Teste de Arquitetura (Respostas)

# Questão 1

* Seja escalável
  + Utilizar arquitetura de microserviços, separando-os por contexto, por exemplo Web Application, Estoque, Promoções, Gateway de Pagamentos, Régua de Relacionamento, Logística, etc.
* Seja segura
  + Elaborar um Plano de Segurança, com foco em ataques, acesso indevido, fraudes, etc.
* Seja flexível
  + Utilizar frameworks que facilitem o desenvolvimento e manutenção das aplicações, como Bootstrap, AngularJS, etc.
* Rode em ambiente Cloud
  + Utilizar soluções que facilitem o desenvolvimento, como NoSQL, Spring Cloud, Spring Boot, Kubernetes, MQ, Redis, etc.
* Tenha resiliência
  + Aplicar plano de Disaster Recovery, focado em nuvem.
* Seja evolutiva, manutenível e testes
  + A utilização de microserviços já torna as aplicações evolutivas, podendo dimensionar recursos para pontos críticos do sistema. Uma arquitetura bem definida e com ferramentas de detecção de desvios (por exemplo utilizar SonarQube). Aplicar metodologia de testes automatizados, como Testes Unitários, Integrados e de Sistema.
* Tenha baixa latência a BD
  + Pode-se adotar CQRS, banco de dados em memória (Redis por exemplo) ou ferramenta mais poderosa, conforme a necessidade.
* Como desenho da solução de arquitetura, foi criado um documento (DrawIO - draw.io) onde é possível visualizar, de forma simplificada, um diagrama de componentes do sistema (arquivo “Netshoes - Questão 1.xml”). É claro que foram omitidos componentes, pois o objetivo é demonstrar como seria a estrutura básica dos microserviços e, dependendo das especificações, pode-se adotar outras ferramentas e soluções.

# Questão 2

Para solucionar o problema, pode-se escalar a aplicação de Back Office e ajustar os sistemas que se integram a ele.

Uma solução para ajustar a integração seria a utilização de filas de mensagens (por exemplo ActiveMQ), com Divert para distribuir as mensagens e, assim, paralelizar o Back Office.

Para a solução, o Design Pattern a ser utilizado é o Proxy, pois permitirá a distribuição das transações entre as filas.

Veja arquivo “Netshoes - Questão 2.xml”

# Questão 3

Microserviço é uma maneira de separar a aplicação em módulos, organizados por contexto e, que os tornam independentes.

A utilização é aplicada quando precisamos escalar partes críticas do sistema ou evitar dificuldade de desenvolvimento e manutenção entre os módulos.

Prós - Possibilidade de escalar os módulos, conforme a necessidade de recursos. Fácil adequação, manutenção e substituição dos módulos. Fácil detecção à falha.

Contra - Complexidade dos testes. Controle de transação e tratamento de falhas (entre módulos). Tratamento de chamadas síncronas.

Para tratar a questão do Single Entry Point, pode-se utilizar a solução de API Gateway. Essa solução pode ser melhorada, aplicando API Gateway para cada tipo de acesso (por exemplo: Mobile, Web Application, System Integration, etc)

# Questão 4

Essa questão requer uma análise mais detalhada do requisito, porque as possibilidades de solução são muitas.

* Pode-se aplicar CQRS, a fim de separar as transações de escrita e leitura.
* Para os casos de consultas constantes e dados pouco atualizados (por exemplo: consulta de cep, endereços, cidades, GEO, etc), podemos aplicar Cache ou BD em memória.
* Se for um BD relacional, podemos migrar tabelas críticas para Big Data.
* Pode-se segregar o armazenamento, paralelizando os BDs.
* Para geração de relatórios, pode-se adotar um Business Intelligence.

# Questão 5

Como a questão trata somente da disponibilidade do produto em estoque, a solução é centralizar o saldo de estoque, em uma aplicação de alto desempenho.

Como cada canal cuida de seu estoque, podemos criar microserviços que “escutam” as transações ocorridas em cada estoque e, atualiza um banco centralizado e disponibilizando o saldo global entre os canais. Esse banco centralizado pode ser otimizado com soluções do tipo NoSQL, Redis ou Kinesis.

# Questão 6

Poderia pegar esse código, fazendo uma busca simples na Internet. De verdade, efetuei a busca e, não foi nada difícil encontrar várias soluções prontas para essa questão (algumas bem duvidosas e que não funcionavam).

Prefiro descrever que, em casos assim (e já passei por vários), não é papel do Arquiteto de Software ter o conhecimento de todos os tipos de lógica de programação. Somente em momentos específicos são necessários os conhecimentos para a solução de algum problema.

Vou relatar um caso do GuiaMais. Estávamos implementando a busca de anunciantes e, veio o desejo do PO, que tivéssemos a busca dos resultados por distância. Então pesquisei na Internet sobre cálculo de distância, usando latitude/longitude (cálculo complexo levando em consideração a curvatura da terra, diferença de distância entre as longitudes, ângulo entre 2 pontos, etc). Aprendi no momento, implantei, testei e nunca mais precisei fazer manutenção.

Se mesmo assim, for necessário o projeto, segue:

<https://github.com/lfchaim/ns-teste>

# Questão 7

Ao pressionar enter, uma chamada ao caminho do servidor (URI) é executada e, algumas informações da camada Cliente são trafegadas, através do Header da requisição (como dados do navegador, tipo de conteúdo, tipo de requisição, idioma, etc).

Em resposta à chamada, o servidor retorna dados sobre a requisição como, status da resposta (200, 404, 503, 502, etc), informações do servidor, tipo de conteúdo, idioma, etc).

Na grande maioria, são trafegados dados tipo text/html, porém outro tipos podem ser transmitidos como multipart/form-data (transferência de arquivo).