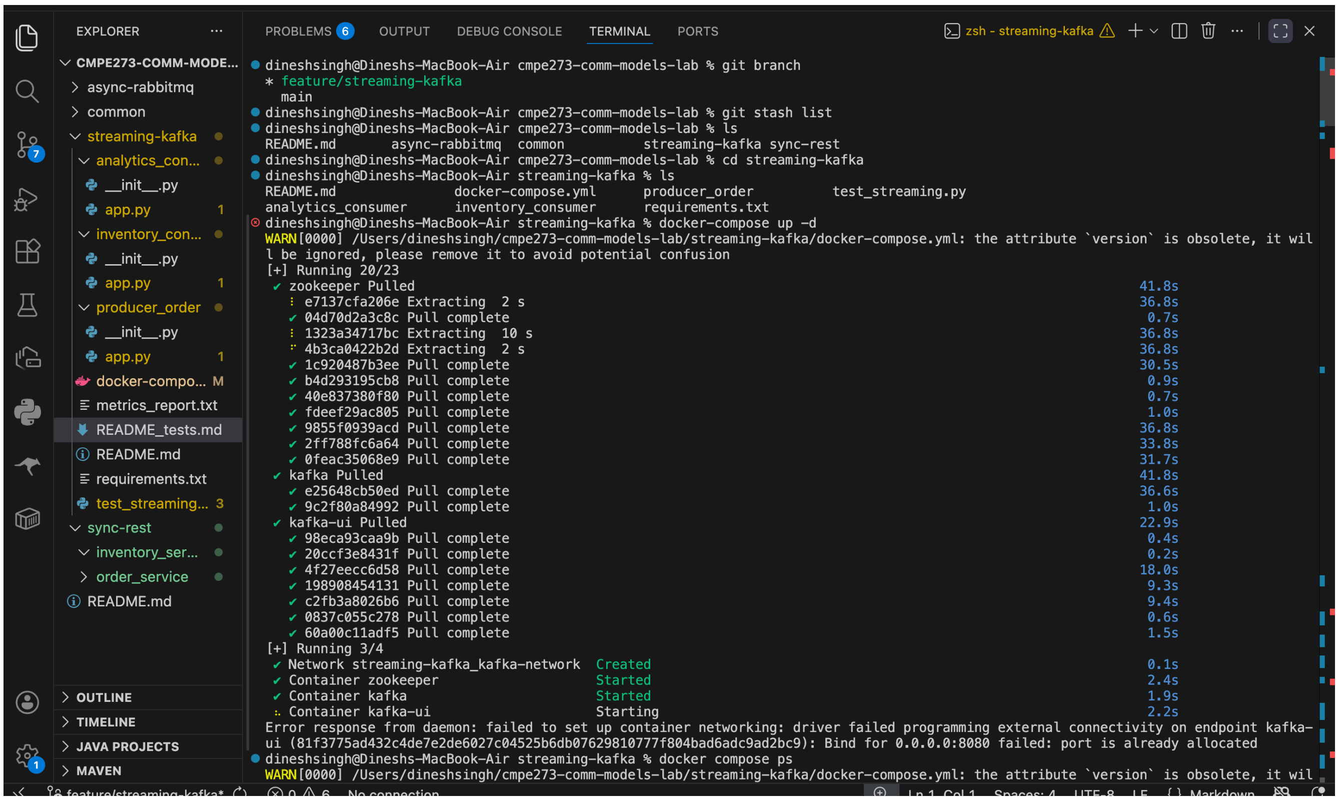
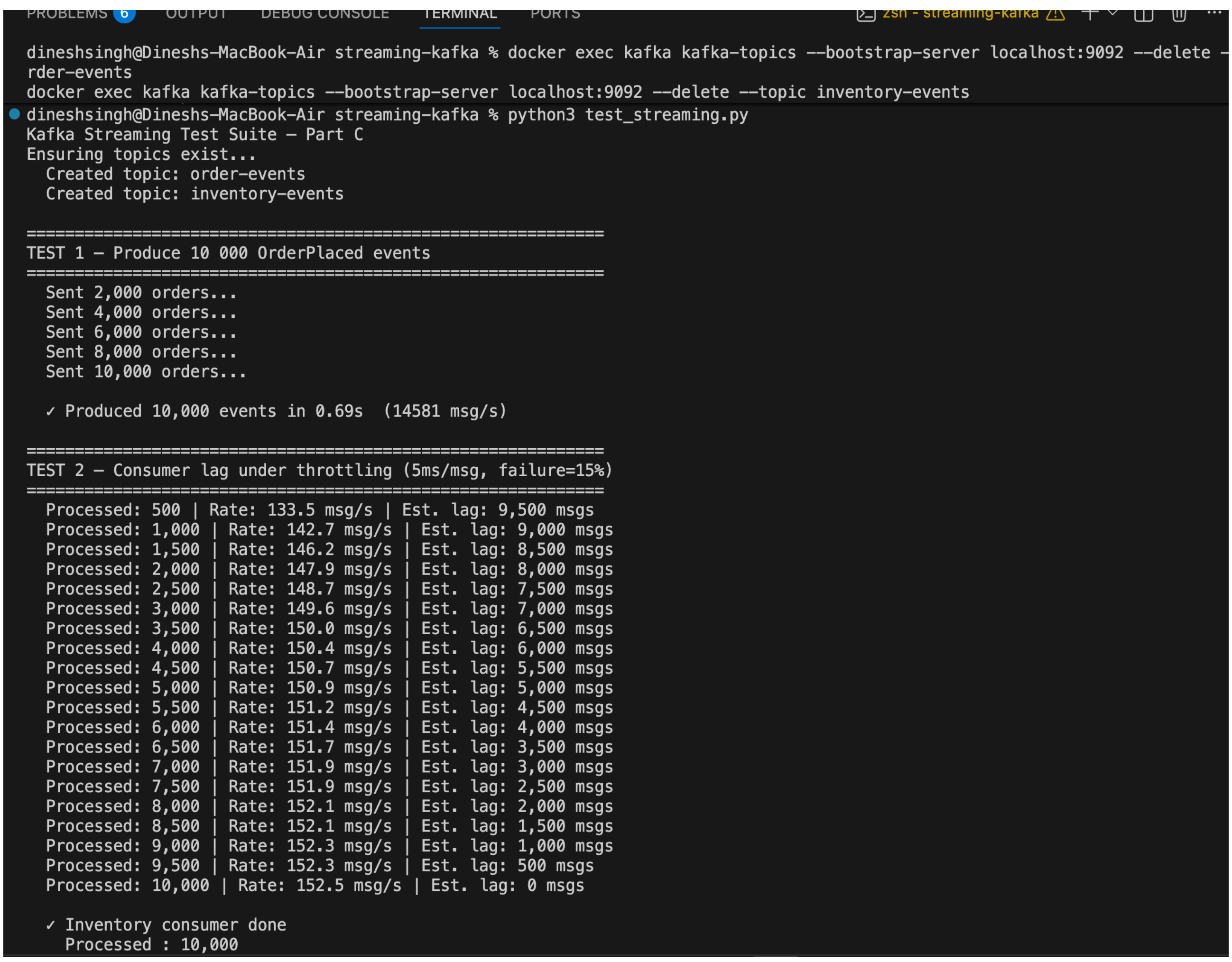
273 Screen shots for Testing - Part C

- Produce 10k events
- Show consumer lag under throttling
- Show replay producing consistent metrics (or explain why not)



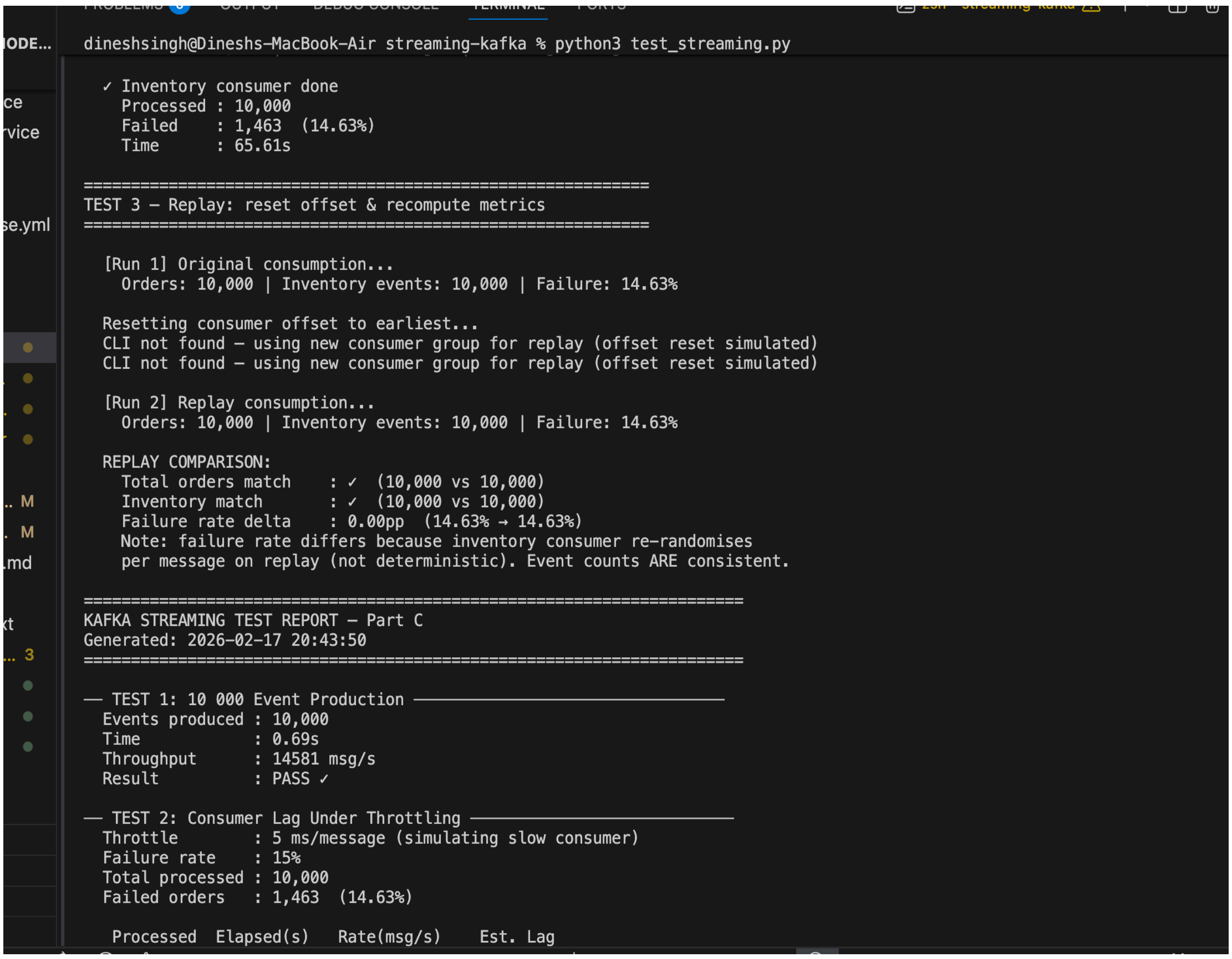
```
dineshsingh@Dineshs-MacBook-Air cmpe273-comm-models-lab % git branch
* feature/streaming-kafka
  main
dineshsingh@Dineshs-MacBook-Air cmpe273-comm-models-lab % git stash list
dineshsingh@Dineshs-MacBook-Air cmpe273-comm-models-lab % ls
README.md       async-rabbitmq  common          streaming-kafka sync-rest
dineshsingh@Dineshs-MacBook-Air cmpe273-comm-models-lab % cd streaming-kafka
dineshsingh@Dineshs-MacBook-Air streaming-kafka % ls
README.md               docker-compose.yml      producer_order          test_streaming.py
analytics_consumer      inventory_consumer      requirements.txt
dineshsingh@Dineshs-MacBook-Air streaming-kafka % docker-compose up -d
WARN[0000] /Users/dineshsingh/cmpe273-comm-models-lab/streaming-kafka/docker-compose.yml: the attribute `version` is obsolete, it wil
l be ignored, please remove it to avoid potential confusion
[+] Running 20/23
 ✔ zookeeper Pulled                                                                                41.8s
   ⠸ e7137cfa206e Extracting  2 s                                                                   36.8s
   ✔ 04d70d2a3c8c Pull complete                                                                     0.7s
   ⠸ 1323a34717bc Extracting  10 s                                                                  36.8s
   ⠙ 4b3ca0422b2d Extracting  2 s                                                                   36.8s
   ✔ 1c920487b3ee Pull complete                                                                    30.5s
   ✔ b4d293195cb8 Pull complete                                                                     0.9s
   ✔ 40e837380f80 Pull complete                                                                     0.7s
   ✔ fdeef29ac805 Pull complete                                                                     1.0s
   ✔ 9855f0939acd Pull complete                                                                    36.8s
   ✔ 2ff788fc6a64 Pull complete                                                                    33.8s
   ✔ 0feac35068e9 Pull complete                                                                    31.7s
 ✔ kafka Pulled                                                                                    41.8s
   ✔ e25648cb50ed Pull complete                                                                    36.6s
   ✔ 9c2f80a84992 Pull complete                                                                     1.0s
 ✔ kafka-ui Pulled                                                                                 22.9s
   ✔ 98eca93caa9b Pull complete                                                                     0.4s
   ✔ 20ccf3e8431f Pull complete                                                                     0.2s
   ✔ 4f27eecc6d58 Pull complete                                                                    18.0s
   ✔ 198908454131 Pull complete                                                                     9.3s
   ✔ c2fb3a8026b6 Pull complete                                                                     9.4s
   ✔ 0837c055c278 Pull complete                                                                     0.6s
   ✔ 60a00c11adf5 Pull complete                                                                     1.5s
[+] Running 3/4
 ✔ Network streaming-kafka_kafka-network  Created                                                   0.1s
 ✔ Container zookeeper                    Started                                                   2.4s
 ✔ Container kafka                        Started                                                   1.9s
 ⠼ Container kafka-ui                     Starting                                                  2.2s
Error response from daemon: failed to set up container networking: driver failed programming external connectivity on endpoint kafka-
ui (81f3775ad432c4de7e2de6027c04525b6db07629810777f804bad6adc9ad2bc9): Bind for 0.0.0.0:8080 failed: port is already allocated
dineshsingh@Dineshs-MacBook-Air streaming-kafka % docker compose ps
WARN[0000] /Users/dineshsingh/cmpe273-comm-models-lab/streaming-kafka/docker-compose.yml: the attribute `version` is obsolete, it wil
```

273 Screen shots for Testing - Part C



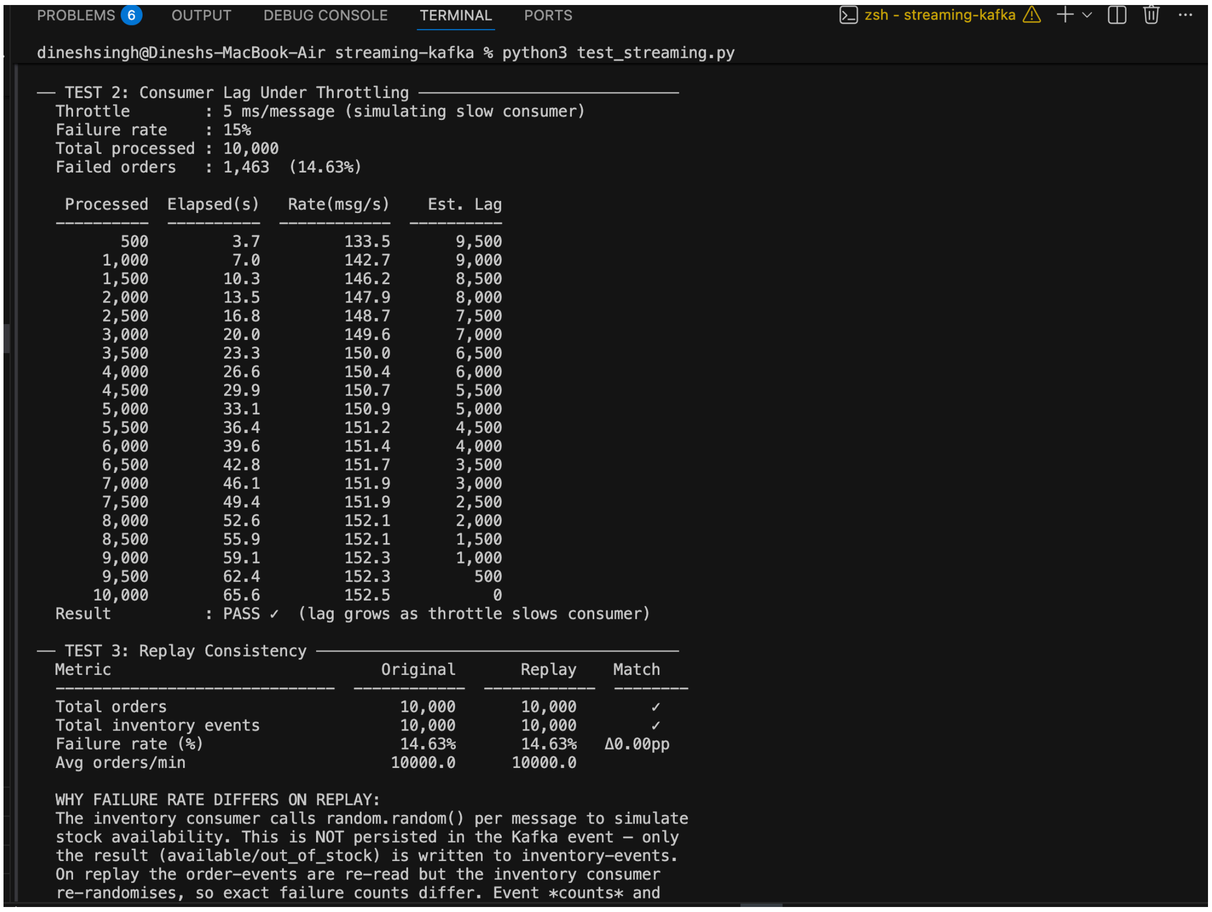
```
dineshsingh@Dineshs-MacBook-Air streaming-kafka % docker exec kafka kafka-topics --bootstrap-server localhost:9092 --delete -
rder-events
docker exec kafka kafka-topics --bootstrap-server localhost:9092 --delete --topic inventory-events
dineshsingh@Dineshs-MacBook-Air streaming-kafka % python3 test_streaming.py
Kafka Streaming Test Suite — Part C
Ensuring topics exist...
  Created topic: order-events
  Created topic: inventory-events

============================================================
TEST 1 — Produce 10 000 OrderPlaced events
============================================================
  Sent 2,000 orders...
  Sent 4,000 orders...
  Sent 6,000 orders...
  Sent 8,000 orders...
  Sent 10,000 orders...

  ✓ Produced 10,000 events in 0.69s  (14581 msg/s)

============================================================
TEST 2 — Consumer lag under throttling (5ms/msg, failure=15%)
============================================================
  Processed: 500 | Rate: 133.5 msg/s | Est. lag: 9,500 msgs
  Processed: 1,000 | Rate: 142.7 msg/s | Est. lag: 9,000 msgs
  Processed: 1,500 | Rate: 146.2 msg/s | Est. lag: 8,500 msgs
  Processed: 2,000 | Rate: 147.9 msg/s | Est. lag: 8,000 msgs
  Processed: 2,500 | Rate: 148.7 msg/s | Est. lag: 7,500 msgs
  Processed: 3,000 | Rate: 149.6 msg/s | Est. lag: 7,000 msgs
  Processed: 3,500 | Rate: 150.0 msg/s | Est. lag: 6,500 msgs
  Processed: 4,000 | Rate: 150.4 msg/s | Est. lag: 6,000 msgs
  Processed: 4,500 | Rate: 150.7 msg/s | Est. lag: 5,500 msgs
  Processed: 5,000 | Rate: 150.9 msg/s | Est. lag: 5,000 msgs
  Processed: 5,500 | Rate: 151.2 msg/s | Est. lag: 4,500 msgs
  Processed: 6,000 | Rate: 151.4 msg/s | Est. lag: 4,000 msgs
  Processed: 6,500 | Rate: 151.7 msg/s | Est. lag: 3,500 msgs
  Processed: 7,000 | Rate: 151.9 msg/s | Est. lag: 3,000 msgs
  Processed: 7,500 | Rate: 151.9 msg/s | Est. lag: 2,500 msgs
  Processed: 8,000 | Rate: 152.1 msg/s | Est. lag: 2,000 msgs
  Processed: 8,500 | Rate: 152.1 msg/s | Est. lag: 1,500 msgs
  Processed: 9,000 | Rate: 152.3 msg/s | Est. lag: 1,000 msgs
  Processed: 9,500 | Rate: 152.3 msg/s | Est. lag: 500 msgs
  Processed: 10,000 | Rate: 152.5 msg/s | Est. lag: 0 msgs

  ✓ Inventory consumer done
    Processed : 10,000
```

273 Screen shots for Testing - Part C



```
dineshsingh@Dineshs-MacBook-Air streaming-kafka % python3 test_streaming.py

  ✓ Inventory consumer done
    Processed : 10,000
    Failed    : 1,463  (14.63%)
    Time      : 65.61s

============================================================
TEST 3 — Replay: reset offset & recompute metrics
============================================================

  [Run 1] Original consumption...
    Orders: 10,000 | Inventory events: 10,000 | Failure: 14.63%

  Resetting consumer offset to earliest...
  CLI not found — using new consumer group for replay (offset reset simulated)
  CLI not found — using new consumer group for replay (offset reset simulated)

  [Run 2] Replay consumption...
    Orders: 10,000 | Inventory events: 10,000 | Failure: 14.63%

  REPLAY COMPARISON:
    Total orders match    : ✓  (10,000 vs 10,000)
    Inventory match       : ✓  (10,000 vs 10,000)
    Failure rate delta    : 0.00pp  (14.63% → 14.63%)
    Note: failure rate differs because inventory consumer re-randomises
    per message on replay (not deterministic). Event counts ARE consistent.

======================================================================
KAFKA STREAMING TEST REPORT — Part C
Generated: 2026-02-17 20:43:50
======================================================================

── TEST 1: 10 000 Event Production ─────────────────────────────────
  Events produced : 10,000
  Time            : 0.69s
  Throughput      : 14581 msg/s
  Result          : PASS ✓

── TEST 2: Consumer Lag Under Throttling ───────────────────────────
  Throttle        : 5 ms/message (simulating slow consumer)
  Failure rate    : 15%
  Total processed : 10,000
  Failed orders   : 1,463  (14.63%)

   Processed  Elapsed(s)   Rate(msg/s)    Est. Lag
```

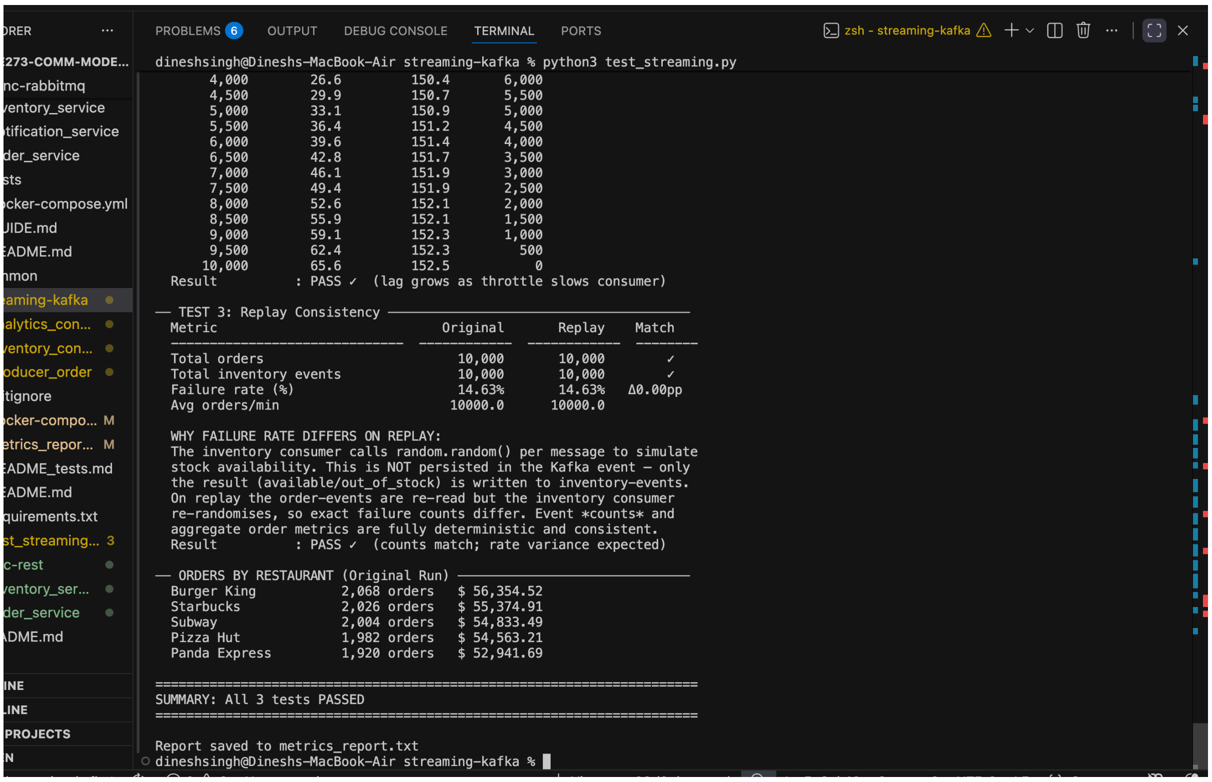
273 Screen shots for Testing - Part C



PROBLEMS 6   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   zsh - streaming-kafka

```
dineshsingh@Dineshs-MacBook-Air streaming-kafka % python3 test_streaming.py

── TEST 2: Consumer Lag Under Throttling ────────────────────────────
  Throttle        : 5 ms/message (simulating slow consumer)
  Failure rate    : 15%
  Total processed : 10,000
  Failed orders   : 1,463  (14.63%)

   Processed  Elapsed(s)   Rate(msg/s)    Est. Lag
  ──────────  ──────────  ────────────  ──────────
         500         3.7         133.5       9,500
       1,000         7.0         142.7       9,000
       1,500        10.3         146.2       8,500
       2,000        13.5         147.9       8,000
       2,500        16.8         148.7       7,500
       3,000        20.0         149.6       7,000
       3,500        23.3         150.0       6,500
       4,000        26.6         150.4       6,000
       4,500        29.9         150.7       5,500
       5,000        33.1         150.9       5,000
       5,500        36.4         151.2       4,500
       6,000        39.6         151.4       4,000
       6,500        42.8         151.7       3,500
       7,000        46.1         151.9       3,000
       7,500        49.4         151.9       2,500
       8,000        52.6         152.1       2,000
       8,500        55.9         152.1       1,500
       9,000        59.1         152.3       1,000
       9,500        62.4         152.3         500
      10,000        65.6         152.5           0
  Result          : PASS ✓  (lag grows as throttle slows consumer)

── TEST 3: Replay Consistency ───────────────────────────────────────
  Metric                                 Original       Replay     Match
  ──────────────────────────────  ────────────  ────────────  ────────
  Total orders                             10,000       10,000         ✓
  Total inventory events                   10,000       10,000         ✓
  Failure rate (%)                         14.63%       14.63%   Δ0.00pp
  Avg orders/min                          10000.0      10000.0

  WHY FAILURE RATE DIFFERS ON REPLAY:
  The inventory consumer calls random.random() per message to simulate
  stock availability. This is NOT persisted in the Kafka event — only
  the result (available/out_of_stock) is written to inventory-events.
  On replay the order-events are re-read but the inventory consumer
  re-randomises, so exact failure counts differ. Event *counts* and
```

273 Screen shots for Testing - Part C



```
dineshsingh@Dineshs-MacBook-Air streaming-kafka % python3 test_streaming.py
       4,000        26.6         150.4       6,000
       4,500        29.9         150.7       5,500
       5,000        33.1         150.9       5,000
       5,500        36.4         151.2       4,500
       6,000        39.6         151.4       4,000
       6,500        42.8         151.7       3,500
       7,000        46.1         151.9       3,000
       7,500        49.4         151.9       2,500
       8,000        52.6         152.1       2,000
       8,500        55.9         152.1       1,500
       9,000        59.1         152.3       1,000
       9,500        62.4         152.3         500
      10,000        65.6         152.5           0
  Result          : PASS ✓  (lag grows as throttle slows consumer)

── TEST 3: Replay Consistency ───────────────────────────────────────
  Metric                               Original       Replay    Match
  ──────────────────────────────  ────────────  ────────────  ────────
  Total orders                           10,000       10,000        ✓
  Total inventory events                 10,000       10,000        ✓
  Failure rate (%)                       14.63%       14.63%   Δ0.00pp
  Avg orders/min                        10000.0      10000.0

  WHY FAILURE RATE DIFFERS ON REPLAY:
  The inventory consumer calls random.random() per message to simulate
  stock availability. This is NOT persisted in the Kafka event — only
  the result (available/out_of_stock) is written to inventory-events.
  On replay the order-events are re-read but the inventory consumer
  re-randomises, so exact failure counts differ. Event *counts* and
  aggregate order metrics are fully deterministic and consistent.
  Result          : PASS ✓  (counts match; rate variance expected)

── ORDERS BY RESTAURANT (Original Run) ──────────────────────────────
  Burger King           2,068 orders   $ 56,354.52
  Starbucks             2,026 orders   $ 55,374.91
  Subway                2,004 orders   $ 54,833.49
  Pizza Hut             1,982 orders   $ 54,563.21
  Panda Express         1,920 orders   $ 52,941.69

======================================================================
SUMMARY: All 3 tests PASSED
======================================================================

Report saved to metrics_report.txt
dineshsingh@Dineshs-MacBook-Air streaming-kafka % 
```