# 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.

**R: O controller não deveria ter a responsabilidade da regra de negócio, e sim o service , pois ele quem e responsável por repassar o objeto pronto para o repositor para que assim persista os dados no BD.**

**Deveria ter sido feito teste unitários afim de facilitar correções futuras. Por mas que seja complexo, seria bom desenvolver em microsserviços para futuramente não ter que para a aplicação por completa para uma manutenção.**

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

**R: Complexidade, governança, segurança, integração com aplicação, monolíticas legadas.**

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.

**R: As configurações de servidor de aplicação passam a ser individuais, sendo necessário para cada aplicação configurar logs, monitoramento e segurança e atualizações. Não é possível compartilhar recursos entre aplicações, tendo que modificar a aplicação para que ela se torne stateless. E sempre que for atualizar a aplicação deve se levar em conta que será necessário o restart do servidor, independente da mudança.**

1. Teste prático (em anexo)

https://github.com/MuriloPK/mini-autorizador-vrbeneficios.git