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).

Arquivo NotaFiscal.cs, linhas abaixo do comentário // Persistência de dados XML

|  |  |
| --- | --- |
|  | 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.

Arquivo NotaFiscal.cs, linhas abaixo do comentário // insertando os valores P\_NOTA\_FISCAL e

// insertando os valores P\_NOTA\_FISCAL\_ITEM

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.

Arquivo NotaFiscalItem.cs para criação do get e set das variáveis.

Arquivo NotaFiscal.cs para cálculos de imposto Ipi, linhas abaixo do comentário /\*\*\*\*\*\*\* Adiciono valores 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.

Script criado com o nome V\_NOTA\_FISCAL\_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.

Arquivo NotaFiscal.cs, linhas abaixo do comentário // correção do erro: variáveis invertidas

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.

Arquivo FormImposto, criação do método LimparDados(), que é chamado no evento buttonGerarNotaFiscal\_Click(), após o evento MessageBox.Show("Operação efetuada com sucesso"), linhas abaixo do comentário // limpamos os dados da tela após registro no banco de dados

7 – Exercício (Novo recurso) - Ok

Criar um campo desconto no item da nota fiscal.  
Preenchê-lo com a seguinte regra: Clientes do sudeste (campo “EstadoDestino”) tem 10% de desconto.

Arquivo NotaFiscal.cs, linhas abaixo do comentário // Valor Desconto para clientes onde EstadoDestino seja Sudoeste.

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.

Arquivo NotaFiscal.cs, linhas abaixo do comentário // melhora da complexidade ciclomática

|  |  |
| --- | --- |
|  | 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.

Objetivo

A tela “Cálculo de Impostos” foi criada para gerar as Notas Fiscais dos pedidos realizados pelos clientes, aplicando diferentes regras de cálculo segundo o negócio.

Alterações

Foi solicitado aplicar novos cálculos de impostos (cálculos IPI), assim como a geração de um arquivo XML para a persistência os dados após registro no banco de dados (estrutura da Nota Fiscal).

Além disso, foi necessário ajustar a performance no funcionamento da tela (criação de novos métodos para limpeza da tela e conexões ao servidor para agilizar a resposta do serviço).

