Stream Processing using Spark Streaming

School of Science and Engineering

Nouamane Zanboui

Meriem Lmoubariki

Dr. Tajjeeddine Rachidi

Al Akhawayn University in Ifrane

November 25 , 2024

Table of Contents

In this assignment, we used **Apache Kafka** and **Apache Spark** to process streaming data in real-time. Kafka was used to create a topic and produce messages, while Spark processed and displayed the messages. By running these technologies in Docker containers, we explored how modern tools work together for distributed data processing.

2. Question 1: Kafka and Spark Streaming Integration

2.1. Environment Setup

Step 1: **Set up Docker containers**
We created Docker containers for Kafka, Zookeeper, and Spark to handle streaming and processing.

Commands:

1. **Create a network for communication between the containers**:
2. docker network create kafka-net
3. **Run the Zookeeper container**:
4. docker run -d --name zookeeper --network kafka-net -p 2181:2181 -e ALLOW_ANONYMOUS_LOGIN=yes wurstmeister/zookeeper
5. **Run the Kafka container**:
6. docker run -d --name kafka --network kafka-net -p 9092:9092 -e KAFKA_BROKER_ID=1 -e KAFKA_ZOOKEEPER_CONNECT=zookeeper:2181 -e KAFKA_ADVERTISED_LISTENERS=PLAINTEXT://localhost:9092 wurstmeister/kafka

Step 2: **Verify Kafka and Zookeeper**

- Check the running containers:
- docker ps

This ensures all services (Kafka, Zookeeper, and Spark) are running properly.

2.2. Commands and Steps

Step 1: **Create a Kafka topic**

- Create a topic named testTopic:
- docker exec -it kafka /opt/kafka/bin/kafka-topics.sh --create --topic testTopic --bootstrap-server localhost:9092 --partitions 1 --replication-factor 1

Step 2: **Launch Spark Shell**

- Open the Spark shell with Kafka integration:
- docker exec -it spark-container /opt/bitnami/spark/bin/spark-shell --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.5.3

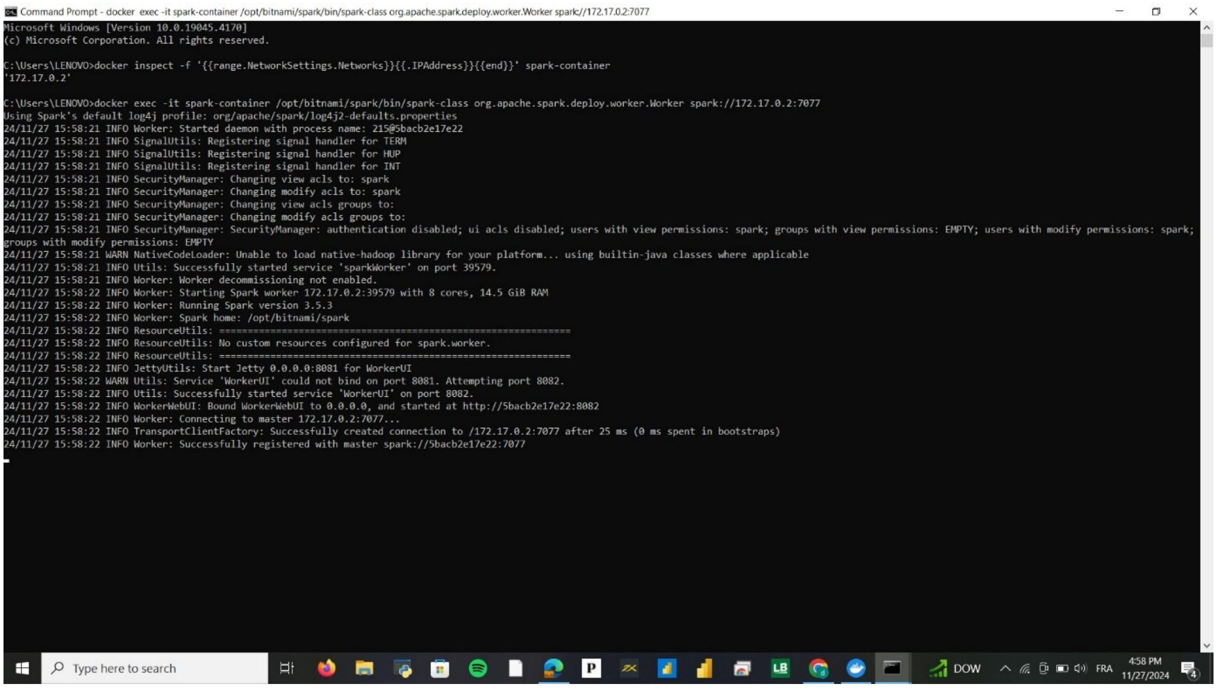
Step 3: **Produce messages to Kafka**

- Produce messages to the testTopic:
- docker exec -it kafka /opt/kafka/bin/kafka-console-producer.sh --bootstrap-server localhost:9092 --topic testTopic

Example messages:

Hello

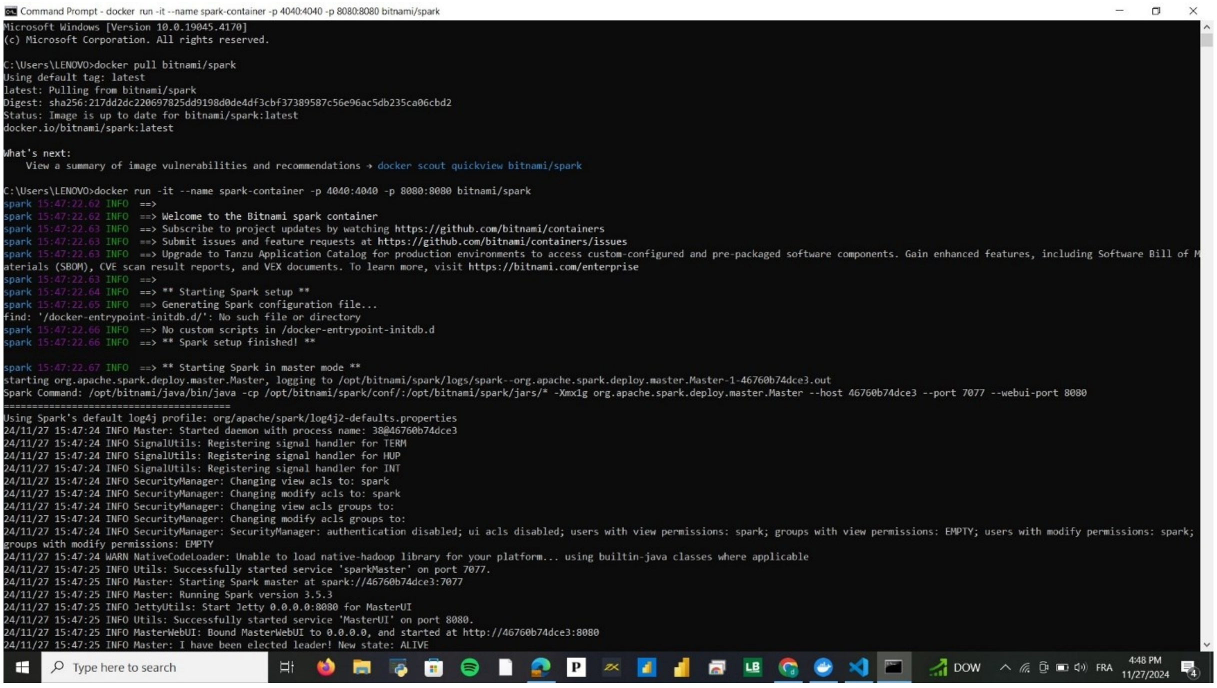
Spark and Kafka are awesome!

1. 

Command Prompt - docker exec -it spark-container /opt/bitnami/spark/bin/spark-class org.apache.spark.deploy.worker.Worker spark://172.17.0.2:7077

```
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LENOVO>docker inspect -f '{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' spark-container
'172.17.0.2'

C:\Users\LENOVO>docker exec -it spark-container /opt/bitnami/spark/bin/spark-class org.apache.spark.deploy.worker.Worker spark://172.17.0.2:7077
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
24/11/27 15:58:21 INFO Worker: Started daemon with process name: 215@5bacb2e17e22
24/11/27 15:58:21 INFO SignalUtils: Registering signal handler for TERM
24/11/27 15:58:21 INFO SignalUtils: Registering signal handler for HUP
24/11/27 15:58:21 INFO SignalUtils: Registering signal handler for INT
24/11/27 15:58:21 INFO SecurityManager: Changing view acls to: spark
24/11/27 15:58:21 INFO SecurityManager: Changing modify acls to: spark
24/11/27 15:58:21 INFO SecurityManager: Changing view acls groups to:
24/11/27 15:58:21 INFO SecurityManager: Changing modify acls groups to:
24/11/27 15:58:21 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: spark; groups with view permissions: EMPTY; users with modify permissions: spark;
groups with modify permissions: EMPTY
24/11/27 15:58:21 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/11/27 15:58:21 INFO Utils: Successfully started service 'sparkWorker' on port 39579.
24/11/27 15:58:21 INFO Worker: Worker decommissioning not enabled.
24/11/27 15:58:22 INFO Worker: Starting Spark worker 172.17.0.2:39579 with 8 cores, 14.5 GiB RAM
24/11/27 15:58:22 INFO Worker: Running Spark version 3.5.3
24/11/27 15:58:22 INFO Worker: Spark home: /opt/bitnami/spark
24/11/27 15:58:22 INFO ResourceUtils: ==============================================================
24/11/27 15:58:22 INFO ResourceUtils: No custom resources configured for spark.worker.
24/11/27 15:58:22 INFO ResourceUtils: ==============================================================
24/11/27 15:58:22 INFO JettyUtils: Start Jetty 0.0.0.0:8081 for WorkerUI
24/11/27 15:58:22 WARN Utils: Service 'WorkerUI' could not bind on port 8081. Attempting port 8082.
24/11/27 15:58:22 INFO Utils: Successfully started service 'WorkerUI' on port 8082.
24/11/27 15:58:22 INFO WorkerWebUI: Bound WorkerWebUI to 0.0.0.0, and started at http://5bacb2e17e22:8082
24/11/27 15:58:22 INFO Worker: Connecting to master 172.17.0.2:7077...
24/11/27 15:58:22 INFO TransportClientFactory: Successfully created connection to /172.17.0.2:7077 after 25 ms (0 ms spent in bootstraps)
24/11/27 15:58:22 INFO Worker: Successfully registered with master spark://5bacb2e17e22:7077
```

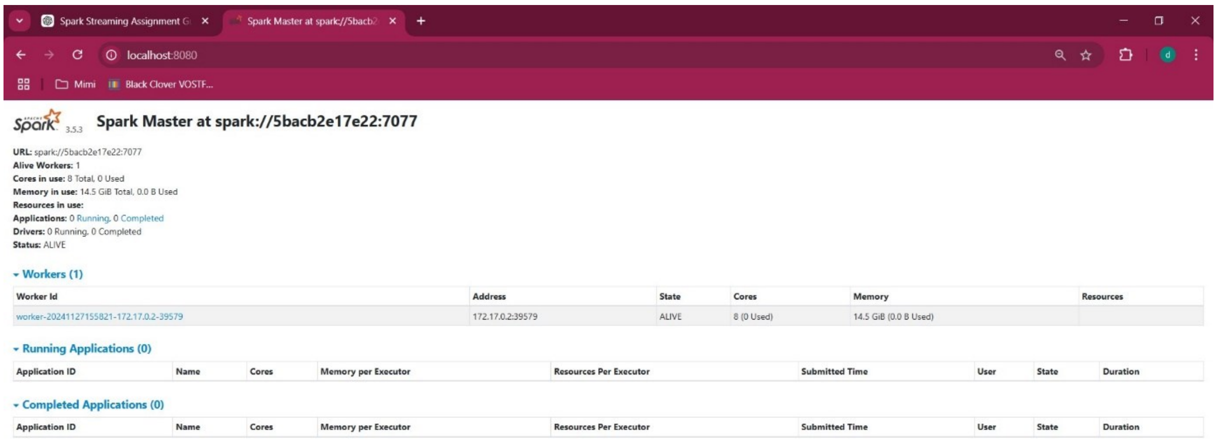
Type here to search DOW FRA 4:58 PM 11/27/2024

2. 

Command Prompt - docker run -it --name spark-container -p 4040:4040 -p 8080:8080 bitnami/spark

```
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LENOVO>docker pull bitnami/spark
Using default tag: latest
latest: Pulling from bitnami/spark
Digest: sha256:217dd2dc220697825dd9198d0de4df3cbf37389587c56e96ac5db235ca06cbd2
Status: Image is up to date for bitnami/spark:latest
docker.io/bitnami/spark:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview bitnami/spark

C:\Users\LENOVO>docker run -it --name spark-container -p 4040:4040 -p 8080:8080 bitnami/spark
spark 15:47:22.62 INFO  ==>
spark 15:47:22.62 INFO  ==> Welcome to the Bitnami spark container
spark 15:47:22.63 INFO  ==> Subscribe to project updates by watching https://github.com/bitnami/containers
spark 15:47:22.63 INFO  ==> Submit issues and feature requests at https://github.com/bitnami/containers/issues
spark 15:47:22.63 INFO  ==> Upgrade to Tanzu Application Catalog for production environments to access custom-configured and pre-packaged software components. Gain enhanced features, including Software Bill of M
aterials (SBOM), CVE scan result reports, and VEX documents. To learn more, visit https://bitnami.com/enterprise
spark 15:47:22.63 INFO  ==>
spark 15:47:22.64 INFO  ==> ** Starting Spark setup **
spark 15:47:22.65 INFO  ==> Generating Spark configuration file...
find: '/docker-entrypoint-initdb.d/': No such file or directory
spark 15:47:22.66 INFO  ==> No custom scripts in /docker-entrypoint-initdb.d
spark 15:47:22.66 INFO  ==> ** Spark setup finished! **

spark 15:47:22.67 INFO  ==> ** Starting Spark in master mode **
starting org.apache.spark.deploy.master.Master, logging to /opt/bitnami/spark/logs/spark--org.apache.spark.deploy.master.Master-1-46760b74dce3.out
Spark Command: /opt/bitnami/java/bin/java -cp /opt/bitnami/spark/conf/:/opt/bitnami/spark/jars/* -Xmx1g org.apache.spark.deploy.master.Master --host 46760b74dce3 --port 7077 --webui-port 8080
========================================
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
24/11/27 15:47:24 INFO Master: Started daemon with process name: 38@46760b74dce3
24/11/27 15:47:24 INFO SignalUtils: Registering signal handler for TERM
24/11/27 15:47:24 INFO SignalUtils: Registering signal handler for HUP
24/11/27 15:47:24 INFO SignalUtils: Registering signal handler for INT
24/11/27 15:47:24 INFO SecurityManager: Changing view acls to: spark
24/11/27 15:47:24 INFO SecurityManager: Changing modify acls to: spark
24/11/27 15:47:24 INFO SecurityManager: Changing view acls groups to:
24/11/27 15:47:24 INFO SecurityManager: Changing modify acls groups to:
24/11/27 15:47:24 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: spark; groups with view permissions: EMPTY; users with modify permissions: spark;
groups with modify permissions: EMPTY
24/11/27 15:47:24 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/11/27 15:47:25 INFO Utils: Successfully started service 'sparkMaster' on port 7077.
24/11/27 15:47:25 INFO Master: Starting Spark master at spark://46760b74dce3:7077
24/11/27 15:47:25 INFO Master: Running Spark version 3.5.3
24/11/27 15:47:25 INFO JettyUtils: Start Jetty 0.0.0.0:8080 for MasterUI
24/11/27 15:47:25 INFO Utils: Successfully started service 'MasterUI' on port 8080.
24/11/27 15:47:25 INFO MasterWebUI: Bound MasterWebUI to 0.0.0.0, and started at http://46760b74dce3:8080
24/11/27 15:47:25 INFO Master: I have been elected leader! New state: ALIVE
```

Type here to search DOW FRA 4:48 PM 11/27/2024



Spark Streaming Assignment G | Spark Master at spark://5bacb2

localhost:8080

Spark 3.5.3 **Spark Master at spark://5bacb2e17e22:7077**

URL: spark://5bacb2e17e22:7077
Alive Workers: 1
Cores in use: 8 Total, 0 Used
Memory in use: 14.5 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory	Resources
worker-20241127155821-172.17.0.2-39579	172.17.0.2:39579	ALIVE	8 (0 Used)	14.5 GiB (0.0 B Used)	

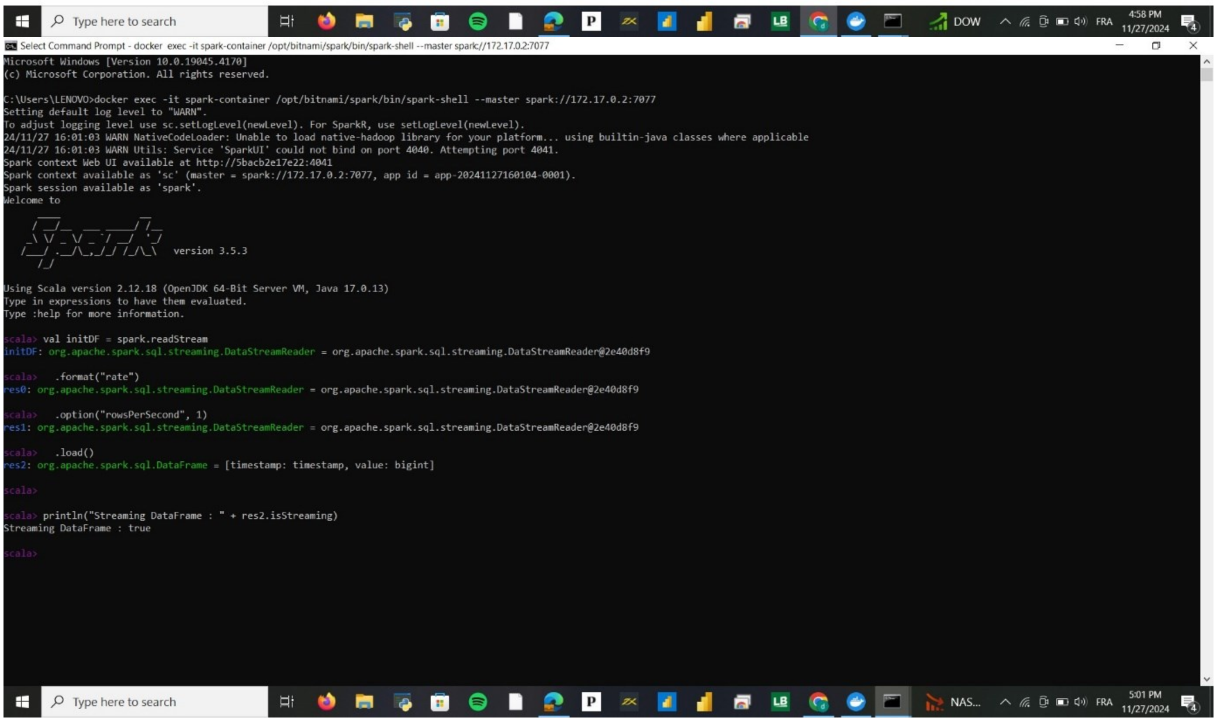
Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

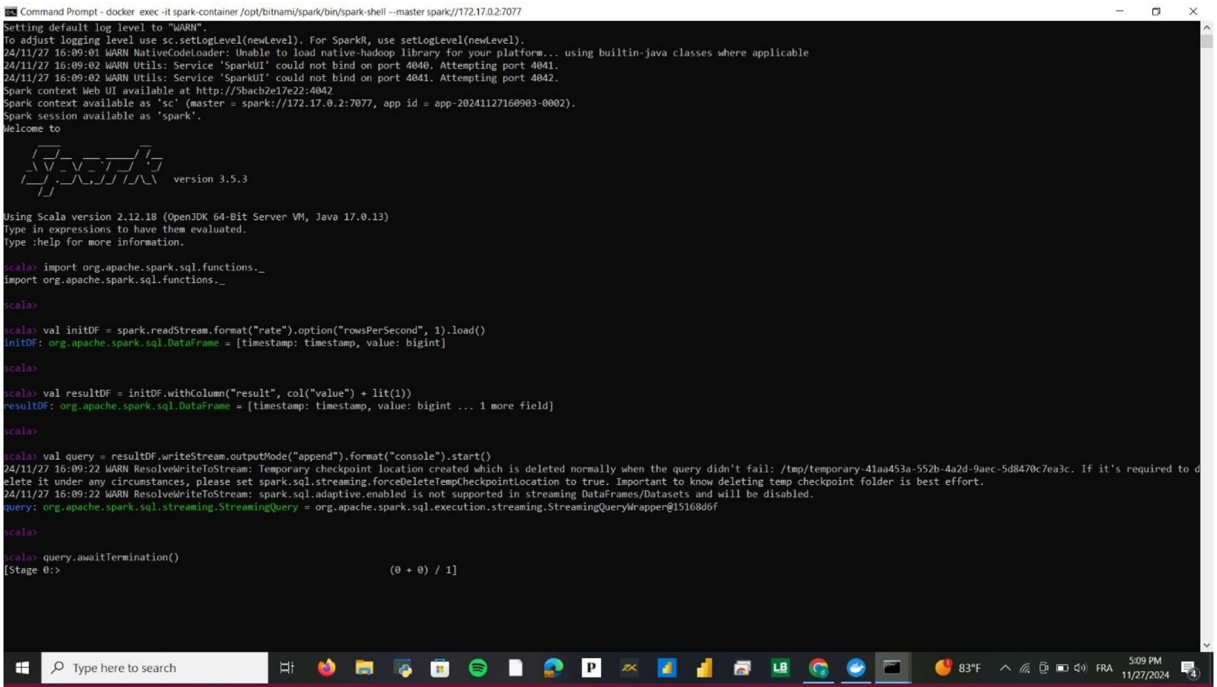
3.



Select Command Prompt - docker exec -it spark-container /opt/bitnami/spark/bin/spark-shell --master spark://172.17.0.2:7077

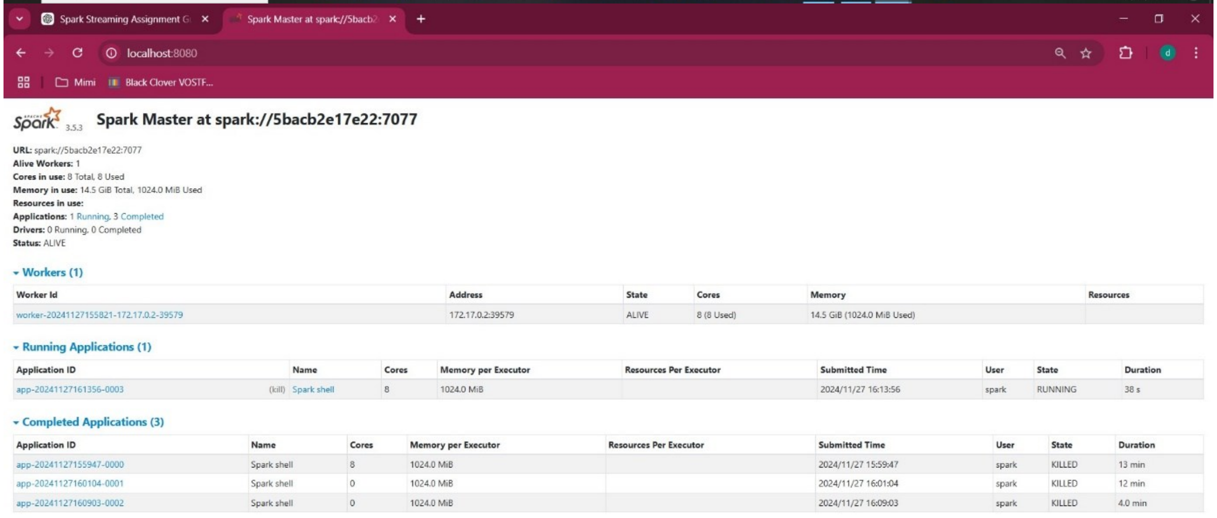
```
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LENOVO>docker exec -it spark-container /opt/bitnami/spark/bin/spark-shell --master spark://172.17.0.2:7077
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/11/27 16:01:03 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/11/27 16:01:03 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
Spark context Web UI available at http://5bacb2e17e22:4041
Spark context available as 'sc' (master = spark://172.17.0.2:7077, app id = app-20241127160104-0001).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.3
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 17.0.13)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val initDF = spark.readStream
initDF: org.apache.spark.sql.streaming.DataStreamReader = org.apache.spark.sql.streaming.DataStreamReader@2e40d8f9

scala>   .format("rate")
res0: org.apache.spark.sql.streaming.DataStreamReader = org.apache.spark.sql.streaming.DataStreamReader@2e40d8f9

scala>   .option("rowsPerSecond", 1)
res1: org.apache.spark.sql.streaming.DataStreamReader = org.apache.spark.sql.streaming.DataStreamReader@2e40d8f9

scala>   .load()
res2: org.apache.spark.sql.DataFrame = [timestamp: timestamp, value: bigint]

scala>

scala> println("Streaming DataFrame : " + res2.isStreaming)
Streaming DataFrame : true

scala>
```

4.



```
Command Prompt - docker  exec -it spark-container /opt/bitnami/spark/bin/spark-shell --master spark://172.17.0.2:7077
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/11/27 16:09:01 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
24/11/27 16:09:02 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
24/11/27 16:09:02 WARN Utils: Service 'SparkUI' could not bind on port 4041. Attempting port 4042.
Spark context Web UI available at http://5bacb2e17e22:4042
Spark context available as 'sc' (master = spark://172.17.0.2:7077, app id = app-20241127160903-0002).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.3
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 17.0.13)
Type in expressions to have them evaluated.
Type :help for more information.

scala> import org.apache.spark.sql.functions._
import org.apache.spark.sql.functions._

scala>

scala> val initDF = spark.readStream.format("rate").option("rowsPerSecond", 1).load()
initDF: org.apache.spark.sql.DataFrame = [timestamp: timestamp, value: bigint]

scala>

scala> val resultDF = initDF.withColumn("result", col("value") + lit(1))
resultDF: org.apache.spark.sql.DataFrame = [timestamp: timestamp, value: bigint ... 1 more field]

scala>

scala> val query = resultDF.writeStream.outputMode("append").format("console").start()
24/11/27 16:09:22 WARN ResolveWriteToStream: Temporary checkpoint location created which is deleted normally when the query didn't fail: /tmp/temporary-41aa453a-552b-4a2d-9aec-5d8470c7ea3c. If it's required to d
elete it under any circumstances, please set spark.sql.streaming.forceDeleteTempCheckpointLocation to true. Important to know deleting temp checkpoint folder is best effort.
24/11/27 16:09:22 WARN ResolveWriteToStream: spark.sql.adaptive.enabled is not supported in streaming DataFrames/Datasets and will be disabled.
query: org.apache.spark.sql.streaming.StreamingQuery = org.apache.spark.sql.execution.streaming.StreamingQueryWrapper@15168d6f

scala>

scala> query.awaitTermination()
[Stage 0:>                                                          (0 + 0) / 1]
```

5.



Spark Master at spark://5bacb2e17e22:7077

URL: spark://5bacb2e17e22:7077
Alive Workers: 1
Cores in use: 8 Total, 8 Used
Memory in use: 14.5 GiB Total, 1024.0 MiB Used
Resources in use:
Applications: 1 Running, 3 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory	Resources
worker-20241127155821-172.17.0.2-39579	172.17.0.2:39579	ALIVE	8 (8 Used)	14.5 GiB (1024.0 MiB Used)	

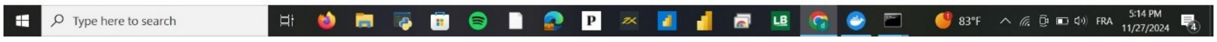
Running Applications (1)

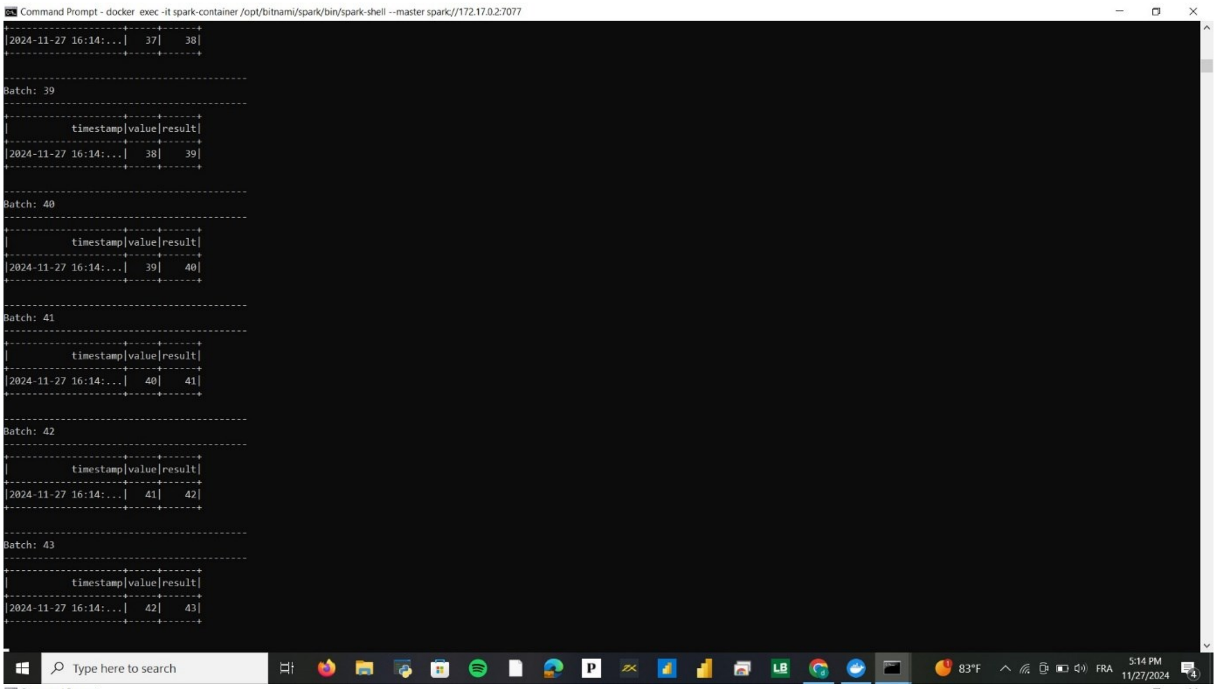
Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20241127161356-0003 (kill)	Spark shell	8	1024.0 MiB		2024/11/27 16:13:56	spark	RUNNING	38 s

Completed Applications (3)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20241127155947-0000	Spark shell	8	1024.0 MiB		2024/11/27 15:59:47	spark	KILLED	13 min
app-20241127160104-0001	Spark shell	0	1024.0 MiB		2024/11/27 16:01:04	spark	KILLED	12 min
app-20241127160903-0002	Spark shell	0	1024.0 MiB		2024/11/27 16:09:03	spark	KILLED	4.0 min

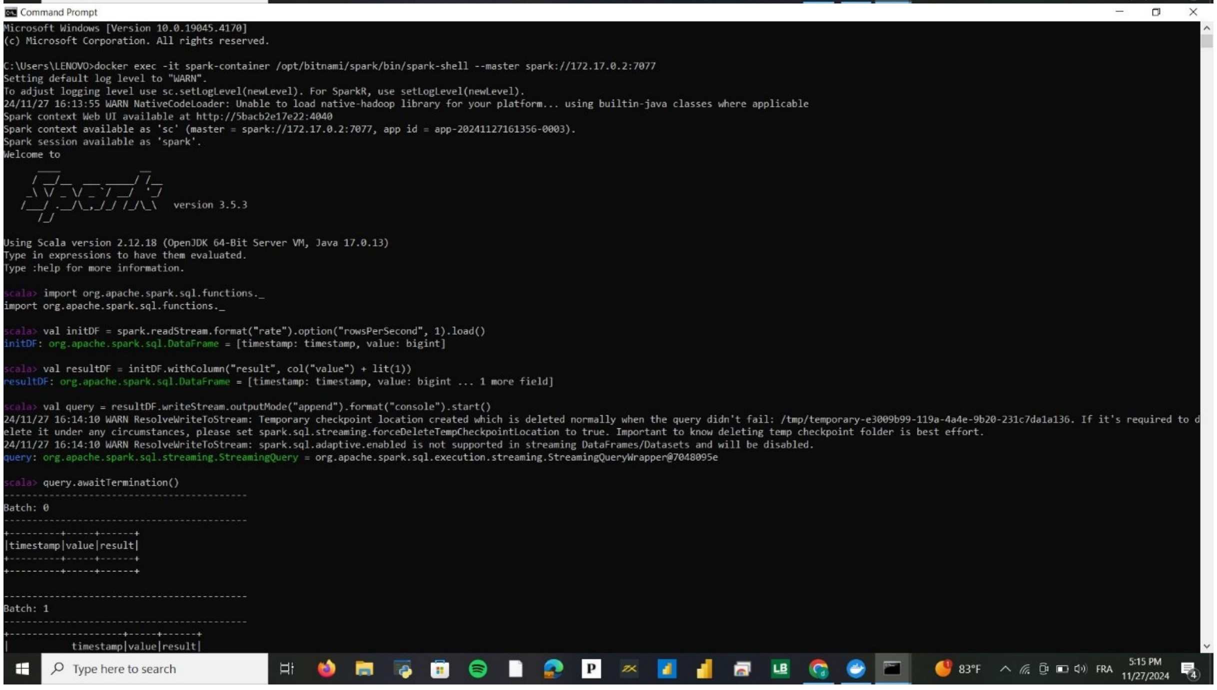
6.





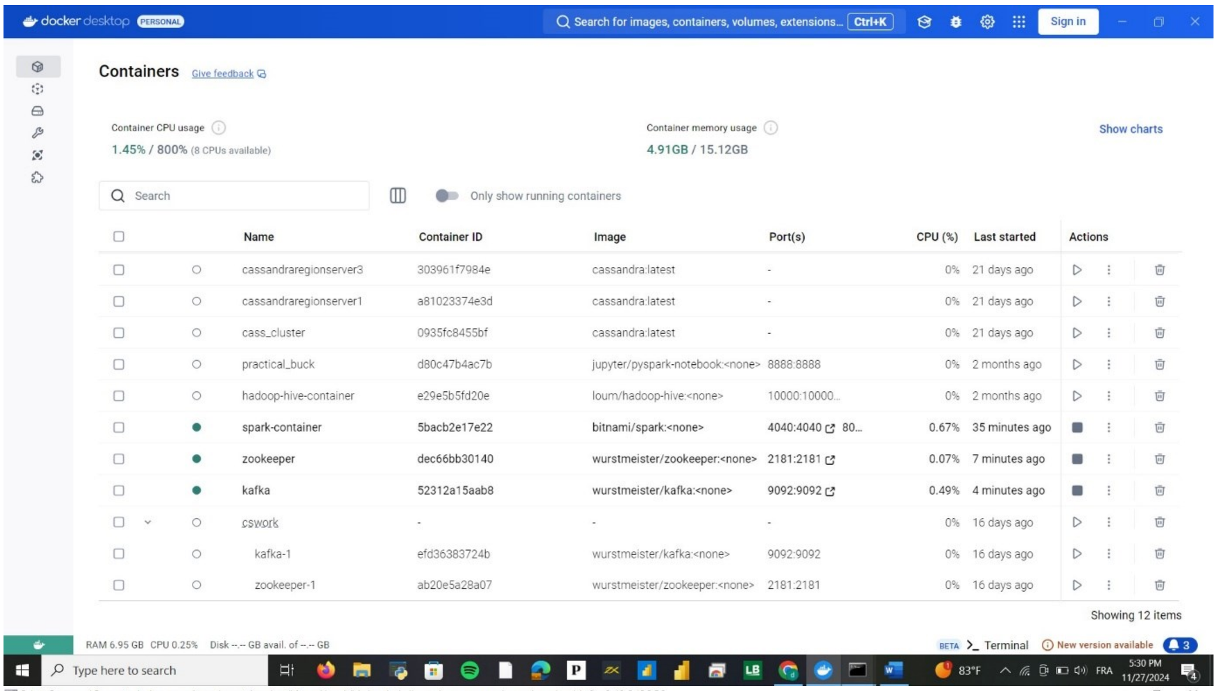
```
Command Prompt - docker  exec -it spark-container /opt/bitnami/spark/bin/spark-shell --master spark://172.17.0.2:7077
+--------------------+-----+------+
|2024-11-27 16:14:...|   37|    38|
+--------------------+-----+------+

-------------------------------------------
Batch: 39
-------------------------------------------
+--------------------+-----+------+
|           timestamp|value|result|
+--------------------+-----+------+
|2024-11-27 16:14:...|   38|    39|
+--------------------+-----+------+

-------------------------------------------
Batch: 40
-------------------------------------------
+--------------------+-----+------+
|           timestamp|value|result|
+--------------------+-----+------+
|2024-11-27 16:14:...|   39|    40|
+--------------------+-----+------+

-------------------------------------------
Batch: 41
-------------------------------------------
+--------------------+-----+------+
|           timestamp|value|result|
+--------------------+-----+------+
|2024-11-27 16:14:...|   40|    41|
+--------------------+-----+------+

-------------------------------------------
Batch: 42
-------------------------------------------
+--------------------+-----+------+
|           timestamp|value|result|
+--------------------+-----+------+
|2024-11-27 16:14:...|   41|    42|
+--------------------+-----+------+

-------------------------------------------
Batch: 43
-------------------------------------------
+--------------------+-----+------+
|           timestamp|value|result|
+--------------------+-----+------+
|2024-11-27 16:14:...|   42|    43|
+--------------------+-----+------+
```

7.



```
Command Prompt
Microsoft Windows [Version 10.0.19045.4170]
(c) Microsoft Corporation. All rights reserved.

C:\Users\LENOVO>docker exec -it spark-container /opt/bitnami/spark/bin/spark-shell --master spark://172.17.0.2:7077
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
24/11/27 16:13:55 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Spark context Web UI available at http://5bacb2e17e22:4040
Spark context available as 'sc' (master = spark://172.17.0.2:7077, app id = app-20241127161356-0003).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.3
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 17.0.13)
Type in expressions to have them evaluated.
Type :help for more information.

scala> import org.apache.spark.sql.functions._
import org.apache.spark.sql.functions._

scala> val initDF = spark.readStream.format("rate").option("rowsPerSecond", 1).load()
initDF: org.apache.spark.sql.DataFrame = [timestamp: timestamp, value: bigint]

scala> val resultDF = initDF.withColumn("result", col("value") + lit(1))
resultDF: org.apache.spark.sql.DataFrame = [timestamp: timestamp, value: bigint ... 1 more field]

scala> val query = resultDF.writeStream.outputMode("append").format("console").start()
24/11/27 16:14:10 WARN ResolveWriteToStream: Temporary checkpoint location created which is deleted normally when the query didn't fail: /tmp/temporary-e3009b99-119a-4a4e-9b20-231c7da1a136. If it's required to d
elete it under any circumstances, please set spark.sql.streaming.forceDeleteTempCheckpointLocation to true. Important to know deleting temp checkpoint folder is best effort.
24/11/27 16:14:10 WARN ResolveWriteToStream: spark.sql.adaptive.enabled is not supported in streaming DataFrames/Datasets and will be disabled.
query: org.apache.spark.sql.streaming.StreamingQuery = org.apache.spark.sql.execution.streaming.StreamingQueryWrapper@7048095e

scala> query.awaitTermination()
-------------------------------------------
Batch: 0
-------------------------------------------
+---------+-----+------+
|timestamp|value|result|
+---------+-----+------+
+---------+-----+------+

-------------------------------------------
Batch: 1
-------------------------------------------
+--------------------+-----+------+
|           timestamp|value|result|
```

8.



9.

docker desktop PERSONAL

Containers Give feedback

Container CPU usage 1.45% / 800% (8 CPUs available)

Container memory usage 4.91GB / 15.12GB

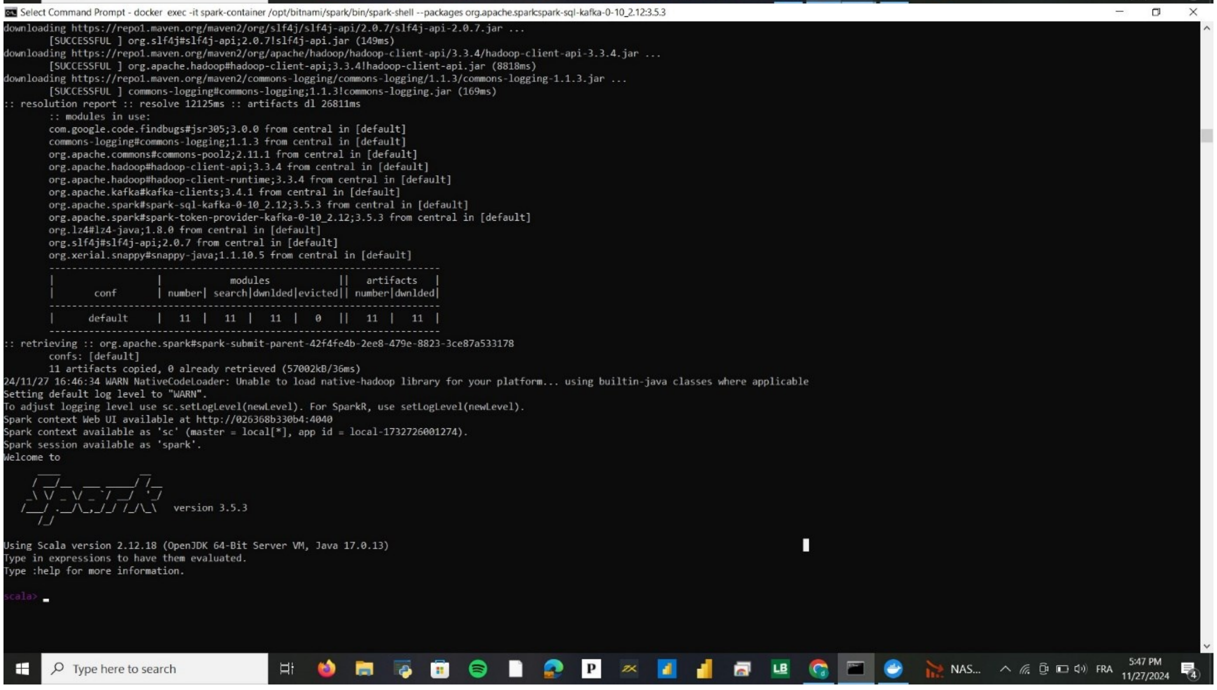
Show charts

Only show running containers

Name	Container ID	Image	Port(s)	CPU (%)	Last started
cassandraregionserver3	303961f7984e	cassandra:latest	-	0%	21 days ago
cassandraregionserver1	a81023374e3d	cassandra:latest	-	0%	21 days ago
cass_cluster	0935fc8455bf	cassandra:latest	-	0%	21 days ago
practical_buck	d80c47b4ac7b	jupyter/pyspark-notebook:<none>	8888:8888	0%	2 months ago
hadoop-hive-container	e29e5b5fd20e	loum/hadoop-hive:<none>	10000:10000...	0%	2 months ago
spark-container	5bacb2e17e22	bitnami/spark:<none>	4040:4040 80...	0.67%	35 minutes ago
zookeeper	dec66bb30140	wurstmeister/zookeeper:<none>	2181:2181	0.07%	7 minutes ago
kafka	52312a15aab8	wurstmeister/kafka:<none>	9092:9092	0.49%	4 minutes ago
cswork	-	-	-	0%	16 days ago
kafka-1	efd36383724b	wurstmeister/kafka:<none>	9092:9092	0%	16 days ago
zookeeper-1	ab20e5a28a07	wurstmeister/zookeeper:<none>	2181:2181	0%	16 days ago

Showing 12 items

RAM 6.95 GB CPU 0.25% Disk --.-- GB avail. of --.-- GB



10.

Select Command Prompt - docker exec -it spark-container /opt/bitnami/spark/bin/spark-shell --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.5.3

```
downloading https://repo1.maven.org/maven2/org/slf4j/slf4j-api/2.0.7/slf4j-api-2.0.7.jar ...
        [SUCCESSFUL ] org.slf4j#slf4j-api;2.0.7!slf4j-api.jar (149ms)
downloading https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-client-api/3.3.4/hadoop-client-api-3.3.4.jar ...
        [SUCCESSFUL ] org.apache.hadoop#hadoop-client-api;3.3.4!hadoop-client-api.jar (8818ms)
downloading https://repo1.maven.org/maven2/commons-logging/commons-logging/1.1.3/commons-logging-1.1.3.jar ...
        [SUCCESSFUL ] commons-logging#commons-logging;1.1.3!commons-logging.jar (169ms)
:: resolution report :: resolve 12125ms :: artifacts dl 26811ms
        :: modules in use:
        com.google.code.findbugs#jsr305;3.0.0 from central in [default]
        commons-logging#commons-logging;1.1.3 from central in [default]
        org.apache.commons#commons-pool2;2.11.1 from central in [default]
        org.apache.hadoop#hadoop-client-api;3.3.4 from central in [default]
        org.apache.hadoop#hadoop-client-runtime;3.3.4 from central in [default]
        org.apache.kafka#kafka-clients;3.4.1 from central in [default]
        org.apache.spark#spark-sql-kafka-0-10_2.12;3.5.3 from central in [default]
        org.apache.spark#spark-token-provider-kafka-0-10_2.12;3.5.3 from central in [default]
        org.lz4#lz4-java;1.8.0 from central in [default]
        org.slf4j#slf4j-api;2.0.7 from central in [default]
        org.xerial.snappy#snappy-java;1.1.10.5 from central in [default]
        ---------------------------------------------------------------------
        |                  |            modules            ||   artifacts   |
        |       conf       | number| search|dwnlded|evicted|| number|dwnlded|
        ---------------------------------------------------------------------
        |      default     |   11  |   11  |   11  |   0   ||   11  |   11  |
        ---------------------------------------------------------------------
:: retrieving :: org.apache.spark#spark-submit-parent-42f4fe4b-2ee8-479e-8823-3ce87a533178
        confs: [default]
        11 artifacts copied, 0 already retrieved (57002kB/36ms)
24/11/27 16:46:34 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://026368b330b4:4040
Spark context available as 'sc' (master = local[*], app id = local-1732726001274).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.3
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 17.0.13)
Type in expressions to have them evaluated.
Type :help for more information.

scala> 
```

3. Question 2: Kafka Stream Processing Using Spark

3.1. Implementation Steps

Step 1: Create a DataFrame from Kafka topic

- In Spark shell, create a DataFrame:
- val df = spark.readStream
- .format("kafka")

- .option("kafka.bootstrap.servers", "kafka:9092")
- .option("subscribe", "testTopic")
- .load()

Step 2: **Check the schema of the DataFrame**

- Print the schema:
- df.printSchema()

Step 3: **Process the data**

- Select and cast the value field as a string:
- val processedDf = df.selectExpr("CAST(value AS STRING) as message")

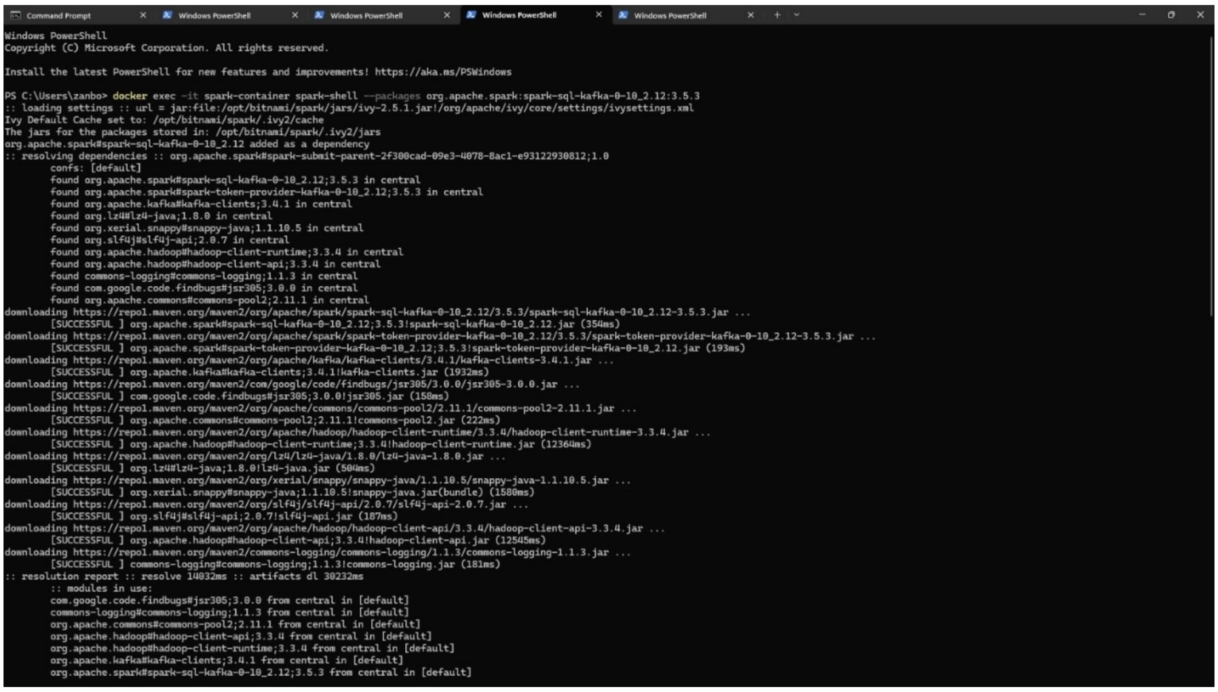
Step 4: **Stream the data to the console**

- Write the processed data to the console:
- val query = processedDf.writeStream
- .outputMode("append")
- .format("console")
- .start()

Step 5: **Keep the query running**

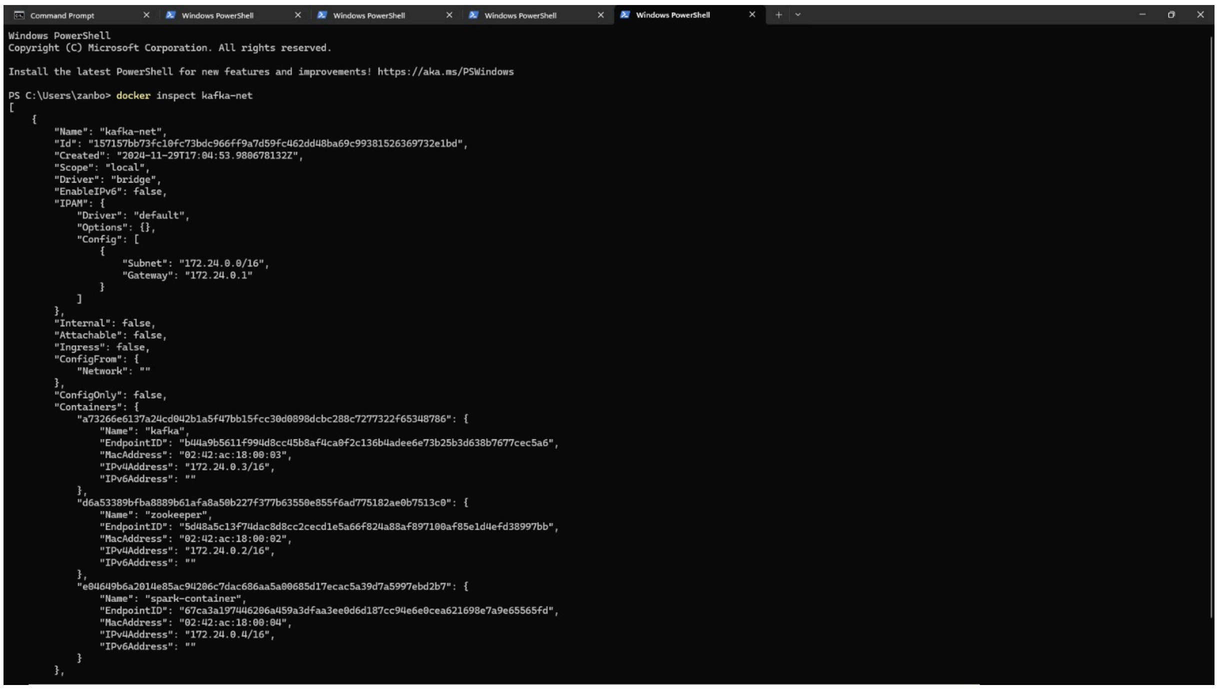
- Run the query:
- query.awaitTermination()

running spark shell



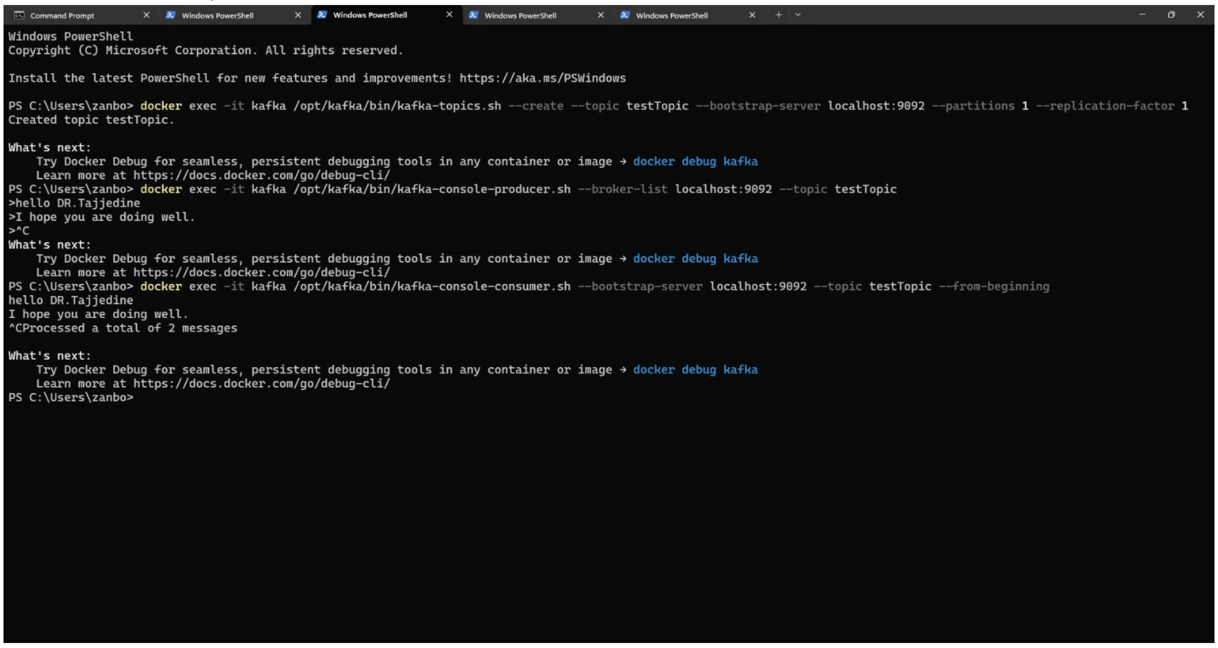
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\zanbo> docker exec -it spark-container spark-shell --packages org.apache.spark:spark-sql-kafka-0-10_2.12:3.5.3
:: loading settings :: url = jar:file:/opt/bitnami/spark/jars/ivy-2.5.1.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /opt/bitnami/spark/.ivy2/cache
The jars for the packages stored in: /opt/bitnami/spark/.ivy2/jars
org.apache.spark#spark-sql-kafka-0-10_2.12 added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-2f300cad-09e3-4078-8ac1-e93122930812;1.0
        confs: [default]
        found org.apache.spark#spark-sql-kafka-0-10_2.12;3.5.3 in central
        found org.apache.spark#spark-token-provider-kafka-0-10_2.12;3.5.3 in central
        found org.apache.kafka#kafka-clients;3.4.1 in central
        found org.lz4#lz4-java;1.8.0 in central
        found org.xerial.snappy#snappy-java;1.1.10.5 in central
        found org.slf4j#slf4j-api;2.0.7 in central
        found org.apache.hadoop#hadoop-client-runtime;3.3.4 in central
        found org.apache.hadoop#hadoop-client-api;3.3.4 in central
        found commons-logging#commons-logging;1.1.3 in central
        found com.google.code.findbugs#jsr305;3.0.0 in central
        found org.apache.commons#commons-pool2;2.11.1 in central
downloading https://repo1.maven.org/maven2/org/apache/spark/spark-sql-kafka-0-10_2.12/3.5.3/spark-sql-kafka-0-10_2.12-3.5.3.jar ...
        [SUCCESSFUL ] org.apache.spark#spark-sql-kafka-0-10_2.12;3.5.3!spark-sql-kafka-0-10_2.12.jar (354ms)
downloading https://repo1.maven.org/maven2/org/apache/spark/spark-token-provider-kafka-0-10_2.12/3.5.3/spark-token-provider-kafka-0-10_2.12-3.5.3.jar ...
        [SUCCESSFUL ] org.apache.spark#spark-token-provider-kafka-0-10_2.12;3.5.3!spark-token-provider-kafka-0-10_2.12.jar (193ms)
downloading https://repo1.maven.org/maven2/org/apache/kafka/kafka-clients/3.4.1/kafka-clients-3.4.1.jar ...
        [SUCCESSFUL ] org.apache.kafka#kafka-clients;3.4.1!kafka-clients.jar (1932ms)
downloading https://repo1.maven.org/maven2/com/google/code/findbugs/jsr305/3.0.0/jsr305-3.0.0.jar ...
        [SUCCESSFUL ] com.google.code.findbugs#jsr305;3.0.0!jsr305.jar (158ms)
downloading https://repo1.maven.org/maven2/org/apache/commons/commons-pool2/2.11.1/commons-pool2-2.11.1.jar ...
        [SUCCESSFUL ] org.apache.commons#commons-pool2;2.11.1!commons-pool2.jar (222ms)
downloading https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-client-runtime/3.3.4/hadoop-client-runtime-3.3.4.jar ...
        [SUCCESSFUL ] org.apache.hadoop#hadoop-client-runtime;3.3.4!hadoop-client-runtime.jar (12364ms)
downloading https://repo1.maven.org/maven2/org/lz4/lz4-java/1.8.0/lz4-java-1.8.0.jar ...
        [SUCCESSFUL ] org.lz4#lz4-java;1.8.0!lz4-java.jar (504ms)
downloading https://repo1.maven.org/maven2/org/xerial/snappy/snappy-java/1.1.10.5/snappy-java-1.1.10.5.jar ...
        [SUCCESSFUL ] org.xerial.snappy#snappy-java;1.1.10.5!snappy-java.jar(bundle) (1580ms)
downloading https://repo1.maven.org/maven2/org/slf4j/slf4j-api/2.0.7/slf4j-api-2.0.7.jar ...
        [SUCCESSFUL ] org.slf4j#slf4j-api;2.0.7!slf4j-api.jar (187ms)
downloading https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-client-api/3.3.4/hadoop-client-api-3.3.4.jar ...
        [SUCCESSFUL ] org.apache.hadoop#hadoop-client-api;3.3.4!hadoop-client-api.jar (12545ms)
downloading https://repo1.maven.org/maven2/commons-logging/commons-logging/1.1.3/commons-logging-1.1.3.jar ...
        [SUCCESSFUL ] commons-logging#commons-logging;1.1.3!commons-logging.jar (181ms)
:: resolution report :: resolve 14032ms :: artifacts dl 30232ms
        :: modules in use:
        com.google.code.findbugs#jsr305;3.0.0 from central in [default]
        commons-logging#commons-logging;1.1.3 from central in [default]
        org.apache.commons#commons-pool2;2.11.1 from central in [default]
        org.apache.hadoop#hadoop-client-api;3.3.4 from central in [default]
        org.apache.hadoop#hadoop-client-runtime;3.3.4 from central in [default]
        org.apache.kafka#kafka-clients;3.4.1 from central in [default]
        org.apache.spark#spark-sql-kafka-0-10_2.12;3.5.3 from central in [default]
```

Ensuring connection between zookeeper and kafka and spark-container:



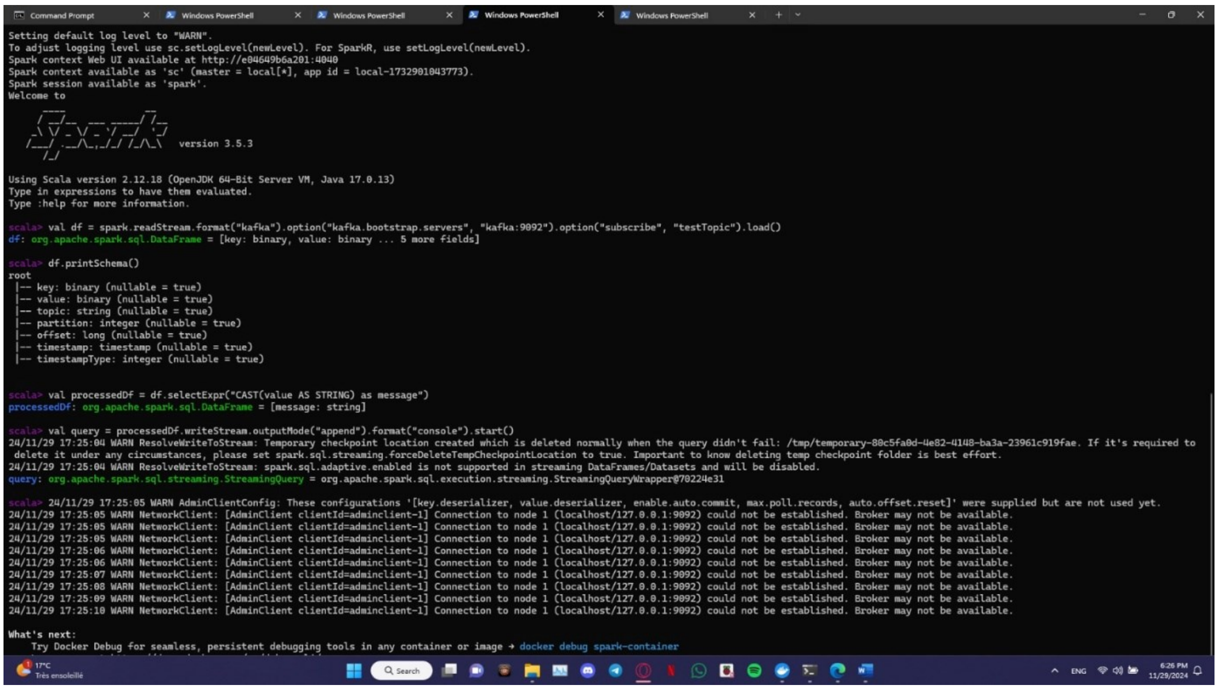
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\zanbo> docker inspect kafka-net
[
    {
        "Name": "kafka-net",
        "Id": "157157bb73fc10fc73bdc966ff9a7d59fc462dd48ba69c99381526369732e1bd",
        "Created": "2024-11-29T17:04:53.980678132Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.24.0.0/16",
                    "Gateway": "172.24.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "a73266e6137a24cd042b1a5f47bb15fcc30d0898dcbc288c7277322f65348786": {
                "Name": "kafka",
                "EndpointID": "b44a9b5611f994d8cc45b8af4ca0f2c136b4adee6e73b25b3d638b7677cec5a6",
                "MacAddress": "02:42:ac:18:00:03",
                "IPv4Address": "172.24.0.3/16",
                "IPv6Address": ""
            },
            "d6a53389bfba8889b61afa8a50b227f377b63550e855f6ad775182ae0b7513c0": {
                "Name": "zookeeper",
                "EndpointID": "5d48a5c13f74dac8d8cc2cecd1e5a66f824a88af897100af85e1d4efd38997bb",
                "MacAddress": "02:42:ac:18:00:02",
                "IPv4Address": "172.24.0.2/16",
                "IPv6Address": ""
            },
            "e04649b6a2014e85ac94206c7dac686aa5a00685d17ecac5a39d7a5997ebd2b7": {
                "Name": "spark-container",
                "EndpointID": "67ca3a197446206a459a3dfaa3ee0d6d187cc94e6e0cea621698e7a9e65565fd",
                "MacAddress": "02:42:ac:18:00:04",
                "IPv4Address": "172.24.0.4/16",
                "IPv6Address": ""
            }
        },
```

Content of the topic:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\zanbo> docker exec -it kafka /opt/kafka/bin/kafka-topics.sh --create --topic testTopic --bootstrap-server localhost:9092 --partitions 1 --replication-factor 1
Created topic testTopic.

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug kafka
    Learn more at https://docs.docker.com/go/debug-cli/
PS C:\Users\zanbo> docker exec -it kafka /opt/kafka/bin/kafka-console-producer.sh --broker-list localhost:9092 --topic testTopic
>hello DR.Tajjedine
>I hope you are doing well.
>^C
What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug kafka
    Learn more at https://docs.docker.com/go/debug-cli/
PS C:\Users\zanbo> docker exec -it kafka /opt/kafka/bin/kafka-console-consumer.sh --bootstrap-server localhost:9092 --topic testTopic --from-beginning
hello DR.Tajjedine
I hope you are doing well.
^CProcessed a total of 2 messages

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug kafka
    Learn more at https://docs.docker.com/go/debug-cli/
PS C:\Users\zanbo>
```

3.2. Streaming Output



```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://e04649b6a201:4040
Spark context available as 'sc' (master = local[*], app id = local-1732901043773).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.3
      /_/

Using Scala version 2.12.18 (OpenJDK 64-Bit Server VM, Java 17.0.13)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val df = spark.readStream.format("kafka").option("kafka.bootstrap.servers", "kafka:9092").option("subscribe", "testTopic").load()
df: org.apache.spark.sql.DataFrame = [key: binary, value: binary ... 5 more fields]

scala> df.printSchema()
root
 |-- key: binary (nullable = true)
 |-- value: binary (nullable = true)
 |-- topic: string (nullable = true)
 |-- partition: integer (nullable = true)
 |-- offset: long (nullable = true)
 |-- timestamp: timestamp (nullable = true)
 |-- timestampType: integer (nullable = true)


scala> val processedDf = df.selectExpr("CAST(value AS STRING) as message")
processedDf: org.apache.spark.sql.DataFrame = [message: string]

scala> val query = processedDf.writeStream.outputMode("append").format("console").start()
24/11/29 17:25:04 WARN ResolveWriteToStream: Temporary checkpoint location created which is deleted normally when the query didn't fail: /tmp/temporary-80c5fa0d-4e82-4148-ba3a-23961c919fae. If it's required to
 delete it under any circumstances, please set spark.sql.streaming.forceDeleteTempCheckpointLocation to true. Important to know deleting temp checkpoint folder is best effort.
24/11/29 17:25:04 WARN ResolveWriteToStream: spark.sql.adaptive.enabled is not supported in streaming DataFrames/Datasets and will be disabled.
query: org.apache.spark.sql.streaming.StreamingQuery = org.apache.spark.sql.execution.streaming.StreamingQueryWrapper@70224e31

scala> 24/11/29 17:25:05 WARN AdminClientConfig: These configurations '[key.deserializer, value.deserializer, enable.auto.commit, max.poll.records, auto.offset.reset]' were supplied but are not used yet.
24/11/29 17:25:05 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:05 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:05 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:06 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:06 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:07 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:08 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:09 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.
24/11/29 17:25:10 WARN NetworkClient: [AdminClient clientId=adminclient-1] Connection to node 1 (localhost/127.0.0.1:9092) could not be established. Broker may not be available.

What's next:
    Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug spark-container
```

The messages produced in Kafka (testTopic) did not appeared in Spark shell as processed output in real-time, we had a problem with the connection between containers I kept searching for solutions I didn't find any unfortunately.

Example Output:

---

Batch: 1

---

message

Hello DR.Tajjedine

---

Batch: 2

---

message

I hope you are doing well.

3.3. Challenges Encountered

1. **Spark Shell Setup Issues**
   - Issue: Errors related to missing Kafka dependencies in Spark.
   - Solution: Used the correct Kafka integration package when launching the Spark shell.
2. **Connectivity Problems**
   - Issue: Initial issues with the Kafka and Spark containers communicating.
   - Solution: Ensured all containers were connected to the same Docker network using:
   - docker network connect kafka-net spark-container but still encountered some problems to stream the output. We didn't know the solution because other groups worked and for us didn't I don't know exactly where is the problem It can be from kafka logs that didn't let the connectivity between the containers.
3. **Schema Verification**
   - Issue: Confusion in identifying the schema of the Kafka data source.
   - Solution: Used df.printSchema() to inspect the schema.

---

4. Conclusion

This assignment demonstrated how Apache Kafka and Apache Spark can work together for real-time data processing. We created a Kafka topic, produced messages, and successfully streamed and processed the data using Spark. This project showcased the efficiency and scalability of distributed streaming systems, highlighting the importance of proper configurations and debugging techniques.

---

5. References

11. Docker Documentation: https://docs.docker.com/
12. Apache Kafka Documentation: https://kafka.apache.org/documentation/
13. Apache Spark Structured Streaming Guide: https://spark.apache.org/docs/latest/structured-streaming-programming-guide.html