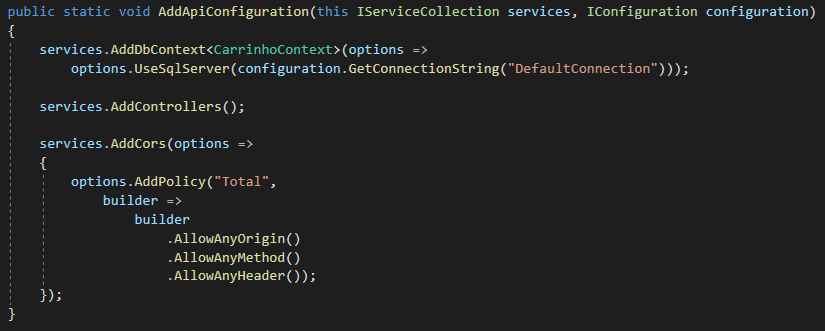
**API de Carrinhos**

* **Setup inicial**

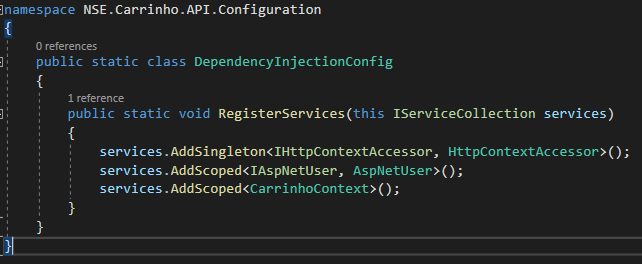
->Criar a pasta Configuration

->Dentro dela, ApiConfig

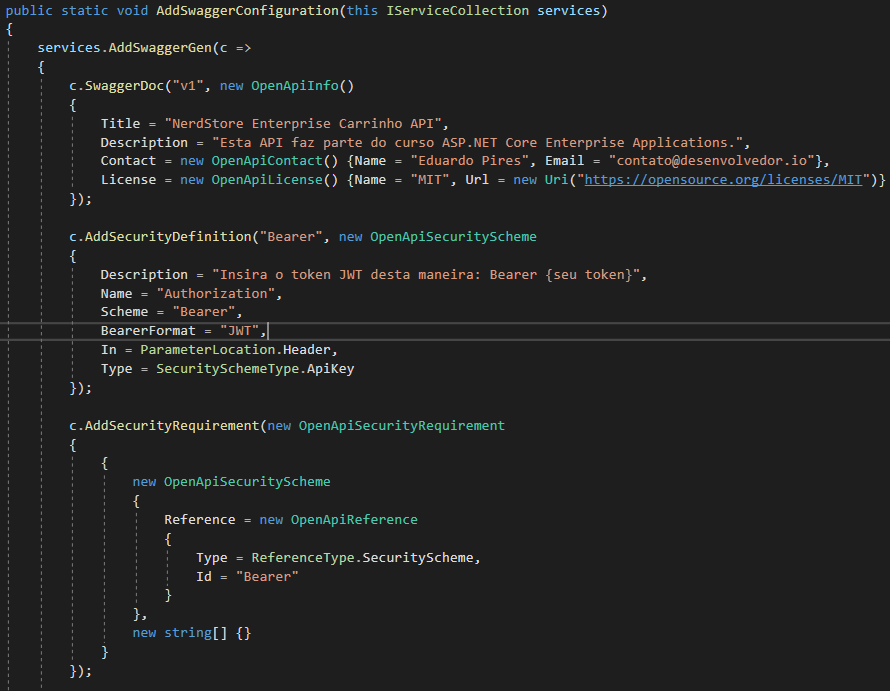




->Criar a DependencyInjectionConfig

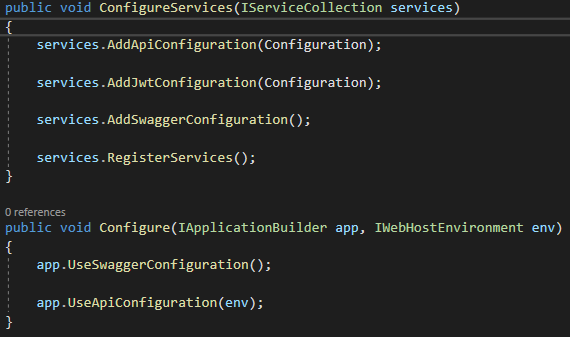


->Criar a SwaggerConfig



->Configurar a StartUp



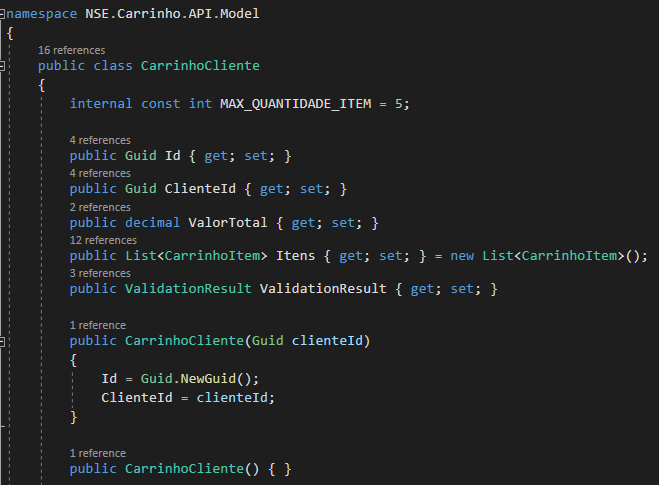


->Criar a pasta Model

->Criar a classe CarrinhoItem

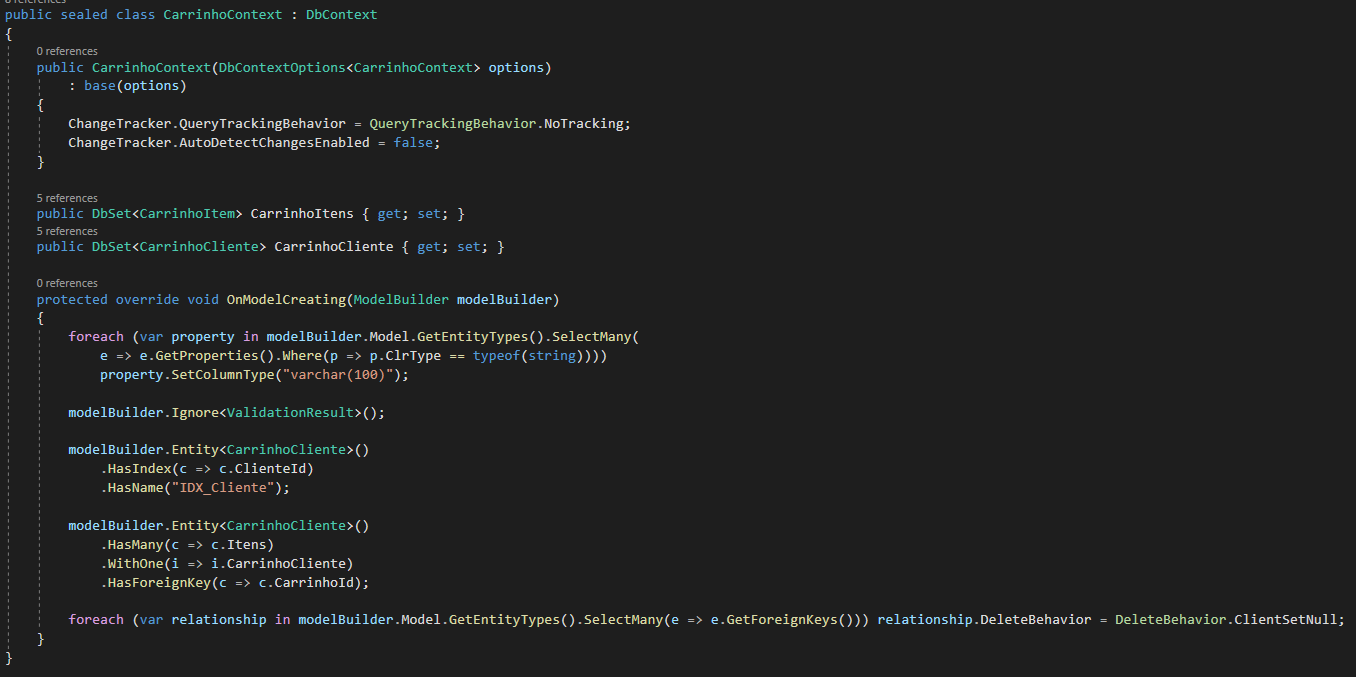


->Criar a classe CarrinhoCliente



->Criar a pasta Data

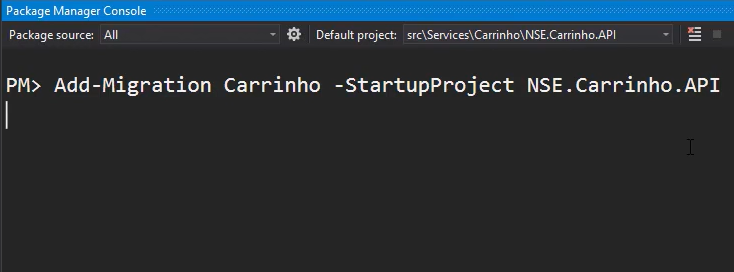
->Dentro dela, criar CarrinhoContext



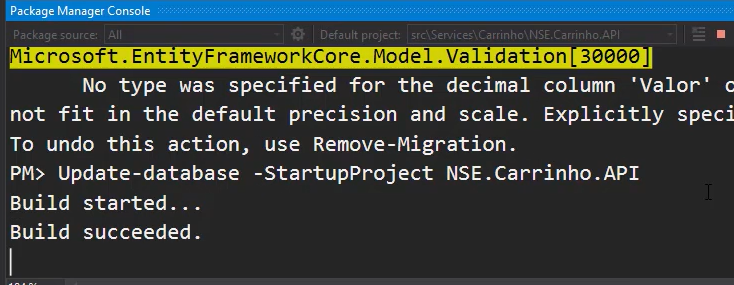
->Apontar o appSettings.Development para a base



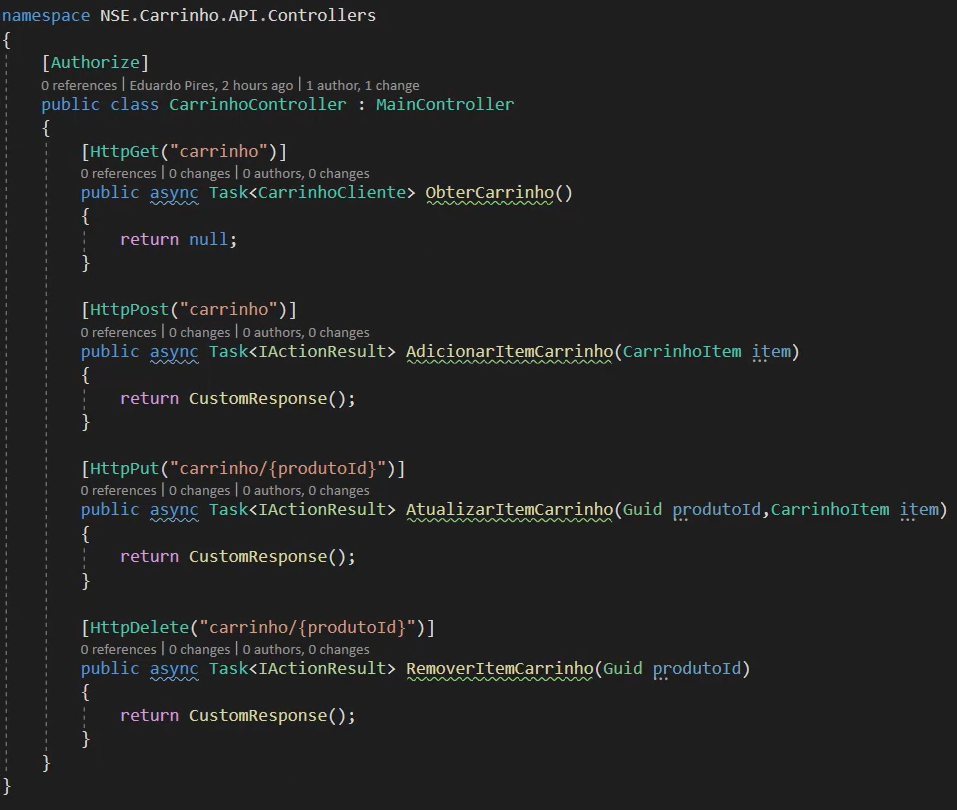
->Adcionar a migration com a start up Project especificada



->Realizar o update-database também com o startup Project especificado



-> Criar a controller com seus métodos



* **Globalização da Interface IUser**

->Em Building Blocks/Services/NSE.WebAPI.Core, criar uma pasta chamada Usuario

->Dentro dela, criar uma classe chamada AspNetUser

->Copiar tudo que dentro da classe Web/MVC/NSE.WebApp.MVC/Extensions/IUser e colar nessa classe criada anteriormente

->Substituir o nome da interface e da classe para AspNetUser

->Extrair a interface para um arquivo separado

->Extrair a classe ClaimsPrincipalExtensions para um arquivo separado

->Isso inicialmente quebrará a aplicação

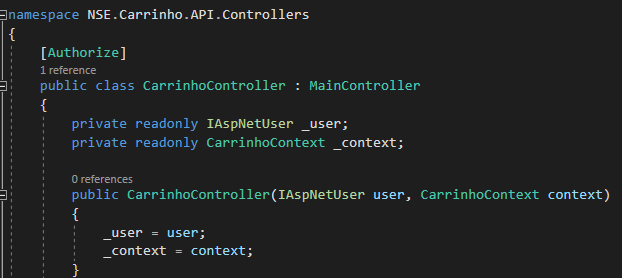
->Dar um build para identifica onde foi que quebrou

->Referenciar corretamente dessa vez já com a interface globalizada

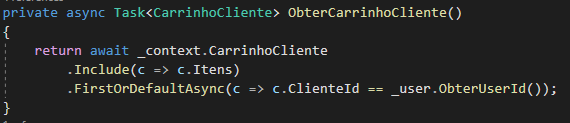
->Excluir a classe IUser da aplicação mvc(Extensions/IUser.cs)

* **Implementando Task ObterCarrinho**

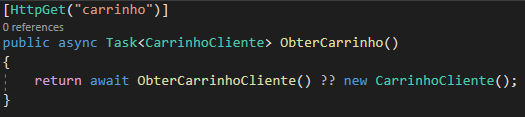
->Injetar o IAspNetUser e o CarrinhoContext na controller



->Criar o Método ObterCarrinhoCliente

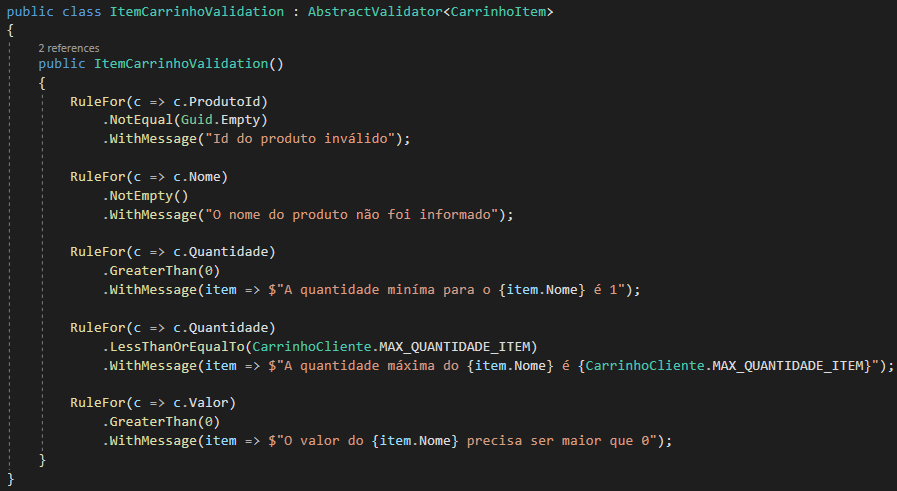


->Implementá-lo no método ObterCarrinho

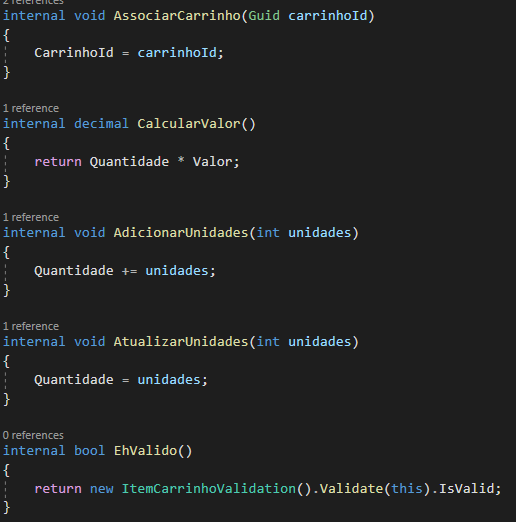


* **Criando métods base na classe CarrinhoItem**

->Criar uma classe de validação aninhada Chamada ItemCarrinhoValidation

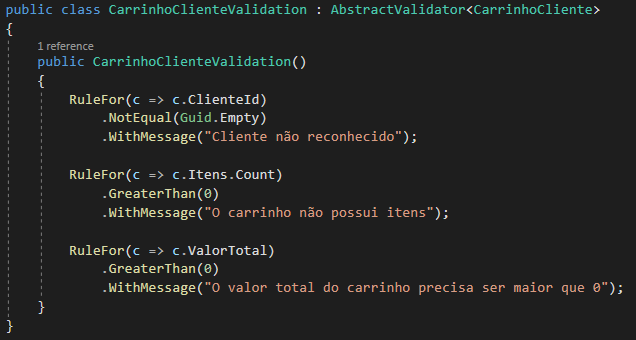


->Depois disso, criar os métodos base para incrementar os métodos da controller e da Classe CarrinhoController

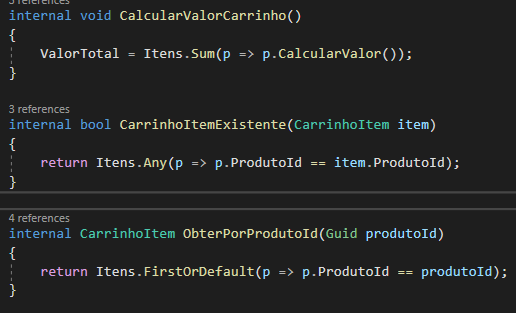


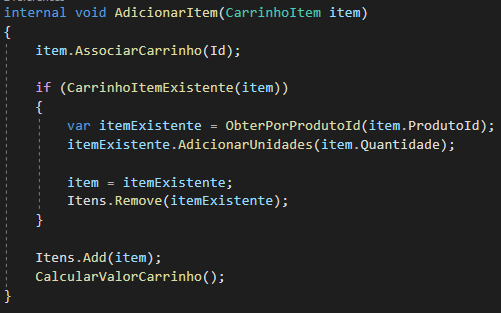
* **Criando métodos Base na classe CarrinhoCliente**

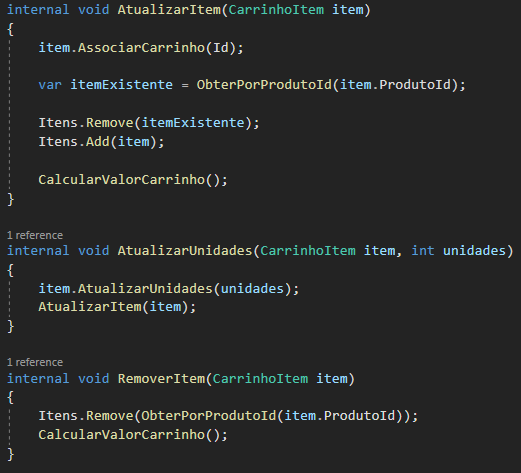
->Criar uma classe de validação aninhada Chamada CarrinhoClienteValidation

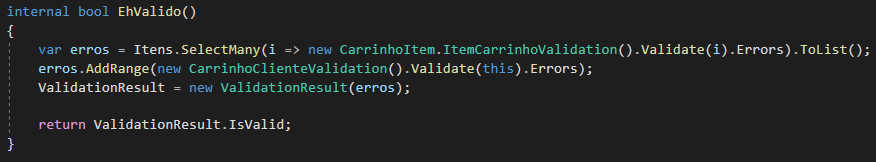


->Criar métodos base para serem utilizados na Controller





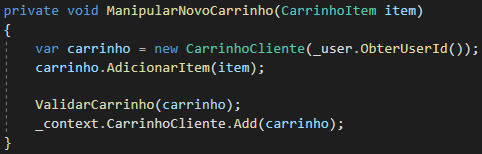




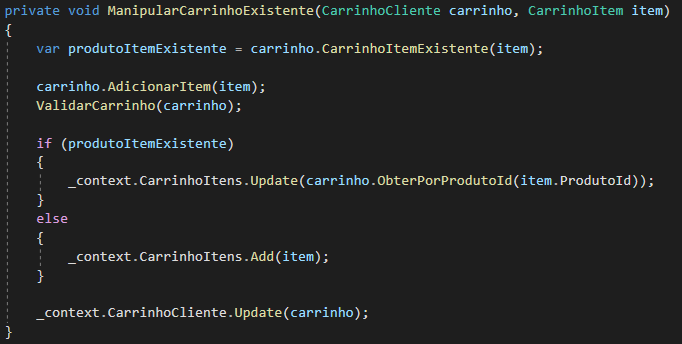
* **Implementando Task AdicionarItemCarrinho**

->Na controller, criar os métodos bases para Implementar os métodos

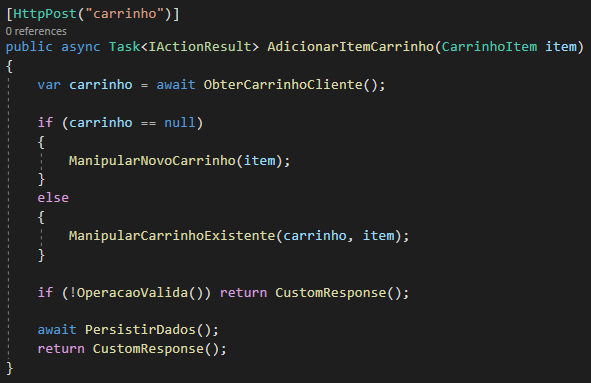
->Criar o método ManipularNovoCarrinho



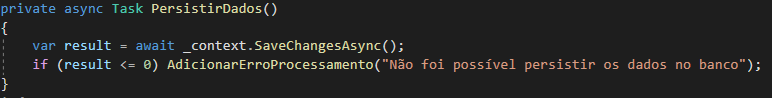
->Criar o método ManipularCarrinhoExistente



->Criar a devidas validações para implementar a Task AdicionarItemCarrinho

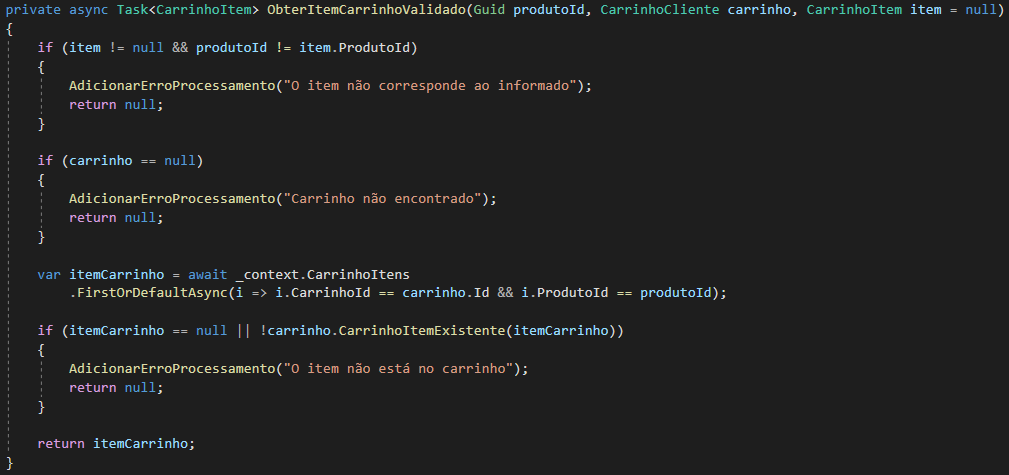


->Criar método PersistirDados

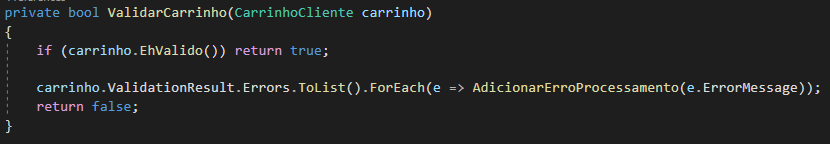


* **Implementando Task AtualizarItemCarrinho**

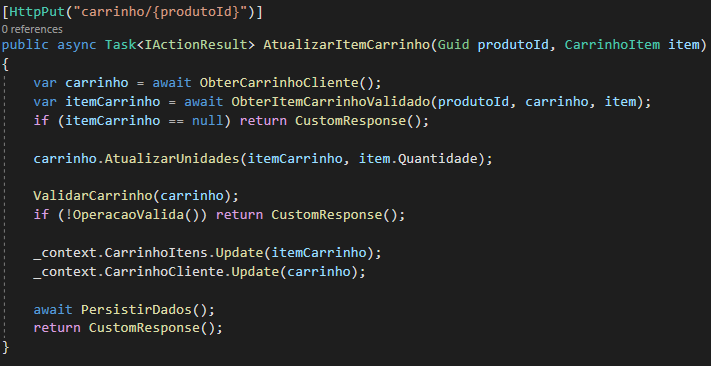
->Criar método ObterItemCarrinhoValidado



->Criar método validar Carrinho



->Implementar Task AtualizarItemCarrinho



* **Implementando Task RemoverItemCarrinho**

->Implementar a Task de remover

