# Desafio VR Desenvolvimento

1. Recebemos um código desenvolvido por terceiros de um sistema que possui alto volume de lógica de negócio e apresenta as seguintes características:

- O sistema recebe requisições REST, está dividido em camadas e possui classes de domínio;

- O controller recebe a requisição e está com toda lógica de negócio. Monta e repassa o domínio para a aplicação;

- A aplicação tem a responsabilidade de repassar o objeto pronto para o repositório;

- O repositório apenas persiste os objetos mapeados do hibernate através de spring data;

- O domínio apenas faz o mapeamento para o BD;

- Nenhum teste unitário foi escrito.

- O sistema está escrito em java para rodar como spring boot.

Apresente observações/problemas sobre essa solução.

Comente qual(is) a(s) sua(s) estratégia(s) para melhorar este sistema em termos de qualidade e manutenção. Justifique suas decisões.

Primeiramente eu removeria toda regra de negócio da camada de controller e levaria para uma camada de negócio.

Depois criaria pastas para separar objeto de request e response da api, objeto da camada de negócio e objeto para a camada de repositório.

Além disso adicionaria conceitos de arquitetura limpa para deixar a aplicação mais fácil de se criar testar unitários e manter o código também uma vez que cada classe terá uma única responsabilidade.

Trabalhando dessa forma inclusive fica fácil trocar o banco de dados da aplicação caso haja necessidade porque o objeto de repositório é completamente desacoplado do negócio.

E adicionaria testes unitários no projeto cobrindo pelo menos 80% da camada de negócio para garantir mais qualidade nas entregas.

1. Descreva quais são as principais limitações ao se adotar servidores de aplicação em uma arquitetura orientada a microsserviços.

Um produto criado com arquitetura orientada a microserviços é complexo para fazer o gerenciamento. Em teoria a aplicação tende a se manter mais de pé, uma vez que cada microserviço cuida de um pedaço específico do produto.

Porém como serão inúmeros serviços é necessário criar documentações de todas as integrações, trabalhar com múltiplas zonas na aws para minimizar ainda mais a indisponibilidade da aplicação.

E ter um bom gerenciamento das pods e logs para pegar indisponibilidades rápido e corrigir problemas também.

1. Atualmente, diversas aplicações escritas em Java estão deixando de serem desenvolvidas para rodarem em servidores (JBoss, Tomcat), adotando ferramentas que disponibilizam um servidor embutido na própria ferramenta. Quais são os principais desafios ao se tomar uma decisão dessas? Justifique sua resposta.

Nesse cenário a aplicação deve ser configurada para printar logs, todo o monitoramente, segurança e atualizações. Também temos um grande problema porque sempre que precisar atualizar qualquer coisa na aplicação é necessário reiniciar o servidor e isso pode gerar indisponibilidade.

1. Teste prático (em anexo)