Take-home Assignment

Senior Backend Engineer – Royalty Processing

Introduction

We're excited to have you as part of our interview process!
This take-home task is designed to evaluate your technical skills, system design approach, and overall software craftsmanship.

Context

Imagine you are part of the team building a music metadata service for a streaming platform akin to Spotify or Apple Music, that serves millions of users worldwide. Your task is to design and implement a Music Metadata Service that stores and provides metadata about different music tracks and artists. Our goal is to provide a user-friendly interface where customers can effortlessly access this information.

Task Requirements

Your task is to design and implement a system that provides the following user experiences:

1. **Add a New Track**: As a user, you should be able to add a new track to an artist's catalogue, capturing attributes such as track title, genre, length, etc. .
2. **Edit Artist Name**: As a user, you should be able to edit an artist's name to accommodate instances where artists have multiple aliases.
3. **Fetch Artist Tracks**: As a user, you should be able to fetch all tracks associated with a specific artist.
4. **Artist of the Day**: As a user, you should be able to see a different "Artist of the Day" in a cyclical manner on the homepage each day, ensuring a fair rotation through the entire catalogue of artists. This means if there are n artists, after n days, the cycle restarts with the first artist, ensuring an equal chance for each artist to be the "Artist of the Day".

These requirements brief provide an outline for the service. However, how you approach these requirements, how you design and implement the system, and how you anticipate and plan for potential issues is entirely up to you. We encourage you to make assumptions where necessary, but please ensure that you document and justify those assumptions. Please note that you are not expected to submit any UI components.

Extra Information

- **Language:** Your application should be implemented using Scala.
- **Freedom of Choice**: You are free to choose the build tools, libraries, frameworks, etc., that you want to use.
- **Keep it simple.** You don't need to dedicate days to this; simply demonstrate your ability to craft excellent software.
- **Functional:** Please ensure that the code in the submission is fully functional on a local machine, and include instructions for building and running it.
- **Dependencies:** Avoid any external dependencies (remote cloud databases, remote search engines, etc.) to solve the task. This exercise is intended to assess your problem-solving skills rather than creating a system reliant on cloud connections.
- **MVP:** Use this exercise as a guide for design decisions, considering it as the initial prototype of a Minimum Viable Product (MVP) that will require future scaling and production readiness. Be prepared for further discussions regarding how to transition and scale the prototype for future deployment. We understand that "production-ready" may have different interpretations, and we look forward to discussing what it means to you during the review.

Deliverables

- Please submit your solution either as a link to a public repository or as a zip file with all the necessary components.
- Be prepared to present and explain your design and implementation.
- If there are additional artifacts (e.g., slides, Miro-board, etc.) that you wish to use during the presentation, please include them in your submission.

If you have any questions or require further clarification, feel free to reach out to us.
We are looking forward to receiving your solution!