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, (BRUNO.COUTO@superdigital.com.br)

Perguntas:

* Explique com suas palavras o que é domain driven design e sua importância na estratégia de desenvolvimento de software.  
    
    
    
  O projeto orientado a domínio é um padrão de desenvolvimento de software que tem como base o desenvimento em camadas. uMA SOLUTION MUITA MAIS ORGANIZADA E OTIMIZAÇÃO DE CÓDIGO.

* 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.  
    
    
  oS MICRO SERVIÇOS SÃO APIS QUE CONTEM REGRAS DE NEGÓCIO ESPECIFICAS, ESSA ARQUITETURA GERA GANHOS AGILIDADE NAS APLICAÇÕES E OTIMIZAÇÃO DOS TIMES DE DESENVOLVIMENTO, ASISM CADA DESENVOLVEDOR OU EQUIPE FICARIA RESPONSÁVEL POR UMA API
* explique qual a diferença entre comunicação sincrona e assincrona e qual o melhor cenário para utilizar uma ou outra.

COMUNICAÇÃO SINCRONA É REALIZAR UMA TAREFA E INICAIR OUTRA APENAS QUANDO A PRIMEIRA TERMINAR JÁ A ASSINCRONA É REALZIAR AS TAREFAS DE FORMA INDEPENDENTE.

A SINCRONA É IDEAL QUANDO NÃO EXISTEM ROTINAS QUE UTILIZEM MUITOS RECURSOS DE UM BANCO DE DADOS, SIM O USUÁRIO TERIA QUE ESPERAR A LISTA DE PRODUOS CARREGAR PARA INIICAR UMA OUTRA ATIVIDADE.

A ASSINCRONA É O CONTÁRIO O USUÁRIO PODE INICIAR SUAS ATIVIDADES E ASSIM QUE A LISTA DE PRODUTOS TERMINAR DE CARREGAR ELE ERA NOTIFICADO.

Teste prático:

regra de negócio:

criar um microservices que, através de um http post efetue uma operação de debito (origem) e credito (destino) nas contas correntes.

entidades: contacorrente, lancamentos (voce pode incrementar com outras entidades se achar necessário)

Parâmetros de entrada:

conta origem

conta destino

valor

Parâmetros de saída:

http status code

informações adicionais:

* o método “post” devera receber os parametros no body da requisição em formato json
* UTILIZE Domain Driven Design
* serão avaliados critérios de arquitetura como separação de responsabilidade, clean code, segurança e testes
* tecnologias que você pode utilizar .net core 2.X, c#, xunits (testes)
* no término do projeto, publique o código em um repositório no git-hub