Interview Task – Data Engineer

The Challenge

There is a free API https://rapidapi.com/theapiguy/api/free-nba/endpoints that provides historical records for NBA matches. This API provides lots of data, but we need only a small fraction of it and presented in a slightly different format. Please create a micro-service that exposes a REST API and delivers the following functionality:

1. Lists all NBA matches for a given date.
2. Returns data just for a single game specified by gameID
3. Create an endpoint that allows one to download a CSV file with aggregated statistics by team for a requested date period, start_date, end_date, e.g., 2022-01-31
4. The CSV file should contain: Team name, Median score, Percentage of games won. Additional fields are allowed. (**Stretch goal**: put in home and away summaries)

We are only interested in the following data for each match:

1. Game ID
2. Date of the game
3. Home and Away team names
4. Home and Away team scores
5. We would also like to have a sum of how many points each player scored in the game (calculated value). Any player who didn't score a single point in the match can be excluded from the list of players.

In order to reduce network traffic, the data needs to be cached in a local database of your choice.

The NBA API is free to use but you will need to sign-up for a free Rapidapi account.

Objectives

You are expected to:

1. Analyse the NBA API yourself and find out the endpoint(s) and response fields you need to use to complete the exercise
2. Design a clean and intuitive RESTful API
3. Please use Flask (https://flask.palletsprojects.com/en/2.3.x/) to build the RESTful API.
4. Implement the service as it was for critical production use:
    a. Think about operations. What would you need to add for live troubleshooting? (Logging for example)
    b. Are unexpected situations handled properly?
    c. Is there a way you to ensure response time SLA?
5. Provide adequate test coverage
6. Use design patterns where appropriate
7. Use other open source libraries and frameworks where appropriate
8. Make it easy to run the service locally in a Docker container (including the DB).

Technical requirements

Please use Python within a Docker container for the implementation of your micro-service.

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 service development in data engineering.

Deliverables

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

Please complete within **one week or less, the sooner the better**. The task should take no more than 2 hours to complete.

Thank you.