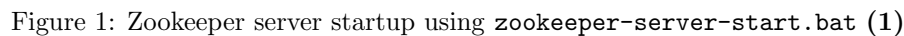


## May 7, 2024

- Installed and configured Apache Kafka on my local machine.
- Started Zookeeper server using `zookeeper-server-start.bat`.



- Started Kafka server using `kafka-server-start.bat`.

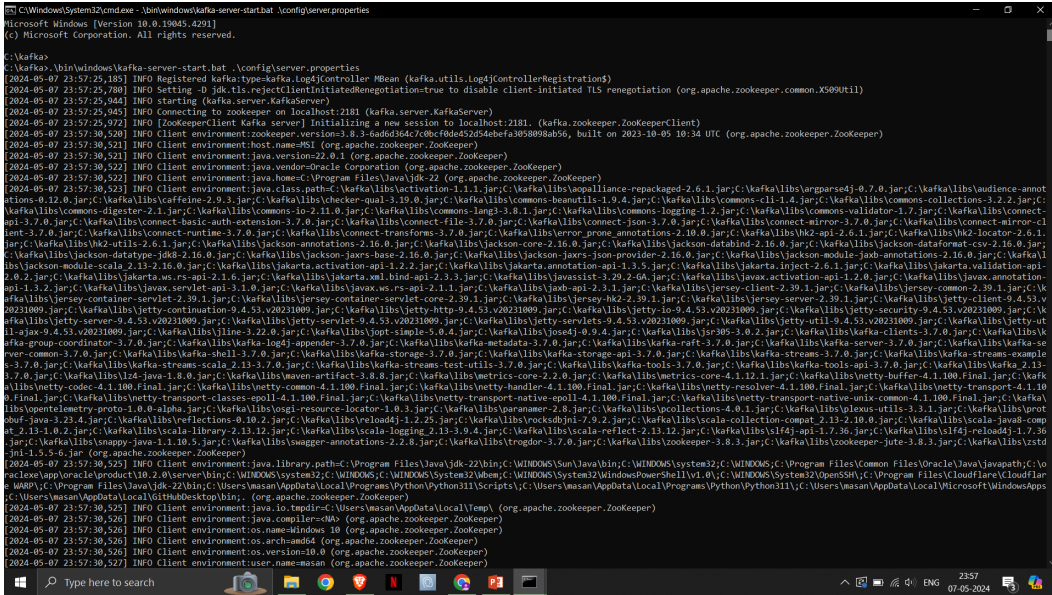


Figure 2: Launching Kafka server with server.properties configuration (2)

## 2 Creating Kafka Topic

- Created a Kafka topic named "test" using the kafka-topics.bat command.

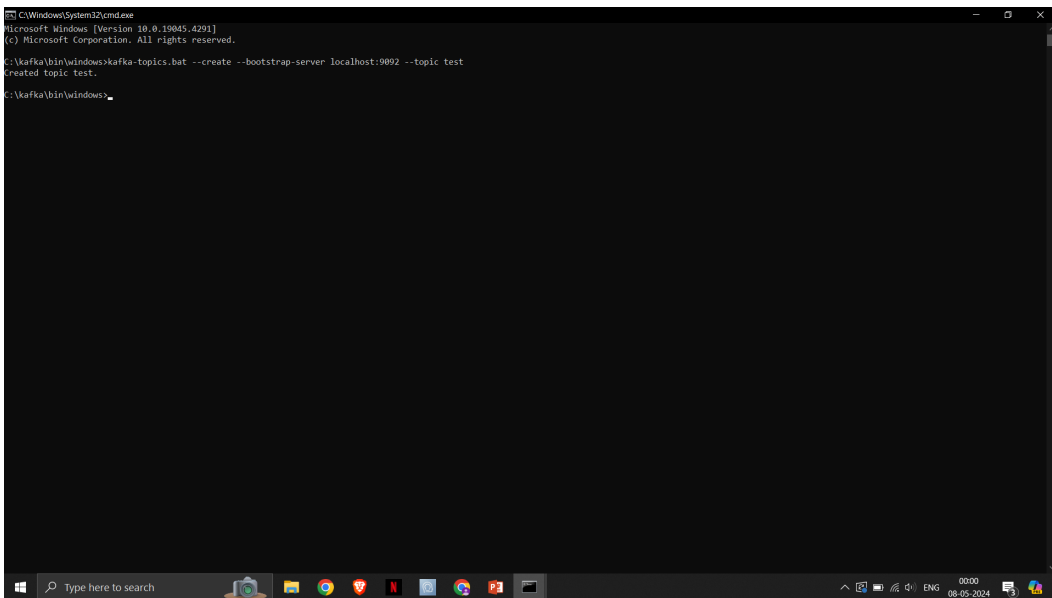


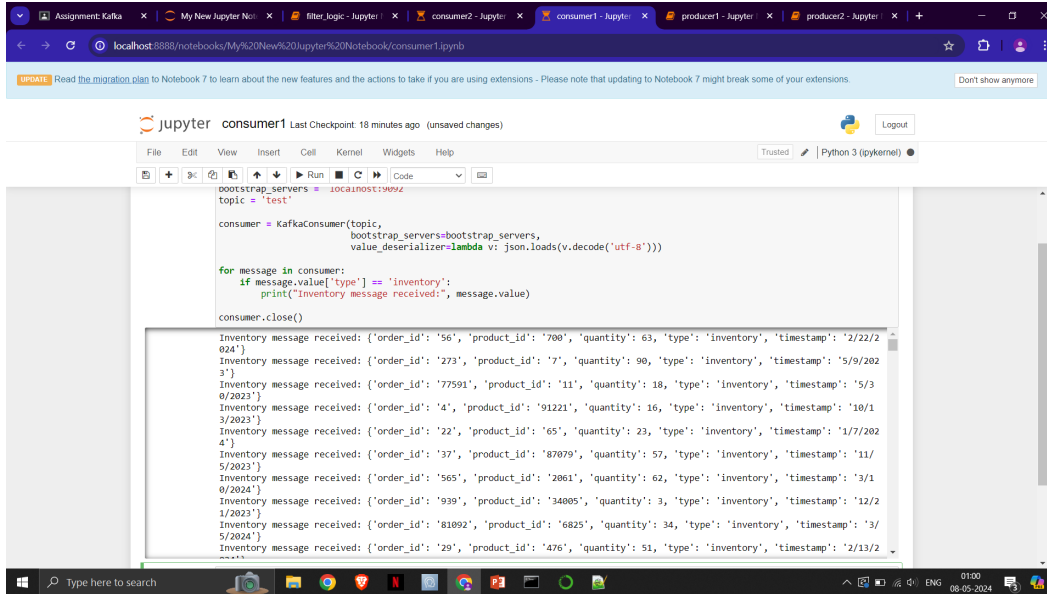
Figure 3: "Topic 'test' created successfully." (3)

## 3 Producing E-commerce Orders

- Developed Python scripts to act as Kafka producers.
- Producer 1: Inventory Orders Producer
  - Subscribed to the Kafka topic "test" to consume inventory messages.
  - Printed received inventory messages to the console.
- Producer 2: Delivery Orders Producer
  - Subscribed to the Kafka topic "test" to consume delivery messages.
  - Printed received delivery messages to the console.

## 4 Consuming E-commerce Orders

- Developed Python scripts to act as Kafka consumers.
- Consumer 1: Inventory Data Consumer
  - Subscribed to the Kafka topic "test" to consume inventory messages.
  - Printed received inventory messages to the console.



The screenshot shows a Jupyter Notebook interface for 'consumer1'. The code defines a Kafka consumer for the 'test' topic, filtering for 'inventory' messages. The output displays a series of received inventory messages, each containing order\_id, product\_id, quantity, type, and timestamp.

```
bootstrap_servers = 'localhost:9092'
topic = 'test'

consumer = KafkaConsumer(topic,
    bootstrap_servers=bootstrap_servers,
    value_deserializer=lambda v: json.loads(v.decode('utf-8')))

for message in consumer:
    if message.value['type'] == 'inventory':
        print("Inventory message received:", message.value)

consumer.close()
```

Inventory message received: {'order\_id': '56', 'product\_id': '780', 'quantity': 63, 'type': 'inventory', 'timestamp': '2/22/2024'}

Inventory message received: {'order\_id': '273', 'product\_id': '7', 'quantity': 90, 'type': 'inventory', 'timestamp': '5/9/2023'}

Inventory message received: {'order\_id': '77591', 'product\_id': '11', 'quantity': 18, 'type': 'inventory', 'timestamp': '5/3/2023'}

Inventory message received: {'order\_id': '4', 'product\_id': '91221', 'quantity': 16, 'type': 'inventory', 'timestamp': '10/13/2023'}

Inventory message received: {'order\_id': '22', 'product\_id': '65', 'quantity': 23, 'type': 'inventory', 'timestamp': '1/7/2024'}

Inventory message received: {'order\_id': '37', 'product\_id': '87079', 'quantity': 57, 'type': 'inventory', 'timestamp': '11/5/2023'}

Inventory message received: {'order\_id': '565', 'product\_id': '2061', 'quantity': 62, 'type': 'inventory', 'timestamp': '3/10/2024'}

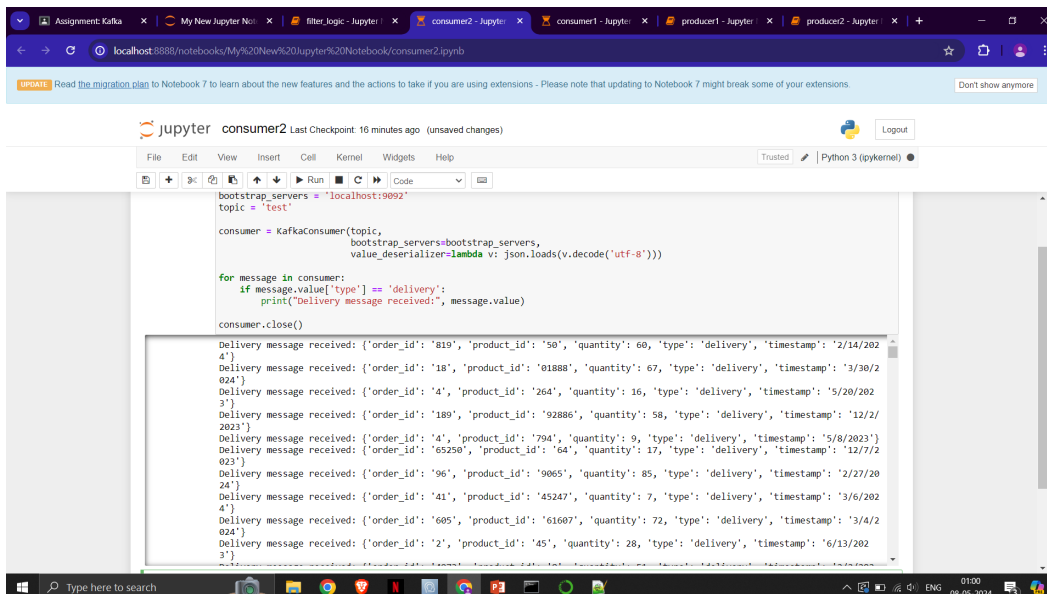
Inventory message received: {'order\_id': '939', 'product\_id': '34005', 'quantity': 3, 'type': 'inventory', 'timestamp': '12/21/2023'}

Inventory message received: {'order\_id': '81092', 'product\_id': '6825', 'quantity': 34, 'type': 'inventory', 'timestamp': '3/5/2024'}

Inventory message received: {'order\_id': '29', 'product\_id': '476', 'quantity': 51, 'type': 'inventory', 'timestamp': '2/13/2024'}

Figure 4: Consumer1 Result (4)

- Consumer 2: Delivery Data Consumer
  - Subscribed to the Kafka topic "test" to consume delivery messages.
  - Printed received delivery messages to the console.



The screenshot shows a Jupyter Notebook interface for 'consumer2'. The code defines a Kafka consumer for the 'test' topic, filtering for 'delivery' messages. The output displays a series of received delivery messages, each containing order\_id, product\_id, quantity, type, and timestamp.

```
bootstrap_servers = 'localhost:9092'
topic = 'test'

consumer = KafkaConsumer(topic,
    bootstrap_servers=bootstrap_servers,
    value_deserializer=lambda v: json.loads(v.decode('utf-8')))

for message in consumer:
    if message.value['type'] == 'delivery':
        print("Delivery message received:", message.value)

consumer.close()
```

Delivery message received: {'order\_id': '819', 'product\_id': '50', 'quantity': 60, 'type': 'delivery', 'timestamp': '2/14/2024'}

Delivery message received: {'order\_id': '18', 'product\_id': '01888', 'quantity': 67, 'type': 'delivery', 'timestamp': '3/30/2024'}

Delivery message received: {'order\_id': '4', 'product\_id': '264', 'quantity': 16, 'type': 'delivery', 'timestamp': '5/20/2023'}

Delivery message received: {'order\_id': '189', 'product\_id': '92886', 'quantity': 58, 'type': 'delivery', 'timestamp': '12/2/2023'}

Delivery message received: {'order\_id': '4', 'product\_id': '794', 'quantity': 9, 'type': 'delivery', 'timestamp': '5/8/2023'}

Delivery message received: {'order\_id': '65250', 'product\_id': '64', 'quantity': 17, 'type': 'delivery', 'timestamp': '12/7/2023'}

Delivery message received: {'order\_id': '96', 'product\_id': '9065', 'quantity': 85, 'type': 'delivery', 'timestamp': '2/27/2024'}

Delivery message received: {'order\_id': '41', 'product\_id': '45247', 'quantity': 7, 'type': 'delivery', 'timestamp': '3/6/2024'}

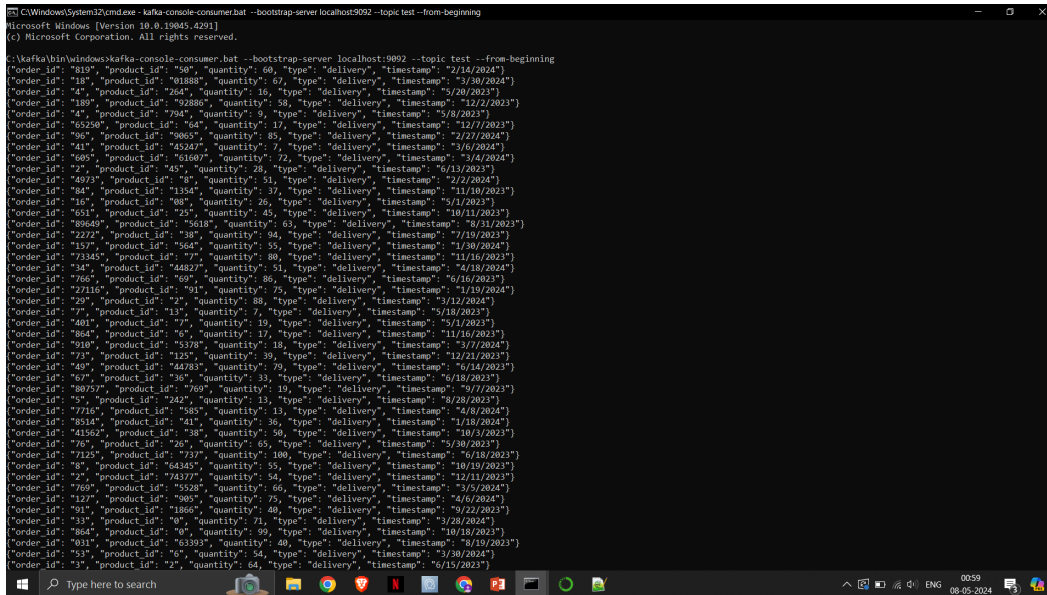
Delivery message received: {'order\_id': '605', 'product\_id': '61607', 'quantity': 72, 'type': 'delivery', 'timestamp': '3/4/2024'}

Delivery message received: {'order\_id': '2', 'product\_id': '45', 'quantity': 28, 'type': 'delivery', 'timestamp': '6/13/2023'}

Figure 5: Consumer2 Result (5)

## 5 Verifying Message Consumption

- Executed the consumer scripts to verify that inventory and delivery messages were being consumed from the Kafka topic "test".
- Confirmed successful consumption of inventory and delivery messages by observing the output in the console.



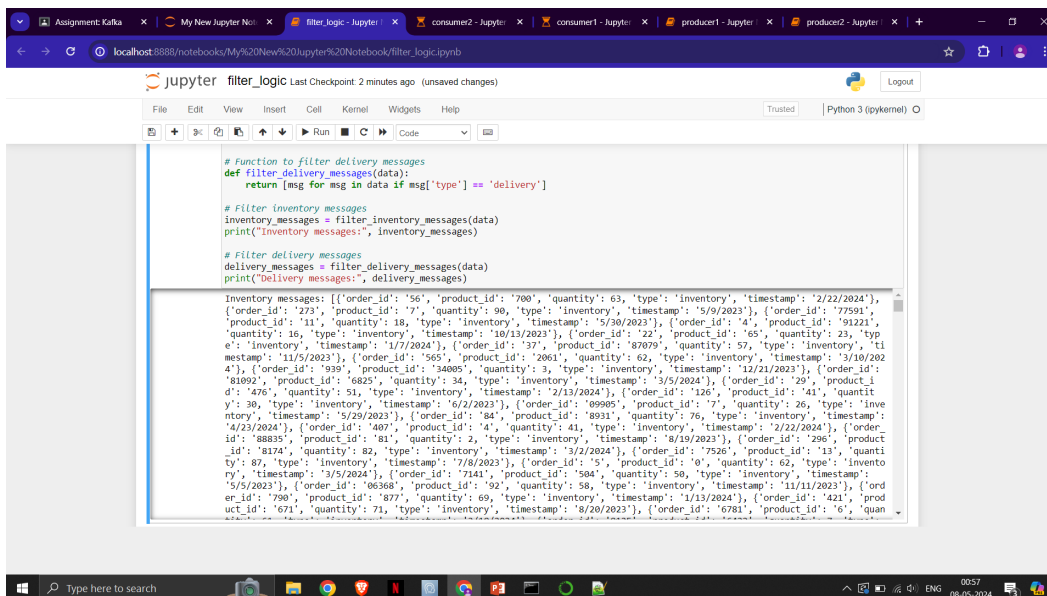
```
C:\Windows\System32\cmd.exe - kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test --from-beginning
Microsoft Windows [Version 10.0.19045.4291]
(c) Microsoft Corporation. All rights reserved.

C:\kafka\bin>kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test --from-beginning
{"order_id": "819", "product_id": "50", "quantity": 60, "type": "delivery", "timestamp": "2/14/2024"}
{"order_id": "119", "product_id": "01888", "quantity": 62, "type": "delivery", "timestamp": "5/30/2024"}
{"order_id": "4", "product_id": "264", "quantity": 36, "type": "delivery", "timestamp": "5/20/2023"}
{"order_id": "189", "product_id": "92886", "quantity": 58, "type": "delivery", "timestamp": "12/2/2023"}
{"order_id": "4", "product_id": "794", "quantity": 6, "type": "delivery", "timestamp": "5/10/2023"}
{"order_id": "65250", "product_id": "64", "quantity": 17, "type": "delivery", "timestamp": "12/7/2023"}
{"order_id": "96", "product_id": "90655", "quantity": 85, "type": "delivery", "timestamp": "2/22/2024"}
{"order_id": "41", "product_id": "42424", "quantity": 7, "type": "delivery", "timestamp": "3/6/2024"}
{"order_id": "685", "product_id": "61607", "quantity": 72, "type": "delivery", "timestamp": "5/4/2024"}
{"order_id": "2", "product_id": "45", "quantity": 28, "type": "delivery", "timestamp": "6/13/2023"}
{"order_id": "8023", "product_id": "8", "quantity": 51, "type": "delivery", "timestamp": "2/1/2024"}
{"order_id": "84", "product_id": "1184", "quantity": 39, "type": "delivery", "timestamp": "11/10/2023"}
{"order_id": "16", "product_id": "08", "quantity": 26, "type": "delivery", "timestamp": "5/1/2023"}
{"order_id": "651", "product_id": "25", "quantity": 45, "type": "delivery", "timestamp": "10/11/2023"}
{"order_id": "80660", "product_id": "5019", "quantity": 63, "type": "delivery", "timestamp": "8/31/2023"}
{"order_id": "2272", "product_id": "38", "quantity": 94, "type": "delivery", "timestamp": "7/19/2023"}
{"order_id": "137", "product_id": "564", "quantity": 55, "type": "delivery", "timestamp": "1/30/2024"}
{"order_id": "71345", "product_id": "7", "quantity": 88, "type": "delivery", "timestamp": "11/16/2023"}
{"order_id": "14", "product_id": "44827", "quantity": 51, "type": "delivery", "timestamp": "4/18/2024"}
{"order_id": "766", "product_id": "69", "quantity": 86, "type": "delivery", "timestamp": "6/16/2023"}
{"order_id": "27116", "product_id": "93", "quantity": 29, "type": "delivery", "timestamp": "2/19/2024"}
{"order_id": "29", "product_id": "2", "quantity": 89, "type": "delivery", "timestamp": "3/12/2024"}
{"order_id": "77", "product_id": "13", "quantity": 7, "type": "delivery", "timestamp": "5/18/2023"}
{"order_id": "401", "product_id": "75", "quantity": 19, "type": "delivery", "timestamp": "5/1/2023"}
{"order_id": "864", "product_id": "6", "quantity": 17, "type": "delivery", "timestamp": "11/16/2023"}
{"order_id": "910", "product_id": "5378", "quantity": 18, "type": "delivery", "timestamp": "3/7/2024"}
{"order_id": "75", "product_id": "115", "quantity": 39, "type": "delivery", "timestamp": "12/21/2023"}
{"order_id": "89", "product_id": "44781", "quantity": 70, "type": "delivery", "timestamp": "6/14/2023"}
{"order_id": "67", "product_id": "36", "quantity": 33, "type": "delivery", "timestamp": "6/18/2023"}
{"order_id": "88757", "product_id": "769", "quantity": 19, "type": "delivery", "timestamp": "9/7/2023"}
{"order_id": "4", "product_id": "242", "quantity": 13, "type": "delivery", "timestamp": "9/24/2023"}
{"order_id": "7716", "product_id": "585", "quantity": 13, "type": "delivery", "timestamp": "4/8/2024"}
{"order_id": "8514", "product_id": "41", "quantity": 36, "type": "delivery", "timestamp": "1/18/2024"}
{"order_id": "41562", "product_id": "38", "quantity": 50, "type": "delivery", "timestamp": "10/3/2023"}
{"order_id": "76", "product_id": "26", "quantity": 65, "type": "delivery", "timestamp": "5/30/2023"}
{"order_id": "7125", "product_id": "717", "quantity": 100, "type": "delivery", "timestamp": "6/18/2023"}
{"order_id": "19", "product_id": "41465", "quantity": 55, "type": "delivery", "timestamp": "10/19/2023"}
{"order_id": "2", "product_id": "74377", "quantity": 54, "type": "delivery", "timestamp": "12/11/2023"}
{"order_id": "769", "product_id": "5528", "quantity": 66, "type": "delivery", "timestamp": "3/5/2024"}
{"order_id": "122", "product_id": "905", "quantity": 79, "type": "delivery", "timestamp": "4/6/2024"}
{"order_id": "91", "product_id": "1866", "quantity": 48, "type": "delivery", "timestamp": "9/22/2023"}
{"order_id": "33", "product_id": "0", "quantity": 71, "type": "delivery", "timestamp": "3/28/2024"}
{"order_id": "864", "product_id": "87", "quantity": 99, "type": "delivery", "timestamp": "10/18/2023"}
{"order_id": "011", "product_id": "6335", "quantity": 40, "type": "delivery", "timestamp": "1/15/2023"}
{"order_id": "53", "product_id": "6", "quantity": 54, "type": "delivery", "timestamp": "3/30/2024"}
{"order_id": "3", "product_id": "2", "quantity": 64, "type": "delivery", "timestamp": "6/15/2023"}
```

Figure 6: Output of Kafka Console Consumer displaying delivery orders from the 'test' topic (6)

## 6 Implementing Message Filtering Logic

- Developed a standalone Python script to filter inventory and delivery messages from the provided JSON data.
- Implemented separate functions for filtering inventory and delivery messages based on their "type" field.
- Printed the filtered inventory and delivery messages to the console.



```
# Function to filter delivery messages
def filter_delivery_messages(data):
    return [msg for msg in data if msg['type'] == 'delivery']

# Filter inventory messages
inventory_messages = filter_inventory_messages(data)
print("Inventory messages:", inventory_messages)

# Filter delivery messages
delivery_messages = filter_delivery_messages(data)
print("Delivery messages:", delivery_messages)

Inventory messages: [{'order_id': '56', 'product_id': '780', 'quantity': 63, 'type': 'inventory', 'timestamp': '2/22/2024'},
{'order_id': '273', 'product_id': '7', 'quantity': 98, 'type': 'inventory', 'timestamp': '5/9/2023'}, {'order_id': '77591',
'product_id': '11', 'quantity': 18, 'type': 'inventory', 'timestamp': '5/50/2023'}, {'order_id': '4', 'product_id': '91221',
'quantity': 16, 'type': 'inventory', 'timestamp': '10/13/2023'}, {'order_id': '22', 'product_id': '65', 'quantity': 23, 'type': 'inventory', 'timestamp': '1/7/2024'}, {'order_id': '37', 'product_id': '87079', 'quantity': 57, 'type': 'inventory', 'timestamp': '11/5/2023'}, {'order_id': '565', 'product_id': '2061', 'quantity': 62, 'type': 'inventory', 'timestamp': '3/10/2024'}, {'order_id': '939', 'product_id': '14005', 'quantity': 3, 'type': 'inventory', 'timestamp': '12/21/2023'}, {'order_id': '81092', 'product_id': '6825', 'quantity': 34, 'type': 'inventory', 'timestamp': '3/5/2024'}, {'order_id': '29', 'product_id': '476', 'quantity': 51, 'type': 'inventory', 'timestamp': '2/13/2024'}, {'order_id': '126', 'product_id': '41', 'quantity': 30, 'type': 'inventory', 'timestamp': '6/12/2023'}, {'order_id': '09005', 'product_id': '7', 'quantity': 26, 'type': 'inventory', 'timestamp': '5/29/2023'}, {'order_id': '84', 'product_id': '8931', 'quantity': 76, 'type': 'inventory', 'timestamp': '4/23/2024'}, {'order_id': '407', 'product_id': '4', 'quantity': 41, 'type': 'inventory', 'timestamp': '2/22/2024'}, {'order_id': '88835', 'product_id': '81', 'quantity': 2, 'type': 'inventory', 'timestamp': '8/19/2023'}, {'order_id': '296', 'product_id': '18174', 'quantity': 82, 'type': 'inventory', 'timestamp': '3/2/2024'}, {'order_id': '7526', 'product_id': '13', 'quantity': 87, 'type': 'inventory', 'timestamp': '7/8/2023'}, {'order_id': '5', 'product_id': '0', 'quantity': 62, 'type': 'inventory', 'timestamp': '3/5/2024'}, {'order_id': '7141', 'product_id': '504', 'quantity': 50, 'type': 'inventory', 'timestamp': '5/5/2023'}, {'order_id': '06368', 'product_id': '92', 'quantity': 58, 'type': 'inventory', 'timestamp': '11/11/2023'}, {'order_id': '780', 'product_id': '977', 'quantity': 69, 'type': 'inventory', 'timestamp': '1/15/2024'}, {'order_id': '421', 'product_id': '671', 'quantity': 71, 'type': 'inventory', 'timestamp': '8/20/2023'}, {'order_id': '6781', 'product_id': '16', 'quantity': 6, 'type': 'inventory', 'timestamp': '6/15/2023'}
```

Figure 7: Filtered inventory messages displayed as part of the filter logic output (7)

## 7 Report Summary

- Successfully set up Apache Kafka environment on the local machine.
- Created a Kafka topic named "test" for handling e-commerce orders.
- Developed Kafka producer and consumer scripts to handle inventory and delivery messages in real-time.
- Tested the system by producing and consuming e-commerce orders, ensuring correct processing and output.