



# Real-Time Social Media Sentiment Analysis

Kafka

Spark

MongoDB

Streamlit

MOUTAOUAFFIQ Sidi Mohamed

SEKKAT Amine

LAMLOUM Ayoub

# | Pourquoi ce projet ?



## **Explosion des Données**

Volume massif de messages générés chaque seconde sur les réseaux sociaux. L'analyse statique ne suffit plus.



## **Besoin Temps Réel**

Nécessité de capter la tendance "maintenant". Réagir à l'instant T sur des sujets critiques (tech, politique).



## **Défi Technique**

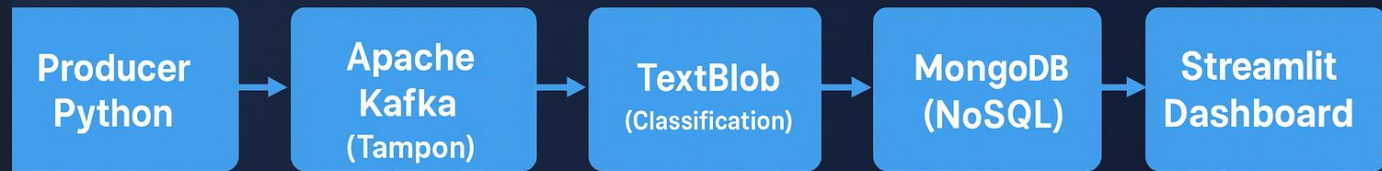
Construire une chaîne Big Data complète : Ingestion, Streaming, NLP, Stockage et Visualisation.

# Architecture End-to-End

- > **Source** : Producer Python (Simulation Faker)
- > **Broker** : Apache Kafka (Tampon)
- > **Processing** : Spark Structured Streaming
- > **NLP** : TextBlob (Classification)
- > **Storage** : MongoDB (NoSQL)
- > **Viz** : Streamlit Dashboard

*Flux continu de gauche à droite.*

## Architecture End-to-End



Flux continu de gauche à droite.

# 1. Ingestion : Génération de Données

## Le "Mock" Twitter

Simulation d'un flux API via un script Python.

- Utilisation de la librairie **Faker**.
- Format **JSON** structuré.
- Topics variés : Tech, Politics, Sport, Music.
- Débit : ~1 message/seconde vers le topic .

`twitter_stream`

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Function to generate Mandelbrot fractal
5 def mandelbrot(c, max_iter):
6     z = c
7     for n in range(max_iter):
8         if abs(z) > 2:
9             return n
10        z = z**2 + c
11    return max_iter
12
13 # Image dimensions
14 width, height = 800, 800
15
16 # Display area parameters
17 re_min, re_max = -2.0, 1.0
18 im_min, im_max = -1.5, 1.5
19
20 # Maximum number of iterations
21 max_iter = 256
```



## 2. Messaging : Apache Kafka

Le système nerveux central de l'architecture.

- **Rôle** : Tampon haute performance pour découpler la production de la consommation.
- **Topic** : `twitter_stream`
- **Avantages** : Résilience aux pannes et scalabilité horizontale.
- Permet d'ajouter d'autres consommateurs futurs sans impacter la prod.

# | 3. Processing : Spark & NLP

## Spark Structured Streaming

Lecture du flux Kafka en temps réel et application du schéma JSON.

### Logique NLP (UDF) :

```
if polarity > 0 → Positif  
if polarity < 0 → Négatif  
else → Neutre
```



## 4. Stockage : MongoDB



### Base NoSQL Orientée Document

Flexibilité totale pour stocker les objets JSON enrichis.

**DB:**

twitter\_db

**Collection:**

sentiments

**Document :** { texte, topic, timestamp, sentiment }

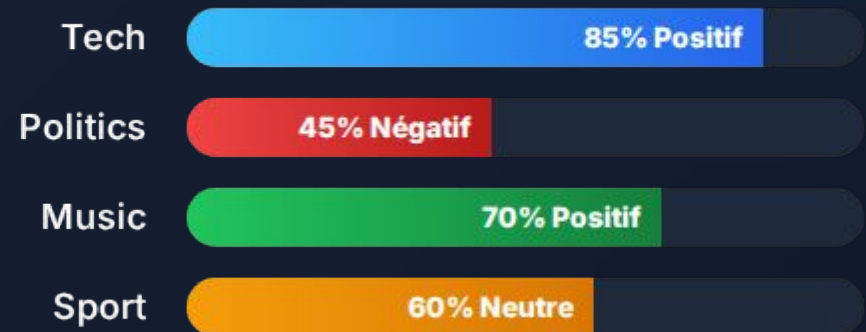
# | 5. Visualisation : Dashboard

## Indicateurs Temps Réel

Connexion directe à MongoDB avec Streamlit  
(rafraîchissement auto).

- > Distribution des sentiments (Pie Chart)
- > Histogrammes par Topic
- > Heatmap : Sentiment vs Topic
- > Feed des derniers tweets

### Analyse par Topic (Live Preview)



## Scénario de Démonstration

1

# Infra

## Docker Compose (Kafka, Zookeeper, Mongo)

2

# Spark

## Lancement du processeur

spark\_processor.py

3

## Source

## Génération des tweets

producer.py

4



# Lancement Dashboard

```
streamlit run app.py
```



# Challenges Techniques

## ❌ Problèmes

**Spark sur Windows** : Erreurs Hadoop et winutils.exe manquants.

**Typage JSON** : Le timestamp était mal interprété (String vs Double).

**Latence Dashboard** : Streamlit rechargeait toute la page à chaque update.

## ✅ Solutions

Configuration stricte des variables d'env  
HADOOP\_HOME.

Définition d'un schéma Spark StructType explicite.

Optimisation des requêtes Mongo et usage des  
caches Streamlit.

# UX & Design du Dashboard

## Objectif : Clarté

Passer de la donnée brute à l'information exploitable.

- > **Aggrégation** : Préférence pour les % plutôt que les valeurs absolues.
- > **Filtrage** : Vues dynamiques par Topic.
- > **Storytelling** : Quel topic domine ? Quelle est l'humeur globale ?
- > Palette de couleurs intuitive (Vert/Rouge).

## Streamlit Dashboard That Looks Like a SaaS Product



# | Limites & Améliorations



## Source de Données

Remplacer Faker par l'API officielle Twitter (X) ou Reddit pour des cas réels.



## NLP Avancé

Remplacer TextBlob (basique) par des modèles Transformers (BERT, CamemBERT).



## Industrialisation

Déploiement Cloud (AWS/Azure) et ajout de monitoring (Prometheus/Grafana).



# Conclusion

---

Réussite de la mise en place d'un pipeline **Big Data complet**.

Maîtrise de l'écosystème .

`Kafka - Spark - Mongo`

Merci de votre attention.