Netshoes

Teste para analista desenvolvedor

|  |  |
| --- | --- |
|  | Foi construído um projeto fictício com algumas regras de negócio já desenvolvidas, o teste consiste basicamente em entender o projeto e evoluir conforme as orientações citadas no decorrer deste documento.  O objetivo do teste é simular as atividades do dia-a-dia afim de avaliar as seguintes áreas:   * Desenvolvimento de novos recursos * Correções de BUG’s. * Melhorias técnicas * Analise das atividades |

Visão geral

O projeto consiste em uma tela cujo objetivo é digitar um pedido, e baseado nos dados da tela fazer a persistência dos dados.

|  |  |
| --- | --- |
|  | Dica: A base de código é bem reduzida, entenda a estrutura do projeto antes de iniciar a solução dos problemas. |

1 – Exercício (Novo recurso)

Está em desenvolvimento uma tela para emissão de notas fiscais, o cálculo dos impostos já está desenvolvido, porém é necessário fazer a persistência da nota fiscal em um arquivo XML, seguindo exatamente a mesma estrutura da classe: NotaFiscal.

O diretório onde serão gerados os XML’s é configurado pelo time de infra estrutura, o usuário não pode altera-lo (o usuário tem acesso apenas ao executável).

|  |  |
| --- | --- |
|  | Dica: Pode ser utilizado serialização de objetos para XML nesse caso. |

2 – Exercício (Novo recurso)

|  |  |
| --- | --- |
|  | Será necessário criar um banco de dados localmente utilizando o SQL Server Express. |

Na pasta SQL foi disponibilizado um conjunto de scripts, para que seja possível criar e popular o banco de dados.

Utilizar as procedures P\_NOTA\_FISCAL e P\_NOTA\_FISCAL\_ITEM para fazer a persistência dos dados no banco.  
A persistência só pode acontecer se o XML foi gerado com sucesso.

3 – Exercício (Novo recurso)

Adicionar na classe de itens da nota fiscal os campos relacionados ao imposto de IPI:

Base de cálculo de IPI: Igual ao valor total do produto.  
Alíquota de IPI: Se for brinde alíquota é igual a 0% se não brinde alíquota é igual a 10%.   
Valor de IPI: Base de cálculo \* Alíquota de IPI.

Esses campos serão persistidos na tabela de Itens de nota fiscal e também no arquivo XML.

4 – Exercício (Novo recurso)

Desenvolver uma stored procedure para ser utilizada por outros sistemas, o objetivo dessa procedure é devolver a seguinte estrutura:

CFOP | Valor Total da Base de ICMS | Valor Total do ICMS | Valor Total da Base de IPI | Valor Total do IPI

Deve estar agrupado por CFOP.

5 – Exercício (Correção de BUG)

Para o estado de origem SP e destino RO o sistema deveria definir a CFOP como 6.006, corrigir o erro no sistema para que seja definido a CFOP correta.

6 – Exercício (Correção de BUG)

A tela tem alguns pontos de melhoria a serem implementados, por exemplo: Ao salvar a tela não está limpando os campos, é possível indicar estados inválidos tanto na origem quanto no destino.

Levantar os pontos de melhoria e executar as correções.

R:

- Adicionar select/combobox com os estados para que o usuário não escolha estados inválidos.

- Iniciar valores booleanos como false para os valores que não foram clicados na grid.

- Validar de campo valor é um campo numérico na grid.

- Validar nome do cliente.

- Colocar cursor waiting.

- Colocar botão deletar na grid de pedidos para deletar caso necessário.

7 – Exercício (Novo recurso)

Criar um campo desconto no item da nota fiscal.  
Preenchê-lo com a seguinte regra: Clientes do sudeste (campo “EstadoDestino”) tem 10% de desconto.   
Esse campo deve ser incluso no XML e no banco de dados.

|  |  |
| --- | --- |
|  | Dica: É sempre bem-vindo a criação de novas classes com apenas uma responsabilidade, utilizando os conceitos SOLID. |

8 – Exercício (Melhoria técnica)

O código tem uma complexidade ciclomática muito alta, refatorar o código para que seja possível entende-lo melhor e simplificar a manutenção.

|  |  |
| --- | --- |
|  | Dica: Não tem nenhum tipo de restrição com relação a reestruturação, sinta-se à vontade para criar mais classes, projetos, métodos e etc. |

9 – Exercício (Melhoria técnica)

Estão previstas muitas alterações nas regras desta aplicação, crie testes unitários para diminuir o risco e melhorar o design.

10 – Exercício (Analise)

É necessário gerar um report técnico das atividades desenvolvidas, descreva com suas palavras qual é o objetivo da tela e como foi implementado as alterações solicitadas.

R:

O objetivo da tela foi gerar uma nota fiscal(xml/database) com o nome do cliente e os devidos impostos e descontos de produtos conforme o estado.

Foram ajustados ou adicionados alguns itens abaixo:  
  
Erros nas classes

- Ao emitir nota fiscal, precisava adicionar os itens da nota a uma lista.

- Na classe FormImposto, precisava inicializar sempre o pedido quando for Gerar Nota Fiscal.

- A gravação de origem e destino de estados estavam invertidos.

- Foi criada uma procedure no qual gera o próximo valor/número da nota fiscal

Erros de banco

- P\_NOTA\_FISCAL: Voltar Campo id para ter resultado

Erros de Tela:  
 - Conforme exercício 6.

Para exemplificar um conceito de design DDD, foram criados alguns projetos novos.

Está na linha .net e não .net core pois já havia projetos criados como dot net framework 4.5.

Foram criadas tabelas necessárias para tirar os inúmeros IFs que havia para o cálculo de alguns impostos.

Foram adicionado campos e procedures conforme solicitado nos exercícios.

Foram criados projetos de testes para validar minhas services.

Foi criado no app.config do projeto uma setting com o caminho físico onde gera a nota fiscal. A chave é folder-infra-notafiscal.

A nota fiscal foi gerada em uma pasta via desrealização e gravação de arquivo.

Subi o projeto para o git hub https://github.com/ascarelli/testnet