CatoriTech Technical Assessment

Welcome to our technical assessment, designed specifically for the role of a Java Developer at our company. This exercise is an opportunity for you to showcase your expertise in Java and Quarkus, your ability to integrate with external services, and your skills in designing and implementing microservices that are both robust and scalable. As part of our team, you will play a crucial role in developing solutions that meet the dynamic needs of our global user base. For us it is important that the solution delivered is a fully functional and working solution. The only technical requirement is to use the Java Quarkus framework, for databases and other technologies there are no constraints.

How to Deliver

The assessment solution can be delivered either by inviting this email antonio.cacciotti@catoritech.com to your git repository or by sending a zip file to the same email. The solution should be delivered within 3/5 working days.

Challenge 1 : Multitenancy

Overview

You are building a Quarkus application that needs to support multitenancy. Using a schema-per-tenant strategy, demonstrate how to configure the application to switch between different tenant schemas based on a tenant identifier.

Requirements:

- Set up multiple tenants with distinct schemas in an H2 or PostgreSQL database.
- Implement a custom tenant resolver that determines which schema to use based on a tenant ID passed in the request headers.

Expected Deliverables:

- A MultiTenantResolver class that resolves the correct schema for each tenant.
- Configuration in application.properties to manage the multitenancy setup.

- A simple REST API with two endpoints: /tenant/{id}/info that returns data specific to that tenant, and /tenant/{id}/update to modify tenant-specific data.

Assessment Criteria:

- Proper use of Quarkus multitenancy features (e.g., the quarkus-hibernate-orm extension).
- Accuracy of tenant-specific database operations.
- Code organization and adherence to Quarkus best practices.

Useful resources:

https://quarkus.io/guides/security-openid-connect-multitenancy

https://quarkus.io/guides/hibernate-orm#multitenancy

https://quarkus.io/guides/security-keycloak-authorization#multi-tenancy

Challenge 1 : Bonus Microservice Implementation (optional)

Our gaming platform aims to enhance user engagement by rewarding players with bonuses upon login. The architecture comprises two microservices: the Login Service and the Bonus Service. These services interact via a Kafka messaging system to facilitate the bonus distribution process.

Objective:

Your task is to develop a Bonus Microservice that listens for login events from the Login Service, calculates the appropriate bonus, and communicates with the Login Service to update the player's bonus in the database.

Requirements:

Login Service

- Emits login events to a Kafka topic named player-login-events upon user login.
- Listens to the player-bonus-updates Kafka topic for bonus update events.

Bonus Microservice:

- Event Subscription: Subscribes to the player-login-events Kafka topic to receive login notifications.
- Bonus Calculation:
    - Queries the bonus table to determine the bonus amount for the player based on predefined criteria.
    - The bonus table schema includes columns: player_id, bonus_amount, and any other necessary fields.

    - Bonus Distribution:
        - Produces an event to the player-bonus-updates topic, instructing the Login Service to update the player's bonus.
        - Ensures idempotency to prevent bonus duplication.

  Database:

    - Utilize a relational database of your choice.
    - Includes a player_bonus table with columns: player_id, total_bonus, and any other relevant fields.

Bonus Challenges (Optional):

- Scalability: Describe or implement strategies to scale the microservices to handle high volumes of login events.
- Security: Outline security measures for protecting the communication between services and securing sensitive data.