Designing and Implementing a data pipeline to ingest, process and extract real-time insights from a Twitter Stream

This project focuses on simulating a stream of tweets, processing them in real-time through Apache Kafka and Apache Spark Streaming, and finally storing and visualizing the insights derived from the tweets in MongoDB. Students will begin by simulating a dynamic tweet stream from a file, integrating with Kafka for efficient data handling. A Spark Streaming application will be developed to consume and process these tweets, applying relevant transformations. Processed tweets will then be indexed into MongoDB, serving as a scalable and fault-tolerant storage solution. The latter part of the project involves the development of a web application using a framework like Flask or Django. This application will offer a user-friendly interface to visualize insights extracted from the tweet stream, incorporating real-time updates.

Students should follow the following steps:

1. Simulate the arrival of a stream of tweets by reading bunches of them from a file each second where each bunch consists of 100 tweets. You can use this python script to accomplish that:

```
import gzip
import time
i = 1
with gzip.open('[path_of_gzipped_file_of_tweets]','rt') as f:
    for line in f:
        if i%1000!=0:
            line = line.replace("\'", "\"")
            attribute_details = line.split(',')
            tweet = {
                "id":attribute_details[1],
                "date":int(time.time()*1000),
                "user":attribute_details[4],
                "text":attribute_details[5],
                "retweets": int(random.random()*10)
            }

        else:
            sleep(1)
        i = i + 1
```

You can download the file using this link:
https://www.kaggle.com/kazanova/sentiment140/data#

2. Download and run Zookeeper [link]
3. Download and run Apache Kafka [link]

Apache Zookeeper:
https://commandstech.com/how-to-install-zookeeper-on-windows-10-with-pictures/
Apache KafKa:
https://towardsdatascience.com/running-zookeeper-kafka-on-windows-10-14fc70dcc77

4. After running Kafka, create a new topic and name it. [link]
5. Using kafka-python library to create a twitter stream producer. The tweets read in Point-1 should be published to the created topic on Kafka. Use this link to install and use this library by calling the "produce" method in the right place of the script in Point-1
6. Run MongoDB and create a new collection 'tweets'. However, You need to have a field of type "Date" referring to the time on which the tweets were published
7. Use Structured Spark Streaming to read batches of tweets from Kafka. You can benefit from this scala code:

```
import org.apache.log4j.BasicConfigurator
import org.apache.log4j.varia.NullAppender
import org.apache.spark.sql.{SparkSession, functions}
import org.apache.spark.sql.streaming.Trigger

object KafkaConsumer_tweets {

  def main(args: Array[String]): Unit = {

    val nullAppender = new NullAppender
    BasicConfigurator.configure(nullAppender)


    val spark = SparkSession
      .builder
      .config(ConfigurationOptions.ES_NODES, "127.0.0.1")
      .config(ConfigurationOptions.ES_PORT, "9200")
      .master("local[8]")
      .appName("StructuredNetworkTweets")
      .getOrCreate()

    import spark.implicits._

    spark.conf.set("spark.sql.shuffle.partitions", 2)
    val df = spark
      .readStream
      .format("kafka")

      .option("kafka.bootstrap.servers", "localhost:9092")
      .option("subscribe", "[topic_name]")
      .load()
    val df1 = df.selectExpr("CAST(value AS STRING)")
    .select(functions.json_tuple($"value","id","date", "user", "text",
"retweets"))
      .toDF("id","date", "user", "text", "retweets")

    val query = df1.writeStream
      .outputMode("append")
      .format("console")
      .trigger(Trigger.ProcessingTime("5 seconds"))
      .start()


    df1.writeStream
      .outputMode("append")
      .format(".....”)
      .option("checkpointLocation", "[_path_of_checkpointLoc_logs]")
      .start("[index_name]")
      .awaitTermination()
  }
}
```

8. Run the spark consumer to start subscribing and consuming bunches of tweets from Kafka brokers, namely run the script in Point-7
9. Run the Python Script to start simulating the process of generating a stream of tweets, namely, run the script prepared in Point-1
10. Prepare a simple web Application that can connect to MongoDB to retrieve the following information with a suitable refresh rate to have a realtime dashboard:
    a. Create a “Pie chart” that depicts the top 20 users in terms of the number of published tweets.
    b. Prepare a trend chart that shows the distribution of tweets over time (day-wise aggregated) after filtering the tweets based on user query.