# DTS 304 – Big Data Computing Project

## A Report on: Integrated Big Data Pipeline for Social Media Sentiment Analysis using Spark, MongoDB, and PostgreSQL

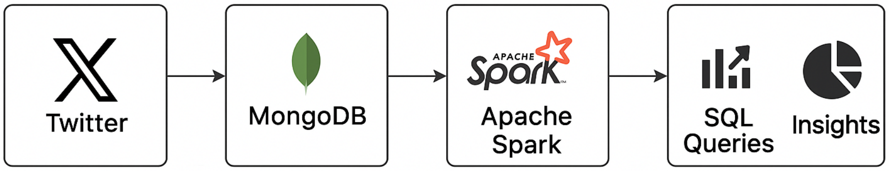
**Group Name**: 12 Nodes of Insight

**Member's Student**:

| ID | Name | Email |
|---|---|---|
| 30001429 | Amanda Gana | amanda.gana@miva.edu.ng |
| 30002640 | Christiana Richards | christiana.richards@miva.edu.ng |
| 30000799 | Comfort Sophia Iwo Odike | comfort.odike@miva.edu.ng |
| 30010860 | King Richard | king.richard@miva.edu.ng |
| 30080232 | Aduragbemi Kinoshi | kinoshi.aduragbemi@miva.edu.ng |
| 30012194 | Margaret Oliver | Margaret.oliver@miva.edu.ng |
| 30002898 | Ojo Ilesanmi | ojo.ilesanmi@miva.edu.ng |
| 30000733 | Okon Enang | okon.enang@miva.edu.ng |
| 30025749 | Olumide Akinboyewa | olumide.akinboyewa@miva.edu.ng |
| 30062596 | Solomon Ayuba | solomon.ayuba@miva.edu.ng |
| 30006647 | Susan Ogidan | susan.ogidan@miva.edu.ng |
| 30078164 | Yomi Aledare | yomi.aledare@miva.edu.ng |

**Date of Submission**: Wednesday, 13th August 2025

**Project Repository**: GitHub



# 1. Introduction

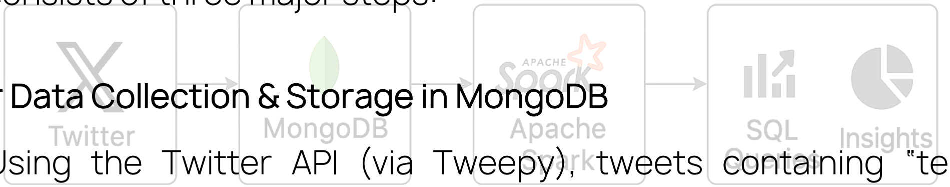
This project implements a mini big data pipeline that collects, stores, processes, analyzes, and queries real-time Twitter data. The system integrates three core technologies:

- **MongoDB** for raw tweet storage (NoSQL database)
- **Apache Spark** for large-scale data cleaning and sentiment classification
- **PostgreSQL** for structured storage and relational queries

The chosen theme for data collection was **"technology"** and **"climate"**, both relevant and trending topics.

# 2. System Architecture

The pipeline consists of three major steps:



1. **Twitter Data Collection & Storage in MongoDB**
   - Using the Twitter API (via Tweepy), tweets containing "technology" or "climate" are fetched in real-time and stored in MongoDB.
   - Raw data is preserved in JSON format for flexibility in downstream processing.
2. **Data Processing & Sentiment Analysis with Apache Spark**
   - Tweets are loaded from MongoDB into Spark.
   - Text is cleaned by removing URLs, mentions, hashtags, special characters, and extra spaces.
   - Sentiment classification is applied using an opinion lexicon (positive/negative word lists).
   - Final output is a structured dataset with a *sentiment* column.
3. **PostgreSQL Integration & Analytical Queries**
   - The processed dataset is written into a PostgreSQL table.
   - SQL queries generate insights such as sentiment distribution, most active users, and time-based sentiment trends.

# 3. Step-by-Step Implementation

## 3.1 Step 1 – Twitter Data Collection (Tweepy + MongoDB)

**Objective:** To acquire real-time tweets based on keywords and store them for processing.

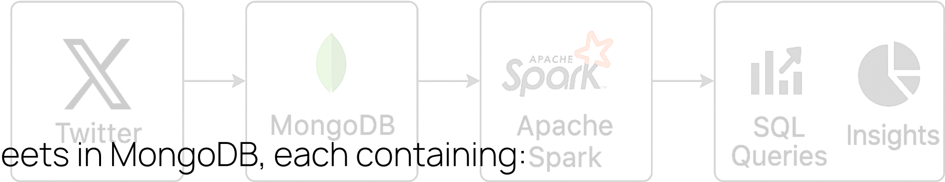
Key Actions:

- Connected to Twitter API v2 using a Bearer Token.
- Search query: *technology OR climate -is:retweet lang:en*

  (retrieves English tweets about technology or climate, excluding retweets).

- Stored tweets in a JSON file (*technology_climate_tweets.json*) and inserted them into *tweet_db.tweets* in MongoDB.

Output:



- ~80 tweets in MongoDB, each containing:
  - Tweet ID, text, author ID, and creation timestamp.
- Raw dataset available for Spark processing.

## 3.2 Step 2 - Data Processing & Sentiment Classification (Apache Spark)

**Objective:** To clean tweet text and label each tweet with a sentiment category.

Key Actions:

- Loaded data from MongoDB into Spark using the MongoDB Spark Connector.
- Applied regular expression rules to:
  - Remove URLs, mentions, hashtags.
  - Remove special characters and newline breaks
  - Trim leading/trailing spaces.

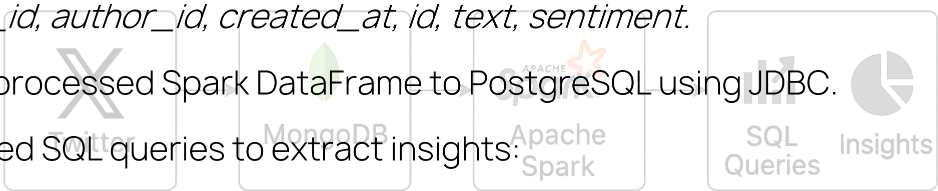
- Implemented **opinion lexicon-based sentiment analysis**:
  - Used predefined positive and negative word lists.
  - Assigned sentiment:
    - **Positive**: contains positive words.
    - **Negative**: contains negative words.
    - **Neutral**: contains neither.
- Output: Spark DataFrame with a new **sentiment** column.

### 3.3 Step 3 - PostgreSQL Integration & Analysis

**Objective:** To store processed data in PostgreSQL and run analytical queries.

**Key Actions:**

- Created PostgreSQL table *tweets* with fields:
  - *_id, author_id, created_at, id, text, sentiment.*



- Wrote processed Spark DataFrame to PostgreSQL using JDBC.
- Executed SQL queries to extract insights:
  1. Count of tweets per sentiment: measures overall tone.
  2. Top 10 active users: identifies key contributors.
  3. Sentiment distribution over time: trends by day, hour, and minute.

**Insight:**

- Positive tweets dominated the dataset, indicating a generally optimistic discussion.
- The most active user contributed 3 tweets.

# 4. Results & Findings

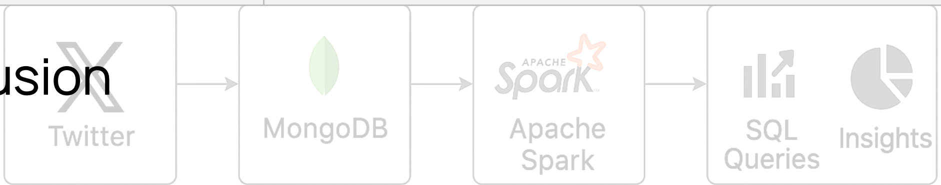
- Successfully implemented a real-time social media pipeline.
- Demonstrated seamless integration of NoSQL (MongoDB), big data processing (Spark), and relational databases (PostgreSQL)

- The system can be adapted for:
    - Political opinion tracking.
    - Brand monitoring.
    - Event-driven sentiment changes.

## 5. Challenges & Solutions

| Challenge | Solution |
|---|---|
| Twitter API rate limits | Limited tweet collection to 80 for demonstration. |
| Large lexicon performance in Spark | Used chunked condition building to avoid performance bottlenecks. |
| Text noise in tweets | Applied regex-based cleaning to remove unwanted elements. |

## 6. Conclusion



This project demonstrated how to build an end-to-end big data pipeline integrating real-time data collection, large-scale processing, and structured storage for analysis. The approach is scalable and adaptable to other domains.

## 7. Future Improvements

- Integrate real-time streaming with Spark Structured Streaming.
- Use machine learning-based sentiment models for higher accuracy.
- Build a dashboard (Power BI / Tableau) for real-time sentiment visualization.
- Expand to multi-language sentiment analysis