Exercicio Sistema de Nota Fiscais (Versao 1)

Sistema de notas fiscais (Questão em 3 partes)

Obedecer aos requisitos é critico, eles representam restrições legais. Uma auditoria examinará periodicamente todas as NF do sistema e falhas resultam em multa.

Conceitos (classes):

1. Nota Fiscal contem itens de venda.
2. NF deve sempre conter informação do cliente.
3. Item de venda: Deve estar associado a **um** produto ou serviço. Os tipos de produtos e serviços já são pré-definidos e armazenados em um BDProdutos, que deve ser mokado. O IV define a quantidade de produto nesta NF. O BDProduto contem apenas dados sobre os Produtos/Serviços e os preços unitários.
4. Cliente: dados de um cliente. contem CPF.
5. Cadastro: Unica classe que acessa BDClientes, possui metodos para encontrar clientes e para cadastrar novo cliente. Nao deve cadastrar cliente se o CPF estiver sujo no SPC. O BD deve ser mokado.
6. VerificadorCPF: acessa o SPC remotamente para verificar antecedentes. Deve recusar clientes com nome sujo. Deve ser mokado.

Restrições e Requisitos

1. Nota fiscal não pode ter zero itens de venda. Deve ter 1 ou mais.
2. Todo item de venda deve pertencer a exatamente uma nota fiscal. Apenas uma NF pode criar itens de venda, e gerenciar os proprios itens de venda.
3. Todo item de venda se referirá a exatamente um produto ou serviço.
4. Nota fiscal deve estar associada a exatamente um cliente pré-cadastrado.
5. deve ser facil de estender o sistema para especificar novas categorias de produtos e servicos no futuro.

6. uma vez criada uma NF, os seus itens de venda devem ser modificados, adicionados ou deletados apenas pelos metodos apropriados. Deve-se cuidar que não haja acesso de escrita inapropriado a lista de itens por outros meios.
7. O objeto Cadastro é o único que pode criar Clientes, ou acessar o BD de clientes.

Atributos das Classes

Há vários atributos naturais para as classes, mas só implemente o estritamente necessário para o demo. Implemente apenas um campo por classe e acrescente outros se houver necessidade.

- Nota Fiscal: numero, valor (deve ser a soma dos valores dos itens), condições e data de entrega, além dos dados do cliente.
- Cliente: CPF, nome, endereco, telefone. Necessários para emitir a NF.
- Item de venda: quantidade, desconto, condições e datas diferenciadas para esta venda especificamente.
- Produto: nome, preço unitário, setor responsável, informações sobre o processo e matérias-primas, etc.
- Servico: nome, preço por hora, setor responsavel, natureza (consultoria, treinamento, etc.)

Mocks

BDProduto; BDCliente : mockam acesso a BD. BDProduto precisaa apenas de de operacoes de acesso, pois o seu subsistema não é responsavel por preencher os dados. BDCliente precisa de todas as operacoes CRUD, pois o seu subsistema eh o responsavel por preencher os dados.

ValidadorCPF: mocka um serviço remoto. O seu subsistema apenas submete CPFs e recebe sim ou nao. O seu subsistema nao eh responsavel por preencher os dados.

Note que para mockar, eh necessario criar uma interface para um objeto que nao sera implementado. Dessa forma, estamos definindo uma interface para o subsistema que nao implementaremos.

[OBRIGATORIO] Parte I) Diagrama UML [2.0]

Defina o diagrama UML correspondente ao sistema, considerando as classes e requisitos/restrições acima (sem detalhar atributos). Para os seguintes relacionamentos, a notação UML deve indicar o tipo de relacionamento (associações, agregações, composições e heranças - vale criar classes auxiliares), e a multiplicidade das associações:

a) NF - Item

b) Item - Produto / Servico

c) NF - cadastro

d) Cadastro - ValidadorCPF

[IMPLEMENTAÇÃO] Parte II) Código [2.5]

Implemente um esqueleto de codigo com as associações deste sistema considerando:

Não precisa detalhar campos das classes, nem no papel, nem na UML, nem no codigo. Os campos existem para explicar a necessidade das classes para o aluno e ajudar a dar contexto. Pode implementar uma classe vazia ou apenas um campo simplificado representando os diversos dados da entidade quando escrever os outros campos seria apenas adicionar algumas atribuicoes. Por exemplo, a classe Cliente pode ter apenas um inteiro CPF, e omitir nome e endereco, com um formato simples para o CPF por brevidade. Outro exemplo: O IV pode ter apenas uma quantidade de um produto/serviço, e apenas multiplicar o preço do produto/servico.

É apenas necessário implementar

i) um esqueleto demonstrando as especificidades dos tipos de relacionamento (associacoes, agregações, composição, multiplicidades), como nos exemplos na aula, ou seja, apenas o código implícito ou arquitetural, para a parte II

ii) o minimo necessario para passar nos testes, para a parte III, ainda sem precisar detalhar campos.

[IMPLEMENTAÇÃO] Parte III) Testes [2.0]

a) cadastro de cliente (com CPF limpo no SPC, precisa passar)

Precisa verificar que o validador de CPF foi chamado

c) cadastro de cliente (com CPF sujo no SPC, precisa falhar)

Precisa verificar que o validador de CPF foi chamado

d) cria NF com itens de compra, depois deleta itens e adiciona novos itens

verifica se os itens estao corretamente presentes na nota fiscal

verifica se o valor da nota eh correto

(valor de um IV eh quantidade * valor unitario do produto)

e) crie uma NF com APENAS um item de compra

troque esse item por outro item

verifique se o item e o valor esta correto depois da troca.

Lembre que de forma nenhuma o sistema pode permitir que exista uma NF vazia.

f) tente adicionar um item de venda contendo um produto nao existente no BDProdutos

Deve ser lançada uma exceção com uma mensagem apropriada.

g) tente criar uma NF vazia: nao deve ser possivel

h) tente remover o ultimo item de venda de uma NF, nao deve ser possivel

Dica:
Fazer o mock do mockito lançar exceções
http://2min2code.com/articles/mockito_intro/stubbing_method_throw_exception