# 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.

O correto um controller rest para receber as requisições e fazer as validações básicas - @RestController

Depois tempos outro controller para processar as regras de negócio - @Service

Não é necessário uma camada somente para repositório (como o DAO) se está se utilizando o spring data com hibernate. Seria necessária uma camada ou aplicação como esta somente se houve legados com detalhes de acesso ao banco, como stored procedures, selects complexos, etc.

Testes unitários devem ser sempre escritos para evitar que novas funcionalidades impactem o funcionamento de funcionalidades antigas. Também garante que os serviços desenvolvidos por outras equipes que consomem nossos serviços não quebrarão nada.

Realizei a divisão em camadas conforme mostrado no teste prático. Também foram implementados os testes unitários

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

Aumento da complexidade com o tempo

Reutilização de forma incorreta pode aumentar a dependência entre todo o sistema e alterações podem trazer impactos não previstos.

Escalabilidade limitada

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.

Múltiplos servidores para manter e gerenciar – Pode-se mitigar este problema com ferramentas de gerenciamento com uma boa interface gráfica como o openshift.

Múltiplos logs – Logs espalhados são difíceis para ler os erros. Pode-se mitigar este problema com ferramentas como o kibana

Múltiplas portas e URLs para monitorar – Cada servidor teria que ter sua porta e url, o que seria muito custoso para gerenciar. Ferramentas como openshift gerenciam um endereço virtual e porta para cada um dos pods.

1. Teste prático (em anexo)