Interview Task – Senior Java Developer

The Challenge

As the Sportsbook Data Team, we are responsible for providing a REST API for publishing Sporting Event data and serve this data to the frontends. The task will mimic this in a simplified manner.

Data storing service

Build a service that provides a REST API for inserting, updating and deleting Sporting Event data and stores them in a DB of your choice.

Data retrieving service

Build a service that provides a REST API to retrieve Sport Events from the DB with the following endpoints:

- retrieve a list of non-settled Sport Events (without Markets and Outcomes) that can be filtered by Sport, sorted by Start Time
- retrieve a single Sport Event by ID

To improve performance the data should be cached and evicted only when it changes (where applicable) or X minutes after last access.

Design the services with the assumption that they don’t have access to each other.

Sport Event data

We are interested in the following data for each Sport Event:

- id (auto-generated)
- description (e.g.: “Arsenal v Manchester United”)
- home team (e.g.: “Arsenal”)
- away team (e.g.: “Manchester United”)
- start time (e.g.: 31/01/2025 18:00)
- sport (e.g.: “Football”)
- country (e.g.: “England”)
- competition (e.g.: “Premier League”)
- settled (Boolean)
- markets (List)
    - id (auto-generated)
    - description (e.g.: “Match Betting”)
    - status (OPEN/CLOSE)
    - settled (Boolean)
    - outcomes (List)
        - id (auto-generated)
        - description (e.g.: “Arsenal to win”)
        - settled (Boolean)
        - price (e.g.: 2.0)
        - result (null/win/lose)

Objectives

You are expected to:

1. Design a clean and intuitive RESTful API
2. Implement the services with observability in mind
3. Provide adequate test coverage
4. Use open-source libraries and frameworks where appropriate
5. Make it easy to run the services locally (including the DB).

Bonus objectives (optional)

1. Add health indicator(s) assuming the service to be deployed in Kubernetes
2. Add data auditing
3. Keep version history of the data

Technical requirements

You can use Java or Kotlin with Spring (preferably Spring Boot) for the implementation. The project should be built with Maven or Gradle.

Notes

Our aim is to get an idea about how you do analysis, design and implementation. We would expect to see a well structured and easily readable code. We will pay attention to your choice of libraries, design patterns used, test strategies used and other things that reflect your personal approach to software development.

Deliverables

1. Zipped source code
2. Documentation on how to deploy and run it.

Please complete within **one week**.

Thank you.