EGE UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT
HIGH DATA PROCESSING

High-Performance Analysis of Twitter Data

Term Project Report

05210000261 – Bahrihan Torpil

05220000295 – Altuğ Emre Tosun

…

1. Part 2 Track Selection

This project implements **Part 2 – Track 2A: The Apache Kafka Ecosystem**.
For real-time stream processing, **Apache Kafka Streams** was used to detect negative tweets in real time, while **Kafka Connect (HDFS 3 Sink Connector)** was used to persist streaming data into **HDFS in Avro format**.

---

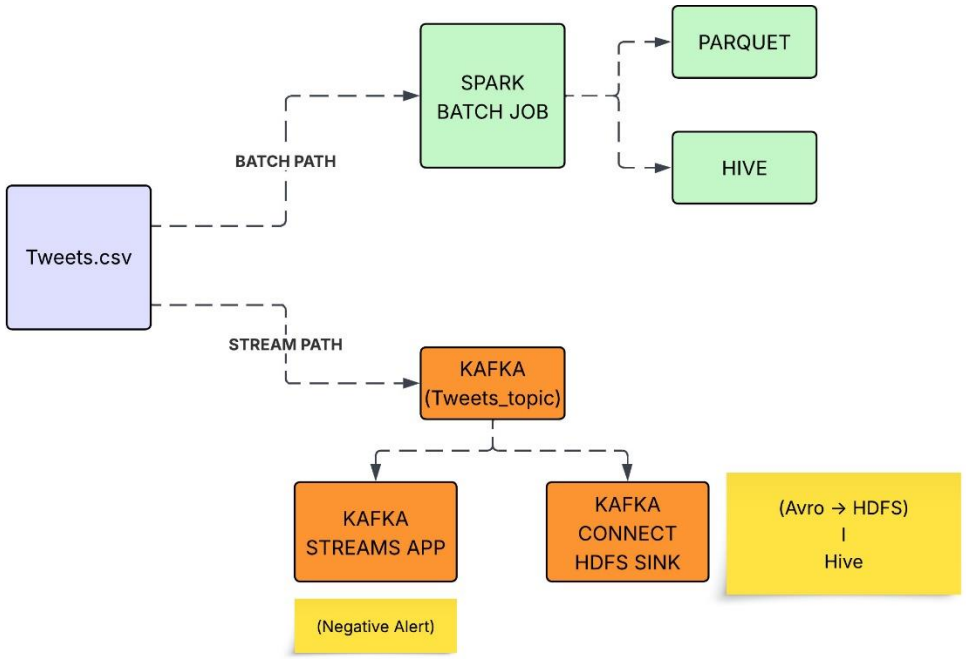
2. System Architecture Overview

The system follows a **Lambda-like architecture** consisting of two parallel processing paths: **Batch Processing** and **Stream Processing**, both unified through **Apache Hive** as a metadata catalog.

High-Level Architecture Components

- **Data Source**: Static Tweets.csv dataset
- **Storage Layer**: HDFS
- **Batch Processing**: Apache Spark
- **Stream Processing**: Apache Kafka + Kafka Streams
- **Stream Persistence**: Kafka Connect HDFS Sink
- **Data Catalog & Query Layer**: Apache Hive

Architectural Flow



This architecture enables both historical analytics and real-time insights using a shared data foundation.

---

3. Project Structure & Deliverables

3.1 Source Code (Deliverable 1)

The submitted .zip file contains the following source code:

Part 1 – Spark Batch Job

- PySpark job that:
  - Reads data from tweets_raw_csv Hive table
  - Aggregates sentiment counts per airline
  - Calculates negative sentiment ratio
  - Writes results as Parquet to HDFS
- Output table: batch_airline_sentiment

Kafka Producer (Stream Simulation)

- **TweetAvroProducer.java**
  - Reads Tweets.csv
  - Converts each row into an Avro record
  - Publishes records to Kafka topic tweets_topic
  - Registers schema in Schema Registry

Part 2 – Kafka Streams Application

- **NegativeTweetAlertApp.java**
  - Consumes Avro messages from tweets_topic
  - Filters tweets with airline_sentiment = "negative"
  - Prints real-time alerts to the console

3.2 Configuration Files (Deliverable 2)

The project includes required configuration files for streaming persistence:

- **Kafka Connect HDFS 3 Sink Configuration**
  - File: tweets-hdfs3-sink.json
  - Writes Avro data from tweets_topic to:
  - /project/streamed_tweets_avro/
  - Uses:
    - Avro format

- Schema Registry
- Time-based partitioning (daily)

---

4. How to Compile and Run the Project

Step 1 – Start Infrastructure

Start all required services using Docker Compose:

```
docker compose up -d
```

Ensure the following services are running:

- **HDFS (NameNode, DataNode)**
- **Kafka**
- **Schema Registry**
- **Kafka Connect**
- **Hive Metastore & HiveServer2**
- **Spark**

Step 2 – Upload Raw Data to HDFS

```
docker exec -it namenode hdfs dfs -put Tweets.csv /project/raw/
```

Step 3 – Create Hive Tables

Using Beeline:

```
CREATE EXTERNAL TABLE tweets_raw_csv (...);
CREATE EXTERNAL TABLE tweets_stream_avro (... PARTITIONED BY (dt
STRING));
CREATE EXTERNAL TABLE batch_airline_sentiment (...);
```

Step 4 – Run Spark Batch Job

```
docker exec -it spark-master \
  /opt/spark/bin/spark-submit \
  /opt/spark_batch_job.py
```

Step 5 – Start Kafka Connect HDFS Sink

```
curl -X POST http://localhost:8083/connectors \
  -H "Content-Type: application/json" \
```

-d @kafka-connect/tweets-hdfs3-sink.json

Step 6 – Start Kafka Streams Application

mvn clean package

mvn exec:java

Step 7 – Start Avro Producer (Stream Simulation)

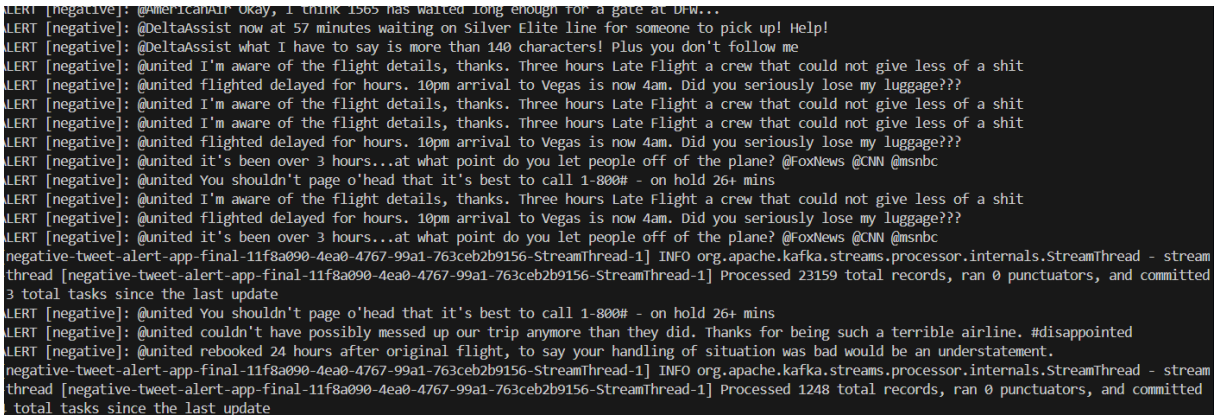
mvn exec:java

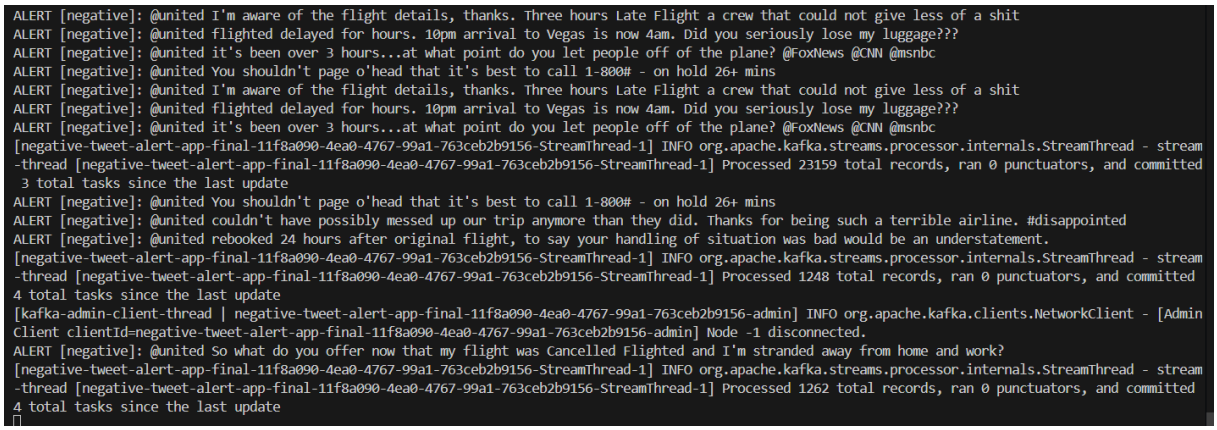
This step replays the dataset as a simulated real-time stream.

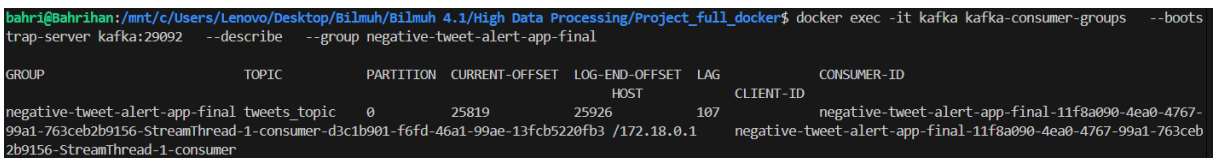
---

5. Results (Deliverable 3 – Screenshots Required)

5.1 Stream Processing Output (Kafka Streams)

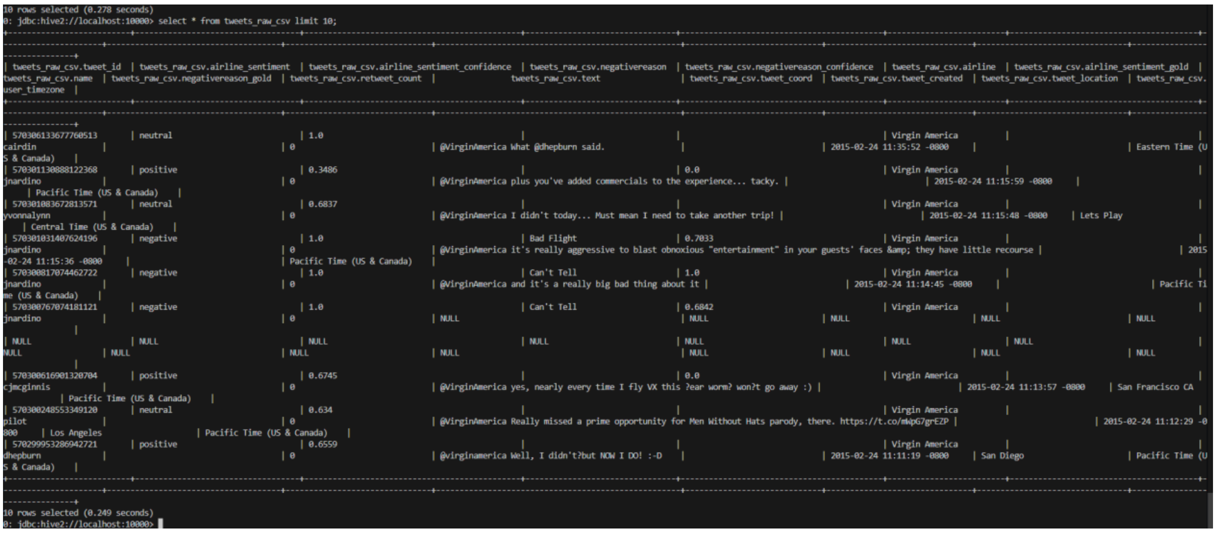






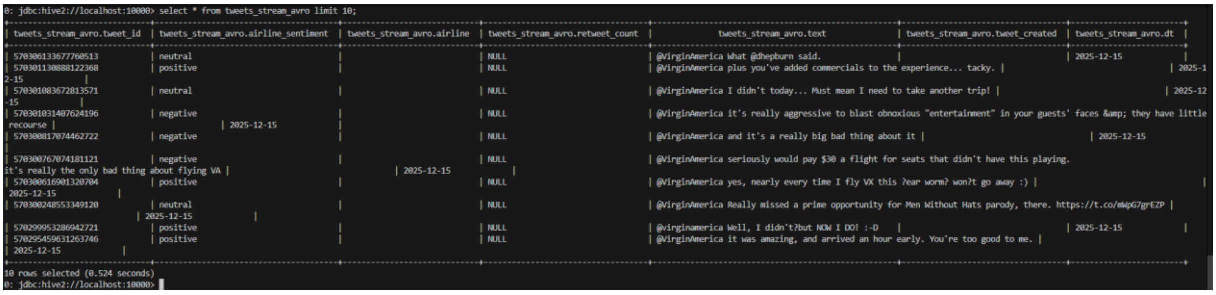
5.2 Hive Query Results

SELECT * FROM tweets_raw_csv LIMIT 10;



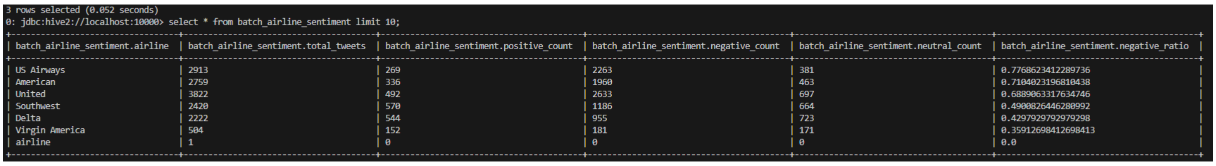
```
10 rows selected (0.278 seconds)
0: jdbc:hive2://localhost:10000> select * from tweets_raw_csv limit 10;
| tweets_raw_csv.tweet_id  | tweets_raw_csv.airline_sentiment  | tweets_raw_csv.airline_sentiment_confidence  | tweets_raw_csv.negativereason  | tweets_raw_csv.negativereason_confidence  | tweets_raw_csv.airline  | tweets_raw_csv.airline_sentiment_gold  |
tweets_raw_csv.name  | tweets_raw_csv.negativereason_gold  | tweets_raw_csv.retweet_count  |  tweets_raw_csv.text  | tweets_raw_csv.tweet_coord  | tweets_raw_csv.tweet_created  | tweets_raw_csv.tweet_location  | tweets_raw_csv.
user_timezone  |
| 570306133677760513  | neutral  | 1.0  |  |  | Virgin America  |  |
cairdin  |  | 0  | @VirginAmerica What @dhepburn said.  |  | 2015-02-24 11:35:52 -0800  |  | Eastern Time (U
S & Canada)  |
| 570301130888122368  | positive  | 0.3486  |  | 0.0  | Virgin America  |  |
jnardino  |  | 0  | @VirginAmerica plus you've added commercials to the experience... tacky. |  | 2015-02-24 11:15:59 -0800  |
  | Pacific Time (US & Canada)  |
| 570301083672813571  | neutral  | 0.6837  |  |  | Virgin America  |  |
yvonnalynn  |  | 0  | @VirginAmerica I didn't today... Must mean I need to take another trip! |  | 2015-02-24 11:15:48 -0800  | Lets Play
  | Central Time (US & Canada)  |
| 570301031407624196  | negative  | 1.0  | Bad Flight  | 0.7033  | Virgin America  |  |
jnardino  |  | 0  | @VirginAmerica it's really aggressive to blast obnoxious "entertainment" in your guests' faces &amp; they have little recourse |  | 2015
-02-24 11:15:36 -0800  |  | Pacific Time (US & Canada)  |
| 570300817074462722  | negative  | 1.0  | Can't Tell  | 1.0  | Virgin America  |  |
jnardino  |  | 0  | @VirginAmerica and it's a really big bad thing about it |  | 2015-02-24 11:14:45 -0800  |  | Pacific Ti
me (US & Canada)  |
| 570300767074181121  | negative  | 1.0  | Can't Tell  | 0.6842  | Virgin America  |  |
jnardino  |  | 0  | NULL  | NULL  | NULL  | NULL  | NULL
  |
| NULL  | NULL  | NULL  | NULL  | NULL  | NULL  | NULL  |
NULL  | NULL  | NULL  | NULL  | NULL  | NULL  | NULL  | NULL
  |
| 570300616901320704  | positive  | 0.6745  |  | 0.0  | Virgin America  |  |
cjmcginnis  |  | 0  | @VirginAmerica yes, nearly every time I fly VX this ?ear worm? won?t go away :) |  | 2015-02-24 11:13:57 -0800  | San Francisco CA
  | Pacific Time (US & Canada)  |
| 570300248553349120  | neutral  | 0.634  |  |  | Virgin America  |  |
pilot  |  | 0  | @VirginAmerica Really missed a prime opportunity for Men Without Hats parody, there. https://t.co/mWpG7grEZP |  | 2015-02-24 11:12:29 -0
800  | Los Angeles  | Pacific Time (US & Canada)  |
| 570299953286942721  | positive  | 0.6559  |  |  | Virgin America  |  |
dhepburn  |  | 0  | @virginamerica Well, I didn't?but NOW I DO! :-D  |  | 2015-02-24 11:11:19 -0800  | San Diego  | Pacific Time (U
S & Canada)  |
10 rows selected (0.249 seconds)
0: jdbc:hive2://localhost:10000>
```

SELECT * FROM tweets_stream_avro LIMIT 10;



```
0: jdbc:hive2://localhost:10000> select * from tweets_stream_avro limit 10;
| tweets_stream_avro.tweet_id  | tweets_stream_avro.airline_sentiment  | tweets_stream_avro.airline  | tweets_stream_avro.retweet_count  |  tweets_stream_avro.text  | tweets_stream_avro.tweet_created  | tweets_stream_avro.dt  |
| 570306133677760513  | neutral  |  | NULL  | @VirginAmerica What @dhepburn said.  |  | 2025-12-15  |
| 570301130888122368  | positive  |  | NULL  | @VirginAmerica plus you've added commercials to the experience... tacky. |  | 2025-1
2-15  |
| 570301083672813571  | neutral  |  | NULL  | @VirginAmerica I didn't today... Must mean I need to take another trip! |  | 2025-12
-15  |
| 570301031407624196  | negative  |  | NULL  | @VirginAmerica it's really aggressive to blast obnoxious "entertainment" in your guests' faces &amp; they have little
 recourse |  | 2025-12-15  |
| 570300817074462722  | negative  |  | NULL  | @VirginAmerica and it's a really big bad thing about it |  | 2025-12-15
|
| 570300767074181121  | negative  |  | NULL  | @VirginAmerica seriously would pay $30 a flight for seats that didn't have this playing.
it's really the only bad thing about flying VA |  | 2025-12-15  |
| 570300616901320704  | positive  |  | NULL  | @VirginAmerica yes, nearly every time I fly VX this ?ear worm? won?t go away :) |  |
 2025-12-15  |
| 570300248553349120  | neutral  |  | NULL  | @VirginAmerica Really missed a prime opportunity for Men Without Hats parody, there. https://t.co/mWpG7grEZP |
  | 2025-12-15  |
| 570299953286942721  | positive  |  | NULL  | @virginamerica Well, I didn't?but NOW I DO! :-D  |  | 2025-12-15  |
| 570295459631263746  | positive  |  | NULL  | @VirginAmerica it was amazing, and arrived an hour early. You're too good to me. |
| 2025-12-15  |
10 rows selected (0.524 seconds)
0: jdbc:hive2://localhost:10000>
```

SELECT * FROM batch_airline_sentiment;



3 rows selected (0.052 seconds)
0: jdbc:hive2://localhost:10000> select * from batch_airline_sentiment limit 10;

batch_airline_sentiment.airline	batch_airline_sentiment.total_tweets	batch_airline_sentiment.positive_count	batch_airline_sentiment.negative_count	batch_airline_sentiment.neutral_count	batch_airline_sentiment.negative_ratio
US Airways	2913	269	2263	381	0.7768623412289736
American	2759	336	1960	463	0.7104023196810438
United	3822	492	2633	697	0.6889063317634746
Southwest	2420	570	1186	664	0.4900826446280992
Delta	2222	544	955	723	0.4297929792979298
Virgin America	504	152	181	171	0.35912698412698413
airline	1	0	0	0	0.0

6. Discussion

This project involved integrating multiple distributed data processing components, each with its own configuration requirements and operational constraints. While the overall architecture worked as intended, several non-trivial challenges were encountered during development and integration.

6.1 Configuration and Serialization Challenges

One of the most significant challenges was ensuring **serialization compatibility across Kafka Producers, Kafka Streams, Kafka Connect, and Hive**.

Initially, a JSON-based Kafka producer was used, which worked correctly with Kafka Streams for real-time alerting. However, when Kafka Connect HDFS Sink was introduced, it required **Avro-formatted messages** along with a properly configured **Schema Registry**. Mixing JSON and Avro producers on the same Kafka topic resulted in errors such as:

- Unknown magic byte
- Avro deserialization failures in Kafka Connect tasks

This issue required:

- Recreating the Kafka topic
- Ensuring that **only Avro producers** published to the topic
- Verifying schema registration via the Schema Registry

This highlighted the importance of **data format consistency** in stream-processing pipelines.

6.2 Kafka Connect and HDFS Integration Issues

Another major challenge was configuring **Kafka Connect HDFS 3 Sink Connector** correctly.

Key difficulties included:

- Correctly setting hdfs.url to use the internal Docker network (hdfs://namenode:9000)
- Ensuring Kafka Connect had write permissions on HDFS directories
- Handling connector task failures caused by schema or serialization mismatches

Additionally, Kafka topics could not always be deleted immediately due to:

- Kafka's asynchronous topic deletion behavior
- Active consumer groups or connector offsets still referencing the topic

This required:

- Explicit verification of connector and consumer shutdown
- Waiting for Kafka's background deletion process to complete
- Recreating topics cleanly before restarting the pipeline

6.3 Hive External Table and Partition Management Problems

Hive integration introduced its own set of challenges, particularly when querying **streamed Avro data**.

Kafka Connect writes data into **time-based directory structures** such as:
/project/streamed_tweets_avro/tweets_topic/2025-12-15/
However, Hive expects partitioned tables to follow the key=value directory naming convention (e.g., dt=2025-12-15).
As a result:

- MSCK REPAIR TABLE initially failed to discover partitions
- SHOW PARTITIONS returned empty results
- Queries returned no rows even though Avro files existed in HDFS

This issue was resolved by:

- Defining the Hive table as **partitioned**
- Pointing the table location correctly
- Allowing Hive to infer partitions once the correct directory structure was recognized

This emphasized the importance of understanding how **Hive metadata discovery works in relation to external tools like Kafka Connect**.

6.4 Hive Repository and Execution Engine Limitations

Another limitation encountered was related to Hive configuration itself:

- Certain configuration parameters (e.g., recursive directory reading) were unavailable or unsupported in the deployed Hive version.
- Hive-on-MR warnings appeared, indicating deprecated execution engines.

Despite these warnings, Hive functioned correctly as a **metadata catalog and SQL query layer**, which was its intended role in this project. No advanced Hive processing was required beyond external table definitions and basic SELECT queries.

6.5 Spark and HDFS Permission Issues

During batch processing, Spark initially failed to write output data to HDFS due to permission errors:

- Spark ran under the spark user
- HDFS directories were owned by root

This caused write failures when saving Parquet results.
The issue was resolved by adjusting HDFS directory permissions to allow group or public write access.

This highlighted the importance of:

- User and permission management in distributed file systems
- Aligning Spark execution context with HDFS access policies

6.6 Architectural Trade-offs and Track Selection

Track 2A (Kafka Streams + Kafka Connect) was chosen over Apache Flink for several reasons:

- Clear separation between **real-time processing** and **data persistence**
- Simpler operational model for small-to-medium streaming workloads
- Strong ecosystem integration with Kafka, Schema Registry, and HDFS

Kafka Streams provided an efficient and lightweight solution for message-level processing and alerting, while Kafka Connect handled reliable, scalable data persistence without embedding file-system logic inside the stream processor.

6.7 Overall Evaluation

Despite the challenges, the final system achieved:

- Correct batch analytics using Apache Spark
- Real-time negative sentiment alerting via Kafka Streams
- Reliable streaming data persistence in Avro format via Kafka Connect
- Unified SQL access through Apache Hive

The encountered issues were representative of **real-world data engineering problems**, particularly around configuration management, schema evolution, and cross-system compatibility.

---

7. Conclusion

This project successfully demonstrates a **high-performance, two-path data processing pipeline** combining batch analytics and real-time stream processing. Using Apache Spark, Kafka, Kafka Streams, Kafka Connect, HDFS, and Hive, the system delivers both historical insights and live sentiment alerts over the same dataset.