Teste - Desenvolvimento de Software

1. Crie um fork desse repositório, faça os testes, responda as perguntas e depois submeta um pull request.

2. Os testes envolvendo código devem ser feitos preferencialmente em **C# ou Java/Kotlin (Android)**. Cada codificação deve estar em uma pasta com o nome que está entre perênteses nas questões. As questões teóricas devem ser respondidas em um **pdf** que também será adicionado ao GIT. O primeiro teste será pegar os códigos e dar um run. Tem que rodar de primeira.
3. Como você se atualiza tecnicamente?
   Através de web-cursos.

4. Crie uma função para calcular o *n-ésimo* elemento da Sequência de Fibonacci (fibonacci).

    i. **Qual solução é mais performática, iterativa ou recursiva?**
    Iterativo.

    Por que?
    O baixo consumo de recursos, visto que vetor de dados não é "relido" e calculado, uma vez que ele obterá o valor atual e o anterio, apenas, não recalculando toda a cadeia de valores.

    Opcional: Qual é o 5287º elemento da sequência?
    R: -1888262963

5. **O que significa SOLID?**
   É um acrônimo que representa os 5 princípios da programação orientada a objetos.
6. **O que são design patterns?**
   São padrões de projeto que definem em alto nível um problema comum, dando uma rota de acordo com a melhor prática.
    i. **Quais são os tipos de design patterns?**
    Existem uma série de DP que podem ser utilizados, porém os mais comuns são os do modelo GoF
    ii. **Com quais você está familiarizado?**
    Criação; Padrões arquiteturais (MVC).
    Qual é a função deles?
    Os designs de criação são responsáveis por estruturar a cadeia de classes de uso de uma maneira mais flexível possível,

encapsulando as necessidades da aplicação e permitindo que ela seja portada entre plataformas, não havendo necessidade de uso exclusive de plataforma uma determinada plataforma X.
Enquanto o padrão de Arquitetura é responsável por "criar uma forma de uso" para a aplicação, através de interface de uso.

iii. ***Opcional:*** **Qual é sua opinião quanto ao uso de design patterns?**
Os designers Patterns permitem uma melhor manutenção de código entre times e recursos, pois utilizam-se de padrões e boas práticas.

7. **Qual foi o último livro técnico que você leu?**
[Clean Code by Robert C Martin]
Quando foi isso?
Comecei a leitura semana passada.
   i. *Observação: se já tivermos lido e você for chamado para uma entrevista, perguntas poderão ser feitas a respeito do mesmo.*
8. **Cite 3 maneiras diferentes de implementar Dependency Inversion.**

9. **O que são ORMs?**
Uma técnica para aproximação dos bancos-de-dados relacionais ao mapeamento objeto-relacional
   i. Quais você conhece bem?
   C# - Entity Framework; .Net Dapper
   ii. *Opcional:* Cite pelo menos 2 vantagens e 2 desvantagens de seu uso.
   Vantagens: Código mais elegante; Fácil manutenção.
   Desvantagens: Alto consumo de recursos; desenho da solução torna-se mais complexo.

10. **O que são microsserviços?**
São partes de um sistema completo. Enquanto em um modelo anterior de desenvolvimento, todo sistema era pensado como se fosse vendido em uma caixa única, os microsserviços quebram essa visão inteira e transformam em pequenos módulos escalonáveis, não precisando de um infra robusta para um único produto e sim, por modelo de consumo de dados.
   i. **Quais são suas vantagens e desvantagens?**
   Vantagens: Alto rendimento para Squads; Limites bem definidos

Desvantagens: Maior custo na serialização; Desemprenho modularmente dependente.

11. **Com a seguinte representação de produto (crud):**
    Nunca atuei em dev de API, apenas realizei as validações de qualidade. Li a documentação para criação, porém possuo maior experiencia direta em .net e VB.NET para dev.

```
{
    "sku": 43264,
    "name": "Batata frita Ruffles Cebola & Salsa",
    "inventory": {
        "quantity": 15,
        "warehouses": [
            {
                "locality": "SP",
                "quantity": 12,
                "type": "ECOMMERCE"
            },
            {
                "locality": "MOEMA",
                "quantity": 3,
                "type": "PHYSICAL_STORE"
            }
        ]
    },
    "isMarketable": true
}
```

Crie endpoints para as seguintes ações:

- Criação de produto onde o payload será o json informado acima (exceto as propriedades isMarketable e inventory.quantity)
- Edição de produto por sku
- Recuperação de produto por sku
- Deleção de produto por sku

Requisitos:

- Toda vez que um produto for recuperado por sku deverá ser calculado a propriedade: inventory.quantity
- A propriedade inventory.quantity é a soma da quantity dos warehouses
- Toda vez que um produto for recuperado por sku deverá ser calculado a propriedade: isMarketable
- Um produto é marketable sempre que seu inventory.quantity for maior que 0
- Caso um produto já existente em memória tente ser criado com o mesmo sku uma exceção deverá ser lançada
- Dois produtos são considerados iguais se os seus skus forem iguais

- Ao atualizar um produto, o antigo deve ser sobrescrito com o que esta sendo enviado na requisição
- A requisição deve receber o sku e atualizar com o produto que tbm esta vindo na requisição

Não é necessário o uso de bancos de dados.

Testes são bem vindos.

Você não deve levar mais do que 4 horas para o teste todo.