Real-Time E-commerce Transaction Monitoring

Submitted to:
Aditya Dua

Submitted by:
Vandana Jain 055058

E-Commerce Real-Time Analytics Pipeline Report

1. Introduction

The exponential growth of e-commerce platforms has created an urgent need for real-time analytics to understand customer behavior, detect fraud, and optimize sales strategies. Traditional batch processing systems are insufficient when businesses require immediate insights to act on customer interactions, inventory changes, or transaction risks.
This project builds a **real-time data pipeline** using **Apache Kafka, MongoDB, and Python-based consumers** to process and analyze e-commerce transaction data as it occurs. The pipeline ensures continuous ingestion, processing, enrichment, and storage of data, which can then be visualized on business dashboards for actionable insights.

---

2. Business Context

E-commerce companies generate millions of events per day, including transactions, clicks, reviews, and payments. Real-time monitoring provides several competitive advantages:

- **Fraud Detection:** Detect suspicious transactions instantly (e.g., high-value purchases from unusual locations).
- **Customer Experience:** Monitor failed payments or repeated cart abandonments and trigger interventions.
- **Sales Optimization:** Identify top-selling products and regions in real time to adjust promotions.
- **Operational Efficiency:** Track inventory depletion and restock proactively.

The implemented pipeline supports both **customer-centric insights** (e.g., purchase sentiment, repeat buyer patterns) and **business-centric insights** (e.g., revenue streams, risk alerts).

---

3. Pipeline Architecture

The real-time data pipeline follows a **four-stage architecture**:

Stage 1: Data Ingestion

- Transaction data (user, product, price, payment method, location, timestamp) is generated either from CSV simulation or live streams.
- A **Kafka Producer** pushes this data into the Kafka topic transactions.raw.

Stage 2: Data Processing & Enrichment

- A **Kafka Consumer** processes the raw stream.
- Business rules are applied:
    - Classification of transactions (low, medium, high-value).
    - Flagging of suspicious patterns (e.g., unusual geolocation, excessive refunds).
    - Aggregation of product-level sales.
- Enriched transactions are forwarded to transactions.enriched.

Stage 3: Real-Time Storage

- Data is stored in **MongoDB** under collections transactions and alerts.
- This enables queries, dashboards, and advanced analytics using MongoDB Compass or external BI tools.

Stage 4: Delivery & Visualization

- Business dashboards display KPIs such as:
  - Real-time revenue trend.
  - Top products and regions.
  - Alerts on suspicious transactions.
- Visual dashboards enable management teams to make **immediate decisions**.

---

4. Example Flow Diagram

Pipeline Flow:
[Producer → Kafka Topic (transactions.raw) → Consumer Processing → Kafka Topic (transactions.enriched) → MongoDB Storage → Business Dashboard]

---

5. Sample Data & Processing

A sample transaction record:

```
{
  "transaction_id": "T101",
  "user": "alice",
  "product": "Smartphone",
  "amount": 550,
  "payment_method": "Credit Card",
  "location": "New York",
  "timestamp": "2025-09-30 14:32:00"
}
```

Enriched Output:

```
{
  "transaction_id": "T101",
  "user": "alice",
  "product": "Smartphone",
  "amount": 550,
  "payment_method": "Credit Card",
  "location": "New York",
  "timestamp": "2025-09-30 14:32:00",
  "value_category": "High-Value",
  "alert_flag": false
}
```

If a transaction is suspicious (e.g., $5000 purchase from an unusual country), an **alert document** is created and stored in the
MongoDB alerts collection.

---

6. Business Insights

From the pipeline, a company can derive **real-time KPIs**:

- **Revenue Growth by Minute:** Continuous monitoring of total sales.
- **Product Performance:** Detect fast-selling SKUs to adjust stock.
- **Geographical Trends:** Identify new high-demand regions instantly.
- **Fraud Alerts:** Stop fraudulent transactions in seconds rather than hours.

This provides a **data-driven advantage** to the company by enabling **real-time decision-making** instead of reactive batch reports.

---

7. Visual Dashboards

Potential dashboards in MongoDB or BI tools:

- **Transaction Volume Trend:** Line chart showing transactions over time.
- **Revenue by Product:** Bar chart comparing sales by product categories.
- **Geographic Heatmap:** Map visualization of purchases by region.
- **Alerts Feed:** Table listing all flagged transactions.

---

8. Conclusion

The designed **real-time e-commerce analytics pipeline** empowers businesses to act instantly on customer behavior, operational bottlenecks, and potential risks.
Unlike traditional reporting systems, this pipeline:

- Captures **streaming transactions** from multiple sources.
- Processes and enriches them **in milliseconds**.
- Stores actionable data in **MongoDB**.
- Delivers **dashboards and alerts** for immediate response.

This ensures **better fraud prevention, enhanced customer experience, and improved revenue strategies**, making it a powerful asset for any e-commerce business.

Producer file

```python
from kafka import KafkaProducer
import pandas as pd
import json
import time

KAFKA_BROKER = "localhost:9092"
TOPIC = "transactions.raw"

producer = KafkaProducer(
    bootstrap_servers=KAFKA_BROKER,
    value_serializer=lambda v: json.dumps(v).encode("utf-8")
)

df = pd.read_csv("transactions.csv")

for _, row in df.iterrows():
    transaction = row.to_dict()
    producer.send(TOPIC, transaction)
    print(f"Sent: {transaction}")
    time.sleep(1)  # simulate real-time

```

Consumer file

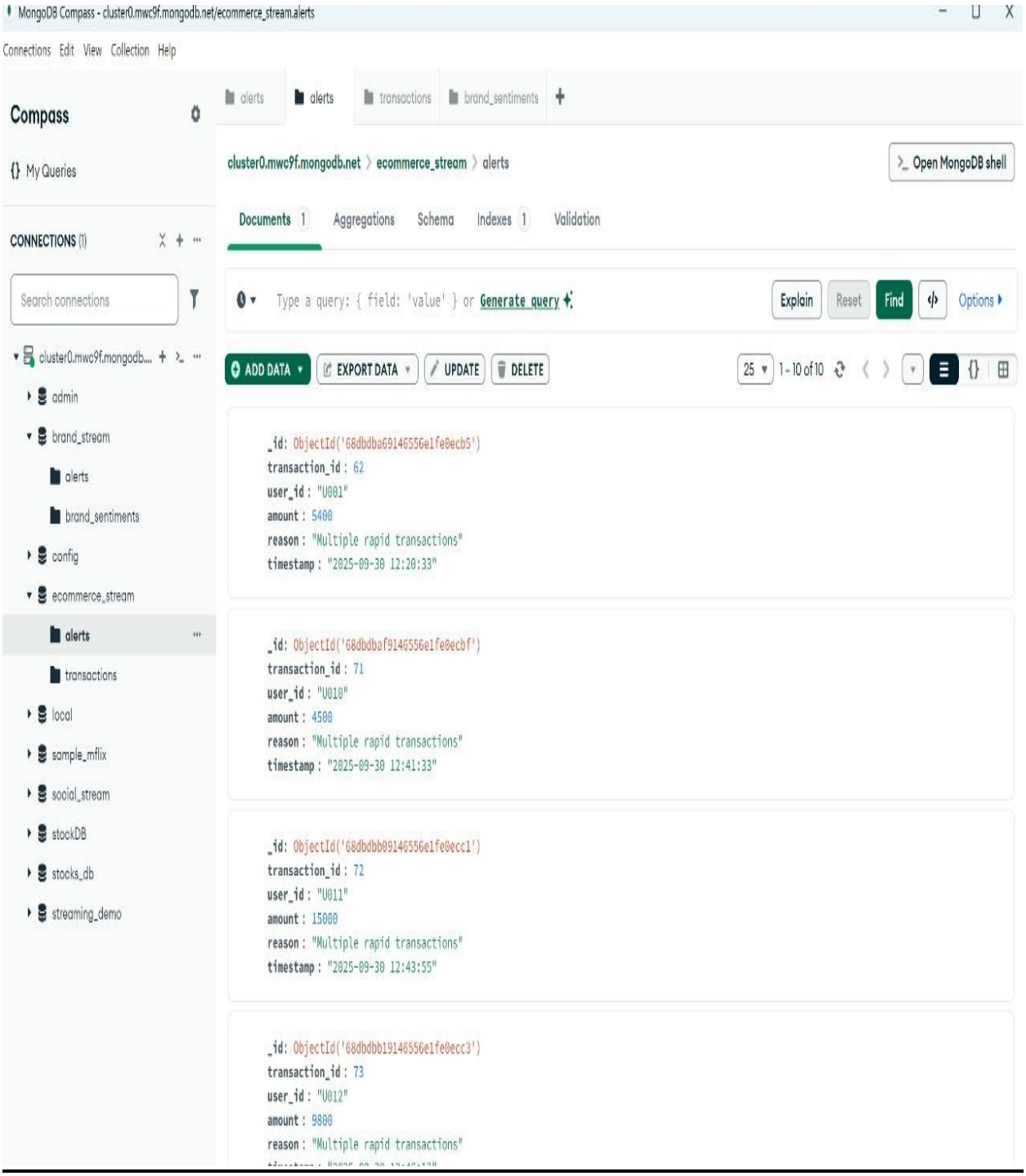
```python
from kafka import KafkaConsumer, KafkaProducer
import json
from datetime import datetime

KAFKA_BROKER = "localhost:9092"
INPUT_TOPIC = "transactions.raw"
OUTPUT_TOPIC = "transactions.processed"

consumer = KafkaConsumer(
    INPUT_TOPIC,
    bootstrap_servers=KAFKA_BROKER,
    auto_offset_reset="earliest",
    value_deserializer=lambda v: json.loads(v.decode("utf-8"))
)

producer = KafkaProducer(
    bootstrap_servers=KAFKA_BROKER,
    value_serializer=lambda v: json.dumps(v).encode("utf-8")
)

for message in consumer:
    txn = message.value
    txn["processed_time"] = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    producer.send(OUTPUT_TOPIC, txn)
    print(f"Processed & forwarded: {txn}")

```

Alerts file

```python
from kafka import KafkaConsumer
from pymongo import MongoClient
import json
from collections import defaultdict
from datetime import datetime

KAFKA_BROKER = "localhost:9092"
INPUT_TOPIC = "transactions.processed"

MONGO_URI = "mongodb+srv://055039:saloni1234@cluster0.mwc9f.mongodb.net/"
DB_NAME = "ecommerce_stream"
TXN_COLLECTION = "transactions"
ALERT_COLLECTION = "alerts"

# Mongo setup
client = MongoClient(MONGO_URI)
db = client[DB_NAME]
transactions = db[TXN_COLLECTION]
alerts = db[ALERT_COLLECTION]

consumer = KafkaConsumer(
    INPUT_TOPIC,
    bootstrap_servers=KAFKA_BROKER,
    auto_offset_reset="earliest",
    value_deserializer=lambda v: json.loads(v.decode("utf-8"))
)

user_activity = defaultdict(list)

for message in consumer:
    txn = message.value
    transactions.insert_one(txn)
    print(f"Saved txn: {txn['transaction_id']}")
```

```python

    suspicious = False
    reason = ""

    # Rule 1: High-value transaction
    if txn["amount"] > 50000:
        suspicious = True
        reason = "High transaction amount"

    # Rule 2: Too many transactions in short time
    user_activity[txn["user_id"]].append(datetime.now())
    if len(user_activity[txn["user_id"]]) > 3:
        suspicious = True
        reason = "Multiple rapid transactions"

    if suspicious:
        alert_doc = {
            "transaction_id": txn["transaction_id"],
            "user_id": txn["user_id"],
            "amount": txn["amount"],
            "reason": reason,
            "timestamp": txn["timestamp"]
        }
        alerts.insert_one(alert_doc)
        print(f"🚨 ALERT: {alert_doc}")

```

Transactions file



	A	B	C	D	E	F
1	transaction_id	user_id	amount	payment_method	location	timestamp
2	1	U001	4500	Credit Card	Mumbai	30-09-2025 10:05
3	2	U002	1200	UPI	Delhi	30-09-2025 10:06
4	3	U001	56000	Debit Card	Mumbai	30-09-2025 10:07
5	4	U003	2300	Net Banking	Bangalore	30-09-2025 10:09
6	5	U004	7800	Wallet	Chennai	30-09-2025 10:11
7	6	U005	15000	Credit Card	Kolkata	30-09-2025 10:12
8	7	U006	9800	UPI	Hyderabad	30-09-2025 10:14
9	8	U007	450	Debit Card	Pune	30-09-2025 10:16
10	9	U008	6700	Net Banking	Mumbai	30-09-2025 10:18
11	10	U009	12300	Wallet	Delhi	30-09-2025 10:20
12	11	U010	5400	Credit Card	Bangalore	30-09-2025 10:23
13	12	U011	8700	UPI	Chennai	30-09-2025 10:25
14	13	U012	2300	Debit Card	Kolkata	30-09-2025 10:27
15	14	U013	4500	Net Banking	Hyderabad	30-09-2025 10:30
16	15	U014	15000	Wallet	Pune	30-09-2025 10:32
17	16	U015	9800	Credit Card	Mumbai	30-09-2025 10:34
18	17	U016	3400	UPI	Delhi	30-09-2025 10:36
19	18	U017	12000	Debit Card	Bangalore	30-09-2025 10:38
20	19	U018	5600	Net Banking	Chennai	30-09-2025 10:41
21	20	U019	7800	Wallet	Kolkata	30-09-2025 10:43
22	21	U020	6700	Credit Card	Hyderabad	30-09-2025 10:45
23	22	U001	2300	UPI	Pune	30-09-2025 10:47
24	23	U002	4500	Debit Card	Mumbai	30-09-2025 10:49
25	24	U003	15000	Net Banking	Delhi	30-09-2025 10:51
26	25	U004	9800	Wallet	Bangalore	30-09-2025 10:53
27	26	U005	5400	Credit Card	Chennai	30-09-2025 10:56
28	27	U006	8700	UPI	Kolkata	30-09-2025 10:58
29	28	U007	2300	Debit Card	Hyderabad	30-09-2025 11:00
30	29	U008	4500	Net Banking	Pune	30-09-2025 11:03
31	30	U009	15000	Wallet	Mumbai	30-09-2025 11:05
32	31	U010	9800	Credit Card	Delhi	30-09-2025 11:07
33	32	U011	5400	UPI	Bangalore	30-09-2025 11:10
34	33	U012	6700	Debit Card	Chennai	30-09-2025 11:12
35	34	U013	2300	Net Banking	Kolkata	30-09-2025 11:14
36	35	U014	4500	Wallet	Hyderabad	30-09-2025 11:17
37	36	U015	15000	Credit Card	Pune	30-09-2025 11:19
38	37	U016	9800	UPI	Mumbai	30-09-2025 11:22
39	38	U017	6700	Debit Card	Delhi	30-09-2025 11:24
40	39	U018	3400	Net Banking	Bangalore	30-09-2025 11:26
41	40	U019	12300	Wallet	Chennai	30-09-2025 11:29
42	41	U020	5400	Credit Card	Kolkata	30-09-2025 11:31
43	42	U001	4500	UPI	Hyderabad	30-09-2025 11:33
44	43	U002	15000	Debit Card	Pune	30-09-2025 11:36
45	44	U003	9800	Net Banking	Mumbai	30-09-2025 11:38
46	45	U004	2300	Wallet	Delhi	30-09-2025 11:40
47	46	U005	6700	Credit Card	Bangalore	30-09-2025 11:43
48	47	U006	5400	UPI	Chennai	30-09-2025 11:45
49	48	U007	4500	Debit Card	Kolkata	30-09-2025 11:47
50	49	U008	15000	Net Banking	Hyderabad	30-09-2025 11:50
51	50	U009	9800	Wallet	Pune	30-09-2025 11:52
52	51	U010	6700	Credit Card	Mumbai	30-09-2025 11:54
53	52	U011	2300	UPI	Delhi	30-09-2025 11:57
54	53	U012	4500	Debit Card	Bangalore	30-09-2025 11:59
55	54	U013	15000	Net Banking	Chennai	30-09-2025 12:01
56	55	U014	9800	Wallet	Kolkata	30-09-2025 12:04
57	56	U015	5400	Credit Card	Hyderabad	30-09-2025 12:06
58	57	U016	6700	UPI	Pune	30-09-2025 12:08
59	58	U017	2300	Debit Card	Mumbai	30-09-2025 12:11
60	59	U018	4500	Net Banking	Delhi	30-09-2025 12:13
61	60	U019	15000	Wallet	Bangalore	30-09-2025 12:15
62	61	U020	9800	Credit Card	Chennai	30-09-2025 12:18
63	62	U001	5400	UPI	Kolkata	30-09-2025 12:20
64	63	U002	6700	Debit Card	Hyderabad	30-09-2025 12:22
65	64	U003	2300	Net Banking	Pune	30-09-2025 12:25
66	65	U004	4500	Wallet	Mumbai	30-09-2025 12:27
67	66	U005	15000	Credit Card	Delhi	30-09-2025 12:29
68	67	U006	9800	UPI	Bangalore	30-09-2025 12:32
69	68	U007	5400	Debit Card	Chennai	30-09-2025 12:34
70	69	U008	6700	Net Banking	Kolkata	30-09-2025 12:36
71	70	U009	2300	Wallet	Hyderabad	30-09-2025 12:39
72	71	U010	4500	Credit Card	Pune	30-09-2025 12:41

transactions

72	71	U010	4500	Credit Card	Pune	30-09-2025 12:41
73	72	U011	15000	UPI	Mumbai	30-09-2025 12:43
74	73	U012	9800	Debit Card	Delhi	30-09-2025 12:46
75	74	U013	5400	Net Banking	Bangalore	30-09-2025 12:48
76	75	U014	6700	Wallet	Chennai	30-09-2025 12:50
77	76	U015	2300	Credit Card	Kolkata	30-09-2025 12:53
78	77	U016	4500	UPI	Hyderabad	30-09-2025 12:55
79	78	U017	15000	Debit Card	Pune	30-09-2025 12:57
80	79	U018	9800	Net Banking	Mumbai	30-09-2025 13:00
81	80	U019	5400	Wallet	Delhi	30-09-2025 13:02
82	81	U020	6700	Credit Card	Bangalore	30-09-2025 13:04
83	82	U001	2300	UPI	Chennai	30-09-2025 13:07
84	83	U002	4500	Debit Card	Kolkata	30-09-2025 13:09
85	84	U003	15000	Net Banking	Hyderabad	30-09-2025 13:11
86	85	U004	9800	Wallet	Pune	30-09-2025 13:14
87	86	U005	5400	Credit Card	Mumbai	30-09-2025 13:16
88	87	U006	6700	UPI	Delhi	30-09-2025 13:18
89	88	U007	2300	Debit Card	Bangalore	30-09-2025 13:21
90	89	U008	4500	Net Banking	Chennai	30-09-2025 13:23
91	90	U009	15000	Wallet	Kolkata	30-09-2025 13:25
92	91	U010	9800	Credit Card	Hyderabad	30-09-2025 13:28
93	92	U011	5400	UPI	Pune	30-09-2025 13:30
94	93	U012	6700	Debit Card	Mumbai	30-09-2025 13:32
95	94	U013	2300	Net Banking	Delhi	30-09-2025 13:35
96	95	U014	4500	Wallet	Bangalore	30-09-2025 13:37
97	96	U015	15000	Credit Card	Chennai	30-09-2025 13:39
98	97	U016	9800	UPI	Kolkata	30-09-2025 13:42
99	98	U017	5400	Debit Card	Hyderabad	30-09-2025 13:44
100	99	U018	6700	Net Banking	Pune	30-09-2025 13:46
101	100	U019	2300	Wallet	Mumbai	30-09-2025 13:49
102						

transactions