290', 'timeuuid_id': '41202211-50bd-11ee-90f0-089798cc0a5d', 'bike': 2, 'truck': 2, 'tryke': 0, 'bus': 1, 'date_saved': '09/12/2023', 'total': 8})
Preparing message: 5
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:07', 'car': 1, 'others': 2, 'jeepney': 1, 'lgu_code': '1290', 'timeuuid_id': '41bbece1-50bd-11ee-a034-089798cc0a5d', 'bike': 4, 'truck': 2, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 11})
Preparing message: 6
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:08', 'car': 3, 'others': 1, 'jeepney': 1, 'lgu_code': '1290', 'timeuuid_id': '4256310f-50bd-11ee-9b2f-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 3, 'bus': 0, 'date_saved': '09/12/2023', 'total': 13})
Preparing message: 7
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:09', 'car': 0, 'others': 1, 'jeepney': 0, 'lgu_code': '1290', 'timeuuid_id': '42f11180-50bd-11ee-b599-089798cc0a5d', 'bike': 5, 'truck': 0, 'tryke': 1, 'bus': 0, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 8
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:10', 'car': 4, 'others': 0, 'jeepney': 0, 'lgu_code': '1290', 'timeuuid_id': '438c1900-50bd-11ee-93c3-089798cc0a5d', 'bike': 0, 'truck': 1, 'tryke': 4, 'bus': 2, 'date_saved': '09/12/2023', 'total': 7})
Preparing message: 9
('Message: ', {'sensor_id': 'sensor_10', 'time_saved': '00:07:11', 'car': 0, 'others': 0, 'jeepney': 2, 'lgu_code': '1290', 'timeuuid_id': '44265d30-50bd-11ee-80b7-089798cc0a5d', 'bike': 4, 'truck': 1, 'tryke': 0, 'bus': 1, 'date_saved': '09/12/2023', 'total': 8})

```

V. Run Consumer

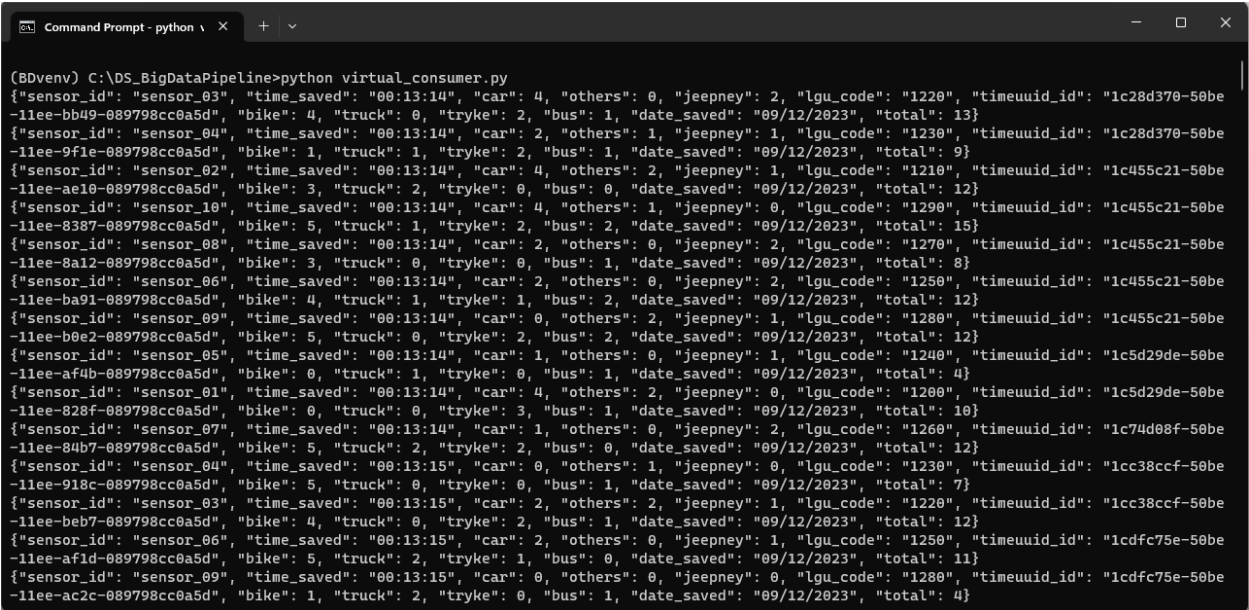
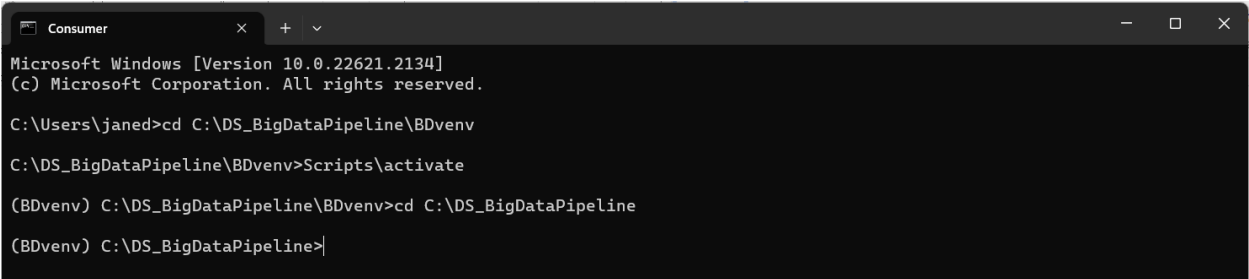
The python script for running the consumer is presented below. KafkaConsumer class is imported from kafka library to create the kafka consumer. KafkaConsumer() creates a new consumer. The consumer subscribes to the topic **bigdatatopic** as specified and will connect to the kafka broker using the loopback address **127.0.0.1:9092**. The consumer will begin reading the latest available

messages and will periodically commit its current position to prevent re-reading messages. A for loop iterates over all messages consumed by the consumer and will print the value of the messages in the prompt.

```
1  # -*- coding: utf-8 -*-
2
3  from kafka import KafkaConsumer
4
5  consumer = KafkaConsumer(
6      *topics: 'bigdatatopic',
7      bootstrap_servers = ['127.0.0.1:9092'],
8      auto_offset_reset = 'latest',
9      enable_auto_commit = True
10 )
11
12 for message in consumer:
13     message = message.value
14     print(message)
```

Run the consumer python script. The command used is **python virtual\_consumer.py** where virtual\_consumer.py is the file name of the script.

The output is presented below where all messages produced and sent by producers with from sensor\_01 to sensor\_10 are consumed by the producer in real-time.



VI. Creating a Cassandra Keyspace and Table

A database is required to store the messages produced by the producer and consumed by the consumer. In this case, a Cassandra keyspace and table will be created.

Activate Cassandra Server

To run the Cassandra server, open a new CLI and access the bin folder within the Cassandra folder. In this case, the command used is **cd C:\apache-cassandra-3.11.16\bin**

Run the Cassandra server using the command **Cassandra**. The output is presented below.

```
Cassandra

Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\janed>cd C:\apache-cassandra-3.11.16

C:\apache-cassandra-3.11.16>
```

```
Cassandra

Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\janed>cd C:\apache-cassandra-3.11.16\bin

C:\apache-cassandra-3.11.16\bin>cassandra
WARNING! Powershell script execution unavailable.
Please use 'powershell Set-ExecutionPolicy Unrestricted'
on this user-account to run cassandra with fully featured
functionality on this platform.
Starting with legacy startup options
Starting Cassandra Server
INFO [main] 2023-09-11 15:31:08,554 YamlConfigurationLoader.java:93 - Configuration location: file:/C:/apache-cassandra-3.11.16/conf/cassandra.yaml
INFO [main] 2023-09-11 15:31:09,134 Config.java:555 - Node configuration:[allocate_tokens_for_keyspace=null; allow_extra_insecure_udfs=false; allow_insecure_udfs=false; authenticator=AllowAllAuthenticator; authorizer=AllowAllAuthorizer; auto_bootstrap=true; auto_snapshot=true; back_pressure_enabled=false; back_pressure_strategy=org.apache.cassandra.net.RateBasedBackPressure{high_ratio=0.9, factor=5, flow=FAST}; batch_size_fail_threshold_in_kb=50; batch_size_warn_threshold_in_kb=5; batchlog_replay_throttle_in_kb=1024; broadcast_address=null; broadcast_rpc_address=null; buffer_pool_use_heap_if_exhausted=true; cache_load_timeout_seconds=30; cas_contention_timeout_in_ms=1000; cdc_enabled=false; cdc_free_space_check_interval_ms=250; cdc_raw_directory=null; cdc_total_space_in_mb=0; check_for_duplicate_rows_during_compaction=true; check_for_duplicate_rows_during_reads=true; client_encryption_options=<REDACTED>; cluster_name=Test Cluster; column_index_cache_size_in_kb=2; column_index_size_in_kb=64; commit_failure_policy=stop; commitlog_compression=null; commitlog_directories=null; commitlog_max_compression_buffers_in_pool=3; commitlog_periodic_queue_size=-1; commitlog_segment_size_in_mb=32; commitlog_sync=periodic; commitlog_sync_batch_window_in_ms=NaN; commitlog_sync_period_in_ms=10000; commitlog_total_space_in_mb=null; compaction_large_partition_warning_threshold_mb=100; compaction_throughput_mb_per_sec=16; concurrent_compactors=null; concurrent_counter_writes=32; concurrent_materialized_view_writes=32; concurrent_reads=32; concurrent_replicates=null; concurrent_writes=32; counter_cache_keys_to_save=2147483647; counter_cache_save_period=7200; counter_cache_size_in_mb=null; counter_write_request_timeout_in_ms=5000; credentials_cache_max_entries=1000; credentials_update_interval_in_ms=-1; credentials_validity_in_ms=2000; cross_node_timeout=false; data_file_directories=[Ljava.lang.String;@30b7c004
```

```
Cassandra

4571763885797588, 7427970688294388521, 7455087776763777395, 7627259355029992220, 7638986752810293836, 7662316695268495246, 7715894053066861611, 7839620904859698790, 7884527306887970129, 7897296876454034635, 8075983109497373081, 8080008522622918040, 8089062502105331355, 8099802892734129893, 811768956185854663, 8145127086500293966, 8181438382786283829, 8188774065480934403, 8271206506230515793, 8284133980205996221, 8328764553764941007, 8336276601691401592, 8339586497872365078, 8390488502367313594, 844815824562247039, 8500802682872902940, 8509731332089148931, 8529792906900181540, 8541826436708451455, 8572889543896019513, 860733043707162272, 8711080274751963510, 8713230546047563574, 8714421643759122572, 8874943769967866562, 8974334059784714224, 8978613215750209875, 8979480835542139709, 9035733777066190311, 907722706275341593, 9129214001988099297, 971314707930377030]
INFO [main] 2023-09-11 15:31:13,721 StorageService.java:1679 - JOINING: Finish joining ring
INFO [main] 2023-09-11 15:31:13,765 SecondaryIndexManager.java:512 - Executing pre-join tasks for: CFS(Keyspace='bdkeyspace', ColumnFamily='prod_messages')
INFO [main] 2023-09-11 15:31:13,766 SecondaryIndexManager.java:512 - Executing pre-join tasks for: CFS(Keyspace='newbdbkeyspace', ColumnFamily='prod_messages')
INFO [main] 2023-09-11 15:31:13,767 SecondaryIndexManager.java:512 - Executing pre-join tasks for: CFS(Keyspace='bigdatakeyspace', ColumnFamily='prod_messages')
INFO [main] 2023-09-11 15:31:13,816 StorageService.java:2604 - Node localhost/127.0.0.1 state jump to NORMAL
INFO [main] 2023-09-11 15:31:14,114 NativeTransportService.java:73 - Netty using Java NIO event loop
INFO [main] 2023-09-11 15:31:14,185 Server.java:158 - Using Netty Version: [netty-buffer=netty-buffer-4.0.44.Final.452812a, netty-codec=netty-codec-4.0.44.Final.452812a, netty-codec-haproxy=netty-codec-haproxy-4.0.44.Final.452812a, netty-codec-http=netty-codec-http-4.0.44.Final.452812a, netty-codec-socks=netty-codec-socks-4.0.44.Final.452812a, netty-common=netty-common-4.0.44.Final.452812a, netty-handler=netty-handler-4.0.44.Final.452812a, netty-tcnative=netty-tcnative-1.1.3.3.Fork26.142ecbb, netty-transport=netty-transport-4.0.44.Final.452812a, netty-transport-native-epoll=netty-transport-native-epoll-4.0.44.Final.452812a, netty-transport-rxtx=netty-transport-rxtx-4.0.44.Final.452812a, netty-transport-sctp=netty-transport-sctp-4.0.44.Final.452812a, netty-transport-udt=netty-transport-udt-4.0.44.Final.452812a]
INFO [main] 2023-09-11 15:31:14,186 Server.java:159 - Starting listening for CQL clients on localhost/127.0.0.1:9042 (unencrypted)...
INFO [main] 2023-09-11 15:31:14,359 CassandraDaemon.java:564 - Not starting RPC server as requested. Use JMX (StorageService->startRPCServer()) or nodetool (enablethrift) to start it
INFO [main] 2023-09-11 15:31:14,360 CassandraDaemon.java:650 - Startup complete
```

To verify that Cassandra is running, open a new CLI and access the bin folder. Enter the command **cqlsh**. The output is shown below.

```
Command Prompt - cqlsh

Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\janed>cd C:\apache-cassandra-3.11.16\bin

C:\apache-cassandra-3.11.16\bin>cqlsh

WARNING: console codepage must be set to cp65001 to support utf-8 encoding on Windows platforms.
If you experience encoding problems, change your console codepage with 'chcp 65001' before starting cqlsh.

Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 3.11.16 | CQL spec 3.4.4 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cqlsh> |
```

### Create Cassandra Keyspace

The python script presented below is used to create a Cassandra Keyspace which will be used to hold the table where messages will be saved. Cluster is imported from `Cassandra.cluster` module and is used to specify the address and port of the Cassandra cluster. The cluster connects

to a session using `connect()`. A keyspace query is created that creates a Cassandra keyspace called **dskeyspacebd** with configurations of SimpleStrategy class and a replication factor of 1. The query is executed using `session.execute()` and the session and cluster are both shutdown.

```
1
2  from cassandra.cluster import Cluster
3
4  cluster = Cluster(contact_points=['127.0.0.1'], port=9042)
5  session = cluster.connect()
6
7  create_keyspace_query = """
8      CREATE KEYSPACE IF NOT EXISTS dskeyspacebd
9      WITH replication = {
10          'class': 'SimpleStrategy',
11          'replication_factor': 1
12      };
13  """
14  session.execute(create_keyspace_query)
15  session.shutdown()
16  cluster.shutdown()
```

To verify that the keyspace is created, the command **DESCRIBE dskeyspacebd;** is used in `cqlsh`. The output is shown below.

```
cqlsh> DESCRIBE dskeyspacebd;

CREATE KEYSPACE dskeyspacebd WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '1'} AND durable_writes = true;

cqlsh> |
```

Create Cassandra Table

The python script shown below creates a table named **prod\_messages** in the keyspace **dskeyspacebd**. Cluster is imported from `Cassandra.cluster` module and is used to specify the address and port of the Cassandra cluster. A session is created that connects to the cluster specified. A query is created that creates a table in **dskeyspacebd** named **prod\_messages** and the column names and datatypes or specified. The column names are the labels of the messages produced by the producer. `session.execute()` executes the query and creates the table. If the table name is already exists in the keyspace, the user is notified that the table already exists, otherwise the user is notified that the table is created. The session and cluster are both shutdown afterwards.

```
1  from cassandra.cluster import Cluster
2
3  cluster = Cluster(contact_points=['127.0.0.1'], port=9042)
4  session = cluster.connect()
5
6  create_table_query = """
7      CREATE TABLE IF NOT EXISTS dskeyspacebd.prod_messages(
8          sensor_id TEXT,
9          time_saved TEXT,
10         car INT,
11         others INT,
12         jeepney INT,
13         lgu_code TEXT,
14         timeuuid_id TEXT PRIMARY KEY,
15         bike INT,
16         truck INT,
17         tryke INT,
18         bus INT,
19         date_saved TEXT,
20         total BIGINT
21     )
22  """
23
24  name = 'prod_messages'
25
26  existing_tables = session.execute("SELECT table_name FROM system_schema.tables WHERE keyspace_name = 'dskeyspacebd'")
27  table_exists = False
28
29  for row in existing_tables:
30      if row.table_name == name:
31          table_exists = True
32          break
33
34  if table_exists:
35      print("Table {} already exists".format(name))
36  else:
37      session.execute(create_table_query)
38      print("Table {} created".format(name))
39
40  session.shutdown()
41  cluster.shutdown()
```

To verify that the table is created, the following commands are executed in `cqlsh`:



USE dskeyspacebd;

SELECT \* FROM prod\_messages;

The output is shown below.

```
cqlsh> USE dskeyspacebd;
cqlsh:dskeyspacebd> select * from prod_messages;

timeuuid_id | bike | bus | car | date_saved | jeepney | lgu_code | others | sensor_id | time_saved | total | truck | tryke
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
(0 rows)
cqlsh:dskeyspacebd> |
```

VII. Fetch and Store Messages to Cassandra Keyspace

The python script below utilizes Pyspark, Kafka, and Cassandra connectors to fetch the latest batch of data sent by the producer and store the messages in the dskeyspacebd keyspace and within the prod\_messages table created earlier. Utilities within pyspark.sql, pyspark.sql.functions, pyspark.sql.types and time libraries are used. The kafka configurations are initialized which encompass the target topic cluster and the broker address and port. The Cassandra configurations are also initialized which encompass the address, port number, target keyspace and target table within the keyspace. A spark application is created and the configurations of the session are specified. The spark connection to Cassandra is specified in the configuration. The spark.jars used for the application are also included in the configurations. These encompass kafka-clients, spark-streaming-kafka, spark-cassandra-connector, as well as the spark.jars.packages which include org.apache.spark:spark-sql-kafka and com.datastax.spark:spark-cassandra-connector which will all be used to fetch messages from kafka and store them in Cassandra keyspace. The schema is prepared which will be used as the framework for organizing the messages collected which will then be sent to Cassandra. The StructFields are initialized with each message attribute. Spark is used to read data from a kafka source and the target topic is specified for subscription. The messages collected are converted to string and initially stored in kafka\_data DataFrame. Afterwards, the messages are parsed into JSON data fitted into the schema. Data is then written to Cassandra using writeStream. Batch, format, target table and keyspace, and mode are specified. The writing process is then carried out using .save(). Upon a keyboard interrupt, i.e. ctrl + c, the query will terminate.

```
1  from pyspark.sql import SparkSession
2  from pyspark.sql.functions import *
3  from pyspark.sql.types import *
4  from pyspark.sql.streaming import DataStreamWriter
5  import time
6
7  kafka_topic_name = "bigdatatopic"
8  kafka_bootstrap_servers = '127.0.0.1:9092'
9
10  cassandra_host_name = '127.0.0.1'
11  cassandra_port_num = '9042'
12  cassandra_keyspace_name = 'dskeyspacebd'
13  cassandra_table_name = 'prod_messages'
14
15  if __name__ == "__main__":
16      print("Welcome!")
17      print("Data Processing Application Started...")
18      print(time.strftime('%Y-%m-%d %H:%M:%S'))
19
20      spark = SparkSession \
21          .builder \
22          .appName("Pyspark Streaming with Kafka and Cassandra") \
23          .master("local[*]") \
24          .config("spark.cassandra.connection.host", cassandra_host_name) \
25          .config("spark.cassandra.connection.port", cassandra_port_num) \
26          .config("spark.jars", "C:\\DS_BigDataPipeline\\lib\\kafka-clients-3.5.1.jar,"
27                          "C:\\DS_BigDataPipeline\\lib\\spark-streaming-kafka-0-10_2.12-3.0.3.jar,"
28                          "C:\\DS_BigDataPipeline\\lib\\spark-cassandra-connector_2.12-3.0.1.jar") \
29          .config("spark.jars.packages", "org.apache.spark:spark-sql-kafka-0-10_2.12:3.0.3,"
30                          "com.datastax.spark:spark-cassandra-connector_2.12:3.0.1") \
31          .getOrCreate()
32      spark.sparkContext.setLogLevel("ERROR")
```

```

33
34     schema = StructType([
35         StructField(name: "sensor_id", StringType(), True),
36         StructField(name: "time_saved", StringType(), True),
37         StructField(name: "car", IntegerType(), True),
38         StructField(name: "others", IntegerType(), True),
39         StructField(name: "jeepney", IntegerType(), True),
40         StructField(name: "lgv_code", StringType(), True),
41         StructField(name: "timeuuid_id", StringType(), True),
42         StructField(name: "bike", IntegerType(), True),
43         StructField(name: "truck", IntegerType(), True),
44         StructField(name: "tryke", IntegerType(), True),
45         StructField(name: "bus", IntegerType(), True),
46         StructField(name: "date_saved", StringType(), True),
47         StructField(name: "total", LongType(), True)
48     ])
49
50     kafka_stream = spark \
51         .readStream \
52         .format("Kafka") \
53         .option("Kafka.bootstrap.servers", kafka_bootstrap_servers) \
54         .option("subscribe", kafka_topic_name) \
55         .load()
56
57     kafka_data = kafka_stream.selectExpr("CAST(value AS STRING)")
58     parsed_data = kafka_data.select(from_json(col: "value", schema).alias("data")).select("data.*")
59
60     cassandra_stream = parsed_data.writeStream \
61         .foreachBatch(lambda df, epoch_id: df.write \
62             .format("org.apache.spark.sql.cassandra") \
63             .options(table=cassandra_table_name, keyspace=cassandra_keyspace_name) \
64             .mode("append") \
65             .save())
66
67     query = cassandra_stream.start()
68
69     try:
70         query.awaitTermination()
71     except KeyboardInterrupt:
72         print("Terminating...")
73         query.stop()

```

Execute the spark application by running the python script in the CLI. In the virtual environment where the script is stored, the command **python main.py** is executed where main.py is the file name of the spark application python script. The output is shown below.

```

(BDvenv) C:\DS_BigDataPipeline>python main.py
Welcome!
Data Processing Application Started...
2023-09-12 01:06:45
Ivy Default Cache set to: C:\Users\janed\ivy2\cache
The jars for the packages stored in: C:\Users\janed\ivy2\jars
:: loading settings :: url = jar:file:/C:/Spark/spark-3.0.3-bin-hadoop3.2/jars/ivy-2.4.0.jar!/org/apache/ivy/core/settings/ivysettings.xml
org.apache.spark#spark-sql-kafka-0-10_2.12 added as a dependency
com.datastax.spark#spark-cassandra-connector_2.12 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-f514d583-2699-4638-ab7e-b6dbe9b5135b;1.0
  confs: [default]
  found org.apache.spark#spark-sql-kafka-0-10_2.12;3.0.3 in central
  found org.apache.spark#spark-token-provider-kafka-0-10_2.12;3.0.3 in central
  found org.apache.kafka#kafka-clients;2.4.1 in central
  found com.github.luben#zstd-jni;1.4.4-3 in central
  found org.lz4#lz4-java;1.7.1 in central
  found org.xerial.snappy#snappy-java;1.1.8.2 in central
  found org.slf4j#slf4j-api;1.7.30 in central
  found org.spark-project.spark#unused;1.0.0 in central
  found org.apache.commons#commons-pool2;2.6.2 in central
  found com.datastax.spark#spark-cassandra-connector_2.12;3.0.1 in central
  found com.datastax.spark#spark-cassandra-connector-driver_2.12;3.0.1 in central
  found com.datastax.oss#java-driver-core-shaded;4.10.0 in central
  found com.datastax.oss#native-protocol;1.4.12 in central
  found com.datastax.oss#java-driver-shaded-guava;25.1-jre-graal-sub-1 in central
  found com.typesafe#config;1.4.1 in central
  found io.dropwizard.metrics#metrics-core;4.1.16 in central
  found org.hdrhistogram#HdrHistogram;2.1.12 in central
  found org.reactivestreams#reactive-streams;1.0.3 in central

```

## VIII. Verify if Data is Saved in Cassandra

To verify that the spark application is working and the messages sent by the producer are fetched and saved in the Cassandra keyspace table, **cqlsh** is run and the following commands are executed:

USE dskeyspacebd;

SELECT \* FROM prod\_messages;

The output is shown below. All message values are successfully stored in the proper columns in the table.

CQLSH

X

+

▼

qqlsh> USE dskeyspacebd;

qqlsh:dskeyspacebd> SELECT \* FROM prod\_messages;

timeuuid_id	bike	bus	car	date_saved	jeepney	lgv_code	others	sensor_id	time_saved	total	truck	tryke
9fd04530-50c5-11ee-9eef-089798cc0a5d	0	1	1	09/12/2023	2	1230	0	sensor_04	01:07:01	4	0	0
a553190f-50c5-11ee-b0ba-089798cc0a5d	3	0	4	09/12/2023	0	1200	0	sensor_01	01:07:10	10	1	2
b2070d0f-50c5-11ee-afe8-089798cc0a5d	5	1	0	09/12/2023	2	1240	2	sensor_05	01:07:32	13	0	3
ad3539b0-50c5-11ee-967d-089798cc0a5d	5	1	4	09/12/2023	0	1290	2	sensor_10	01:07:23	15	1	2
a857f4a1-50c5-11ee-af4b-089798cc0a5d	3	1	2	09/12/2023	0	1210	0	sensor_02	01:07:15	9	0	3
a23ead1e-50c5-11ee-ab43-089798cc0a5d	5	0	1	09/12/2023	1	1270	0	sensor_08	01:07:05	11	2	2
b03e0cde-50c5-11ee-8e87-089798cc0a5d	2	2	0	09/12/2023	1	1250	0	sensor_06	01:07:29	7	0	2
ae60b4cf-50c5-11ee-bd2a-089798cc0a5d	5	1	1	09/12/2023	1	1230	0	sensor_04	01:07:25	10	1	1
b2097e11-50c5-11ee-9e9b-089798cc0a5d	4	2	4	09/12/2023	1	1230	1	sensor_04	01:07:32	14	1	1
ac93f09e-50c5-11ee-907e-089798cc0a5d	4	2	4	09/12/2023	1	1210	1	sensor_02	01:07:22	13	1	0
ba819d1e-50c5-11ee-836e-089798cc0a5d	4	2	0	09/12/2023	0	1220	1	sensor_03	01:07:46	11	1	3
bbbcb530-50c5-11ee-9cbb-089798cc0a5d	4	1	3	09/12/2023	0	1230	2	sensor_04	01:07:48	15	2	3
bc574700-50c5-11ee-9cc9-089798cc0a5d	4	1	3	09/12/2023	2	1230	1	sensor_04	01:07:49	14	2	1
b2a12a30-50c5-11ee-8f28-089798cc0a5d	2	0	2	09/12/2023	2	1240	2	sensor_05	01:07:33	8	0	0
bc526500-50c5-11ee-82f4-089798cc0a5d	0	0	3	09/12/2023	1	1210	0	sensor_02	01:07:49	8	2	2
bebd720f-50c5-11ee-b809-089798cc0a5d	3	1	2	09/12/2023	2	1220	1	sensor_03	01:07:53	12	0	3
ac96af00-50c5-11ee-bb19-089798cc0a5d	5	1	1	09/12/2023	0	1220	1	sensor_03	01:07:22	13	2	3
ad3057b0-50c5-11ee-9549-089798cc0a5d	2	0	0	09/12/2023	1	1000	1	sensor_09	01:07:23	7	0	3
a8e6a00f-50c5-11ee-a73e-089798cc0a5d	1	2	4	09/12/2023	2	1270	0	sensor_08	01:07:16	10	1	0
bff6dc70-50c5-11ee-90bb-089798cc0a5d	0	1	4	09/12/2023	0	1230	2	sensor_04	01:07:55	9	2	0
cddad1e-50c5-11ee-ab95-089798cc0a5d	0	2	0	09/12/2023	1	1250	2	sensor_06	01:08:18	7	1	1
ad3057b0-50c5-11ee-ad08-089798cc0a5d	4	2	3	09/12/2023	2	1230	2	sensor_04	01:07:23	15	1	1
cfae4c70-50c5-11ee-91d7-089798cc0a5d	2	1	4	09/12/2023	2	1250	0	sensor_06	01:08:21	10	1	0
bbbcb530-50c5-11ee-b9d5-089798cc0a5d	1	2	2	09/12/2023	1	1290	2	sensor_10	01:07:48	11	2	1
d5b851b0-50c5-11ee-b2d2-089798cc0a5d	4	1	2	09/12/2023	1	1250	2	sensor_06	01:08:31	12	0	2

CQLSH

X

+

▼

bd887400-50c5-11ee-b3a6-089798cc0a5d

4

1

3

09/12/2023

2

1240

1

sensor\_05

01:07:51

14

0

3

b2a87d2e-50c5-11ee-b1bb-089798cc0a5d

3

1

0

09/12/2023

2

1250

0

sensor\_06

01:07:33

8

2

0

a553190f-50c5-11ee-83c5-089798cc0a5d

1

2

0

09/12/2023

0

1290

0

sensor\_10

01:07:10

4

1

0

af8f2f40-50c5-11ee-913e-089798cc0a5d

5

2

2

09/12/2023

2

1260

2

sensor\_07

01:07:27

15

1

1

ce780b91-50c5-11ee-9192-089798cc0a5d

5

1

3

09/12/2023

1

1250

0

sensor\_06

01:08:19

10

0

0

a550a80f-50c5-11ee-9d94-089798cc0a5d

5

1

3

09/12/2023

2

1220

2

sensor\_03

01:07:10

16

1

2

c1c70091-50c5-11ee-80b1-089798cc0a5d

3

0

2

09/12/2023

0

1230

0

sensor\_04

01:07:58

7

2

0

b89c3b4f-50c5-11ee-b008-089798cc0a5d

5

1

0

09/12/2023

2

1260

0

sensor\_07

01:07:43

10

0

2

be232dde-50c5-11ee-8608-089798cc0a5d

0

2

3

09/12/2023

0

1240

0

sensor\_05

01:07:52

8

2

1

b59a45f0-50c5-11ee-b59f-089798cc0a5d

2

1

1

09/12/2023

2

1260

1

sensor\_07

01:07:38

7

0

0

a7c02170-50c5-11ee-8992-089798cc0a5d

3

2

1

09/12/2023

0

1000

2

sensor\_01

01:07:14

10

0

2

c3969070-50c5-11ee-af7b-089798cc0a5d

2

2

2

09/12/2023

2

1200

0

sensor\_09

01:08:01

10

2

0

b1fb4d40-50c5-11ee-8635-089798cc0a5d

5

1

1

09/12/2023

1

1270

2

sensor\_08

01:07:31

12

1

1

bc59df8f-50c5-11ee-9026-089798cc0a5d

0

2

4

09/12/2023

0

1000

2

sensor\_01

01:07:49

8

0

0

af05db00-50c5-11ee-842d-089798cc0a5d

4

2

4

09/12/2023

1

1290

1

sensor\_10

01:07:27

14

2

0

a857f4a1-50c5-11ee-ba1c-089798cc0a5d

0

2

1

09/12/2023

2

1230

2

sensor\_04

01:07:15

11

1

3

bff1fa70-50c5-11ee-8cfe-089798cc0a5d

4

0

1

09/12/2023

2

1280

2

sensor\_09

01:07:55

9

0

0

cc0df430-50c5-11ee-83ca-089798cc0a5d

4

2

0

09/12/2023

2

1250

2

sensor\_06

01:08:15

15

2

3

a2e2672e-50c5-11ee-8775-089798cc0a5d

0

2

4

09/12/2023

1

1290

1

sensor\_10

01:07:06

9

1

0

ba8a200-50c5-11ee-ac63-089798cc0a5d

1

1

0

09/12/2023

0

1290

1

sensor\_10

01:07:46

8

2

3

bc3b3400-50c5-11ee-91cc-089798cc0a5d

0

1

2

09/12/2023

1

1260

2

sensor\_07

01:07:49

9

1

2

d04905cf-50c5-11ee-8fdc-089798cc0a5d

3

0

2

09/12/2023

0

1250

0

sensor\_06

01:08:22

7

1

1

a7214961-50c5-11ee-b696-089798cc0a5d

2

1

1

09/12/2023

0

1280

2

sensor\_09

01:07:13

9

1

2

a7bd1430-50c5-11ee-b710-089798cc0a5d

2

2

0

09/12/2023

0

1210

2

sensor\_02

01:07:14

7

1

0

ac996ade-50c5-11ee-a800-089798cc0a5d

0

0

3

09/12/2023

0

1290

2

sensor\_10

01:07:22

9

1

3

b02a84de-50c5-11ee-bb24-089798cc0a5d

4

2

4

09/12/2023

0

1270

2

sensor\_08

01:07:28

16

1

3

b2a87d2e-50c5-11ee-a9e6-089798cc0a5d

5

0

1

09/12/2023

1

1290

0

sensor\_10

01:07:33

11

1

3

(654 rows)

qqlsh:dskeyspacebd> |

The Big Data Streaming Pipeline is now complete using Pycharm, Kafka, Pyspark, and Cassandra.

## References:

- Marquez, P. (2020). *Big Data Streaming*. Medium. <https://medium.com/analytics-vidhya/big-data-streaming-c483195751ee>
- Foley, D. (2019). *Let's Build a Streaming Data Pipeline*. Medium. <https://towardsdatascience.com/lets-build-a-streaming-data-pipeline-e873d671fc57>
- Jeyaraman, B. (2023). *Building Real-time Data Pipelines with Kafka and Machine Learning*. LinkedIn. <https://www.linkedin.com/pulse/building-real-time-data-pipelines-kafka-machine-brindha-jeyaraman>
- Bouras, S. (2017). *Reliable and faster integration tests with Apache Cassandra*. <https://medium.com/@saidbouras/running-integration-tests-with-apache-cassandra-42305dc260a6>
- Kaplarevic, V. (2020). *How to Install Cassandra on Windows 10*. PhoenixNAP. <https://phoenixnap.com/kb/install-cassandra-on-windows>
- DataMaking (2019). *Real-Time Spark Project | Data Analysis | Structured Streaming | Part 6.2 | DM | DataMaking*. YouTube. <https://www.youtube.com/watch?v=ZjsObNeUUd4&list=PLe1T0uBrDrfOYE8OwQvooPjmnP1zY3wFe&index=10>
- Cherif, A. N. (2020). *Beginners guide to learn Cassandra-Part 1: Cassandra overview*. Medium. <https://medium.com/@aymannaitcherif/beginners-guide-to-learn-cassandra-part-1-cassandra-overview-bf1634e4ce30>
- Apache Cassandra (n.d.). *The Cassandra Query Language (CQL)*. The Apache Software Foundation. <https://cassandra.apache.org/doc/latest/cassandra/cql/>
- DataStax (n.d.). *Querying tables*. DataStax Documentation. [https://docs.datastax.com/en/cql-oss/3.x/cql/cql\\_using/useQueryDataTOC.html](https://docs.datastax.com/en/cql-oss/3.x/cql/cql_using/useQueryDataTOC.html)
- jumpstartCS (2020). *Apache Cassandra Tutorial*. YouTube. <https://www.youtube.com/playlist?list=PLalrWAGybpB-L1PGA-NfFu2uiWHEsdscD>