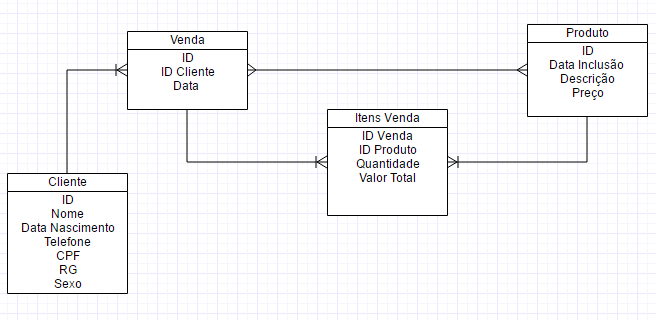
Exercício - CRUD com Entity Framework

**Exercício 1**



Para Cliente considere que todas informações sejam obrigatórias, exceto Telefone e Sexo

Para Produto considere que todas informações sejam obrigatórias

Para Venda considere que todas informações sejam obrigatória

Para ItensVenda considere que todas informações sejam obrigatória

Aproveite a base de dados LOJA (desenvolvida anteriormente) e desenvolva uma API onde seja possível inserir registros nela.

Utilize Migrations para criação das tabelas

Disponibilize rests para INSERIR, DELETAR, ALTERAR e CONSULTAR Clientes

* + Para a pesquisa e remoção, permita o usuário filtrar pelo Id do Cliente (validar a existência desses registros antes de realizar alteração ou remoção)
  + Nome do Cliente deve ter no mínimo 5 letras
  + Aceitar apenas Cpfs válidos
  + Não aceitar cadastros de Cpfs e Nomes repetidos

Disponibilize rests para INSERIR, DELETAR, ALTERAR e CONSULTAR Produtos

* + Não aceitar cadastros de Nomes repetidos
  + Não aceitar preços com valores negativos

Disponibilize rests para cadastrar uma Venda

* O Dto de entrada terá uma lista de ids de produtos com o Id do produto e quantidade e o id do cliente
* Fazer as validações necessárias para criação da venda.
  + Exemplos: Validar se os ids passados existem, quantidade disponível do produto

Disponibilize um rest para consulta de todas as Vendas, exibindo o Nome e CPF do Cliente, Data da Venda, Valor total da Venda e uma lista com o Nomes dos produtos, Preço e quantidade usada na Venda

Rests de Cliente:

/api/v1/cliente - POST

/api/v1/cliente/:id - GET

/api/v1/cliente/:id - PUT

/api/v1/cliente/:id - Delete

Rests de Produto:

/api/v1/produto - POST

/api/v1/produto/:id - GET

/api/v1/produto/:id - PUT

/api/v1/produto/:id - Delete

Rest de venda

/api/v1/venda - POST

Seguir padrão de projeto que estudamos, com as camadas de Interface, Service, Infrastructure, Domain e Common

Criar um tipo de exceção para as regras de Domínio e fazer as tratativas no controller, conforme feito no exercício de API

Criar o Swagger da API

Os rests de Post deverão retornar o Id do registro criado

As injeções de dependência deverão ficar em arquivos separados por contexto (ApplicationService, Repository, UnitOfWork)

Dicas:

Nos métodos de consulta lembrar de usar o AsNoTracking()

Nos mapeamentos, conferir quais campos são obrigatórios e quais não são

Identação do Código

Fazer as validações de objetos que podem nulos

Bônus: Criar um interceptor para tratar as exceções lançadas pela API, ao invés de usar o try e catch dentro do controller