FIAP – Faculdade de Informática e Administração Paulista
Advanced Business Development With .Net
***Professor**: Humberto Oliveira*

CHECKPOINT 2

- O checkpoint 2 deverá ser feito de forma INDIVIDUAL.
- A entrega deverá ser o **link do repositório** que foi feita a implementação, no portal da **FIAP**, área de trabalhos.
- Cada questão vale 1 ponto, os outros 2 pontos restantes serão do exercício que foi passado em aula.
- Projeto: https://gitlab.com/fiap-dotnet/2tdsr/segundo-semestre/checkpoint02
- Requisitos:
    - O projeto deve estar buildando, caso contrário a nota é 0.
    - Será executado o update do banco de dados, caso não funcione a nota é 0.

Sobre o Checkpoint:

Sistema de Gerenciamento de Fornecedores e Vendedores

Nesta prova, você deve implementar um sistema para gerenciar informações de fornecedores e vendedores.

1. **Implementação da Aplicação**:
    - Complete os métodos que estão faltando nas classes de aplicação para **Fornecedor** e **Vendedor**.
2. **Criação das Entidades**:
    - Implemente as entidades FornecedorEntity e VendedorEntity com as seguintes propriedades e anotações:

FornecedorEntity:

- Id: Identificador único do fornecedor (tipo: int).
- Nome: Nome do fornecedor (tipo: string).
- CNPJ: CNPJ do fornecedor (tipo: string).
- Endereco: Endereço do fornecedor (tipo: string).
- Telefone: Telefone de contato do fornecedor (tipo: string).
- Email: Email do fornecedor (tipo: string).
- CriadoEm: Data de criação do registro (tipo: DateTime).

VendedorEntity:

- Id: Identificador único do vendedor (tipo: int).
- Nome: Nome do vendedor (tipo: string).
- Email: Email do vendedor (tipo: string).
- Telefone: Telefone de contato do vendedor (tipo: string).
- DataNascimento: Data de nascimento do vendedor (tipo: DateTime).
- Endereco: Endereço do vendedor (tipo: string).
- DataContratacao: Data de contratação do vendedor (tipo: DateTime).
- ComissaoPercentual: Percentual de comissão do vendedor (tipo: decimal).
- MetaMensal: Meta mensal do vendedor (tipo: decimal).
- CriadoEm: Data de criação do registro (tipo: DateTime).

3. **Implementação do Repositório**:
   - Complete os métodos que estão faltando no repositório, garantindo que todas as operações de CRUD (Create, Read, Update, Delete) estejam implementadas corretamente.
4. **Injeção de Dependência**:
   - Implemente a injeção de dependência para as classes de serviço e repositório e configure o bootstrap na API.
5. **Documentação da API**:
   - Adicione documentação clara e concisa em todos os métodos dos controllers, utilizando comentários XML.
6. **Implementação de DTOs**:
   - Crie e implemente as propriedades necessárias nas classes DTO (Data Transfer Objects) para **Fornecedor** e **Vendedor**.
7. **Validação com FluentValidation**:
   - Implemente as regras de validação na classe de validação dos DTOs utilizando a biblioteca FluentValidation, garantindo que os dados atendam aos critérios necessários antes de serem processados.
8. **Configuração da Conexão**:
   - Adicione a string de conexão necessária no arquivo de configuração da aplicação, garantindo que o acesso ao banco de dados esteja corretamente configurado.