Sporty Group

Overview

The purpose of the assignment is to assess object oriented analysis and modelling skills, Java coding skills, and code structuring. Take your time on the task, but don't get too carried away. If you submit a solution that is in any way incomplete, the parts that you decided to focus on are relevant.

Keeping the objective in mind, you are free to use whatever tools, libraries, frameworks at your disposal. Artificial Intelligence (AI) usage is encouraged. Please include a `README` in any format on how to run and use the program.

Requirements

Your task is to write a backend application that will simulate sports betting event outcome handling and bet settlement via Kafka and RocketMQ.

- **An API endpoint to publish a sports event outcome to Kafka.**
- **A Kafka consumer that listens to `event-outcomes` Kafka named-topic.**
- **Matches the event outcome to bets that need to be settled.**
- **A RocketMQ producer that sends messages to `bet-settlements`.**

The details of these Use Cases can be found in the next page. There are no further defined requirements for the API, it is up to you to design and implement the necessary code in order for the API to support the mentioned operations above. This refers to the entire flow starting from the API, down to the persistence layer.
It is expected for you to spend around 90 minutes to complete the exercise.

Delivery

- The solution needs to be 100% executable
- Provide a link to the GitHub repository where your solution is committed
    - Please make sure it is not private or restricted
- Provide a `README` file (documentation) how to run and use the solution

Sports Betting Settlement Trigger Service

We want to launch a new backend service that will simulate sports betting event outcome handling and bet settlement via Kafka and RocketMQ. Here you have the relevant Use Cases to support:

1. **An API endpoint to publish a sports event outcome to Kafka.**
    a. This endpoint should publish an event outcome to Kafka. An event outcome is represented by:
        i. Event ID
        ii. Event Name
        iii. Event Winner ID
    b. The topic should be `event-outcomes` as commented before.

2. **A Kafka consumer that listens to `event-outcomes` Kafka named-topic.**

3. **Matches the event outcome to bets that need to be settled.**
    a. The System checks if we have bets in our database that can be settled based on Event ID from the event outcome message.
    b. A bet in database is composed by:
        i. Bet ID
        ii. User ID
        iii. Event ID
        iv. Event Market ID
        v. Event Winner ID
        vi. Bet Amount

4. **A RocketMQ producer that sends messages to `bet-settlements`.**
    a. After identifying the bets to be settled in the previous point, here we should produce the messages for RocketMQ so the bet can be settled.

Continues in the next page …

Conditions:

- If the RocketMQ setup is too complex, use mocks for the RocketMQ producer. Just log the payload.
- Use an in-memory database for the bets.