# 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.

**- Criar testes unitários para validar as classes (Controller, Service e Repository) criadas, para garantir que tudo está funcionando corretamente.**

**- Criar testes integrados para validar diferentes cenários.**

**- Criar testes funcionais para garantir que os endpoints REST estão todos funcionando.**

**- Implantar o SonarQube para rodar com a API, para analisar os testes e sua cobertura.**

**- Seguir os principais Design Patters.**

**- Mover a lógica de negócio da classe Controller para a classe Service ou Domínio da aplicação, para assim facilitar seu reuso.**

**- Permitir que o Repositório persista qualquer objeto criado nas classes de Entidade.**

**- Criar classe Model ou DTO para a transferência e conversão de dados, utilizando o Mapper para assim evitar problemas de serialização de dados.**

**- Se possível documentar o que foi feito para facilitar posteriormente a manutenção do projeto.**

1. Descreva quais são as principais limitações ao se adotar servidores de aplicação em uma arquitetura orientada a microsserviços.   
     
   **- Complexidade  
   Estruturação do projeto é mais complexa que a monolítica.  
   - Governança  
   Arquitetura de microsserviços requer um nível maior de gerenciamento.  
   - Utilização de mais recursos de servidor  
   Ao usar microsserviços será necessário mais espaço em disco e tráfego de rede.  
   - Integração com aplicações monolíticas legadas  
   Será preciso dar sustentação a aplicação legada até completar toda a migração dos módulos para os microsserviços.  
   - Segurança  
   A segurança acaba sendo um processo desafiador, pois terá diferentes partes para serem monitoradas.  
   - Adoção da cultura DevOps  
   Necessário uma equipe maior com conhecimentos de DevOps e divisão dos serviços.**
2. 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.

**- Aumento no trabalho para realizar testes.**

**- Perda de performance momentânea (cold-start)**

**- Dificuldade no monitorando e depuração, por não ter acesso a infraestrutura onde os serviços são executados.**

**- Complexidade da arquitetura, sem padrões estabelecidos.**

**- Ao usar a mesma estrutura (serverless) a aplicação fica engessada em um mesmo fornecedor, tendo limites de recursos a serem usados.**

**- Não é possível o compartilhamento de recursos entre as aplicações criadas, precisando assim modificar cada uma de forma isolada.**

1. Teste prático (em anexo)