Desafio Técnico – Desenvolvedor(a) .NET

Plataforma para processamento de transações financeiras

1. Sobre a PagueVeloz

A PagueVeloz é uma empresa de tecnologia voltada para o setor financeiro, especializada em soluções de meios de pagamento, serviços bancários integrados e adquirência. Nosso propósito é facilitar a vida financeira de empresas e empreendedores por meio de produtos robustos, ágeis e seguros. Com uma arquitetura orientada a microsserviços e foco em escalabilidade, operamos em alto volume de transações e buscamos talentos que compartilhem nossa paixão por desempenho, arquitetura limpa e excelência técnica.

2. Contexto do desafio

Você foi alocado em um time responsável por construir o núcleo transacional de uma nova plataforma de adquirência, que fará parte de um ecossistema distribuído. Essa plataforma deve lidar com clientes, contas e operações financeiras em um ambiente onde o volume, concorrência e confiabilidade são críticos.

3. Objetivo

Construa um sistema para processar operações financeiras (créditos, débitos, transferências), com suporte a:

a. Múltiplas contas por cliente
b. Limite de crédito operacional
c. Transações reversíveis
d. Processamento assíncrono
e. Resiliência a falhas e controle de concorrência

O projeto deve ser estruturado para facilitar a divisão futura em microsserviços.

4. Regras de negócio

a. Clientes e contas
- Cada cliente pode possuir N contas.
- Cada conta possui: Saldo disponível, reservado, limite de crédito e status.

b. Operações
- Crédito, Débito, Reserva, Captura, Estorno, Transferência.

- Operações consideram saldo e limite.

c. Locks e concorrência
- Operações concorrentes na mesma conta devem ser bloqueadas.
- Garantir consistência nos eventuais lock.

d. Resiliência e eventos
- Cada operação deve gerar eventos assíncronos.
- Simular falhas na publicação e aplicar retry com backoff exponencial.

e. Consistência eventual
- Atualização de saldo pode ser eventual via eventos.

f. Histórico e auditoria
- Registrar todas as operações com status, tipo e timestamps.

g. Rollback/Idempotência
- Garantir idempotência e rollback em falhas.

5. Requisitos técnicos avaliados

a. Assincronia (async/await)
b. Uso eficiente de memória
c. SOLID, OOP, polimorfismo e etc.
d. Padrões
e. Código testável
f. Arquitetura escalável (Clean, DDD, Onion)
g. Resiliência (retry, fallback)
h. Transações distribuídas
i. Modelagem relacional adequada

6. O que será avaliado além do código

a. Modelagem de domínio
b. Clareza do código e decisões
c. Organização e legibilidade
d. Estratégia de testes e concorrência
e. Criatividade na resolução de problemas
f. Cobertura de testes consistente

7. **Entregáveis**
    a. Projeto público (Git)
    b. README com instruções, vamos rodar seu projeto, então cada detalhe importa!
    c. Código C# .NET 9 com cobertura de testes

8. **Diferenciais**
    a. Uso de Docker
    b. Métricas de performance
    c. Observabilidade e eventos de negócio
    d. Deploy em nuvem ou container