

Avaliação Técnica (C# + Clean Architecture + CQRS + DDD + SQL + MongoDB)

💡 Subir código no Github e nos enviar o link.

O Desafio Desenvolver uma aplicação em C#, na plataforma .NET Core 5 ou superior, obedecendo os seguintes requisitos:

Instruções

1. Desenvolver back-end utilizando Web Api, seguindo o padrão RESTful.
2. Construir o front-end com Single Page Application. (VueJs, React ou Angular)
3. Não é necessário se preocupar com o layout nem com boas práticas visuais, html, css, etc.

Contexto do Projeto

Você vai construir uma API RESTful que gerencia um sistema de **Pedidos** de uma loja online. Os pedidos podem ser realizados pelos clientes, e o sistema deve armazenar tanto os dados principais dos pedidos quanto dados complementares relacionados a itens do pedido e status.

A API terá que interagir com um banco de dados relacional (SQL Server) para armazenar os dados do pedido, e com um banco de dados NoSQL (MongoDB) para armazenar os dados para leitura.

A aplicação deve ser desenvolvida com **Clean Architecture**, utilizando **CQRS** (Command Query Responsibility Segregation) para separar os fluxos de leitura e escrita e deve atender ao **DDD**. Você também utilizará um **ORM** (como o Entity Framework) para o acesso ao banco relacional.

Observações:

- Customer pode ser sempre fixo, consideremos que todos os pedidos serão feitos apenas por um cliente;
- A tabela de produto deve conter apenas três opções.

Especificação de ações:

- Adicionar;
- Remover;
- Atualizar;
- Listar;

- Detalhar pedidos.

Entidades Relacionais:

- **Customer**
 - `Id` (PK)
 - `Name`
 - `Email`
 - `Phone`
- **Order**
 - `Id` (PK)
 - `CustomerId` (FK para Customer)
 - `OrderDate`
 - `TotalAmount`
 - `Status` (Enum - mapeado para `OrderStatus`)
- **OrderItem**
 - `Id` (PK)
 - `OrderId` (FK para Order)
 - `ProductId`
 - `ProductName`
 - `Quantity`
 - `UnitPrice`
 - `TotalPrice`
- **Produto (Product)**
 - `Id` (PK)
 - `Name`
 - `Price`