Big Data dev challenge

The Big Data Divas (BDD) team is responsible for delivering reports to our customers, who require daily, weekly, and monthly insights on their campaigns.

In this task, you will work with a simplified version of our datasets. Your objective is to create a small data pipeline to process both streaming and non-streaming data, ultimately storing to the file system.

Input data sources

- campaigns.csv static file - This is a CSV file stored on the file system. Each campaign is associated with a network. Assume the file size is approximately 10MB. The schema can be inferred from the attached file.
- `view_log` streaming data - Each record in this dataset corresponds to a view event of a banner, linked to a campaign. Assume the daily volume of view_log data is in terabytes.
  The schema is:
  - `view_id` (STRING): Unique ID of the record
  - `start_timestamp` (TIMESTAMP): Timestamp at which the view started
  - `end_timestamp` (TIMESTAMP): Timestamp at which the view ended (also the timestamp when it was pushed to the Kafka topic)
  - `banner_id` (BIGINT): The banner ID
  - `campaign_id` (INT): The campaign ID

Report to generate

Customers need to know the average view duration (in seconds) and the total number of views per time window (hour, day, week) for their campaigns.

For simplicity in this task assume that:

- the report time window should only be 1 minute
- the maximum delay for a `view_log` message in Kafka is 10 seconds (maximum latency)
- the message retention time is 3 minutes

The report entries for each minute window should be stored to a Parquet file with the following schema:

- `campaign_id` (INT): Campaign ID
- `network_id` (INT): Network ID
- `minute_timestamp` (TIMESTAMP): Timestamp of the minute when the report time window starts
- `avg_duration` (DOUBLE): Average view duration for the given minute time window
- `total_count` (INT): Total view count for the given minute time window

The report files should be separated into different folders by `network_id` and `minute_timestamp`

Task

Your task is to build a small system comprising the following components:

- **Apache Kafka instance:** To stream the view_log data.
- **Simple `view_log` generator (Kafka producer):** To simulate incoming traffic. You can use any tool of your choice.
- **Report generator:** A data processing component to generate the report. The choice of tool/framework is up to you.
- **Storage folder:** A folder on the local file system used to store the reports and the campaigns.csv input file.

Expectations:

- The code should be in a private Github repository with access to our developers (username: `vm-bdd`) If you would like to use another git hosting service, let us know.
- The language preference order is Java, then Python, then Scala
- The solution should be fully dockerized and executable with the `docker-compose up` command.
- The solution should be production-ready (tested, well-documented etc). Imagine you're handing this code to a colleague for code review; it should be CI-ready and deployable.
- If any requirements are unclear and cannot be resolved by making reasonable assumptions, please feel free to email us.