MongoDB Assignment Submission Doc

This file details and describes all the attached files for the Kafka Assignment

Tools Used:

1. Python3
2. Confluent Kafka
3. MongoDB Atlas
4. MongoDB Compass
5. Flask

Files Attached:

1. mongo_Assignment.pdf – This file
2. delivery_trip_truck_data_new.csv – The csv raw data used to push to the kafka topic
3. producer_Code.py – Python producer script
4. Consumer_code.py – Python consumer script
5. flaskmongo.py – Python script which creates a Flask App by opening on 5000 port just paste your aggregation pipeline to get the result
6. Dockerfile – Docker file that can be used to build an docker image for flask app.
7. producer.Dockerfile – Docker file that can be used for producer.
8. consumer.Dockerfile-Docker file that can be used for consumer
9. docker-compose.yml-Docker compose file for flask app and mongodb
10. docker-compose.yml– YAML docker compose file that details the docker image, environment details (Kafka config) and the consumer script to parallelize.\
11. .env – for environment variables.
12. Avro_schema.json – which contain avro_schema
13. Index.html – which contain code for frontend of flask app.
14. Sample_mongo_query.py- which contain query for aggregation which can be used in flask app.

Process and File Descriptions:

Step 1

Created a kafka topic called 'logistics_data1' with 6 partitions and I made sure to save the API keys for the producer. I also created an appropriate schema value and key to prepare the kafka topic for data ingestion/retrieval looking at the delivery_trip_truck_data_new.csv file. I especially made sure to handle the 'Nan' values by replacing them with the None value' if the field is string type. And also make all data type to string.

Step 2

I created a producer script called "Producer_code.py" that fetches the data from the above- mentioned Kafka topic. The script also serializes the data into Avro format and uses index value as the key.

Step 3

Created a mongodb database called ‘logistic_data and created an empty collection called ‘delivery_truck_data’ so that data can be stored once a consumer script can be run.

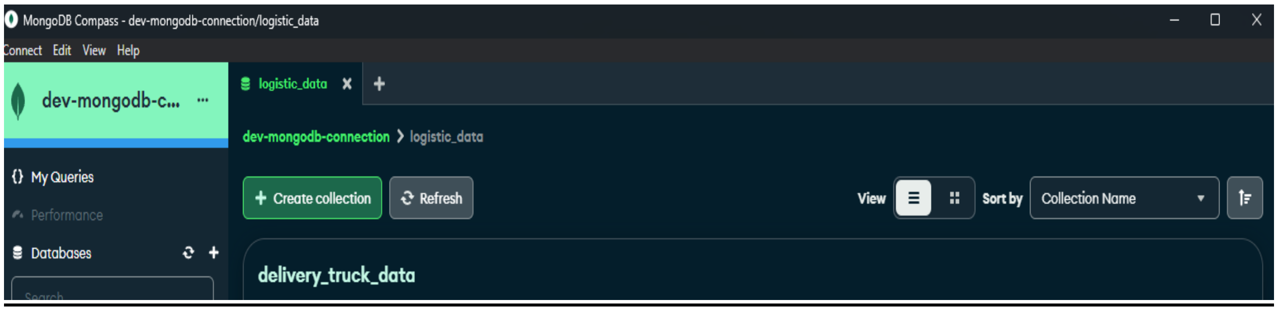
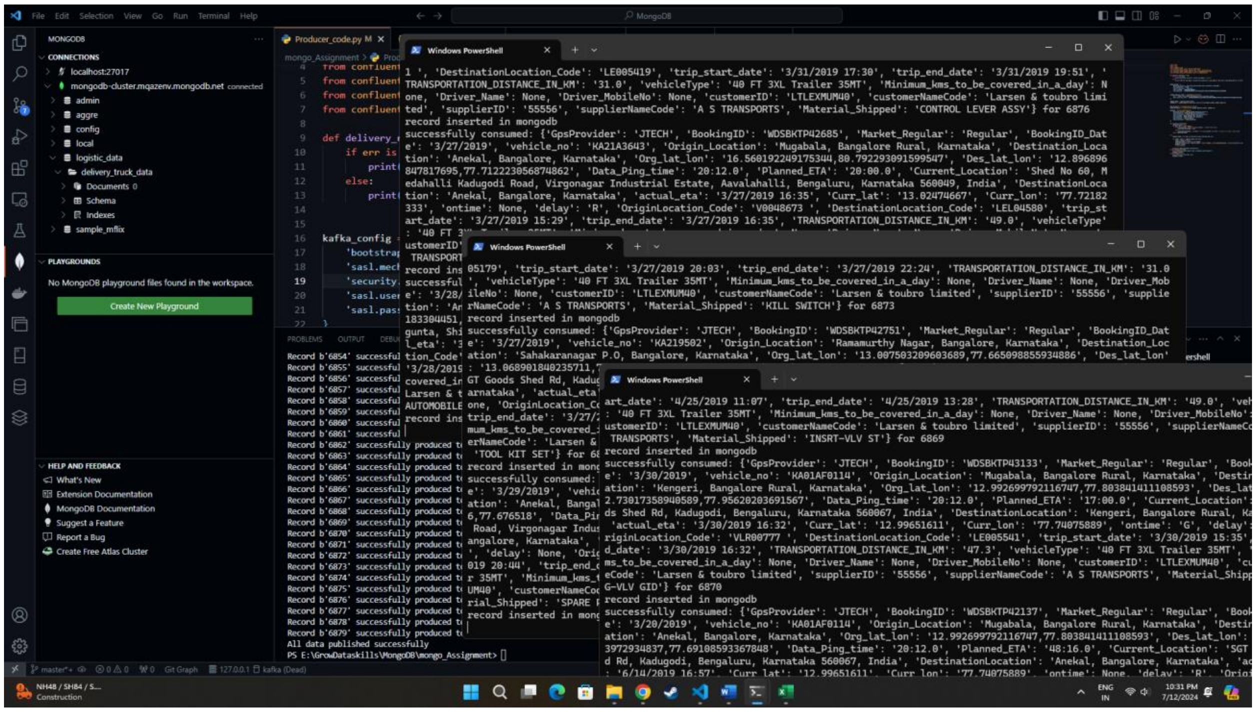


Fig. 1: **MongoDB database name and Collection Name**

Step 4

Then created a “Consumer_code.py” where i wrote the logic for the consumer in which before dumping data to the mongodb collection made sure to convert all the string type data which were published by producer will get convert back to it’s original data type also give the mongodb collection connection in the consumer code.I make sure that there are no duplicate records pushed when the consumer runs.



```
1 ', 'DestinationLocation_Code': 'LE005419', 'trip_start_date': '3/31/2019 17:30', 'trip_end_date': '3/31/2019 19:51', '
TRANSPORTATION_DISTANCE_IN_KM': '31.0', 'vehicleType': '40 FT 3XL Trailer 35MT', 'Minimum_kms_to_be_covered_in_a_day': N
one, 'Driver_Name': None, 'Driver_MobileNo': None, 'customerID': 'LTLEXMUM40', 'customerNameCode': 'Larsen & toubro limi
ted', 'supplierID': '55556', 'supplierNameCode': 'A S TRANSPORTS', 'Material_Shipped': 'CONTROL LEVER ASSY'} for 6876
record inserted in mongodb
successfully consumed: {'GpsProvider': 'JTECH', 'BookingID': 'WDSBKTP42685', 'Market_Regular': 'Regular', 'BookingID_Dat
e': '3/27/2019', 'vehicle_no': 'KA21A3643', 'Origin_Location': 'Mugabala, Bangalore Rural, Karnataka', 'Destination_Loca
tion': 'Anekal, Bangalore, Karnataka', 'Org_lat_lon': '16.560192249175344,80.792293091599547', 'Des_lat_lon': '12.896896
847817695,77.712223056874862', 'Data_Ping_time': '20:12.0', 'Planned_ETA': '20:00.0', 'Current_Location': 'Shed No 60, M
edahalli Kadugodi Road, Virgonagar Industrial Estate, Aavalahalli, Bengaluru, Karnataka 560049, India', 'DestinationLoca
tion': 'Anekal, Bangalore, Karnataka', 'actual_eta': '3/27/2019 16:35', 'Curr_lat': '13.02474667', 'Curr_lon': '77.72182
333', 'ontime': None, 'delay': 'R', 'OriginLocation_Code': 'V0048673 ', 'DestinationLocation_Code': 'LEL04580', 'trip_st
art_date': '3/27/2019 15:29', 'trip_end_date': '3/27/2019 16:35', 'TRANSPORTATION_DISTANCE_IN_KM': '49.0', 'vehicleType'
```

```
05179', 'trip_start_date': '3/27/2019 20:03', 'trip_end_date': '3/27/2019 22:24', 'TRANSPORTATION_DISTANCE_IN_KM': '31.0
', 'vehicleType': '40 FT 3XL Trailer 35MT', 'Minimum_kms_to_be_covered_in_a_day': None, 'Driver_Name': None, 'Driver_Mob
ileNo': None, 'customerID': 'LTLEXMUM40', 'customerNameCode': 'Larsen & toubro limited', 'supplierID': '55556', 'supplie
rNameCode': 'A S TRANSPORTS', 'Material_Shipped': 'KILL SWITCH'} for 6873
record inserted in mongodb
successfully consumed: {'GpsProvider': 'JTECH', 'BookingID': 'WDSBKTP42751', 'Market_Regular': 'Regular', 'BookingID_Dat
e': '3/27/2019', 'vehicle_no': 'KA219502', 'Origin_Location': 'Ramamurthy Nagar, Bangalore, Karnataka', 'Destination_Loc
ation': 'Sahakaranagar P.O, Bangalore, Karnataka', 'Org_lat_lon': '13.007503209603689,77.665098855934886', 'Des_lat_lon'
```

```
art_date': '4/25/2019 11:07', 'trip_end_date': '4/25/2019 13:28', 'TRANSPORTATION_DISTANCE_IN_KM': '49.0', 'veh
: '40 FT 3XL Trailer 35MT', 'Minimum_kms_to_be_covered_in_a_day': None, 'Driver_Name': None, 'Driver_MobileNo':
ustomerID': 'LTLEXMUM40', 'customerNameCode': 'Larsen & toubro limited', 'supplierID': '55556', 'supplierNameCo
 TRANSPORTS', 'Material_Shipped': 'INSRT-VLV ST'} for 6869
record inserted in mongodb
successfully consumed: {'GpsProvider': 'JTECH', 'BookingID': 'WDSBKTP43133', 'Market_Regular': 'Regular', 'Book
e': '3/30/2019', 'vehicle_no': 'KA01AF0114', 'Origin_Location': 'Mugabala, Bangalore Rural, Karnataka', 'Destin
ation': 'Kengeri, Bangalore Rural, Karnataka', 'Org_lat_lon': '12.992699792116747,77.803841411108593', 'Des_lat
2.73017358940589,77.95620203691567', 'Data_Ping_time': '20:12.0', 'Planned_ETA': '17:00.0', 'Current_Location':
ds Shed Rd, Kadugodi, Bengaluru, Karnataka 560067, India', 'DestinationLocation': 'Kengeri, Bangalore Rural, Ka
 'actual_eta': '3/30/2019 16:32', 'Curr_lat': '12.99651611', 'Curr_lon': '77.74075889', 'ontime': 'G', 'delay':
riginLocation_Code': 'VLR00777 ', 'DestinationLocation_Code': 'LE005541', 'trip_start_date': '3/30/2019 15:35',
d_date': '3/30/2019 16:32', 'TRANSPORTATION_DISTANCE_IN_KM': '47.3', 'vehicleType': '40 FT 3XL Trailer 35MT',
ms_to_be_covered_in_a_day': None, 'Driver_Name': None, 'Driver_MobileNo': None, 'customerID': 'LTLEXMUM40', 'cu
eCode': 'Larsen & toubro limited', 'supplierID': '55556', 'supplierNameCode': 'A S TRANSPORTS', 'Material_Shipp
G-VLV GID'} for 6870
record inserted in mongodb
successfully consumed: {'GpsProvider': 'JTECH', 'BookingID': 'WDSBKTP42137', 'Market_Regular': 'Regular', 'Book
e': '3/20/2019', 'vehicle_no': 'KA01AF0114', 'Origin_Location': 'Mugabala, Bangalore Rural, Karnataka', 'Destin
ation': 'Anekal, Bangalore, Karnataka', 'Org_lat_lon': '12.992699792116747,77.803841411108593', 'Des_lat_lon':
3972934837,77.69108593367848', 'Data_Ping_time': '20:12.0', 'Planned_ETA': '48:16.0', 'Current_Location': 'SGT
d Rd, Kadugodi, Bengaluru, Karnataka 560067, India', 'DestinationLocation': 'Anekal, Bangalore, Karnataka', 'ac
: '6/14/2019 16:57', 'Curr_lat': '12.99651611', 'Curr_lon': '77.74075889', 'ontime': None, 'delay': 'R', 'Origi
```

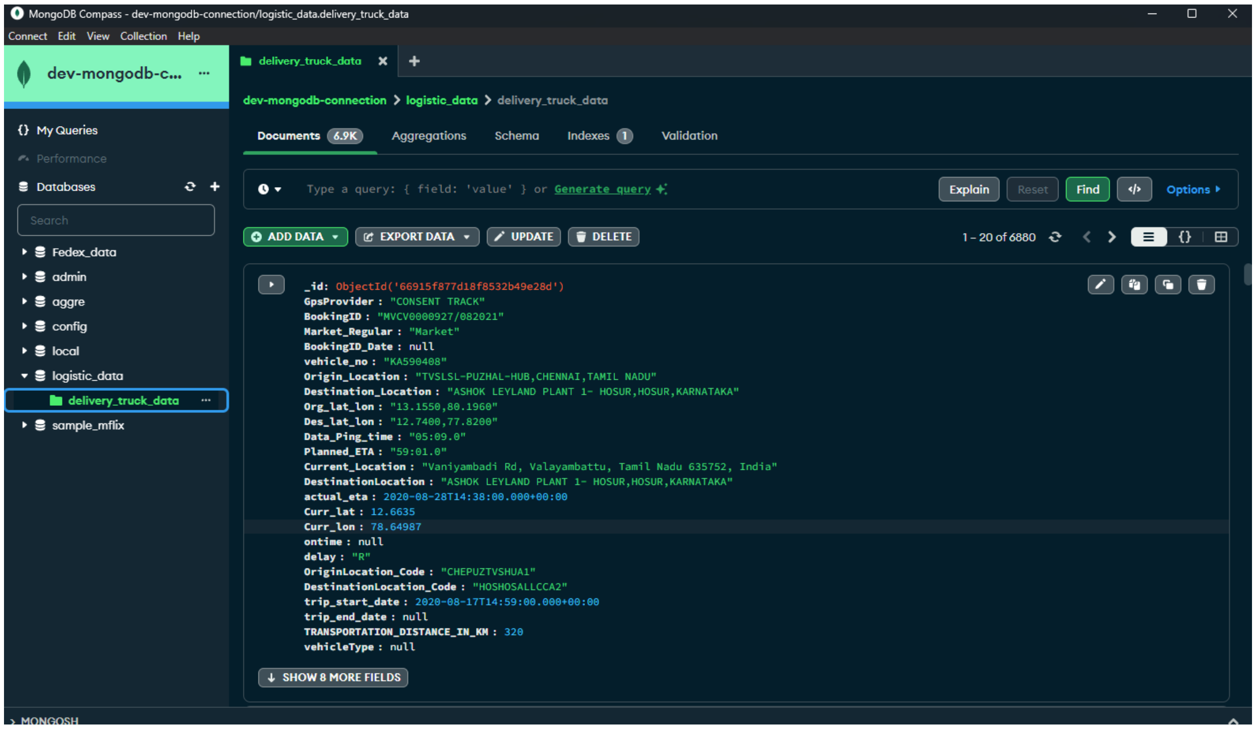
```
Record b'6854' successful
Record b'6855' successful
Record b'6856' successful
Record b'6857' successful
Record b'6858' successful
Record b'6859' successful
Record b'6860' successful
Record b'6861' successful
Record b'6862' successfully produced t
Record b'6863' successfully produced t
Record b'6864' successfully produced t
Record b'6865' successfully produced t
Record b'6866' successfully produced t
Record b'6867' successfully produced t
Record b'6868' successfully produced t
Record b'6869' successfully produced t
Record b'6870' successfully produced t
Record b'6871' successfully produced t
Record b'6872' successfully produced t
Record b'6873' successfully produced t
Record b'6874' successfully produced t
Record b'6875' successfully produced t
Record b'6876' successfully produced t
Record b'6877' successfully produced t
Record b'6878' successfully produced t
Record b'6879' successfully produced t
All data published successfully
PS E:\GrowDataskills\MongoDB\mongo_Assignment>
```

Fig. 2: **A Producer along with three Consumers to parallelize the processing**

Step 5

To achieve scaling, I first created a Dockerfile for producer and one for consumer then create the docker compose file and used both of this dockerfile in it as we need to scaled the system hence created 3 consumer instance in the docker-compose file and also create the dockerfile and docker-compose file for the flask app.

We can also check the data using Mongodb Compass:



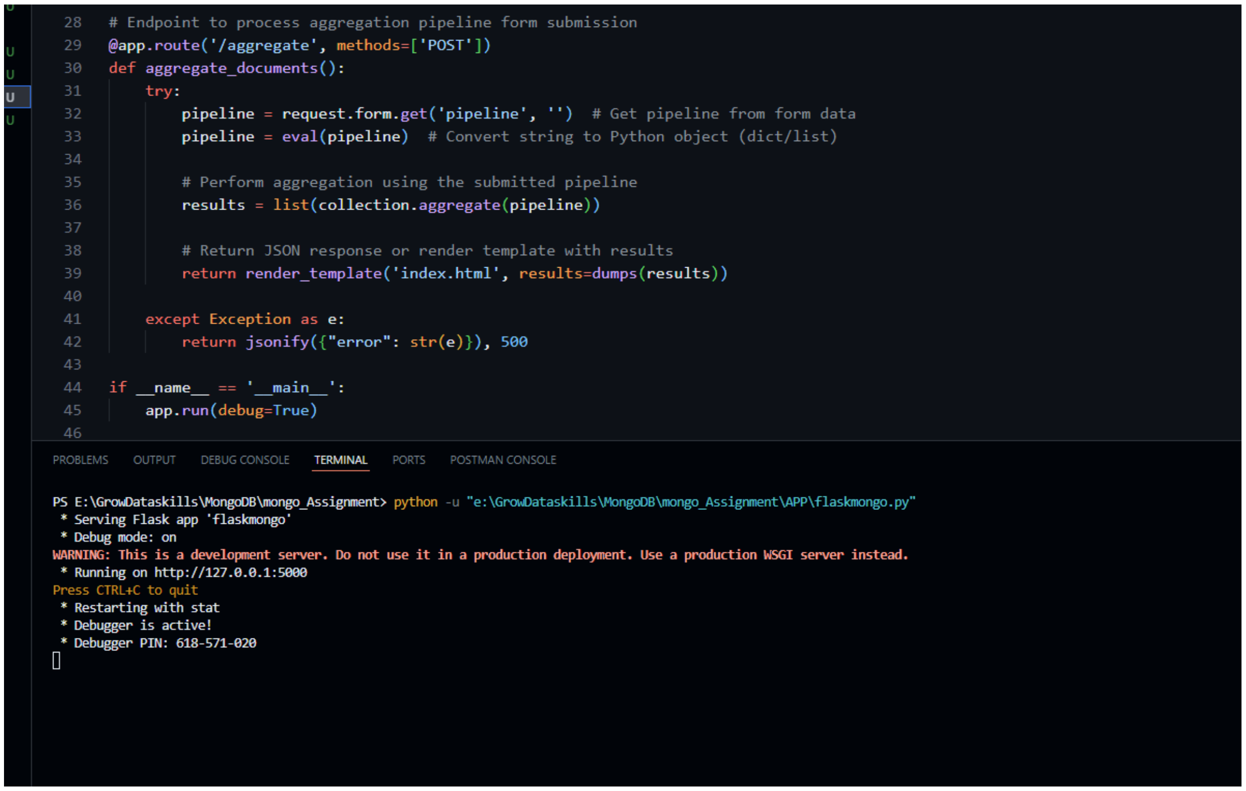
```
_id: ObjectId('66915f877d18f8532b49e28d')
GpsProvider : "CONSENT TRACK"
BookingID : "MVCV0000927/082021"
Market_Regular : "Market"
BookingID_Date : null
vehicle_no : "KA590408"
Origin_Location : "TVSLSL-PUZHAL-HUB,CHENNAI,TAMIL NADU"
Destination_Location : "ASHOK LEYLAND PLANT 1- HOSUR,HOSUR,KARNATAKA"
Org_lat_lon : "13.1550,80.1960"
Des_lat_lon : "12.7400,77.8200"
Data_Ping_time : "05:09.0"
Planned_ETA : "59:01.0"
Current_Location : "Vaniyambadi Rd, Valayambattu, Tamil Nadu 635752, India"
DestinationLocation : "ASHOK LEYLAND PLANT 1- HOSUR,HOSUR,KARNATAKA"
actual_eta : 2020-08-28T14:38:00.000+00:00
Curr_lat : 12.6635
Curr_lon : 78.64987
ontime : null
delay : "R"
OriginLocation_Code : "CHEPUZTVSHUA1"
DestinationLocation_Code : "HOSHOSALLCCA2"
trip_start_date : 2020-08-17T14:59:00.000+00:00
trip_end_date : null
TRANSPORTATION_DISTANCE_IN_KM : 320
vehicleType : null
```

Fig. 3: **MongoDB data after dumping records using 3 consumers.**

Step 6

Using Flask I was able to made an APP by using little bit of Css and html forms which run on 5000 port on localhost by opening in browser we can see the output.

This is a snapshot of the code running.

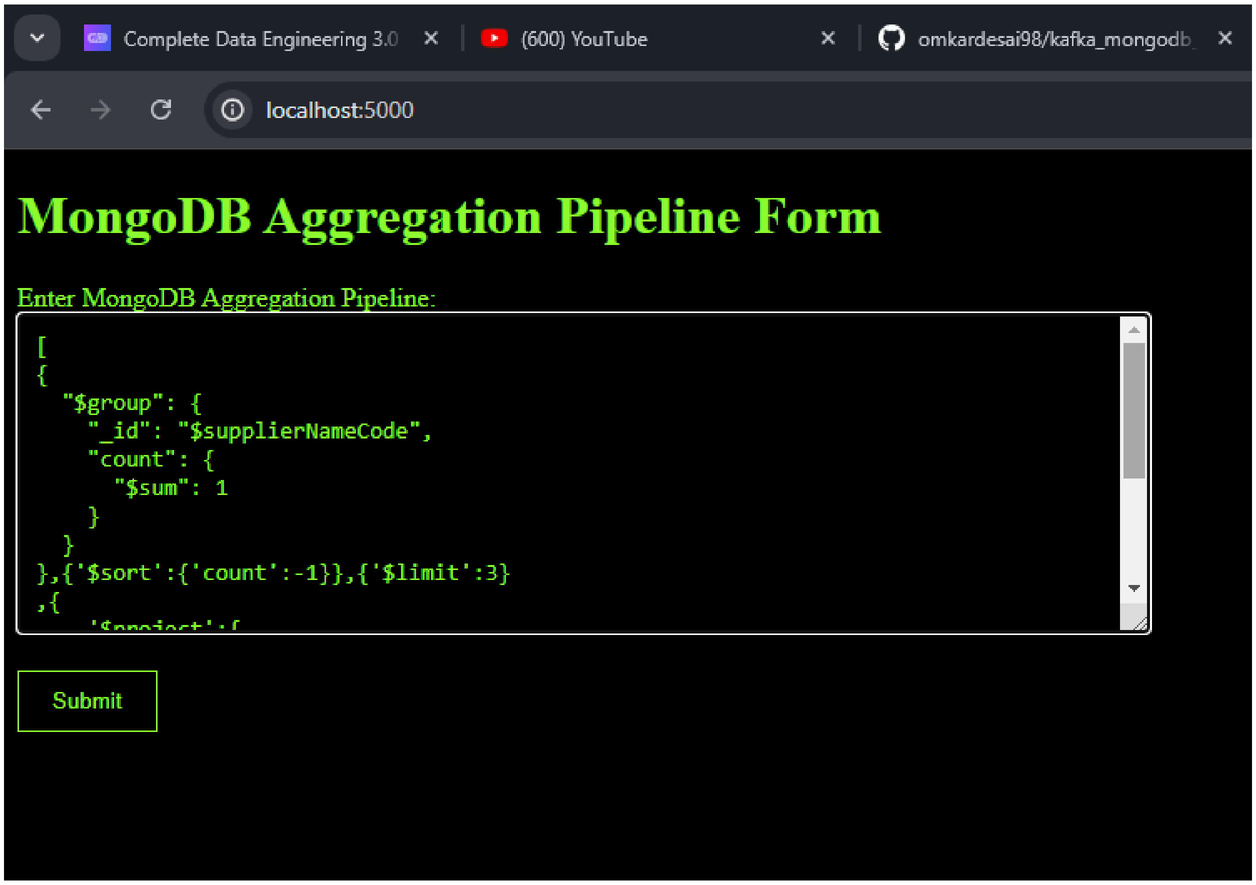


```python
# Endpoint to process aggregation pipeline form submission
@app.route('/aggregate', methods=['POST'])
def aggregate_documents():
    try:
        pipeline = request.form.get('pipeline', '')  # Get pipeline from form data
        pipeline = eval(pipeline)  # Convert string to Python object (dict/list)

        # Perform aggregation using the submitted pipeline
        results = list(collection.aggregate(pipeline))

        # Return JSON response or render template with results
        return render_template('index.html', results=dumps(results))

    except Exception as e:
        return jsonify({"error": str(e)}), 500

if __name__ == '__main__':
    app.run(debug=True)
```

```
PS E:\GrowDataskills\MongoDB\mongo_Assignment> python -u "e:\GrowDataskills\MongoDB\mongo_Assignment\APP\flaskmongo.py"
 * Serving Flask app 'flaskmongo'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 618-571-020
```

Fig. 4: **Running Flask Application on localhost:5000**

Below image shows the flask app with a aggregation pipeline (i.e. to get the top 3 suppliers who provide most vehicles).

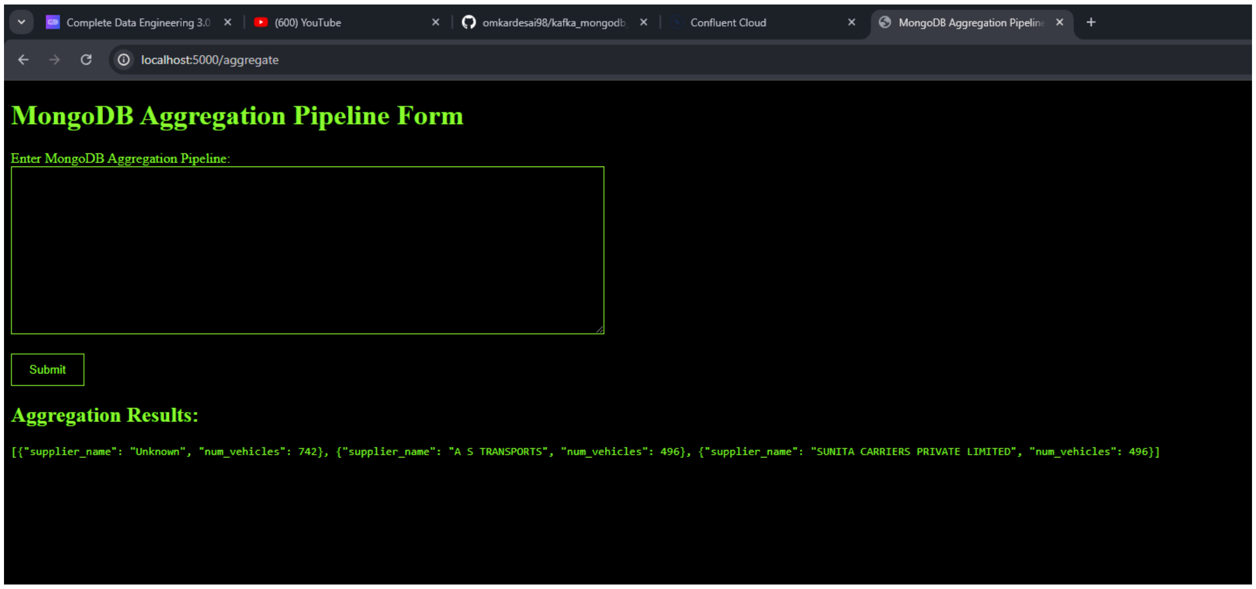


MongoDB Aggregation Pipeline Form

Enter MongoDB Aggregation Pipeline:

```
[
{
  "$group": {
    "_id": "$supplierNameCode",
    "count": {
      "$sum": 1
    }
  }
},{'$sort':{'count':-1}},{'$limit':3}
,{
    '$project':{
```

Fig. 5: **Running Flask Application with aggregation pipeline**


Complete Data Engineering 3.0
(600) YouTube
omkardesai98/kafka_mongodb
Confluent Cloud
MongoDB Aggregation Pipeline
localhost:5000/aggregate
Submit

MongoDB Aggregation Pipeline Form

Enter MongoDB Aggregation Pipeline:

Aggregation Results:

```
[{"supplier_name": "Unknown", "num_vehicles": 742}, {"supplier_name": "A S TRANSPORTS", "num_vehicles": 496}, {"supplier_name": "SUNITA CARRIERS PRIVATE LIMITED", "num_vehicles": 496}]
```

Fig. 6: **Aggregation result after clicking on submit button**