ML Engineer take home test

Summary

The objective of this task is to:

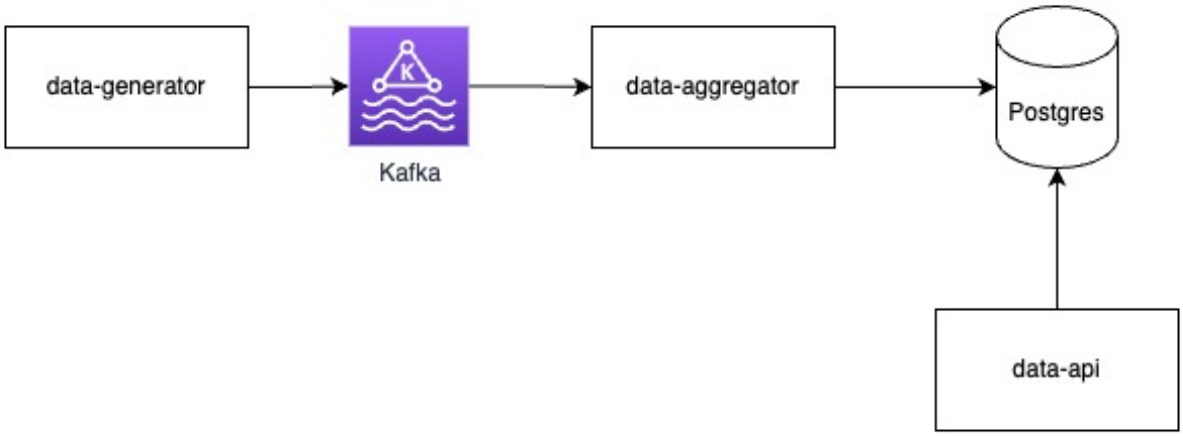
1. Assess your ability to follow a given direction towards a common or known deliverable.
2. Measure the extent to how quickly you learn and adopt new technologies.
3. Measure the acuity of your programming skills.

Problem Description

You are tasked with creating a simple data pipeline, the data pipeline includes:

1. A data generator to generate and stream data.
2. Kafka to stream data between a data generator and a data aggregator.
3. A data aggregator to read from Kafka and persist the information to a database like postgres.
4. A data API which would read from postgres and serve requests.

The overall system would look something like below



Instructions

Create a test configuration, started together with docker-compose tool:

1. **Data generator**:
   Should send a continuous stream of stock ticker data into Kafka topic named “stocks” (10 msgs per sec).
   Data to send are in JSON format:

```
{
    "tickers": [
        {
            "name": "AMZN",
            "price": 1902
        },{
            "name": "MSFT",
            "price": 107
        },{
            "name": "AAPL",
            "price": 215
        }
    ]
}
```

Each message should contain 1-3 stocks, price is integer (generate random integer prices using above provided prices as mean prices +-10%)

2. **Data Aggregator:**
   The container should read from the Kafka topic “stocks”, aggregate ticker price every 30 sec, and persist the average price per stock to a postgres table with a timestamp.
3. **Data API:**
   The API would be responsible for reading the postgres table and serve the requests. The REST API method that is to be implemented is as follows

   {{HOST}}/v1/get-prices
   - The API would return top distinct prices of all tickers if no query params are supplied.
   - There must be a support for query parameter to filter based on stock ticker and time range. Given a time range as startTime and endTime, the API should return all the aggregated data.

Expectation

1. The solution must be fully working using separate docker containers for all the services.
2. A docker-compose file having postgres and kafka is included in the project, this file is to be updated with containers for data-generator, data-aggregator and data-api.
3. A sample python-based placeholders for all the 3 apps is in place. You can choose to either use Python3 or Java to complete the applications. Feel free to use any open-source frameworks to simplify your task.

Furthermore, we expect the code you submit is yours. After you submit a solution, it might be required that you walk us through your solution and any important design details you considered/made.

As a result, please provide:

1. Source code(s) for the data-aggregator, and data-api
2. Docker files for the containers.
3. Docker-compose to run the containers
4. Readme file with any special instructions if needed to run the pipeline.