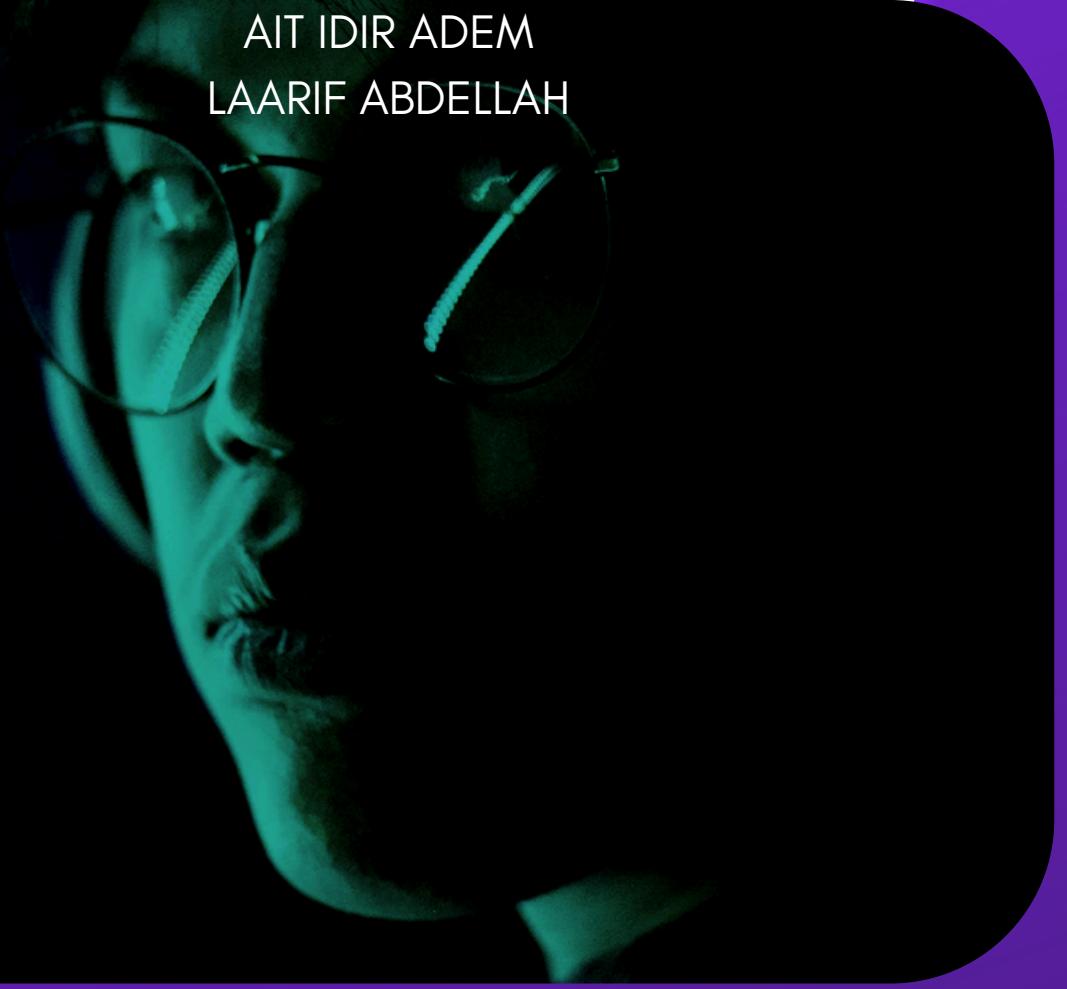


ROQUE LOUIS KINSLEY  
DOROSENCO NADEJDA  
AIT IDIR ADEM  
LAARIF ABDELLAH



TROUVEZ LES MEILLEURS TAGS POUR  
ACCOMPAGNER VOS TIKTOK

# OPTITOK



***POUR QUI?***

INFLUENCEURS

ADOLESCENTS

ARTISTES

MARKETEURS

**Les # les plus  
présents**



TRANSMISSION/ANALYSE/NETTOYAGE  
DES TAGS

CLUSTER

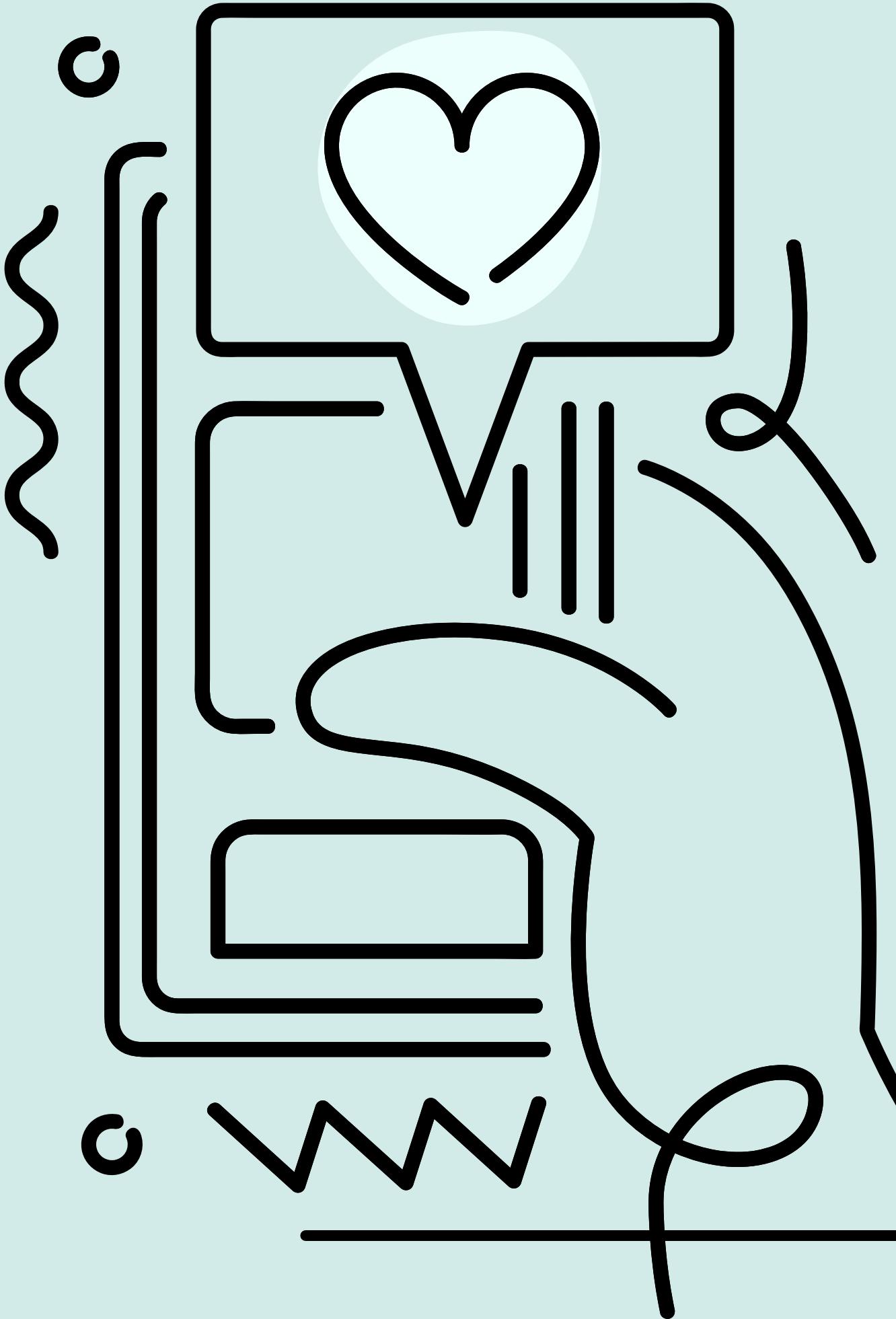
TAG POUR VOIR LE CLUSTER



obtenez le premier  
glossaire de # les plus  
populaires

# OPTITOK

Partie technique



# ÉTAPES PRINCIPALES

---

## EXTRACTION DE DONNEES

- Utilisation de la bibliothèque Apache POI pour lire les données à partir d'un fichier Excel spécifié.
- Création d'une liste de lignes (rows) représentant les enregistrements du fichier Excel.

## ÉCRITURE DANS DES FICHIERS TEXTE

- Utilisation d'un Timer pour planifier l'écriture des données extraites dans des fichiers texte.
- Les données sont extraites en lots de 10 lignes à chaque itération du Timer.
- Les lignes contenant des hashtags sont écrites dans des fichiers texte avec un format approprié.

## CONFIGURATION DE SPARK STREAMING

- Création d'un objet SparkSession pour initialiser l'environnement Spark.
- Création d'un StreamingContext avec un intervalle de lot (batch interval) de 10 secondes.
- Utilisation de textFileStream pour lire les fichiers texte générés en temps réel.

## ANALYSE EN TEMPS RÉEL DES HASHTAGS

- Utilisation de Spark Streaming pour compter le nombre d'occurrences de chaque hashtag.
- Utilisation de updateStateByKey pour maintenir un compteur cumulatif des hashtags.
- Tri des hashtags en fonction de leur compteur et affichage des 10 hashtags les plus populaires à chaque intervalle de lot.

## SUPPRESSION DES WARNINGS

```
// Set the log level to WARN
Logger.getLogger("org").setLevel(Level.ERROR)
Logger.getLogger("akka").setLevel(Level.ERROR)
```

## EXTRACTION DE DONNÉES

```
// Code d'extraction des données
val workbook: Workbook = WorkbookFactory.create(new File("/Users/Nadya/Desktop/ESGI/S2/Spark Streaming/dataset_free-tiktok-scraper_2022-07-27_21-44-20-266.xlsx"))
val sheet: Sheet = workbook.getSheetAt(0)
val rows: Iterator[Row] = sheet.rowIterator().asScala
```

## ÉCRITURE DES HASHTAGS

```
val timer = new Timer()
var count = 0

timer.schedule(new TimerTask {
    override def run() {
        val file = new File(s"/Users/Nadya/Desktop/ESGI/S2/Spark Streaming/streaming-dir/output_$count.txt")
        val writer = new PrintWriter(new FileOutputStream(file, true)) // append to file
        rows.take(10).foreach(row => {
            val cell = row.getCell(47)
            if (cell != null) {
                val hashtags = cell.toString.split(" ").filter(word => word.startsWith("#") && word.length > 1)
                hashtags.foreach(hashtag => writer.write(hashtag + "\n"))
            }
        })
        writer.close()
        count += 1
    }
}, 0, 10000) // 10 seconds
```

CODE POUR ÉCRIRE LES  
HASHTAGS EXTRAITS DANS  
DES FICHIERS TEXTE  
TOUTES LES 10 SECONDES.

## NETTOYAGES DES #VIDE DANS LE FICHIER TXT

```
val hashtags = cell.toString.split(" ").filter(word => word.startsWith("#") && word.length > 1)
```

## SYNONYMES

```
val hashtagSynonyms = Map(  
    "#fyp" -> "#foryoupage",  
    "#foryou" -> "#foryoupage",  
    "#fy"->"#foryoupage",  
    "#foru"->"#foryoupage",  
    "#4u"->"#foryoupage",  
    "fypage"->"#foryoupage",  
    "fypviral"->"#foryoupage",  
    "#humor"->"#comedy",  
    "#fun"->"#funny",  
    "#laugh"->"#funny",  
    "#jokes"->"#prank",  
    "#funnyvideos"->"#funny",  
    "#funnyvideo"->"#funny",  
    "#lol"->"#funny",  
    "#lmao"->"#funny",  
    "#humour"->"#comedy",  
    "#funnymemes"->"#meme",  
    "#memesdaily"->"#meme",  
    "#memestiktok"->"#meme",  
    "#memes"->"#meme",  
    "#animals"->"#animal",  
    "#pets"->"#pet",  
    "#cats"->"#cat",  
    "#catlover"->"#cat",  
    "#dogs" -> "#dog",  
    "#dogslover" -> "#dog",  
    "#pranks"->"prank",  
    "vrgame"->"virtualreality"  
)
```

## NETTOYAGES DES ACCENTS

```
object TextStreamAnalysis {  
    val accentMapping = Map(  
        'é' -> 'e', 'è' -> 'e', 'ê' -> 'e', 'ë' -> 'e',  
        'à' -> 'a', 'â' -> 'a', 'ä' -> 'a',  
        'ù' -> 'u', 'û' -> 'u', 'ö' -> 'o',  
        'ï' -> 'i', 'î' -> 'i',  
        'ö' -> 'o', 'ô' -> 'o'  
        // Ajoutez d'autres mappings si nécessaire  
    )
```

## NETTOYAGES DES CARACTÈRES SPÉCIAUX

```
val alphabet = ('a' to 'z').toSet ++ ('A' to 'Z').toSet ++ ('0' to '9').toSet ++ Set('#')
```

## L'APPLICATION AFFICHE LE TOP 10 DES HASHTAGS LES PLUS UTILISÉS

```
// count des hashtags
val updateCount = (values: Seq[Int], state: Option[Int]) => {
    val currentCount = values.sum
    val previousCount = state.getOrElse(0)
    Some(currentCount + previousCount)
}

// updateStateByKey pour conserver un cumulative count
val cumulativeCounts = hashtagCounts.updateStateByKey(updateCount)

// trie par hashtag count
val sortedCounts = cumulativeCounts.transform(rdd => rdd.sortBy(_._2, ascending = false))

// Affiche le top 10 des hashtags
sortedCounts.foreachRDD { rdd =>
    println("\nTop 10 hashtags:")
    for ((tag, count) <- rdd.take(num = 10)) {
        println(s"$tag: $count")
    }
}
```

POSSIBILITÉ DE RENTRER UN NOUVEAU HASHTAG  
ET VOIR À QUEL CATÉGORIE IL APPARTIENT

```
object KeyboardInput {
    def main(args: Array[String]): Unit = {
        val directoryPath = "/Users/Nadya/Desktop/ESGI/S2/Spark Streaming/streaming-dir"
        while (true) {
            println("Enter something: ")
            val input = readLine()
            input.split(regex = " ").zipWithIndex.foreach { case (hashtag, index) =>
                val pw = new PrintWriter(new File(pathname = s"$directoryPath/file${System.currentTimeMillis()}_$index.txt"))
                pw.write(hashtag + "\n")
                pw.close
            }
        }
    }
}
```

## EXTRACTION EN CSV



DASHBOARD

```
// Collect tous les hashtags et leurs counts
val allHashtagsList = rdd.collect()

val allHashtagsDF = spark.createDataFrame(allHashtagsList).toDF("hashtag", "count")
val csvPath = "/Users/Nadya/Desktop/ESGI/S2/Spark Streaming/all_hashtags"
allHashtagsDF.write.mode("overwrite").csv(csvPath)
```

**METHODE DES K-MEANS  
8 CLUSTER**

FOR YOU PAGE ET  
FUNNY LES TRÈS  
UTILISÉS(2,3)

VIRAL,TIKTOK UTILISÉS

ANIMAL,LOVE(0) PEU  
UTILISÉ

SPORT, TRAVAILE MOIN  
UTILISÉS(5)

SCHOOL,  
INFORMATION,TERROR  
LES TRÉS PEUX UTILISER

ANIMAL,LOVE(0) PEU  
UTILISÉ

