Teste prático de programação - Desenvolvedor Backend Pleno

Descrição geral

Desenvolver uma API utilizando Java 17, Maven e, preferencialmente, Quarkus. A API deve entregar as funcionalidades descritas abaixo seguindo boas práticas de desenvolvimento, com camadas de persistência, modelo e serviço. A criação da estrutura de banco de dados é imprescindível (banco de dados relacional, preferencialmente MySQL).

Use case

Uma das estruturas mais básicas para um cartão de crédito é a parte de conta e cartões. Imagine que toda vez que alguém abre uma conta no alt.bank nós precisamos criar uma conta para esse novo Customer com base em seus dados cadastrais e também disponibilizar um cartão de crédito físico e virtual para o novo cliente.
O cartão de crédito físico deve ser enviado por nós via transportadora para o cliente com base no seu endereço. Somente após a chegada do cartão físico, e sua devida validação, é que o cliente pode solicitar um cartão virtual. Esse processo serve para evitar fraudes.
É muito importante que a nossa base de dados seja o mais concisa possível, siga bons padrões de normalização de dados e seja livre de problemas como dados duplicados, pois vamos ter que passar todo esse conjunto de dados para o pessoal de produto.

Requisitos básicos

- A conta do cliente deve ter dados básicos de cadastro como CPF, e-mail, telefone, etc.;
- Um cartão deve ter os dados básicos, como número, nome do titular, etc.
- É necessário saber qual o endereço do cliente para podermos enviar um cartão físico para ele;
- O mapeamento de entidades na API deve ser feito utilizando Hibernate;
- Não é necessário criar métodos para validar se a numeração de um CPF ou CEP são válidos, porém a modelagem e validação de dados imputados deve fazer sentido;
- Todas as operações devem ser disponibilizadas via endpoints REST. Devem ser disponibilizados endpoints para:
    - Criar uma conta;
    - Listar uma conta;
        - Filtro por documento do cliente;
    - Inativar uma conta **[opcional]**;
    - Criar um novo cartão;
    - Listar todos os cartões de uma conta **[opcional]**;
    - Ativar um cartão;
    - Bloquear um cartão **[opcional]**;

    - Solicitar a reemissão de um cartão (com um motivo para isso, como perda ou roubo) **[opcional]**;
- É necessário criar testes unitários para ao menos uma parte das funcionalidades;

Requisitos extras **[opcionais]**

- Nós precisamos ter 2 webhooks:
    - Resposta da transportadora sobre a entrega do cartão: Quando o cartão é entregue, nós precisamos receber o webhook para ter certeza de que a validação do usuário realmente deve ocorrer.
      `{"tracking_id":"tracking id do cartão","delivery_status":"status da entrega","delivery_date":"local date time","delivery_return_reason":"status about any delivery problem","delivery_address":"delivery address"}`
    - Mudança automática do CVV: O número do CVV do cartão virtual deve mudar periodicamente para evitarmos fraudes.
      `{"account_id":"processor account id","card_id":"processor card id", "next_cvv":123, "expiration_date": "local date time"}`