Data Science CA2

Alireza Karimi 810101492

Mohammad Taha Majlesi 810101504

Mohammad Hossein Mazhari 810101520

Guides for working with terminal and kafka.

```
### link for this commands:
https://kafka.apache.org/quickstart



# This initializes the log metadata directory for KRaft.
$ cd ~/Documents/install_file/kafka/kafka_2.13-4.0.0
# The address to my CA files:
# cd Documents/uni/term6/Data-Science/CA/2/DS-CA2/

# Generate a Cluster UUID
$ KAFKA_CLUSTER_ID="$(bin/kafka-storage.sh random-uuid)"



# Format Log Directories
$ bin/kafka-storage.sh format --standalone -t $KAFKA_CLUSTER_ID -c config/se



# Start the Kafka Server
$ bin/kafka-server-start.sh config/server.properties



# Create a topic to store your events (In a new terminal)
$ bin/kafka-topics.sh --create --topic quickstart-events --bootstrap-server local
```

```
# Write some events into the topic
$ bin/kafka-console-producer.sh --topic quickstart-events --bootstrap-server lo



# Read the events(In a new terminal)
$ bin/kafka-console-consumer.sh --topic quickstart-events --from-beginning --l




# Terminate the Kafka environment(Stop the producer and consumer clients and
$ rm -rf /tmp/kafka-logs /tmp/kraft-combined-logs



#### check that kakfa is running
$ netstat -tuln | grep 9092
or this one:
$ nc -zv localhost 9092



#To run the python code "darooghe_pulse":
python darooghe_pulse.py



lsof -i :9092



#Stop the existing container using port 9000
###Run this to find the container using port 9000:
$ docker ps

###Look for something exposing 0.0.0.0:9000 and note the CONTAINER ID or NA
$docker stop <container_id_or_name>
```

1. List all topics

```
bin/kafka-topics.sh \
  --bootstrap-server localhost:9092 \
  --list

This will print every topic name on your broker.



2. Describe a topic (partitions, replicas, offsets)

bin/kafka-topics.sh \
  --bootstrap-server localhost:9092 \
  --describe \
  --topic <your_topic_name>

That shows you partition counts, leader/ISR, and log size (i.e. high-watermark off



3. Peek at the messages in a topic

bin/kafka-console-consumer.sh \
  --bootstrap-server localhost:9092 \
  --topic <your_topic_name> \
  --from-beginning \
  --max-messages 10
```

Part1: Environment Setup

Input

```
(base) alireza@alirezas-MacBook-Air DS-CA2 % python darooghe_pulse.py
```

output

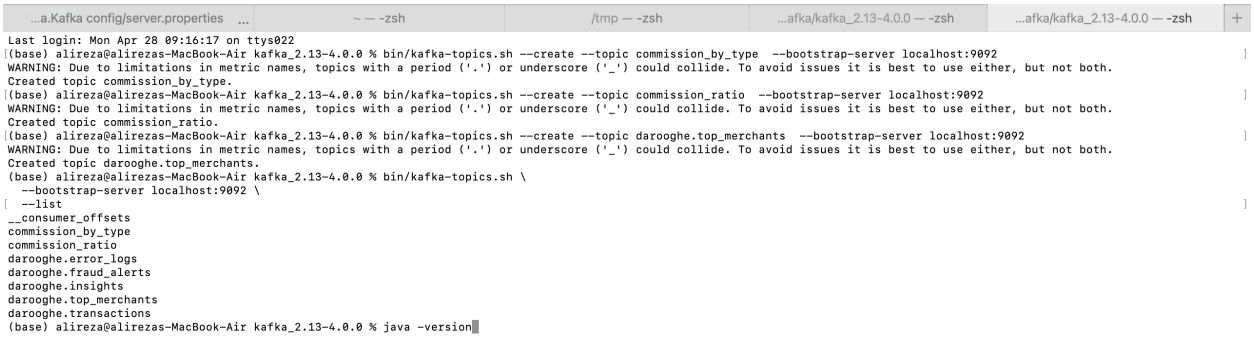
```
2025-04-19 20:08:44,566 INFO Producing 20000 historical events...
2025-04-19 20:08:45,658 INFO Historical events production completed.
2025-04-19 20:08:45,658 INFO Starting continuous event production...
```

---

input

```
(base) alireza@alirezas-MacBook-Air kafka_2.13-4.0.0 % bin/kafka-topics.sh \
--bootstrap-server localhost:9092 \
--list
```

output



Another part for creating kafdrop but unsuccessful

Ahh yep — that error tells us exactly what's up 👇

> 🚫 Port 9000 is already being used on your Mac, probably by a previously launched Kafdrop container that didn't stop cleanly.

✅ Quick Fix Options

🔧 Option 1: Stop the running container using port 9000

Run this to find it:

```
bash
CopyEdit
docker ps
```

Look for the container using **port 9000** (check the "PORTS" column).

Then stop it:

```
bash
CopyEdit
docker stop <container_id>
```

Example:

```
bash
CopyEdit
docker stop aea9b8844959
```

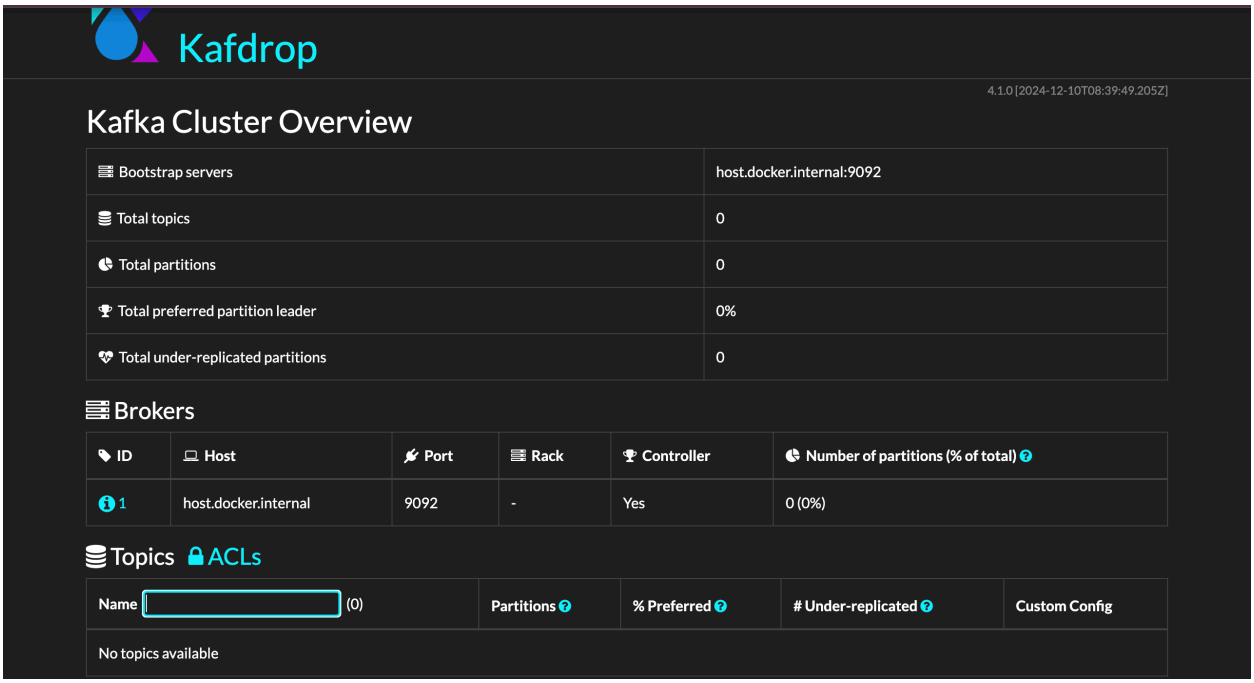
Now try launching Kafdrop again.

🔧 Option 2: Run Kafdrop on a different port (e.g., 9001)

If you want to keep the other container running, just change the port:

```
bash
CopyEdit
docker run --platform linux/amd64 -d --rm -p 9001:9000 \
  -e KAFKA_BROKERCONNECT=host.docker.internal:9092 \
  obsidiandynamics/kafdrop
```

Now you can visit:



Absolutely — you can totally monitor the `darooghe.error_logs` topic straight from the command line! 🔧 Here's how to do it using **Kafka's CLI tools** that come with your Kafka installatio

💻 Option 2: On Your Mac Terminal (If Kafka CLI installed)

If you installed Kafka locally (e.g., via `brew`), just run this on your Mac terminal:

```
kafka-console-consumer \
  --bootstrap-server localhost:9092 \
  --topic darooghe.error_logs \
  --from-beginning \
  --property print.key=true \
  --property print.value=true
```

This is the output, when I run load_data.py:

```
root
 |-- amount: long (nullable = true)
 |-- commission_amount: long (nullable = true)
 |-- commission_type: string (nullable = true)
 |-- customer_id: string (nullable = true)
 |-- customer_type: string (nullable = true)
 |-- device_info: struct (nullable = true)
 |    |-- app_version: string (nullable = true)
 |    |-- device_model: string (nullable = true)
 |    |-- os: string (nullable = true)
 |-- failure_reason: string (nullable = true)
 |-- location: struct (nullable = true)
 |    |-- lat: double (nullable = true)
 |    |-- lng: double (nullable = true)
 |-- merchant_category: string (nullable = true)
```

```
 |-- merchant_id: string (nullable = true)
 |-- payment_method: string (nullable = true)
 |-- risk_level: long (nullable = true)
 |-- status: string (nullable = true)
 |-- timestamp: string (nullable = true)
 |-- total_amount: long (nullable = true)
 |-- transaction_id: string (nullable = true)
 |-- vat_amount: long (nullable = true)

+-------+----------------+--------------+----------+------------+----------
| amount|commission_amount|commission_type|customer_id|customer_type|
+-------+----------------+--------------+----------+------------+----------
| 681670|           13633|        tiered|  cust_498|         CIP|{2.4.1, Samsung G...|
|1514059|           30281|          flat|  cust_169|         CIP|  {NULL, NULL, NULL}|
| 499205|            9984|   progressive|  cust_879|    business|{1.9.5, Google Pi...
| 255756|            5115|   progressive|   cust_28|  individual|{3.1.0, iPhone 15...|
|1327604|           26552|          flat|  cust_321|         CIP|{3.1.0, iPhone 15...|
+-------+----------------+--------------+----------+------------+----------
only showing top 5 rows

+----------------+---------------+-----------------+-------------------------
|merchant_category|total_commission|   avg_commission|commission_to_amour
+----------------+---------------+-----------------+-------------------------
|          retail|       33664610|20255.481347773766|      0.01999950804145335|
|   entertainment|       33235002|20389.571779141104|     0.019999521166901600
|    food_service|       33755412|21031.409345794393|       0.01999954244828
|      government|       33785856|20702.117647058825|     0.0199995276951645
|  transportation|       34291043| 20423.4919594997|     0.0199995088400672
+----------------+---------------+-----------------+-------------------------

(spark-env) (base) alireza@alirezas-MacBook-Air DS-CA2 % python3 load_data.
25/04/21 18:29:45 WARN Utils: Your hostname, alirezas-MacBook-Air.local resolv
25/04/21 18:29:45 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to anoth
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLev
25/04/21 18:29:46 WARN NativeCodeLoader: Unable to load native-hadoop libra
```

📊 Commission Efficiency by Merchant Category:

```
+----------------+---------------+-----------------+------------------------
|merchant_category|total_commission|avg_commission    |commission_to_amour
+----------------+---------------+-----------------+------------------------
|retail          |33664610       |20255.481347773766|0.01999950804145335     |
|entertainment   |33235002       |20389.571779141104|0.019999521166901603
|food_service    |33755412       |21031.409345794393|0.019999542448285
|government      |33785856       |20702.117647058825|0.019999527695164585
|transportation  |34291043       |20423.4919594997  |0.019999508840067227
+----------------+---------------+-----------------+------------------------
```

📈 Commission Model Comparison:

```
+----------------+-----------------+--------+-----------------+
|merchant_category|avg_real         |avg_flat|avg_progressive  |
+----------------+-----------------+--------+-----------------+
|retail          |20255.481347773766|15000.0 |20255.979602888085|
|entertainment   |20389.571779141104|15000.0 |20390.059950920244|
|food_service    |21031.409345794393|15000.0 |21031.890504672887|
|government      |20702.117647058825|15000.0 |20702.606544117636|
|transportation  |20423.4919594997  |15000.0 |20423.993531864206|
+----------------+-----------------+--------+-----------------+
```

⏰ Transactions per Hour:

```
+----+-----+
|hour|count|
+----+-----+
|   0|  110|
|   1|  112|
|   2|  111|
|   3|  122|
|   4|  100|
|   5|  115|
|   6|  120|
|   7|   91|
|   8|  102|
|   9|  110|
```

```
|  10|  101|
|  11|  109|
|  12|  119|
|  13|   91|
|  14|  114|
|  15|  118|
|  16| 2325|
|  17| 3488|
|  18|  110|
|  19|  111|
|  20|  113|
|  21|   95|
|  22|  103|
|  23|  118|
+----+-----+

📅 Transactions per Day of Week:
+-----------+-----+
|day_of_week|count|
+-----------+-----+
|          1|  779|
|          2| 7429|
+-----------+-----+

🧍 Customer Segmentation (Low/Medium/High frequency):
+-------+-----+
|segment|count|
+-------+-----+
|    Low|  672|
| Medium|  328|
+-------+-----+

🛒 Avg Amount and Volume by Merchant Category:
+-----------------+---------+------------------+
|merchant_category|txn_count|avg_amount        |
+-----------------+---------+------------------+
```

```
|transportation   |1679     |1021199.6765932102|
|retail           |1662     |1012798.9801444043|
|government       |1632     |1035130.3272058824|
|entertainment    |1630     |1019502.9975460123|
|food_service     |1605     |1051594.5252336448|
+-----------------+---------+------------------+

🌞 Transactions by Part of Day:
+-----------+-----+
|part_of_day|count|
+-----------+-----+
|  Afternoon| 6255|
|    Evening|  650|
|    Morning|  633|
|      Night|  670|
+-----------+-----+

📊 Daily Spend Trends:
+----------+------------+---------+
|date      |total_amount|txn_count|
+----------+------------+---------+
|2025-04-20|794293966   |779      |
|2025-04-21|7642503989  |7429     |
+----------+------------+---------+
```

After loading data to mongoDB using this simple load_to_mongo.py:

```
import json
from pymongo import MongoClient

# 1. Connect to MongoDB
client = MongoClient("mongodb://localhost:27017/")
db = client["darooghe"]  # Your database name
collection = db["transactions"]  # Collection/table name

# 2. Open your JSONL file (line by line)
```

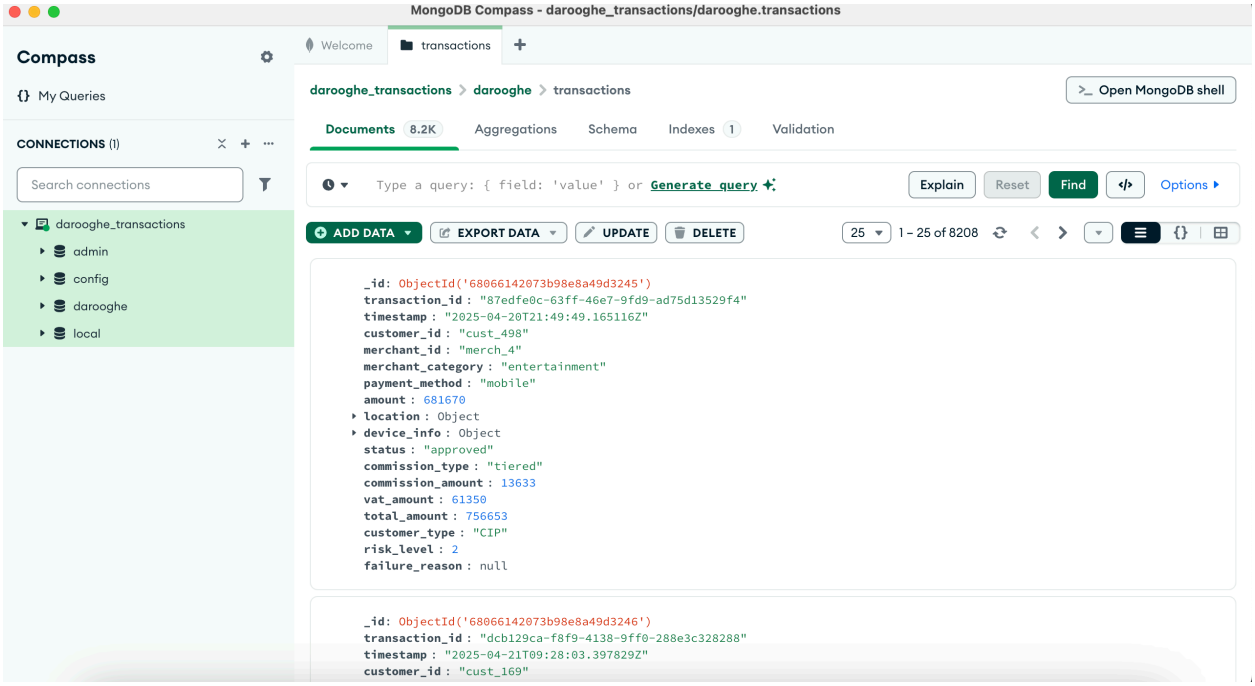
```
with open("transactions.jsonl", "r") as f:
    data = [json.loads(line) for line in f]

# 3. Insert into MongoDB
collection.insert_many(data)

print(f"✅ Inserted {len(data)} documents into MongoDB!")
```

more complex one:

```
import json
from pymongo import MongoClient
from datetime import datetime

# Connect to MongoDB
client = MongoClient("mongodb://localhost:27017/")
db = client["darooghe"]
collection = db["transactions"]

# Optional: Clean old data (during development)
collection.delete_many({})
print("🧹 Old data cleared.")

# Load and enrich JSONL file
with open("transactions.jsonl", "r") as f:
    data = [json.loads(line) for line in f]

# Add 'date' field from 'timestamp'
for tx in data:
    try:
        iso_ts = tx.get("timestamp")
        if iso_ts:
            dt_obj = datetime.fromisoformat(iso_ts.replace("Z", "+00:00"))
            tx["date"] = dt_obj.strftime("%Y-%m-%d")  # Add date as YYYY-MM-DD
    except Exception as e:
        print(f"⚠️ Skipped bad timestamp in tx {tx.get('transaction_id')}: {e}")
```

```
# Insert into MongoDB
if data:
    collection.insert_many(data)
    print(f"✅ Inserted {len(data)} documents into MongoDB with 'date' field.")

    # Add indexes for better performance
    collection.create_index("date")
    collection.create_index("merchant_id")
    collection.create_index([("date", 1), ("merchant_id", 1)])
    print("📌 Indexes on 'date', 'merchant_id', and ('date', 'merchant_id') created."
else:
    print("❌ No data found to insert.")
```

What happened in mongoDB:



✅ 1. Check that documents have the `date` field

In Compass:

- Go to your `darooghe` database → `transactions` collection
- In the **filter bar**, paste this:

```
json
CopyEdit
{ "date": { "$exists": true } }
```

- Press **Enter** or click **"Find"**

This will show only the documents that **do** have a `date` field.

✅ 2. Query by a specific `date` and/or `merchant_id`

Example: Find all transactions from April 20th, 2025

```
json
CopyEdit
{ "date": "2025-04-20" }
```

Example: Find all transactions for a specific merchant

```
json
CopyEdit
{ "merchant_id": "merch_1578" }
```

Example: Combine both

```
json
CopyEdit
{
```

```
  "date": "2025-04-20",
  "merchant_id": "merch_1578"
}
```

📍 Tip: Use the **filter bar** in Compass for all of these — just paste and click **Find**.

✅ 3. Run a Summary Aggregation (e.g., total amount per merchant)

In MongoDB Compass:

Go to the **Aggregation tab** → (add a new stage and) paste this pipeline:

```
json
CopyEdit
[
  {
    "$group": {
      "_id": "$merchant_id",
      "total_amount": { "$sum": "$amount" },
      "transaction_count": { "$sum": 1 }
    }
  },
  { "$sort": { "total_amount": -1 } }
]
```

This gives you:

- Total transaction amount per merchant
- Number of transactions
- Sorted by top spending

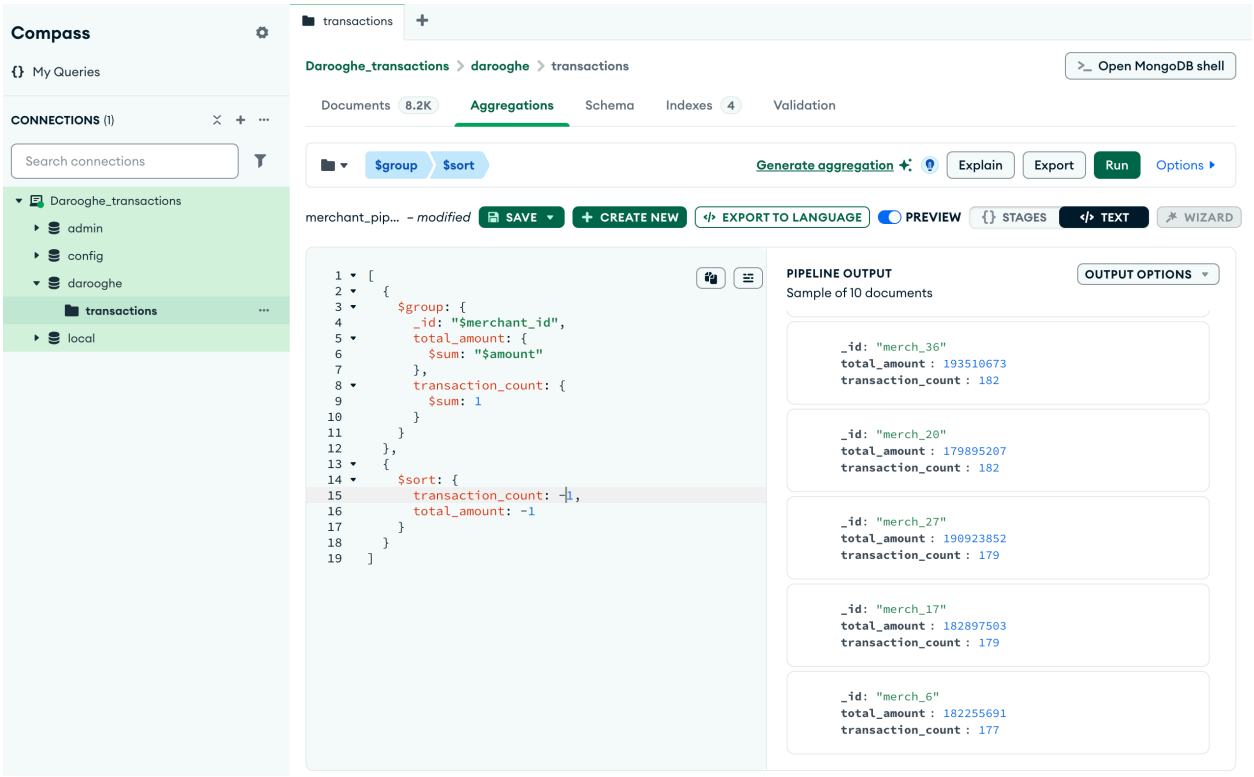
✅ 4. Optional: Show top merchants on a specific date

```
json
CopyEdit
[
  { "$match": { "date": "2025-04-20" } },
  {
    "$group": {
      "_id": "$merchant_id",
      "total_amount": { "$sum": "$amount" }
    }
  },
  { "$sort": { "total_amount": -1 } }
]
```

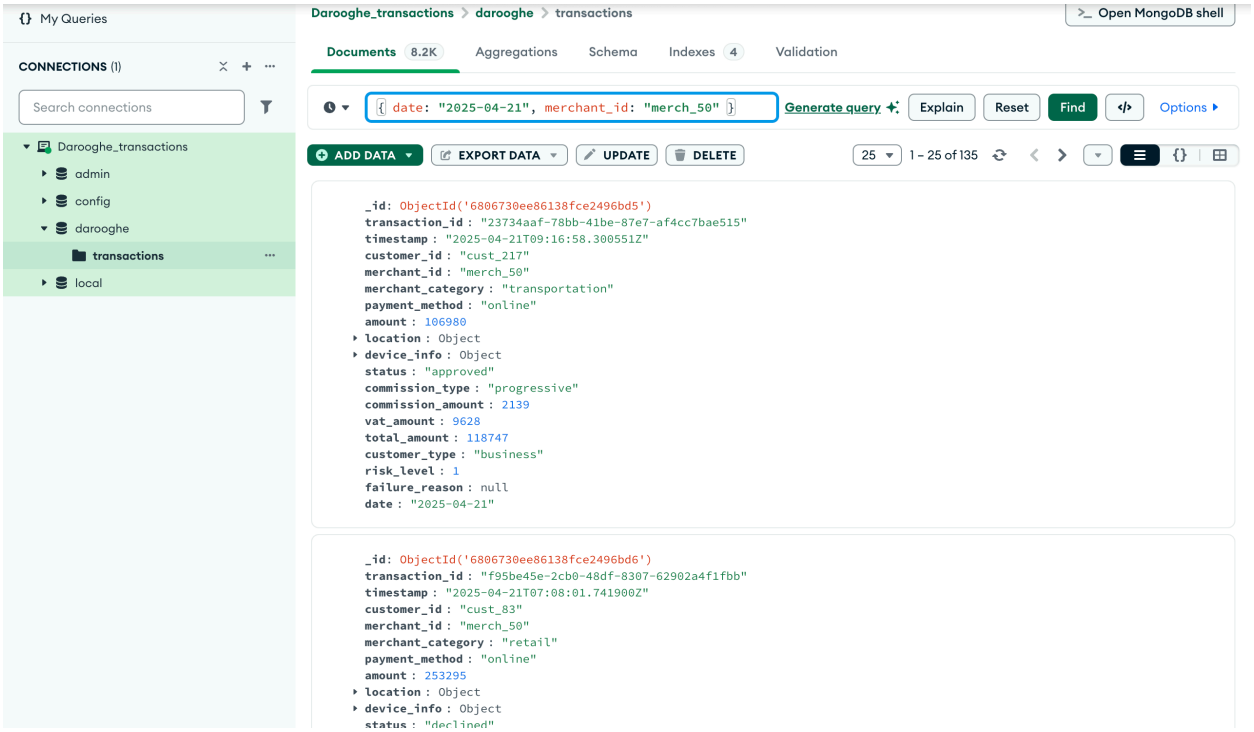
🧠 Extra Query: Find All Transactions With High Risk

```
json
CopyEdit
{ "risk_level": { "$gte": 4 } }
```

generate pipeline



generate query



Daily summery pipeline:

```
[
{
$group: {
_id: {
merchant_id: "$merchant_id",
date: "$date"
},
total_amount: { $sum: "$amount" },
txn_count: { $sum: 1 }
}
},
{ $sort: { "_id.date": 1 } }
]
```

Commission report pipeline:

```
[
 {
  $group: {
   _id: {
    merchant_category: "$merchant_category",
    date: "$date"
   },
   total_commission: {
    $sum: "$commission_amount"
   },
   avg_commission: {
    $avg: "$commission_amount"
   }
  }
 },
```

```
  { $sort: { "_id.date": 1 } }
]
```

Monthly Summery:

```
[
  {
    $group: {
      _id: {
        merchant_id: "$merchant_id",
        month: { $substr: ["$date", 0, 7] } // gives "YYYY-MM"
      },
      total_amount: { $sum: "$amount" },
      txn_count: { $sum: 1 }
    }
  }
]
```

Costomer_pipeline

```
[
    {
        "$group": {
            "_id": "$customer_id",
            "transaction_count": { "$sum": 1 },
            "total_amount": { "$sum": "$amount" }
        }
    },
    {
        "$addFields": {
            "segment": {
                "$switch": {
```

```
            "branches": [
              { "case": { "$gte": ["$transaction_count", 15] }, "then": "High" },
              { "case": { "$gte": ["$transaction_count", 10] }, "then": "Medium" }
            ],
            "default": "Low"
          }
        }
      }
    },
  { "$sort": { "transaction_count": -1 } } // Sort the results by date (ascending)

]
```

We need to change the version of java to 17 or 11 in order to make it compatible with spark

java version change:

```
(venv) (base) alireza@alirezas-MacBook-Air DS-CA2 % java -version
openjdk version "23.0.2" 2025-01-21
OpenJDK Runtime Environment Homebrew (build 23.0.2)
OpenJDK 64-Bit Server VM Homebrew (build 23.0.2, mixed mode, sharing)
(venv) (base) alireza@alirezas-MacBook-Air DS-CA2 % /usr/libexec/java_home -

Matching Java Virtual Machines (4):
    23.0.2 (arm64) "Homebrew" - "OpenJDK 23.0.2" /opt/homebrew/Cellar/openj
    17.0.14 (arm64) "Homebrew" - "OpenJDK 17.0.14" /opt/homebrew/Cellar/openj
    11.0.23 (arm64) "Amazon.com Inc." - "Amazon Corretto 11" /Library/Java/JavaV
    1.8.441.07 (arm64) "Oracle Corporation" - "Java" /Library/Internet Plug-Ins/Jav
/opt/homebrew/Cellar/openjdk/23.0.2/libexec/openjdk.jdk/Contents/Home

(venv) (base) alireza@alirezas-MacBook-Air DS-CA2 % export JAVA_HOME="/Li
export PATH="$JAVA_HOME/bin:$PATH"
```

```
(venv) (base) alireza@alirezas-MacBook-Air DS-CA2 % java -version
openjdk version "11.0.23" 2024-04-16 LTS
OpenJDK Runtime Environment Corretto-11.0.23.9.1 (build 11.0.23+9-LTS)
OpenJDK 64-Bit Server VM Corretto-11.0.23.9.1 (build 11.0.23+9-LTS, mixed mod
```

Always change the java version to 11 when running pyspark. and you don’t need any anacondo diactivated.

For changing to java 17, you can do this

```
(venv) (base) alireza@alirezas-MacBook-Air DS-CA2 % export JAVA_HOME="/o
export PATH="$JAVA_HOME/bin:$PATH"
```

Forcing to change java version in all terminals

As you can see, I added those lines in the .zhsrc file at the end(comment java 11 and now I am using version 17 and if comment both version 11 and 17, it will be java version 23

```
ZSH_THEME="simple"
# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
__conda_setup="$('/Users/alireza/anaconda3/bin/conda' 'shell.zsh' 'hook' 2> /dev/null)"
if [ $? -eq 0 ]; then
    eval "$__conda_setup"
else
    if [ -f "/Users/alireza/anaconda3/etc/profile.d/conda.sh" ]; then
        . "/Users/alireza/anaconda3/etc/profile.d/conda.sh"
    else
        export PATH="/Users/alireza/anaconda3/bin:$PATH"
    fi
fi
unset __conda_setup
# <<< conda initialize <<<

export PATH="/opt/homebrew/Cellar/antlr/4.13.2/bin:$PATH"

___MY_VMOPTIONS_SHELL_FILE="${HOME}/.jetbrains.vmoptions.sh"; if [ -f "${___MY_VMOPTIONS_SHELL_FILE}" ]; then . "${___MY_VMOPTIONS_SHELL_FILE}"; fi
export PATH="/opt/homebrew/opt/openjdk@17/bin:$PATH"
export ARTEMIS_HOME=/opt/artemis
export PATH="$ARTEMIS_HOME/bin:$PATH"
export JAVA_HOME=$(/usr/libexec/java_home -v 23)
export PATH="$JAVA_HOME/bin:$PATH"


# export JAVA_HOME="/Library/Java/JavaVirtualMachines/amazon-corretto-11.jdk/Contents/Home"
# export PATH="$JAVA_HOME/bin:$PATH"

export JAVA_HOME="/opt/homebrew/Cellar/openjdk@17/17.0.14/libexec/openjdk.jdk/Contents/Home"
export PATH="$JAVA_HOME/bin:$PATH"
```

force to java 11:

To ensure that Java 11 is the default version instead of Java 23, follow these steps:

1. Find the Path to Java 11

From your list, you have Java 11 installed at:

2. Update `.zshrc` to Set Java 11 as Default

You need to modify your shell configuration file (`~/.zshrc`) to point to Java 11 as the default version. Open the file in a text editor:

```
bash
CopyEdit
nano ~/.zshrc
```

Then, add the following lines to the file, which will set Java 11 as the default when you open a new terminal:

```
bash
CopyEdit
export JAVA_HOME="/Library/Java/JavaVirtualMachines/amazon-corretto-11.jdk/Contents/Home"
export PATH="$JAVA_HOME/bin:$PATH"
```

3. Apply Changes to `.zshrc`

After saving the changes, apply them by either restarting your terminal or using:

```
bash
CopyEdit
source ~/.zshrc
```

4. Verify the Java Version

Once you've updated and sourced your `.zshrc`, verify that Java 11 is now the default by running:

```
bash
CopyEdit
java -version
```

This should show Java 11 as the active version.

Optional: Check the Java Versions Installed

If you'd like to see all installed Java versions and ensure that the `java` command points to Java 11, you can run:

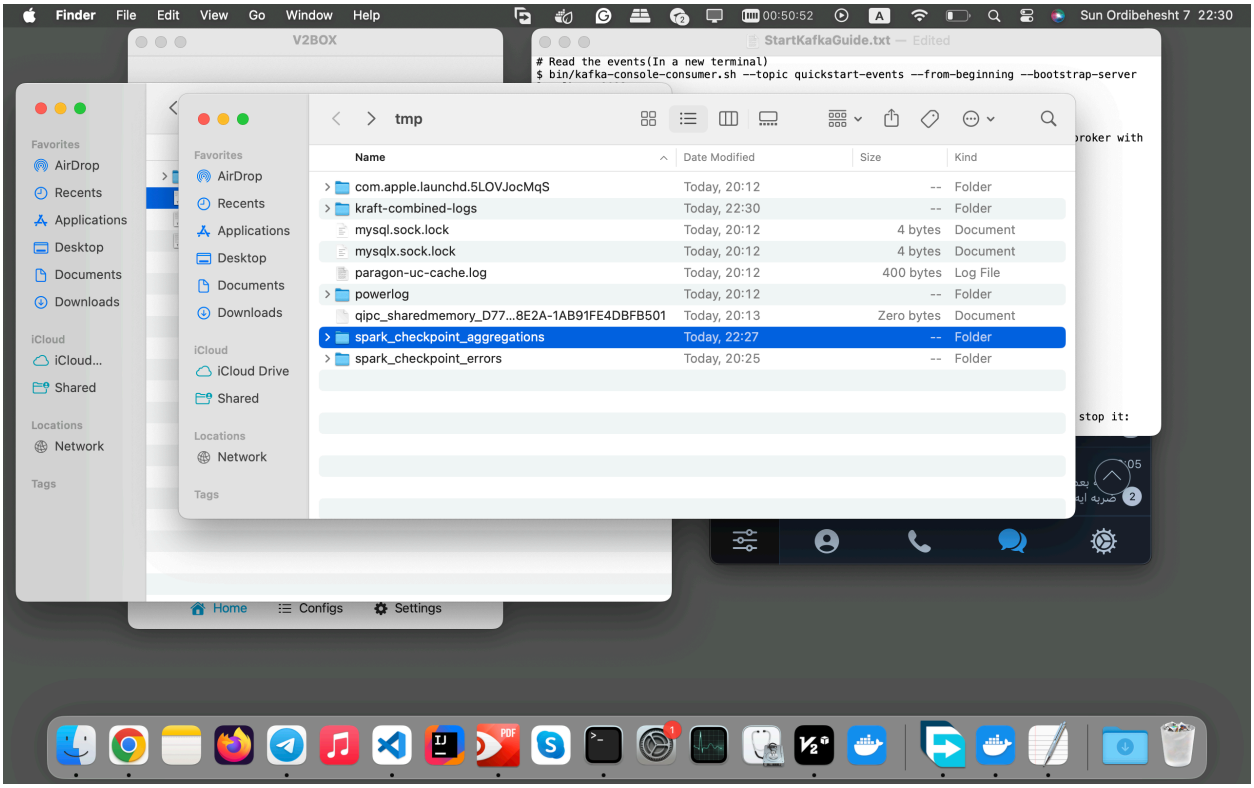
```
bash
CopyEdit
```

```
/usr/libexec/java_home -V
```

This will list all Java versions, and you can confirm that the `JAVA_HOME` path is correct.

Let me know if you encounter any issues!

Remove the /tmp/spark_checkpoints files created at real time processing part

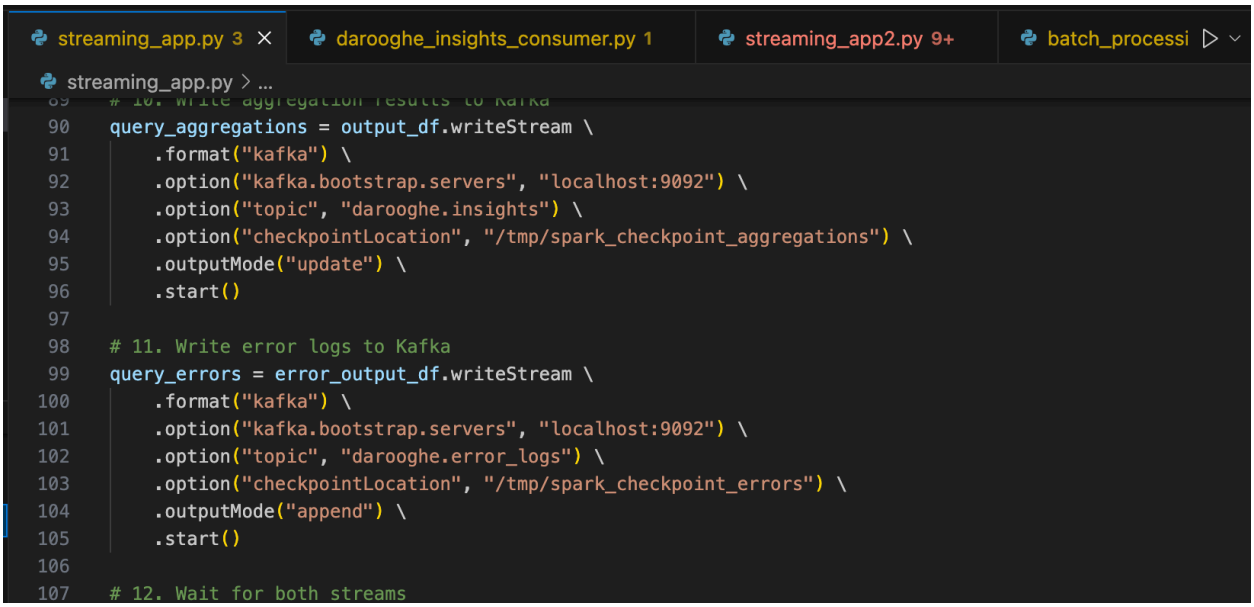


Darooghe.insights for the output of the first part in real time processing layer

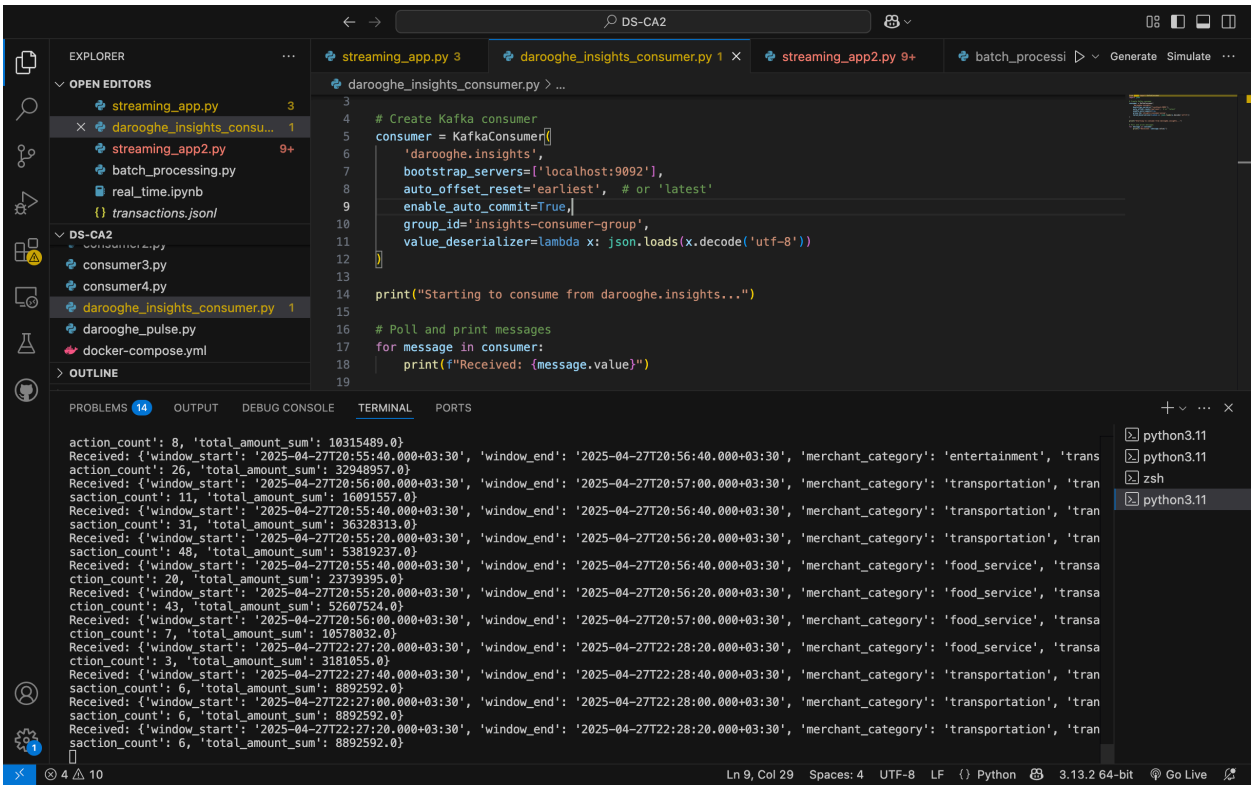
> Spark Streaming Application

> I. Implement a Spark Streaming application that connects to Kafka Consumer.
>
> II. Process data in small time intervals (micro-batches), using time-based windows to aggregate and analyze data over specific periods (e.g., 1 minute), with the window sliding at regular intervals (e.g., every 20 seconds), to ensure timely and efficient data processing in real time. Use the processing to gain some informative insights. Write detected insights to an (or multiple) arbitrary topic (topics).
>
> III. Implement a checkpoint mechanism for fault tolerance.

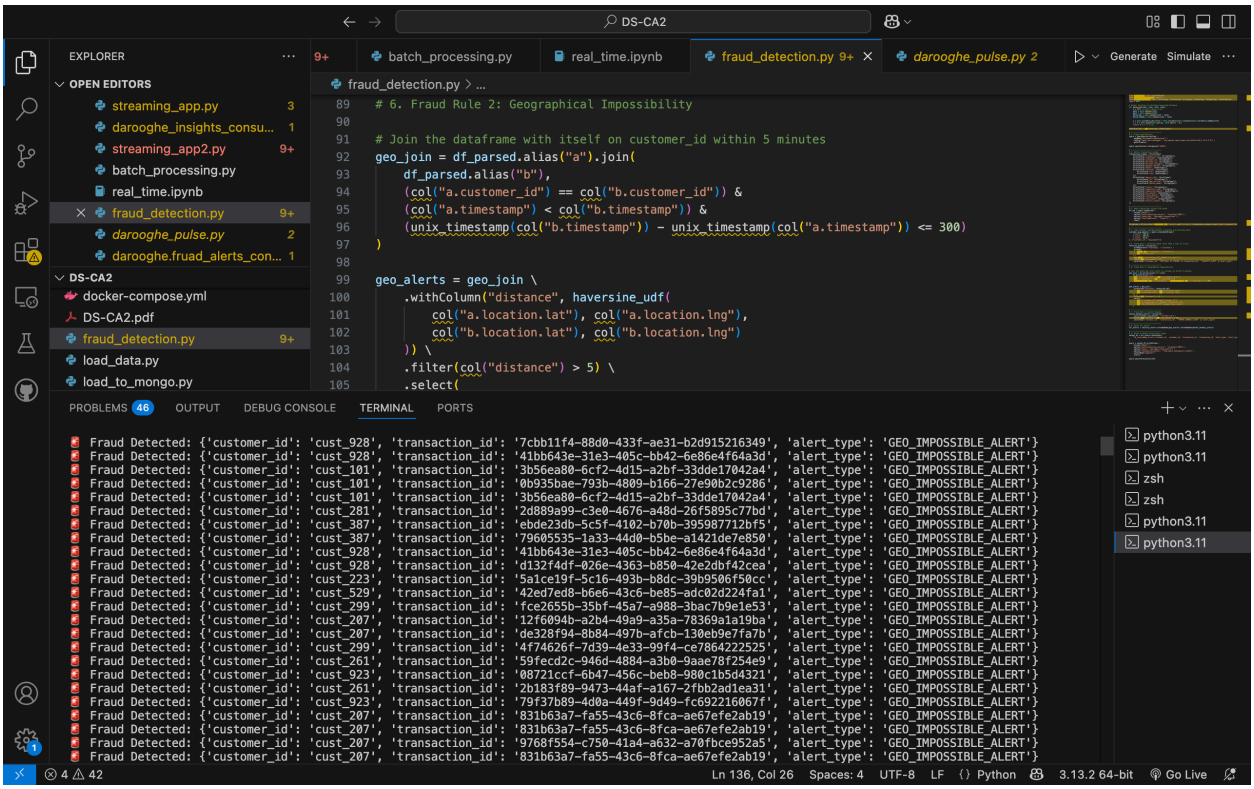
if you run stream_app.py, it will create output into transactions.insight and



And this is the result of its consumer:



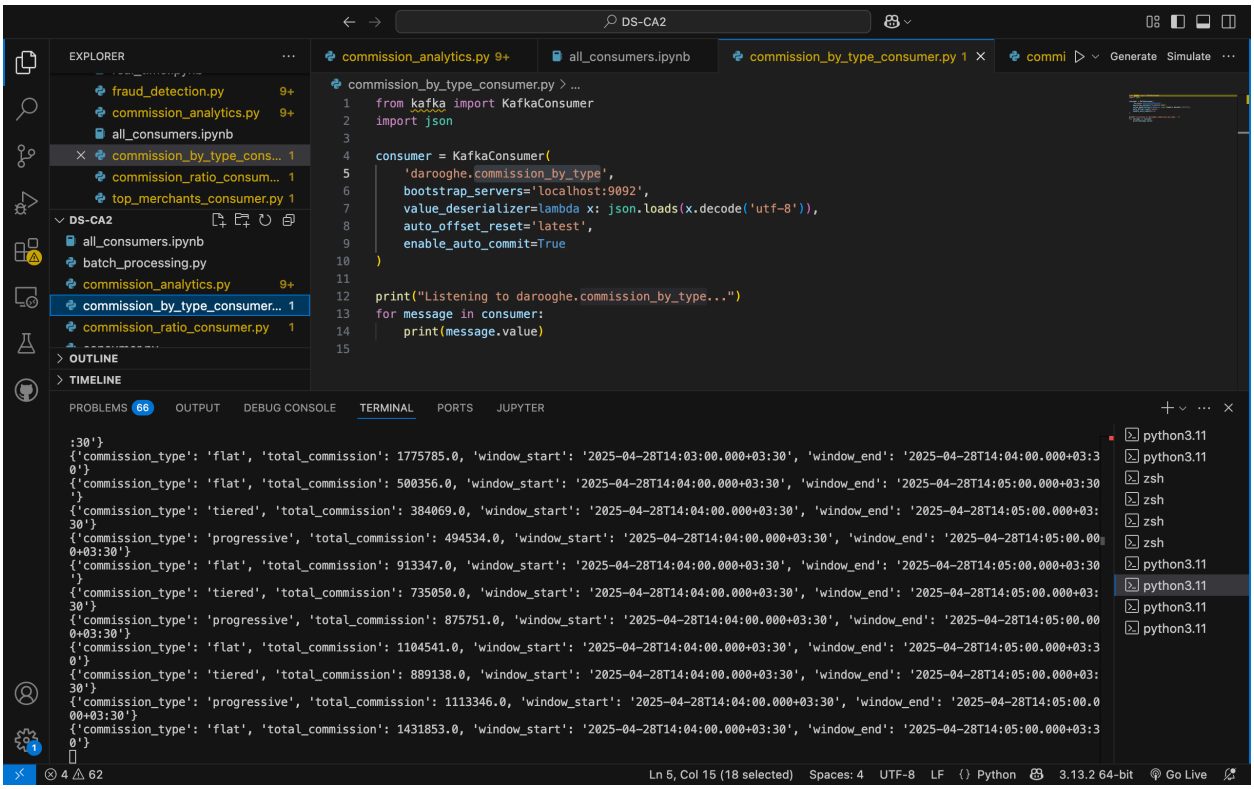
for fraud_Detection part, the “fraud_detection.py create the output into topic “darooghe.fraud_alerts” and the consumer file is darooghe_fraud_detection_consumer.py

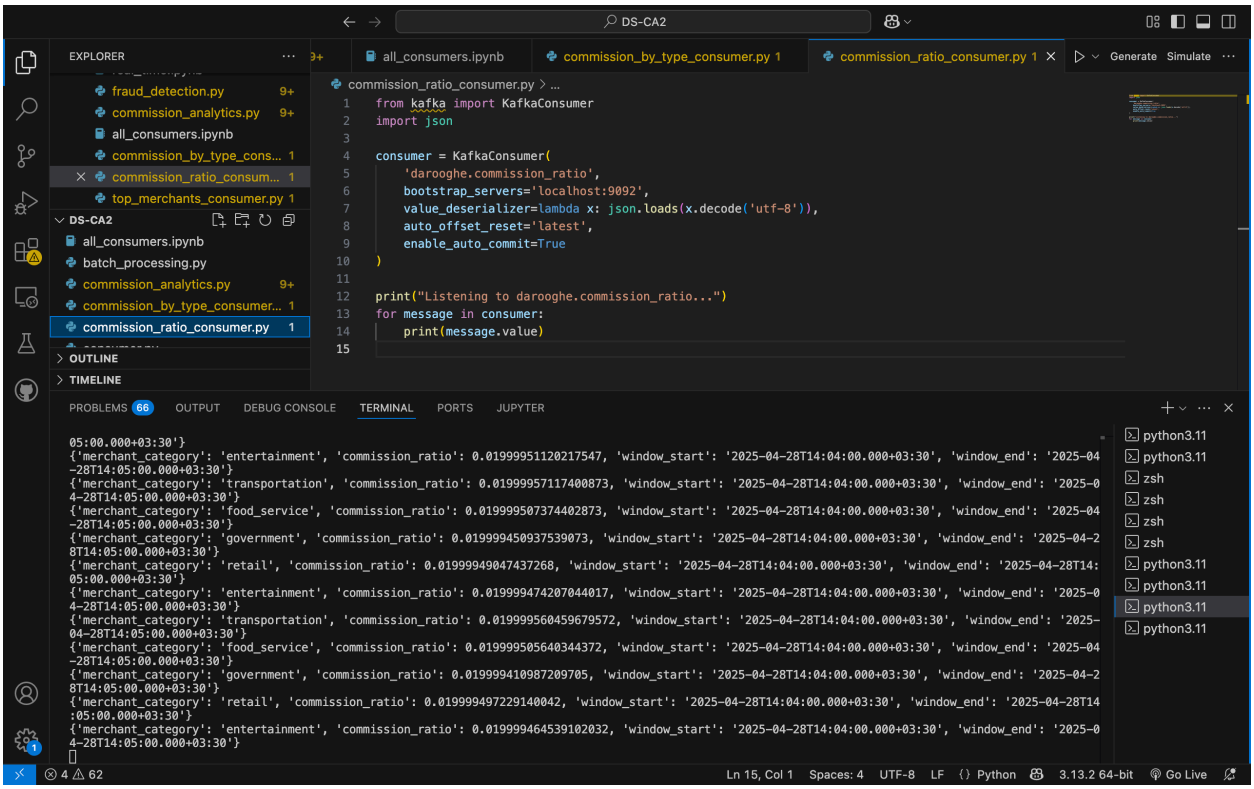


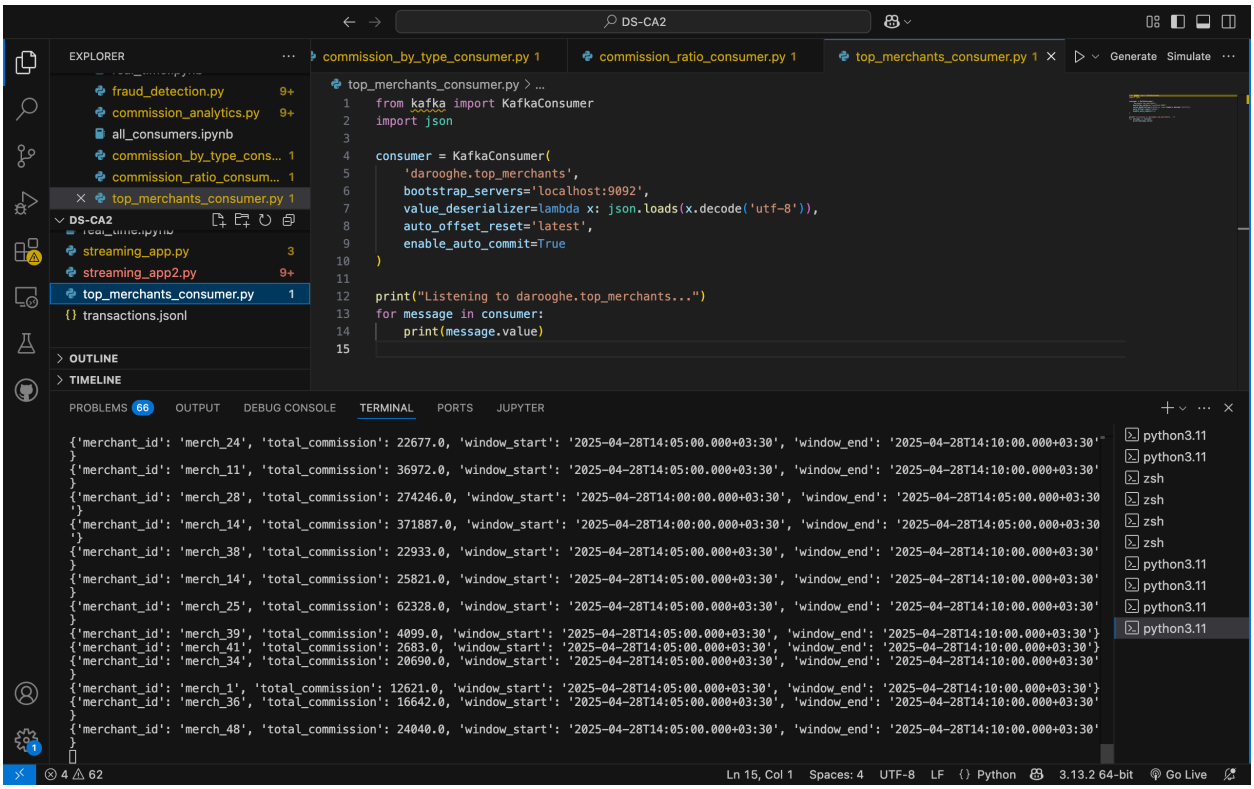
In the commission_analytics.py, I created other three topics said in the :

> Real-Time Commission Analytics
>
> I. Calculate and write to an (or multiple) arbitrary topic (topics), real-time metrics for commissions:
>
> A. Total commission by type per minute.
>
> B. Commission ratio (commission/transaction amount) by merchant category.
>
> C. Highest commission-generating merchants in 5-minute windows.

for each i created a consumer:


DS-CA2
EXPLORER
commission_analytics.py 9+
all_consumers.ipynb
commission_by_type_consumer.py 1
Generate Simulate
fraud_detection.py 9+
commission_analytics.py 9+
all_consumers.ipynb
commission_by_type_cons... 1
commission_ratio_consum... 1
top_merchants_consumer.py 1
DS-CA2
all_consumers.ipynb
batch_processing.py
commission_analytics.py 9+
commission_by_type_consumer... 1
commission_ratio_consumer.py 1
OUTLINE
TIMELINE
commission_by_type_consumer.py > ...
from kafka import KafkaConsumer
import json
consumer = KafkaConsumer(
'darooghe.commission_by_type',
bootstrap_servers='localhost:9092',
value_deserializer=lambda x: json.loads(x.decode('utf-8')),
auto_offset_reset='latest',
enable_auto_commit=True
)
print("Listening to darooghe.commission_by_type...")
for message in consumer:
print(message.value)
PROBLEMS 66
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
JUPYTER
:30'}
{'commission_type': 'flat', 'total_commission': 1775785.0, 'window_start': '2025-04-28T14:03:00.000+03:30', 'window_end': '2025-04-28T14:04:00.000+03:30'}
{'commission_type': 'flat', 'total_commission': 500356.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'tiered', 'total_commission': 384069.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'progressive', 'total_commission': 494534.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'flat', 'total_commission': 913347.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'tiered', 'total_commission': 735050.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'progressive', 'total_commission': 875751.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'flat', 'total_commission': 1104541.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'tiered', 'total_commission': 889138.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'progressive', 'total_commission': 1113346.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'commission_type': 'flat', 'total_commission': 1431853.0, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
python3.11
python3.11
zsh
zsh
zsh
zsh
python3.11
python3.11
python3.11
python3.11
4 62
Ln 5, Col 15 (18 selected)
Spaces: 4
UTF-8
LF
Python
3.13.2 64-bit
Go Live


DS-CA2
EXPLORER
all_consumers.ipynb
commission_by_type_consumer.py 1
commission_ratio_consumer.py 1
Generate Simulate
fraud_detection.py 9+
commission_analytics.py 9+
all_consumers.ipynb
commission_by_type_cons... 1
commission_ratio_consum... 1
top_merchants_consumer.py 1
DS-CA2
all_consumers.ipynb
batch_processing.py
commission_analytics.py 9+
commission_by_type_consumer... 1
commission_ratio_consumer.py 1
OUTLINE
TIMELINE
commission_ratio_consumer.py > ...
from kafka import KafkaConsumer
import json
consumer = KafkaConsumer(
'darooghe.commission_ratio',
bootstrap_servers='localhost:9092',
value_deserializer=lambda x: json.loads(x.decode('utf-8')),
auto_offset_reset='latest',
enable_auto_commit=True
)
print("Listening to darooghe.commission_ratio...")
for message in consumer:
print(message.value)
PROBLEMS 66
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
JUPYTER
05:00.000+03:30'}
{'merchant_category': 'entertainment', 'commission_ratio': 0.01999951120217547, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'transportation', 'commission_ratio': 0.01999957117400873, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'food_service', 'commission_ratio': 0.019999507374402873, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'government', 'commission_ratio': 0.019999450937539073, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'retail', 'commission_ratio': 0.01999949047437268, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'entertainment', 'commission_ratio': 0.019999474207044017, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'transportation', 'commission_ratio': 0.019999560459679572, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'food_service', 'commission_ratio': 0.019999505640344372, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'government', 'commission_ratio': 0.019999410987209705, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'retail', 'commission_ratio': 0.019999497229140042, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
{'merchant_category': 'entertainment', 'commission_ratio': 0.019999464539102032, 'window_start': '2025-04-28T14:04:00.000+03:30', 'window_end': '2025-04-28T14:05:00.000+03:30'}
python3.11
python3.11
zsh
zsh
zsh
zsh
python3.11
python3.11
python3.11
python3.11
4 62
Ln 15, Col 1
Spaces: 4
UTF-8
LF
Python
3.13.2 64-bit
Go Live



Visulization

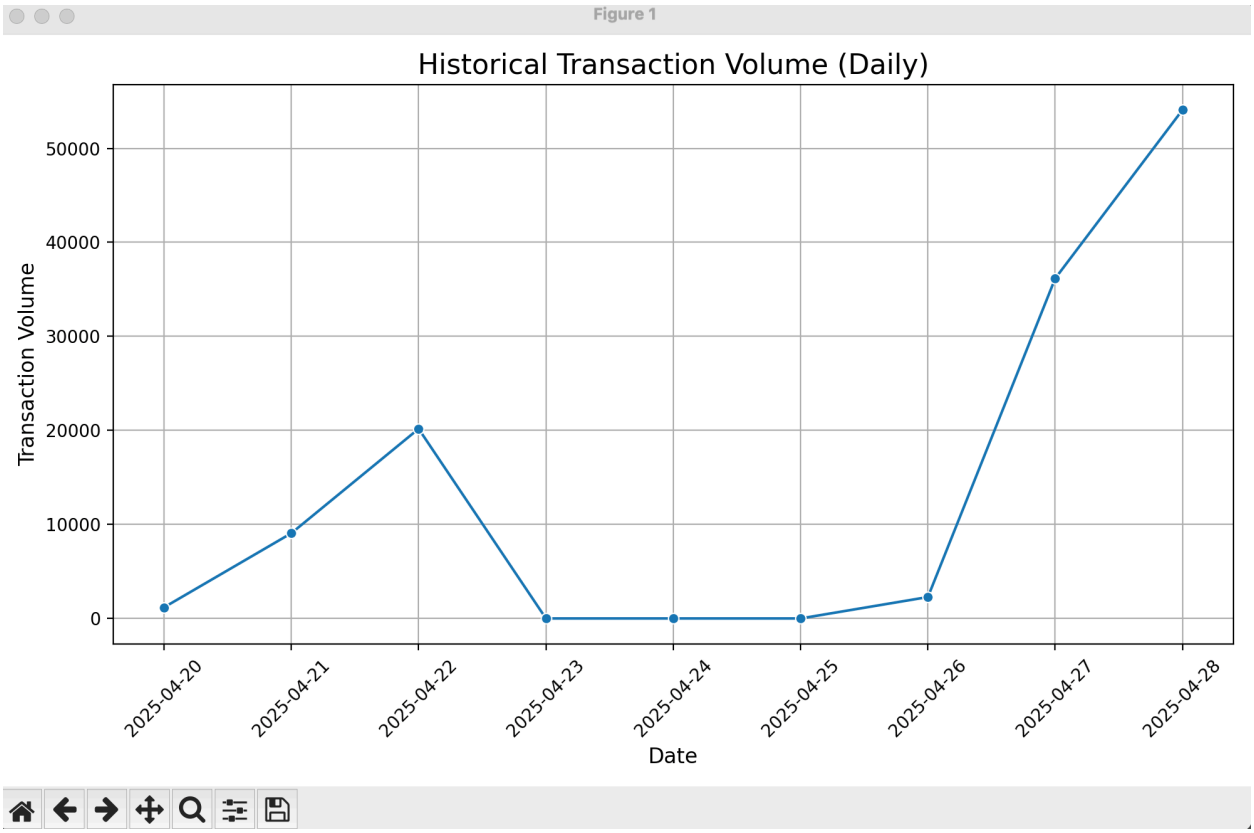


Figure 1
Historical Transaction Volume (Daily)
Transaction Volume
50000
40000
30000
20000
10000
0
2025-04-20
2025-04-21
2025-04-22
2025-04-23
2025-04-24
2025-04-25
2025-04-26
2025-04-27
2025-04-28
Date

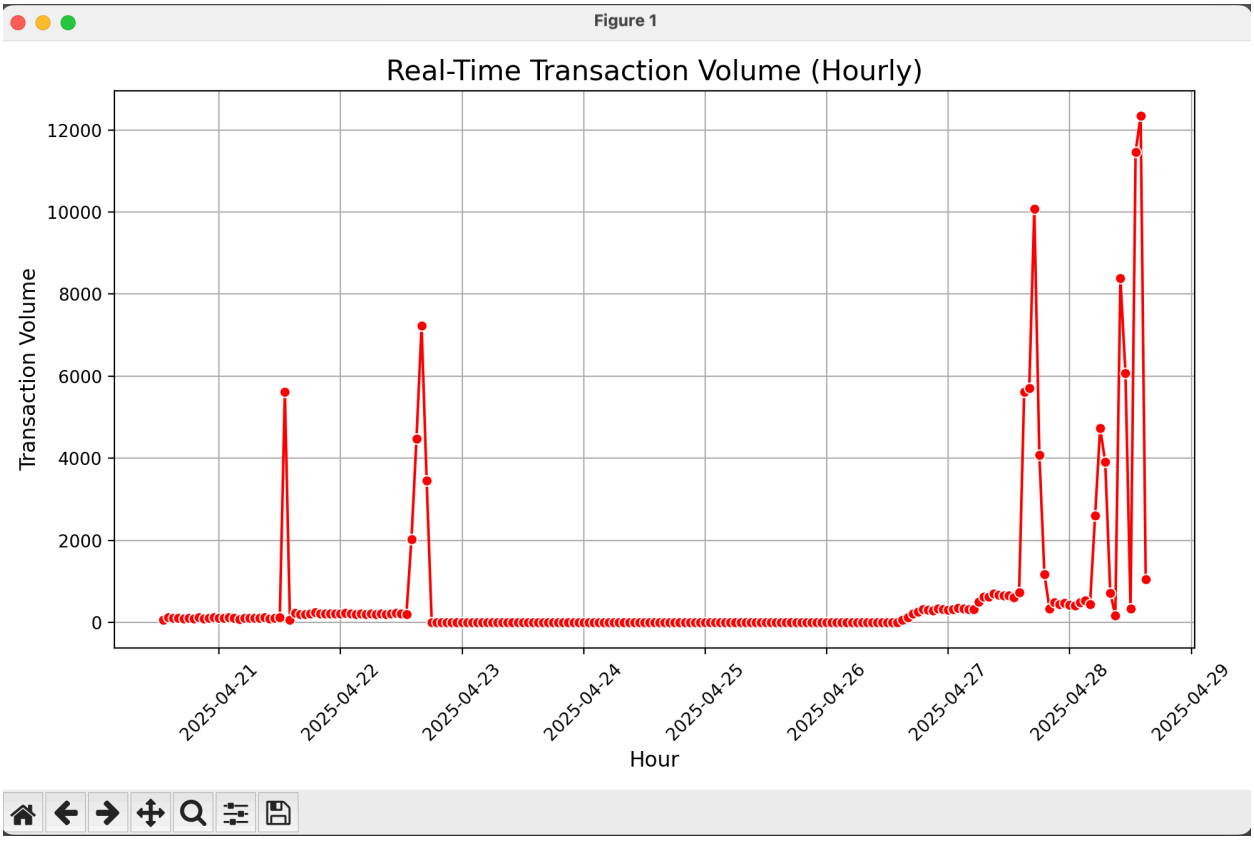


Figure 1
Real-Time Transaction Volume (Hourly)
Transaction Volume
12000
10000
8000
6000
4000
2000
0
2025-04-21
2025-04-22
2025-04-23
2025-04-24
2025-04-25
2025-04-26
2025-04-27
2025-04-28
2025-04-29
Hour

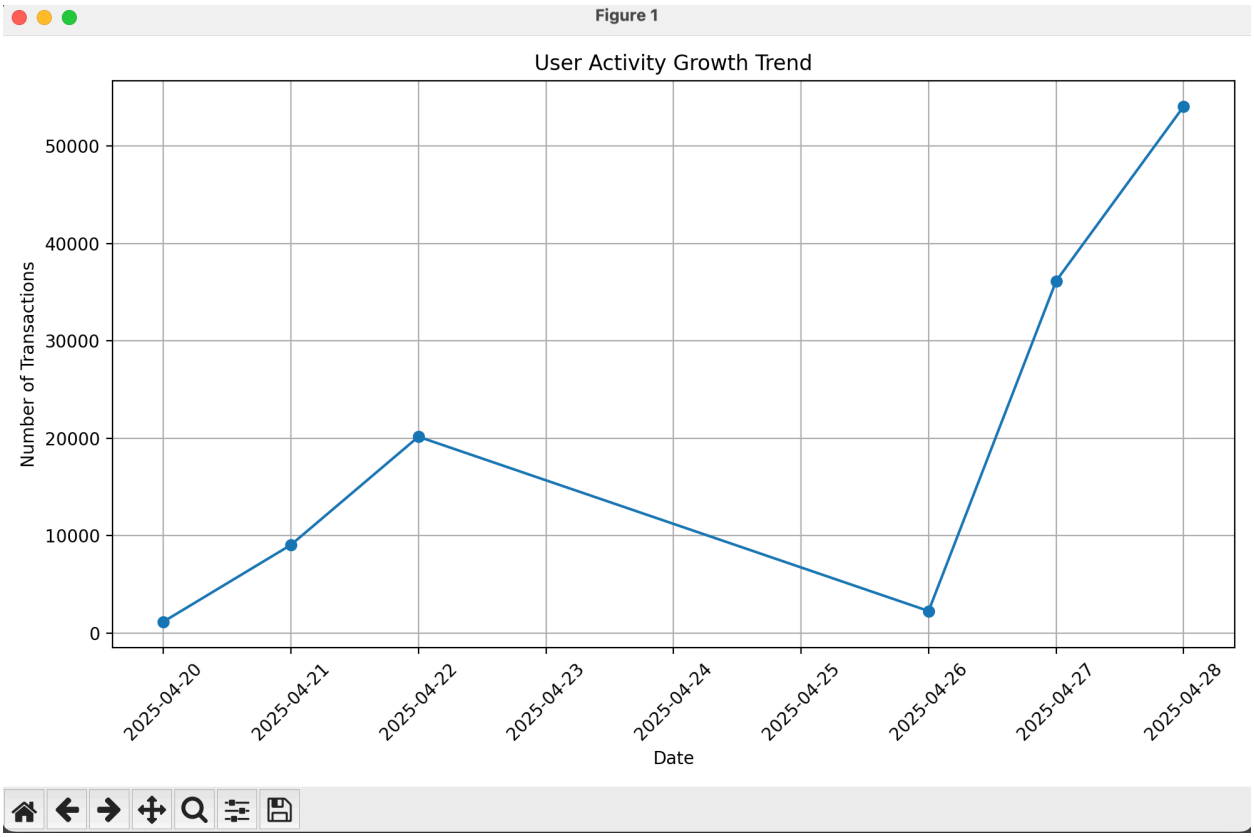


Figure 1
User Activity Growth Trend
Number of Transactions
0
10000
20000
30000
40000
50000
2025-04-20
2025-04-21
2025-04-22
2025-04-23
2025-04-24
2025-04-25
2025-04-26
2025-04-27
2025-04-28
Date

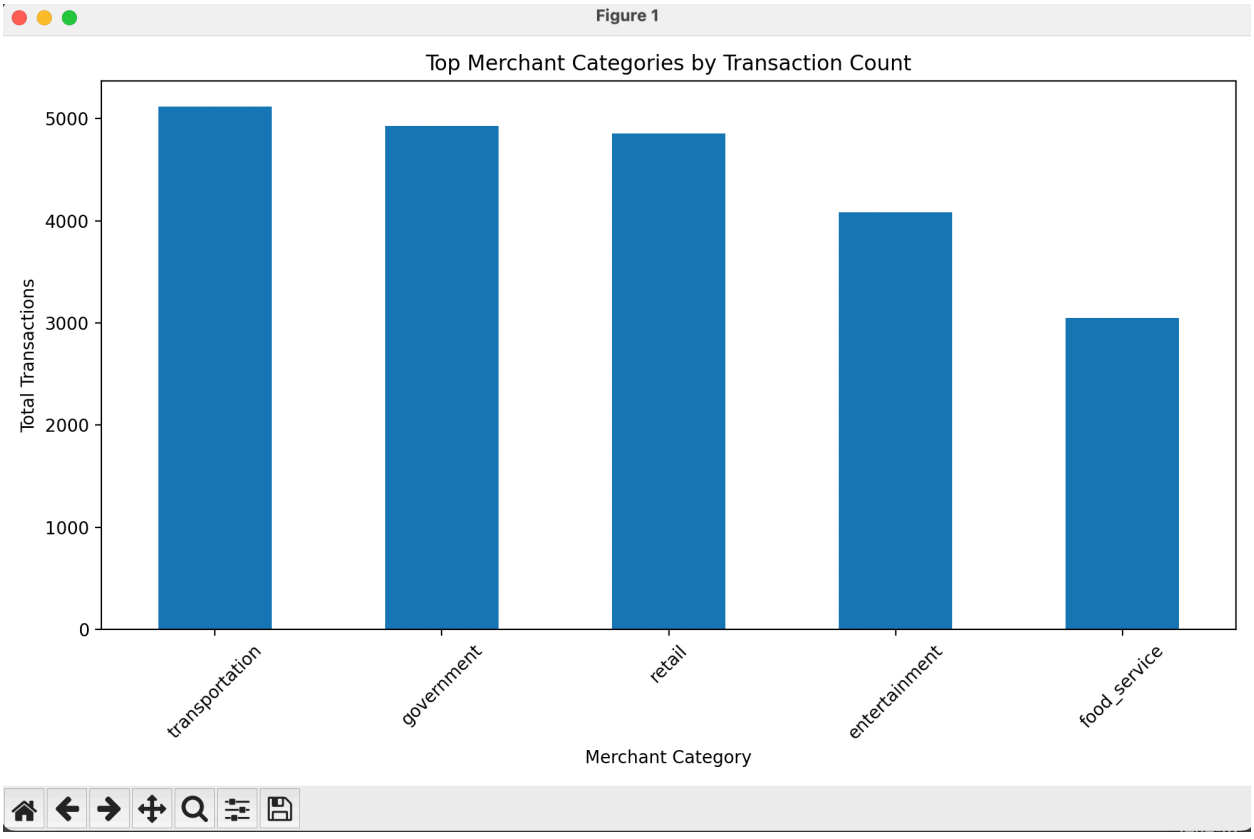


Figure 1
Top Merchant Categories by Transaction Count
Total Transactions
0
1000
2000
3000
4000
5000
transportation
government
retail
entertainment
food_service
Merchant Category

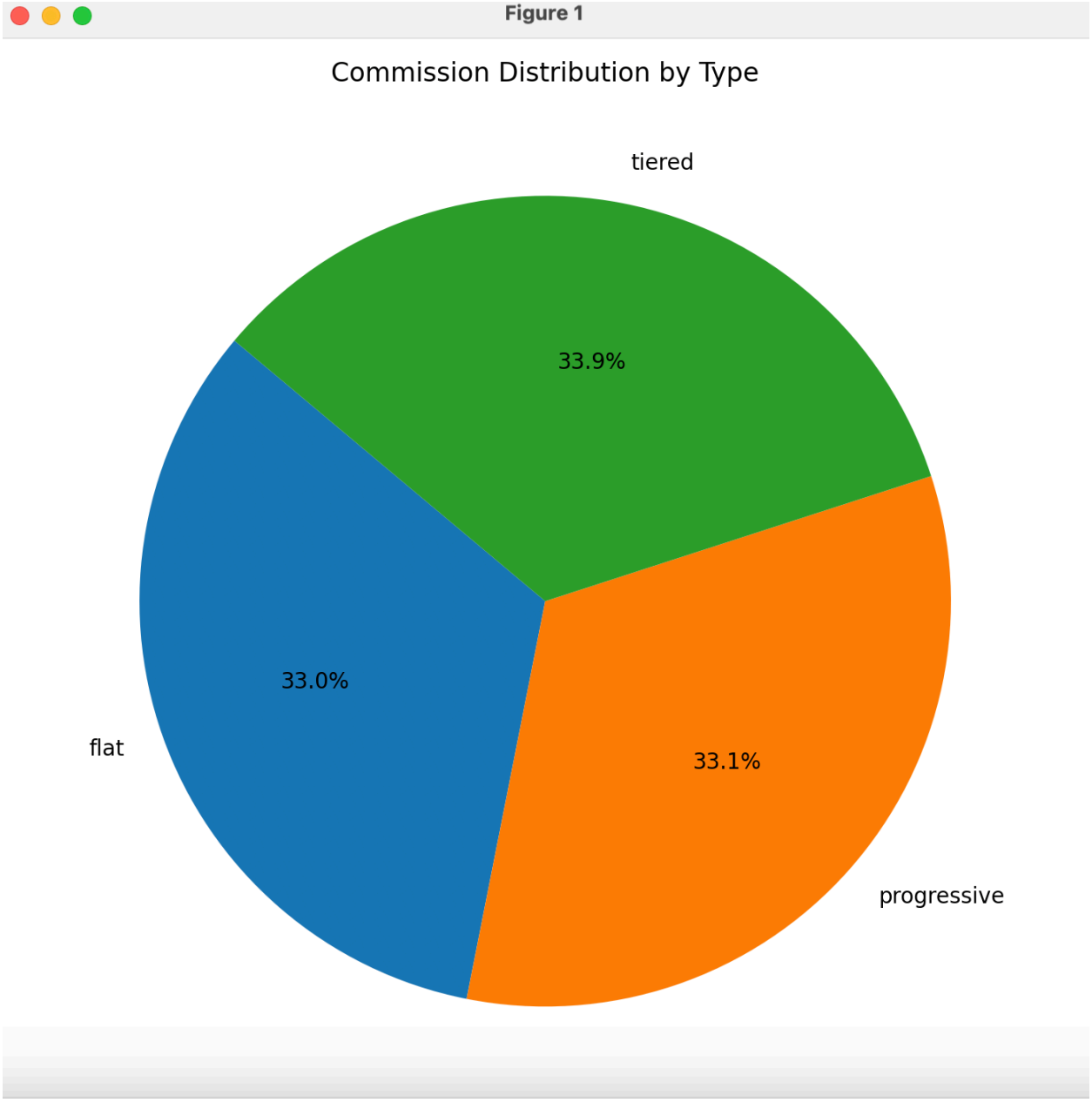


Figure 1
Commission Distribution by Type
tiered
33.9%
33.0%
flat
33.1%
progressive

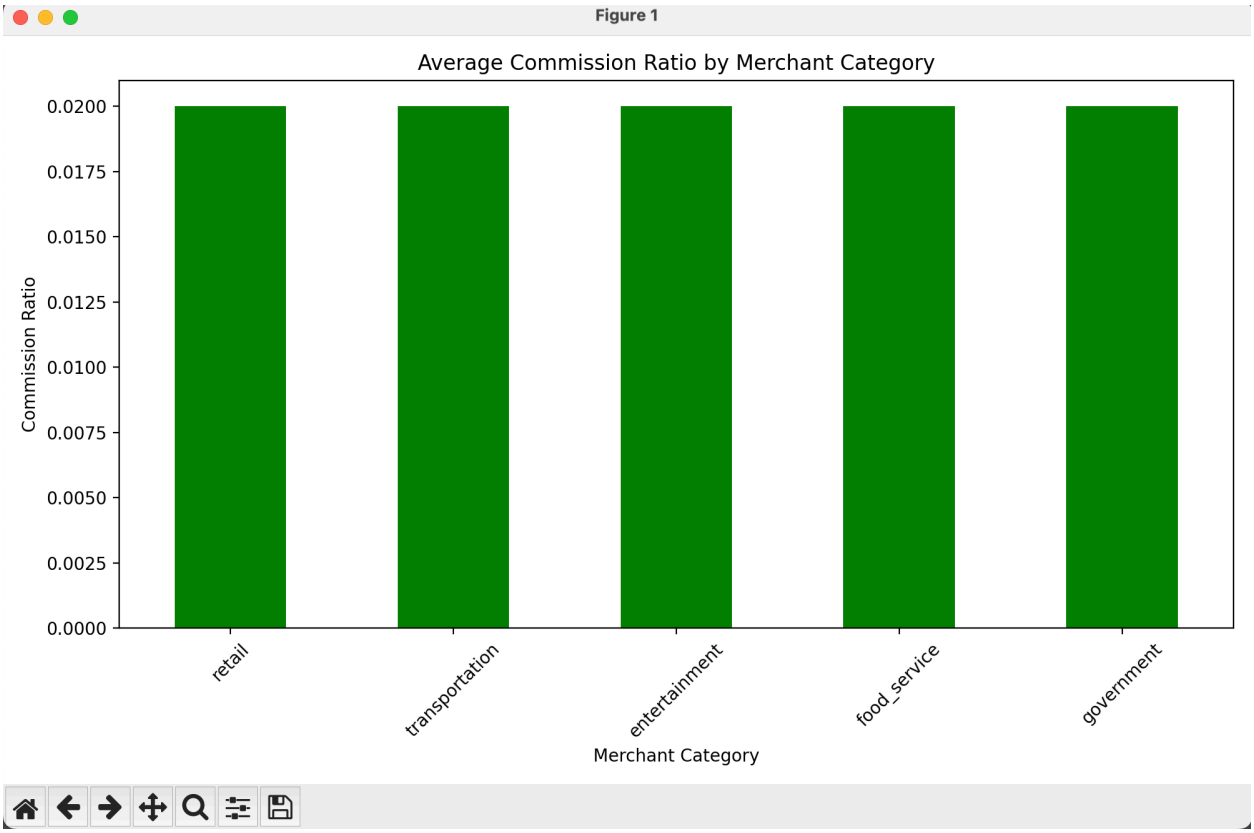


Figure 1
Average Commission Ratio by Merchant Category
Commission Ratio
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
retail
transportation
entertainment
food_service
government
Merchant Category

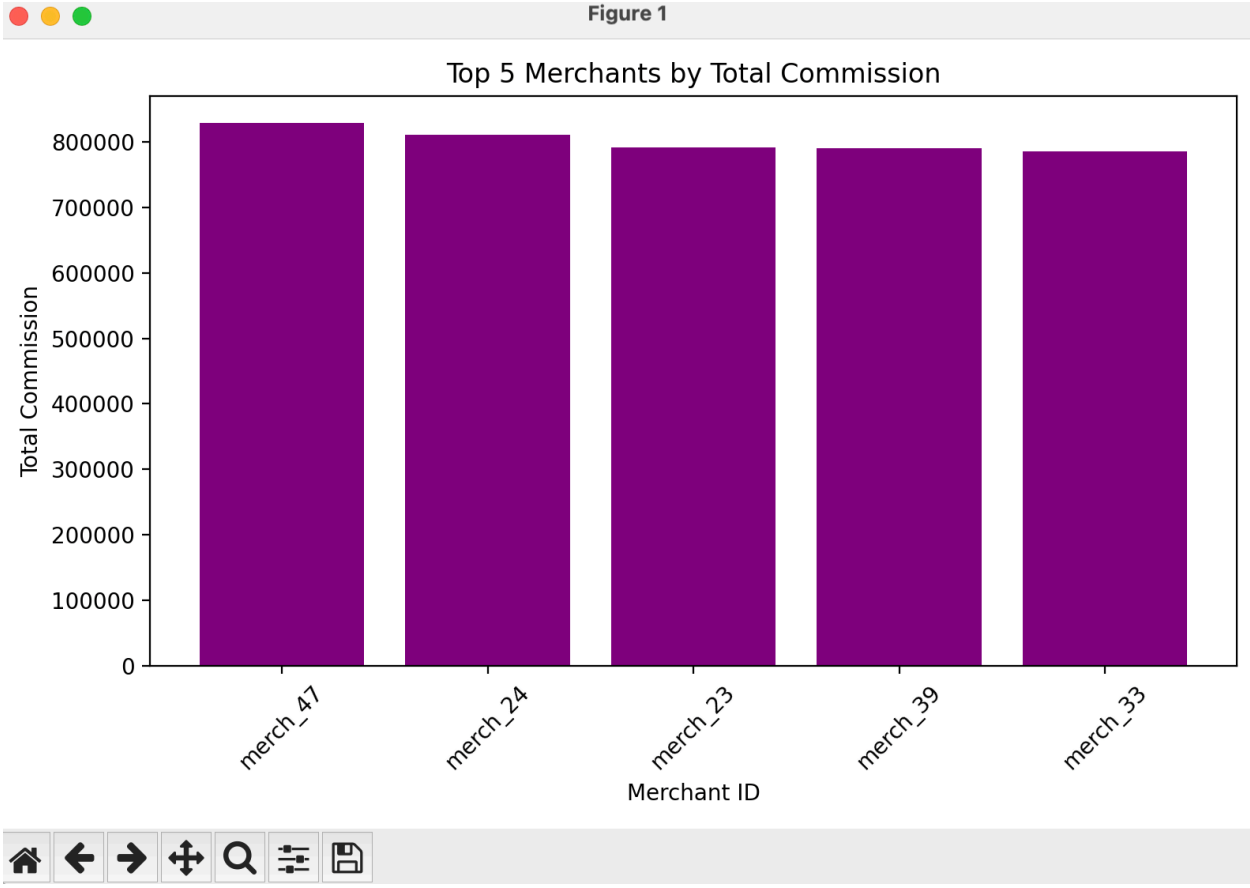


Figure 1
Top 5 Merchants by Total Commission
Total Commission
800000
700000
600000
500000
400000
300000
200000
100000
0
merch_47
merch_24
merch_23
merch_39
merch_33
Merchant ID

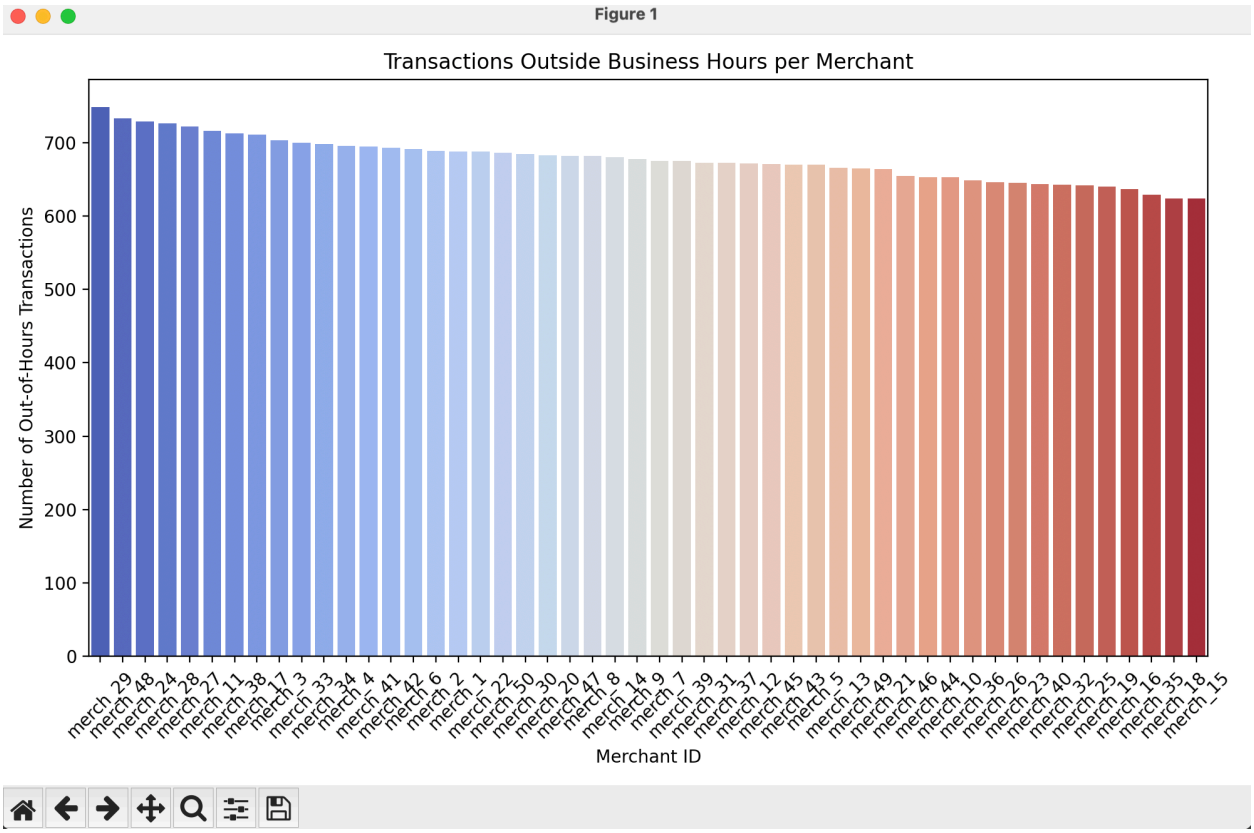


Figure 1
Transactions Outside Business Hours per Merchant
Number of Out-of-Hours Transactions
0
100
200
300
400
500
600
700
merch_29
merch_48
merch_24
merch_28
merch_27
merch_11
merch_38
merch_17
merch_3
merch_33
merch_34
merch_4
merch_41
merch_42
merch_6
merch_2
merch_1
merch_22
merch_50
merch_30
merch_20
merch_47
merch_8
merch_14
merch_9
merch_7
merch_39
merch_31
merch_37
merch_12
merch_45
merch_43
merch_5
merch_13
merch_49
merch_21
merch_46
merch_44
merch_10
merch_36
merch_26
merch_23
merch_40
merch_32
merch_25
merch_19
merch_16
merch_35
merch_18
merch_15
Merchant ID

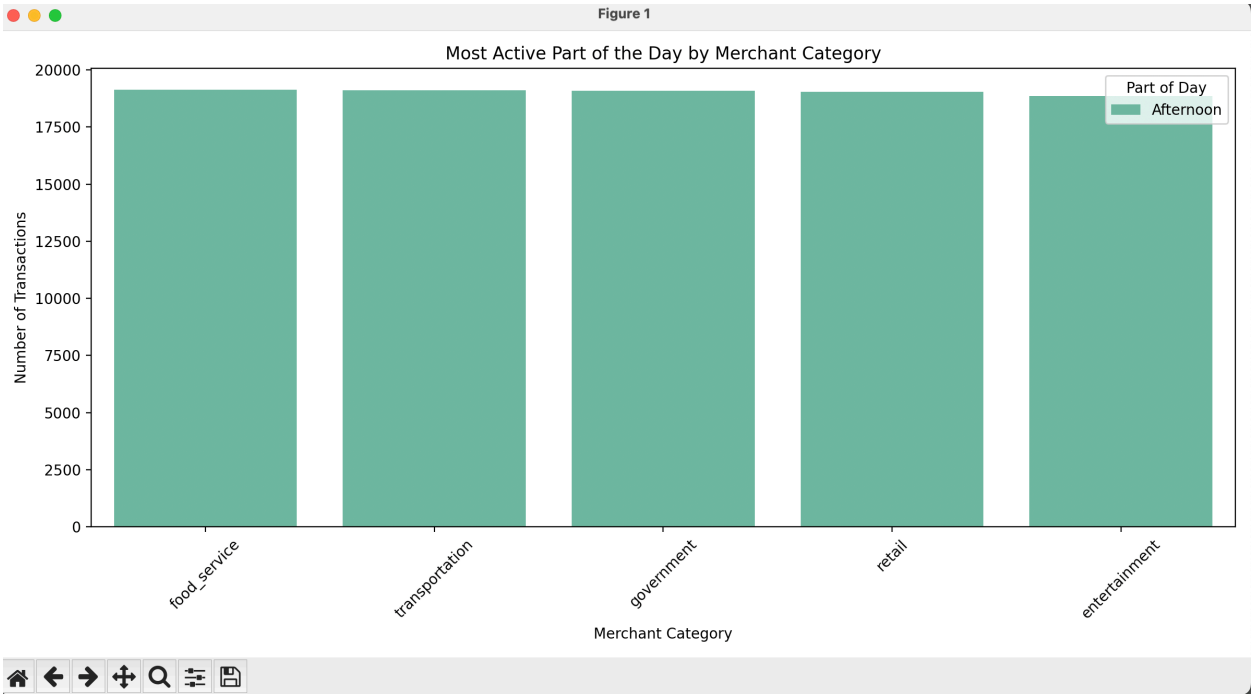
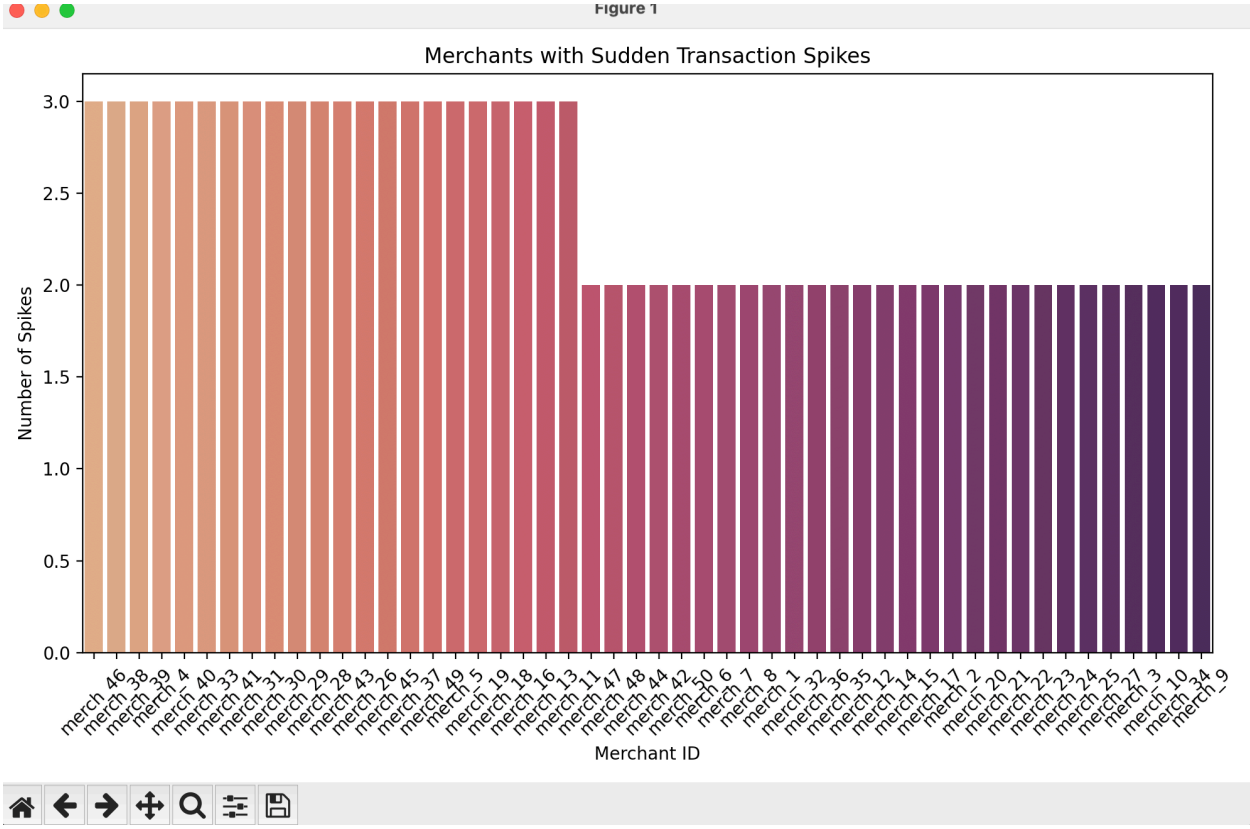


Figure 1
Most Active Part of the Day by Merchant Category
Part of Day
Afternoon
Number of Transactions
0
2500
5000
7500
10000
12500
15000
17500
20000
food_service
transportation
government
retail
entertainment
Merchant Category



Commission Audit System (5 pts)

Dynamic Pricing Simulator

I. Create PySpark UDF to recommend optimal commission type per transaction. (3 pts)

II. Validate against historical profitability data. (2 pts)

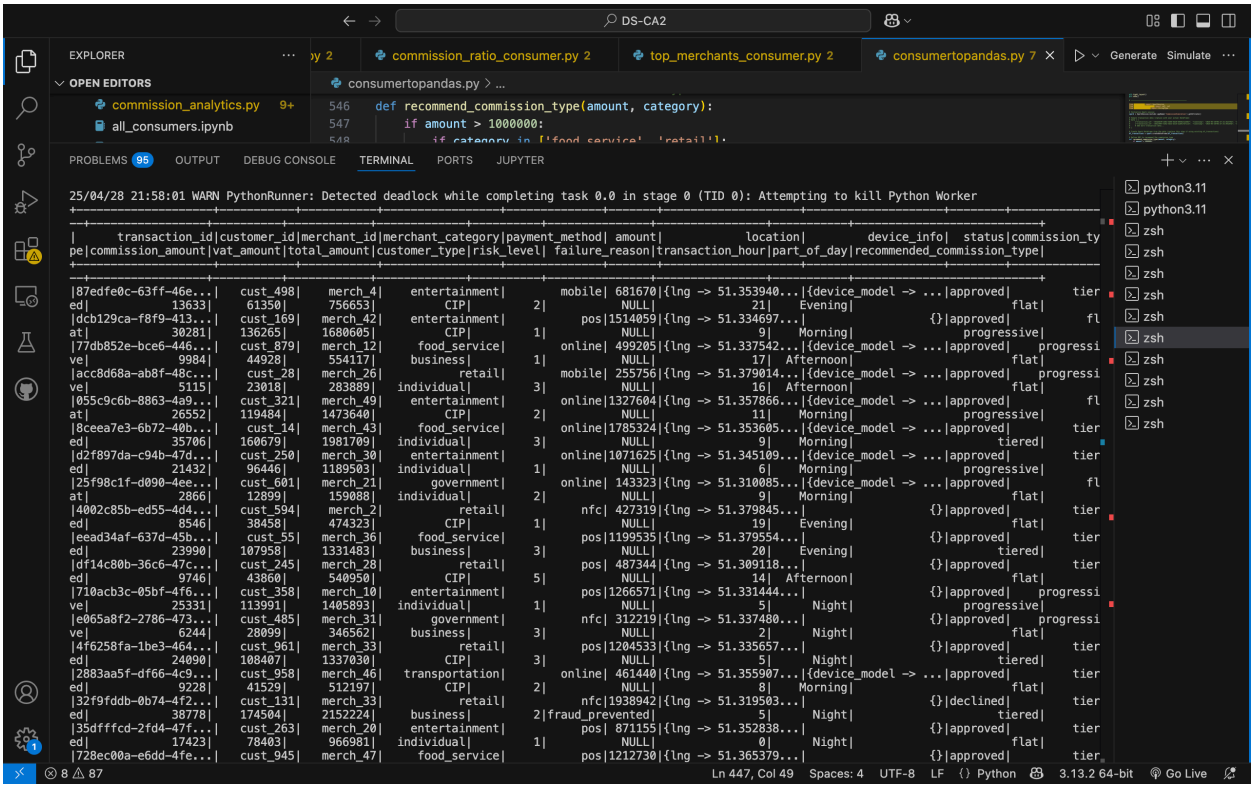
here is the code :

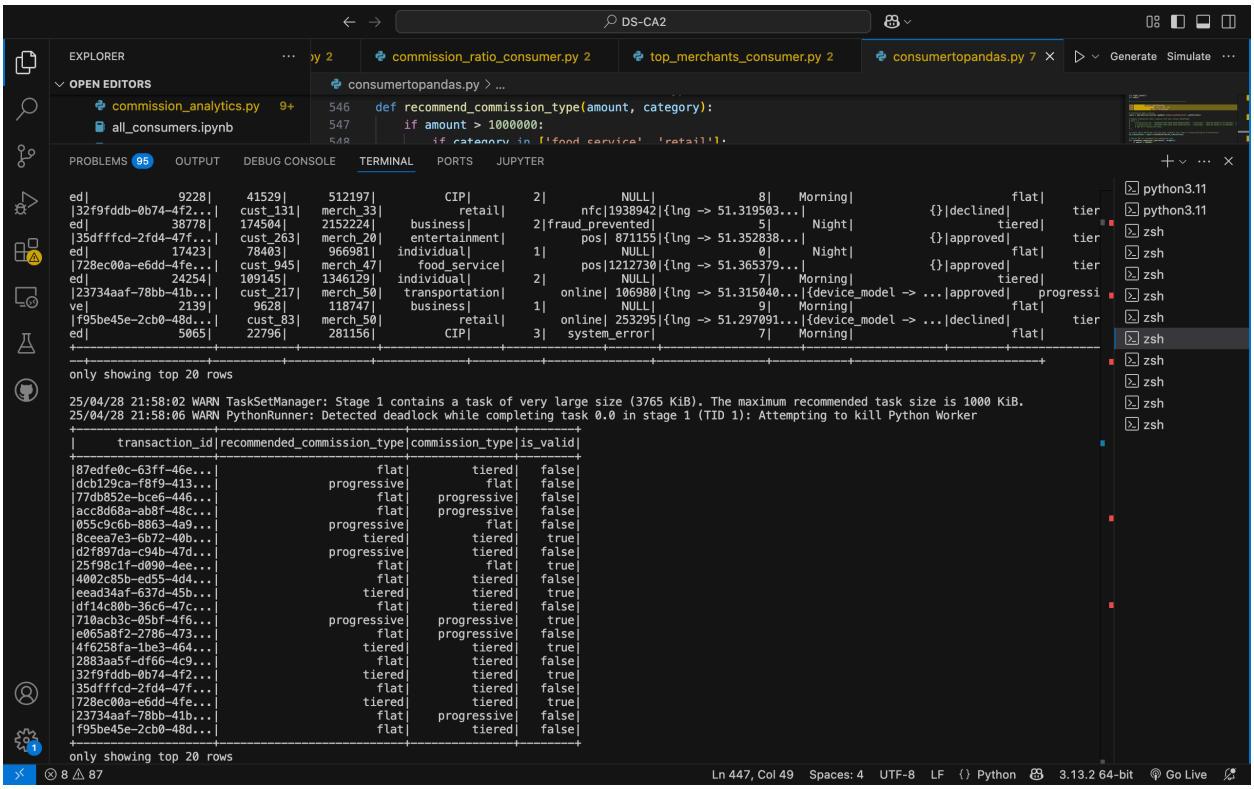
```
from pyspark.sql import SparkSession
from pyspark.sql.functions import udf, col
from pyspark.sql.types import StringType

# Initialize Spark session
```

```
spark = SparkSession.builder.appName('CommissionSimulation').getOrCreate()

# Sample transaction data (replace with your actual DataFrame)
# data = [
#     {"transaction_id": "e64a5870-2d82-4d68-8be0-6850afa10d67", "timestamp
#     {"transaction_id": "a652709f-426a-4bee-b216-ead917d71118", "timestamp":
#     # Add more transaction data...
# ]

# Create Spark DataFrame from the data (replace this step if using existing df_tra
df_transactions = spark.createDataFrame(df_transactions)

# Define UDF to recommend the commission type
def recommend_commission_type(amount, category):
    if amount > 1000000:
        if category in ['food_service', 'retail']:
            return 'tiered'
        else:
            return 'progressive'
    else:
        return 'flat'
# df_out_of_hours
# Register the UDF
recommend_commission_type_udf = udf(recommend_commission_type, StringTy

# Apply UDF to the DataFrame to recommend commission type
df_spark = df_transactions.withColumn('recommended_commission_type',
                                      recommend_commission_type_udf('amount', 'merchant_ca

# Show DataFrame with recommended commission type
df_spark.show()

# Validate the recommended commission type by comparing with actual commis
df_spark = df_spark.withColumn('is_valid',
                               (col('recommended_commission_type') == col('commission_ty
```

```
# Show validation results (checking if the recommendation matches the actual c
df_spark.select('transaction_id', 'recommended_commission_type', 'commission_

# Optionally, calculate profitability difference (recommended commission vs. act
df_spark = df_spark.withColumn('profitability_diff',
                               col('commission_amount') - col('recommended_commission_a

# Show profitability comparison
df_spark.show()
```

this is the output:





Note: always remove the /tmp/checkpoints file after creating them in the realtime part

I created different consumer for different parts of realtime

this is a function for creating consumer who waits until a specific time and gets specific number of massages. then turn them to dataframe for working in visualization part

```
def read_topic_to_dataframe(topic_name, max_messages=1000, timeout_ms=50
    """
```

```
    Reads messages from a Kafka topic into a Pandas DataFrame.

    Args:
        topic_name (str): Name of the Kafka topic
        max_messages (int): Maximum number of messages to read
        timeout_ms (int): How long to wait for messages before timing out (in millise

    Returns:
        pd.DataFrame: DataFrame containing the topic data
    """
    consumer = KafkaConsumer(
        topic_name,
        bootstrap_servers='localhost:9092',
        auto_offset_reset='earliest',
        enable_auto_commit=True,
        value_deserializer=lambda x: json.loads(x.decode('utf-8')),
        consumer_timeout_ms=timeout_ms
    )

    print(f"Consuming from topic: {topic_name}...")

    data = []
    for message in consumer:
        record = message.value
        data.append(record)

        if len(data) >= max_messages:
            break

    consumer.close()

    if data:
        df = pd.DataFrame(data)
        print(f"Read {len(df)} records from {topic_name}")
        return df
    else:
```

```
        print(f"No data found in topic {topic_name}.")
        return pd.DataFrame()  # Return empty DataFrame
```

load_data.py creates the transactions.jsonl and load_to_mongo.py is for loading data to mongoDB.

Last part: pipeline Optimization

Resource Monitoring

I. Implement Prometheus metrics for Kafka consumer lag, Spark executor CPU/MEM, and JVM garbage collection time.

📄 1. `prometheus.yml` file (put this in your project folder)

```
yaml
CopyEdit
global:
  scrape_interval: 5s  # scrape every 5 seconds

scrape_configs:
  - job_name: 'spark'
    static_configs:
      - targets: ['localhost:4040']  # Spark UI

  - job_name: 'kafka_consumer'
    static_configs:
```

```
    - targets: ['localhost:7071']  # Your custom Python consumer exporter
```

✅ This tells Prometheus to:

- Scrape Spark metrics from `localhost:4040`
- Scrape Kafka consumer lag metrics from `localhost:7071`

🐍 2. `kafka_consumer_monitor.py` (your Kafka Consumer Lag exporter)

```
python
CopyEdit
from kafka import KafkaConsumer
from prometheus_client import start_http_server, Gauge
import json
import time

# Create a Prometheus Gauge
consumer_lag_gauge = Gauge('kafka_consumer_lag', 'Kafka Consumer Lag fo
r darooghe.commission_by_type')

# Start Prometheus metrics server
start_http_server(7071)  # Exposes metrics at http://localhost:7071

# Create Kafka consumer
consumer = KafkaConsumer(
    'darooghe.commission_by_type',
    bootstrap_servers='localhost:9092',
    auto_offset_reset='earliest',
    enable_auto_commit=True,
    group_id='monitoring-group',  # Important: Set group_id to track committed
offsets
    value_deserializer=lambda x: json.loads(x.decode('utf-8'))
```

```
)

print("Kafka consumer started. Exposing lag metrics...")

# Assign topic partitions manually
consumer.poll(timeout_ms=1000)  # Poll to join group and get assignments

def get_consumer_lag(consumer):
    for tp in consumer.assignment():
        committed = consumer.committed(tp)
        end_offset = consumer.end_offsets([tp])[tp]
        if committed is not None and end_offset is not None:
            lag = end_offset - committed
            consumer_lag_gauge.set(lag)

# Infinite loop
try:
    while True:
        get_consumer_lag(consumer)
        time.sleep(5)  # Update every 5 seconds
except KeyboardInterrupt:
    consumer.close()
    print("Consumer closed.")
```

✅ This script:

- Connects to your Kafka topic
- Measures **consumer lag** live
- Exposes it at `http://localhost:7071/metrics`
- Prometheus scrapes it every 5 seconds

⚡ 3. Commands to Run Everything

Step	Command
Start Spark streaming application	(your `commission_analytics.py`)
Start Prometheus	`prometheus --config.file=prometheus.yml`
Start Kafka consumer monitor	`python kafka_consumer_monitor.py`

✅ Now visit:

- Prometheus UI: `http://localhost:9090`
- Targets: `http://localhost:9090/targets` (you should see Spark + Kafka consumer green ✅)

✅ You can **query metrics** like:

- `kafka_consumer_lag`
- `jvm_gc_collection_seconds_count` (if you configure JMX properly)

✅ You can **see CPU/MEM stats** from Spark UI itself (`http://localhost:4040`).

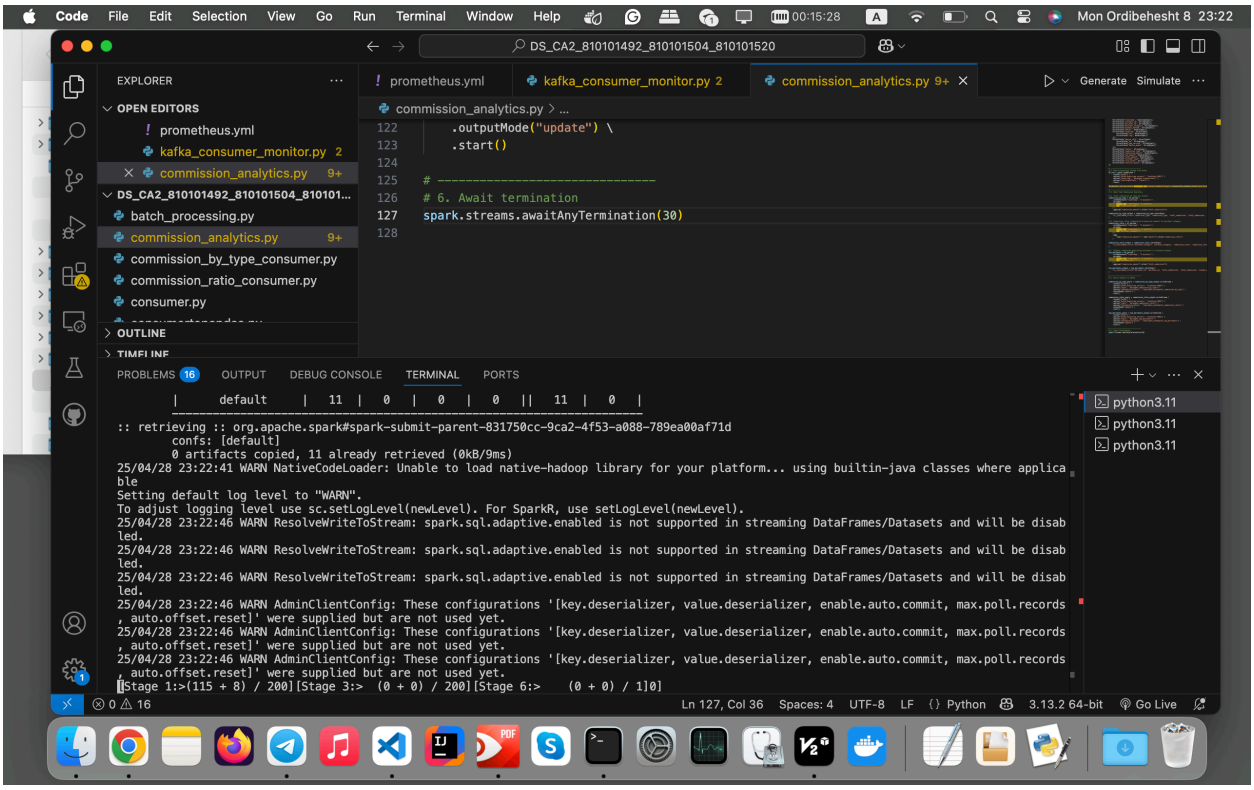
🛠️ Example folder structure:

```
CopyEdit
your_project/
│
├── commission_analytics.py
├── kafka_consumer_monitor.py
├── prometheus.yml
├── spark_checkpoint_commission_by_type/
├── spark_checkpoint_commission_ratio/
├── spark_checkpoint_top_merchants/
└── (other files)
```

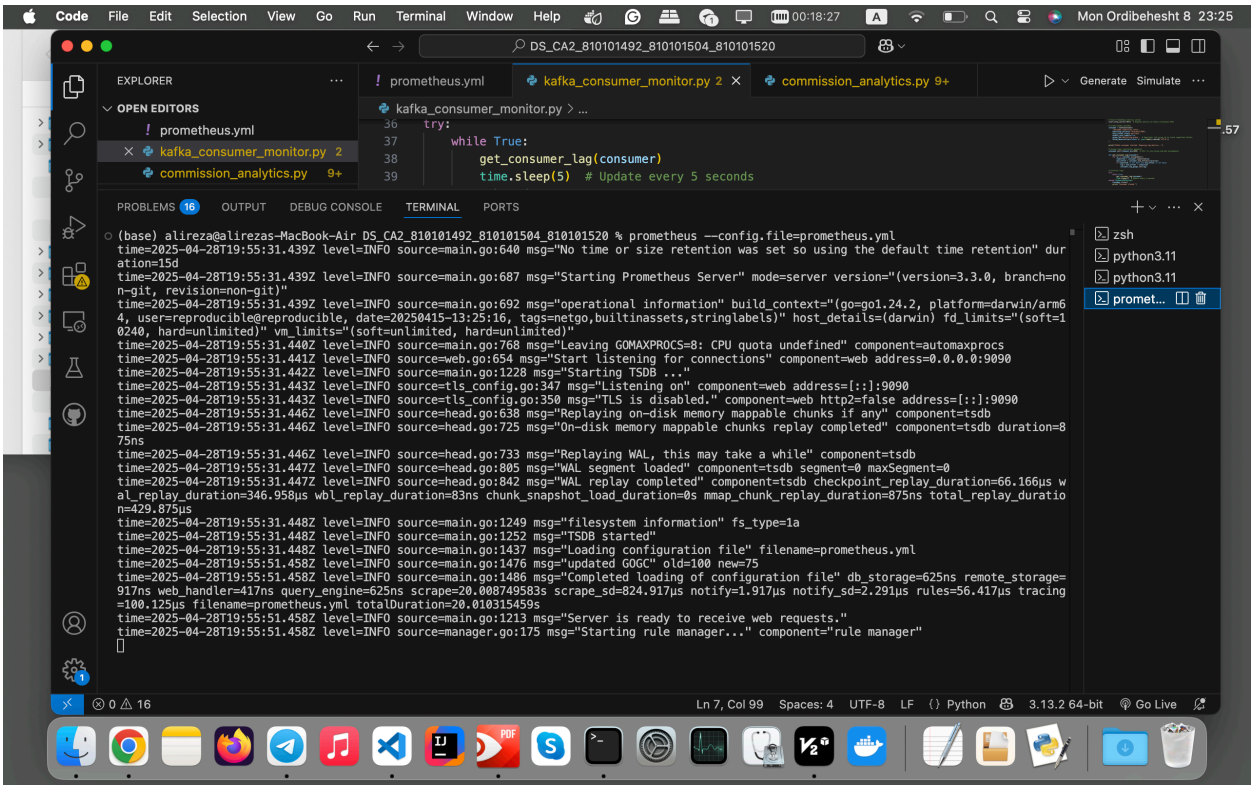
🎯 Summary

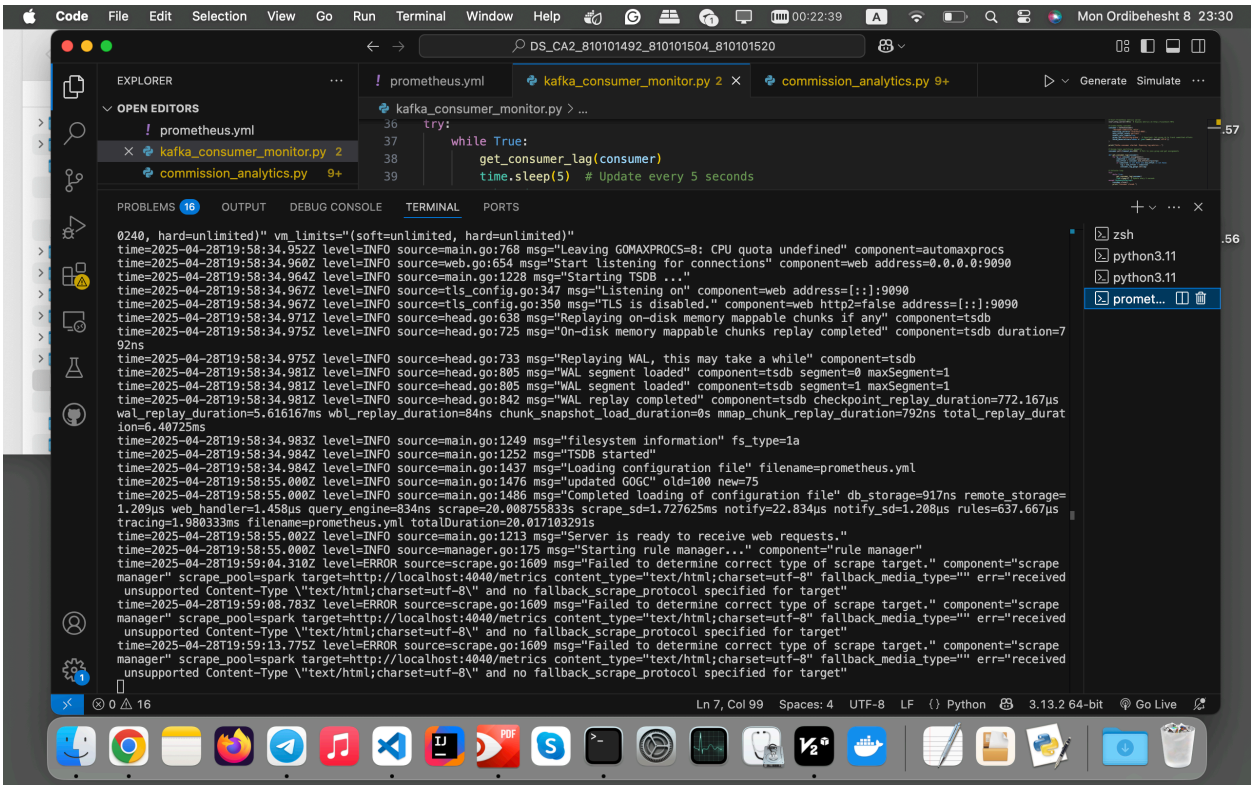
Monitor	How You Do It
Kafka Consumer Lag	`kafka_consumer_monitor.py` exposes via Prometheus
Spark Executor CPU/MEM	Scraped from `localhost:4040`
JVM Garbage Collection	From Spark metrics and Executors tab

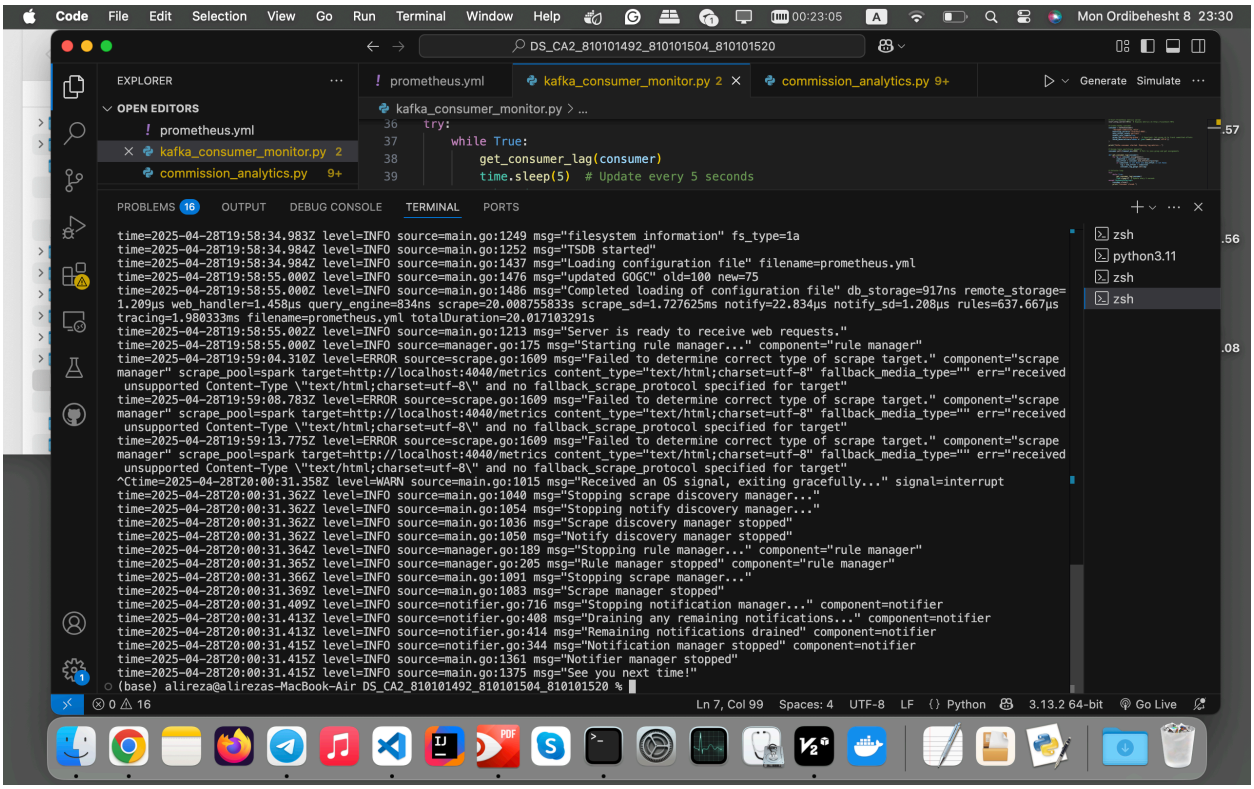
As you can see, I run commission_analytics.py for this



These are the outputs I got after running the yml file:


Code File Edit Selection View Go Run Terminal Window Help 00:18:27 Mon Ordibehesht 8 23:25
DS_CA2_810101492_810101504_810101520
EXPLORER
OPEN EDITORS
prometheus.yml
kafka_consumer_monitor.py 2
commission_analytics.py 9+
prometheus.yml
kafka_consumer_monitor.py 2
commission_analytics.py 9+
Generate Simulate
kafka_consumer_monitor.py > ...
try:
while True:
get_consumer_lag(consumer)
time.sleep(5) # Update every 5 seconds
PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL PORTS
(base) alireza@alirezas-MacBook-Air DS_CA2_810101492_810101504_810101520 % prometheus --config.file=prometheus.yml
time=2025-04-28T19:55:31.439Z level=INFO source=main.go:640 msg="No time or size retention was set so using the default time retention" duration=15d
time=2025-04-28T19:55:31.439Z level=INFO source=main.go:687 msg="Starting Prometheus Server" mode=server version="(version=3.3.0, branch=non-git, revision=non-git)"
time=2025-04-28T19:55:31.439Z level=INFO source=main.go:692 msg="operational information" build_context="(go=go1.24.2, platform=darwin/arm64, user=reproducible@reproducible, date=20250415-13:25:16, tags=netgo,builtinassets,stringlabels)" host_details=(darwin) fd_limits="(soft=10240, hard=unlimited)" vm_limits="(soft=unlimited, hard=unlimited)"
time=2025-04-28T19:55:31.440Z level=INFO source=main.go:768 msg="Leaving GOMAXPROCS=8: CPU quota undefined" component=automaxprocs
time=2025-04-28T19:55:31.441Z level=INFO source=web.go:654 msg="Start listening for connections" component=web address=0.0.0.0:9090
time=2025-04-28T19:55:31.442Z level=INFO source=main.go:1228 msg="Starting TSDB ..."
time=2025-04-28T19:55:31.443Z level=INFO source=tls_config.go:347 msg="Listening on" component=web address=[::]:9090
time=2025-04-28T19:55:31.443Z level=INFO source=tls_config.go:350 msg="TLS is disabled." component=web http2=false address=[::]:9090
time=2025-04-28T19:55:31.446Z level=INFO source=head.go:638 msg="Replaying on-disk memory mappable chunks if any" component=tsdb
time=2025-04-28T19:55:31.446Z level=INFO source=head.go:725 msg="On-disk memory mappable chunks replay completed" component=tsdb duration=875ns
time=2025-04-28T19:55:31.446Z level=INFO source=head.go:733 msg="Replaying WAL, this may take a while" component=tsdb
time=2025-04-28T19:55:31.447Z level=INFO source=head.go:805 msg="WAL segment loaded" component=tsdb segment=0 maxSegment=0
time=2025-04-28T19:55:31.447Z level=INFO source=head.go:842 msg="WAL replay completed" component=tsdb checkpoint_replay_duration=66.166µs wal_replay_duration=346.958µs wbl_replay_duration=83ns chunk_snapshot_load_duration=0s mmap_chunk_replay_duration=875ns total_replay_duration=429.875µs
time=2025-04-28T19:55:31.448Z level=INFO source=main.go:1249 msg="filesystem information" fs_type=1a
time=2025-04-28T19:55:31.448Z level=INFO source=main.go:1252 msg="TSDB started"
time=2025-04-28T19:55:31.448Z level=INFO source=main.go:1437 msg="Loading configuration file" filename=prometheus.yml
time=2025-04-28T19:55:51.458Z level=INFO source=main.go:1476 msg="updated GOGC" old=100 new=75
time=2025-04-28T19:55:51.458Z level=INFO source=main.go:1486 msg="Completed loading of configuration file" db_storage=625ns remote_storage=917ns web_handler=417ns query_engine=625ns scrape=20.008749583s scrape_sd=824.917µs notify=1.917µs notify_sd=2.291µs rules=56.417µs tracing=100.125µs filename=prometheus.yml totalDuration=20.010315459s
time=2025-04-28T19:55:51.458Z level=INFO source=main.go:1213 msg="Server is ready to receive web requests."
time=2025-04-28T19:55:51.458Z level=INFO source=manager.go:175 msg="Starting rule manager..." component="rule manager"
zsh
python3.11
python3.11
promet...
Ln 7, Col 99 Spaces: 4 UTF-8 LF Python 3.13.2 64-bit Go Live


Code File Edit Selection View Go Run Terminal Window Help 00:22:39 Mon Ordibehesht 8 23:30
DS_CA2_810101492_810101504_810101520
EXPLORER
OPEN EDITORS
prometheus.yml
kafka_consumer_monitor.py 2
commission_analytics.py 9+
prometheus.yml
kafka_consumer_monitor.py 2
commission_analytics.py 9+
Generate Simulate
kafka_consumer_monitor.py > ...
try:
while True:
get_consumer_lag(consumer)
time.sleep(5) # Update every 5 seconds
PROBLEMS 16 OUTPUT DEBUG CONSOLE TERMINAL PORTS
0240, hard=unlimited)" vm_limits="(soft=unlimited, hard=unlimited)"
time=2025-04-28T19:58:34.952Z level=INFO source=main.go:768 msg="Leaving GOMAXPROCS=8: CPU quota undefined" component=automaxprocs
time=2025-04-28T19:58:34.960Z level=INFO source=web.go:654 msg="Start listening for connections" component=web address=0.0.0.0:9090
time=2025-04-28T19:58:34.964Z level=INFO source=main.go:1228 msg="Starting TSDB ..."
time=2025-04-28T19:58:34.967Z level=INFO source=tls_config.go:347 msg="Listening on" component=web address=[::]:9090
time=2025-04-28T19:58:34.967Z level=INFO source=tls_config.go:350 msg="TLS is disabled." component=web http2=false address=[::]:9090
time=2025-04-28T19:58:34.971Z level=INFO source=head.go:638 msg="Replaying on-disk memory mappable chunks if any" component=tsdb
time=2025-04-28T19:58:34.975Z level=INFO source=head.go:725 msg="On-disk memory mappable chunks replay completed" component=tsdb duration=792ns
time=2025-04-28T19:58:34.975Z level=INFO source=head.go:733 msg="Replaying WAL, this may take a while" component=tsdb
time=2025-04-28T19:58:34.981Z level=INFO source=head.go:805 msg="WAL segment loaded" component=tsdb segment=0 maxSegment=1
time=2025-04-28T19:58:34.981Z level=INFO source=head.go:805 msg="WAL segment loaded" component=tsdb segment=1 maxSegment=1
time=2025-04-28T19:58:34.981Z level=INFO source=head.go:842 msg="WAL replay completed" component=tsdb checkpoint_replay_duration=772.167µs wal_replay_duration=5.616167ms wbl_replay_duration=84ns chunk_snapshot_load_duration=0s mmap_chunk_replay_duration=792ns total_replay_duration=6.40725ms
time=2025-04-28T19:58:34.983Z level=INFO source=main.go:1249 msg="filesystem information" fs_type=1a
time=2025-04-28T19:58:34.984Z level=INFO source=main.go:1252 msg="TSDB started"
time=2025-04-28T19:58:34.984Z level=INFO source=main.go:1437 msg="Loading configuration file" filename=prometheus.yml
time=2025-04-28T19:58:55.000Z level=INFO source=main.go:1476 msg="updated GOGC" old=100 new=75
time=2025-04-28T19:58:55.000Z level=INFO source=main.go:1486 msg="Completed loading of configuration file" db_storage=917ns remote_storage=1.209µs web_handler=1.458µs query_engine=834ns scrape=20.008755833s scrape_sd=1.727625ms notify=22.834µs notify_sd=1.208µs rules=637.667µs tracing=1.980333ms filename=prometheus.yml totalDuration=20.017103291s
time=2025-04-28T19:58:55.002Z level=INFO source=main.go:1213 msg="Server is ready to receive web requests."
time=2025-04-28T19:58:55.000Z level=INFO source=manager.go:175 msg="Starting rule manager..." component="rule manager"
time=2025-04-28T19:59:04.310Z level=ERROR source=scrape.go:1609 msg="Failed to determine correct type of scrape target." component="scrape manager" scrape_pool=spark target=http://localhost:4040/metrics content_type="text/html;charset=utf-8" fallback_media_type="" err="received unsupported Content-Type \"text/html;charset=utf-8\" and no fallback_scrape_protocol specified for target"
time=2025-04-28T19:59:08.783Z level=ERROR source=scrape.go:1609 msg="Failed to determine correct type of scrape target." component="scrape manager" scrape_pool=spark target=http://localhost:4040/metrics content_type="text/html;charset=utf-8" fallback_media_type="" err="received unsupported Content-Type \"text/html;charset=utf-8\" and no fallback_scrape_protocol specified for target"
time=2025-04-28T19:59:13.775Z level=ERROR source=scrape.go:1609 msg="Failed to determine correct type of scrape target." component="scrape manager" scrape_pool=spark target=http://localhost:4040/metrics content_type="text/html;charset=utf-8" fallback_media_type="" err="received unsupported Content-Type \"text/html;charset=utf-8\" and no fallback_scrape_protocol specified for target"
zsh
python3.11
python3.11
promet...
Ln 7, Col 99 Spaces: 4 UTF-8 LF Python 3.13.2 64-bit Go Live



And this is output in http://localhost:9090/