## Teste para desenvolvedor .net

O teste abaixo visa avaliar o perfil técnico dos candidatos a desenvolvedor backend na Superdigital. Leia atentamente as instruções abaixo e responDa as questões da melhor forma possível.

* O teste será composto de três perguntas técnicas e uma avaliação prática.
* Responda as perguntas com suas palavras, com seu entendimento sobre o tema.
* Publique o teste prático com as respostas em um repositório no git-hub e envie o endereço.

## Perguntas:

## Explique com suas palavras o que é domain driven design e sua importância na estratégia de desenvolvimento de software.

Domain-Driven Design, ao meu ver é uma prática que vai além da entrega de software de qualidade e com o mínimo de bugs possível. Em um cenário onde possamos alcançar um software sem bugs, com uso de testes(TDD) e outras práticas agile, que atenda as necessidades do usuário final podemos ainda acabar com um modelo que não é robusto ou que não reflete totalmente os objetivos do negocio. Para mim DDD auxília a criar verdadeira qualidade em desenvolvimento de software, fazendo com que o próprio modelo reflita como realmente o software funciona, permitindo até que à linguagem entre os desenvolvedores e o BI sejam à mesma e com conhecimento distribuido entre todos. Quando implementamos DDD corretamente chegamos ao ponto onde nosso design é exatamente como o software funciona.

## Explique com suas palavras o que é e como funciona uma arquitetura baseada em microservices. explique ganhos com este modelo e desafios em sua implementação.

Microserviços é uma arquitetura que permite desenhar software como um sitema de componentes independentes, onde cada componente manipula uma única responsabilidade. Para mim sua vantagem é que está alinhado com as principais práticas de desenvolvimento nos dias de hoje como asynchronous patterns, message-based communication, cloud e ambientes de container centralizado, permite entrega continua, permite ser testado por unit test, etc...

Acredito que os desafios principais ou desvantagens sejam a curva de aprendizagem com o conceito e as tecnologias necessárias, como microserviços são sitemas distribuídos, eles sofrem com uma certa lentidão em comparação com os sistemas in-process, microserviços devem ser medidos e monitorados sempre, conforme a quantidade de requisiçoes aumenta, sem um sistema devops adequado, muitos itens podem ir para deploy e isso pode ser custoso se feito manualmente, em um sistema grande refatorar um bloco pode acabar transferindo codigo para outro microserviço, tornando o processo complicado. Cada caso será um caso, como sempre cada projeto deve ser bem conhecido pelo arquiteto, microserviços não será a solução definitiva para todo projeto.

## explique qual a diferença entre comunicação sincrona (http) e assincrona (msmq) e qual o melhor cenário para utilizar uma ou outra.

Bom, HTTP é um protocolo síncrono, resumidamente quer dizer que “um cliente envia uma requisição e aguarda uma resposta”.

Na outra mão uma requisição assíncrona não aguarda uma resposta para continuar á executar outras coisas, no caso do protocolo MSMQ o padrão assíncrono de menssageria permite que cada requisição caia em uma fila para ser processada na ordem da fila, primeiro que entra primeiro que é processado, acredito que o padrão assíncrono seja mais utilizado em serviços de domínio público, projetos enterprise melhor se adequam HTTP mesmo. Porém estou respondendo assim, mas acredito que preciso de maiores informações sobre o projeto e qual dos dois melhor se adequam ao mesmo.

## Teste prático:

## regra de negócio:

## 

## Precisamos de um serviço que , dado duas palavras (alfanuméricas) , calcule o total de movimentações para se transformar a primeira palavra na segunda palavra.

## exemplos:

## dado as palavras “gato”, “rato” a saída deve ser “1” pois:

## substituindo o “G” do gato pelo “R” , transformamos “gato” em “rato”

## exemplo 2:

## dado as palavras “cavalo”, “pato” a saída deve ser “4” pois:

## removendo o “c”, removendo o “a”, substituindo “v” por “p” e substituindo “l” por “t” transformamos “cavalo” em “pato”

## informações adicionais:

## as strings podem ter qualquer tamanho.

## qualquer palavra pode ser informada.

## faça uma api rest com um método “post” recebendo as palavras no body da requisição

## serão avaliados critérios de arquitetura como separação de responsabilidade, clean code, testes

## documente a api com swagger

## tecnologias que você pode utilizar .net core 2.1, c#, xunits (testes)

## no término do projeto, publique o código em um repositório no git-hub