SENAC

Campus Santo Amaro

TADS - Análise Desenvolvimento de Sistemas

POO - Programação Orientada a Objeto

Atividade Aula 7

Aluno: Patryck Vieira Sans - patryck.vsans@senacsp.edu.br
patrycksans@gmail.com

Diagrama de Caso de Uso:

O diagrama de caso de uso representa as interações entre os atores (usuários) e o sistema. No código fornecido, temos um único ator, o "Usuário", que interage com o sistema para realizar várias operações de gerenciamento de produtos.

Diagrama de Classes

Produto

id: int
classificacao: int
nomeProduto: String
precoProduto: double

Produto(id: int, classificacao: int, nomeProduto: String, precoProduto: double)
getId(): int
getClassificacao(): int
setClassificacao(classificacao: int): void
getNomeProduto(): String
setNomeProduto(nomeProduto: String): void
getPrecoProduto(): double
setPrecoProduto(precoProduto: double): void
toString(): String

CadastroProdutos

produtos: ArrayList<Produto>

adicionarProduto(produto: Produto): void
consultarProduto(id: int): Produto
removerProduto(id: int): boolean
alterarProduto(id: int, novaClassificacao: int, novoNome: String, novoPreco: double): boolean
listarProdutos(): ArrayList<Produto>

Encapsulamento:

Os princípios de encapsulamento foram aplicados nas classes Produto e CadastroProdutos: Na classe Produto, os atributos id, classificacao, nomeProduto e precoProduto são privados (private) e acessados por meio de métodos getters e setters apropriados. Isso garante que os atributos sejam encapsulados e que seu acesso seja controlado.
Na classe CadastroProdutos, a lista de produtos (produtos) é privada e gerenciada por métodos públicos que fornecem funcionalidade controlada para adicionar, consultar, remover, alterar e listar produtos. Isso encapsula a lista de produtos e permite que as operações sejam realizadas de forma segura.

Baixo Acoplamento:

O baixo acoplamento entre as classes é alcançado pelo fato de que a classe CadastroProdutos atua como uma interface intermediária para manipular os objetos da classe Produto. As operações que envolvem a classe Produto são realizadas por meio de métodos da classe CadastroProdutos, que abstrai os detalhes internos da implementação da classe Produto. Isso significa que o código que usa a classe CadastroProdutos não precisa conhecer ou se acoplar diretamente com a estrutura interna da classe Produto. Essa abstração reduz o acoplamento e torna o código mais modular e flexível.
Portanto, o código implementa o encapsulamento e alcança um bom nível de baixo acoplamento entre as classes, tornando-o mais fácil de manter e estender no futuro.

UML:

Classe: Produto

Atributos:
- id: int
- classificacao: int
- nomeProduto: String
- precoProduto: double

Métodos:
+ Produto(id: int, classificacao: int, nomeProduto: String, precoProduto: double)
+ getId(): int
+ getClassificacao(): int
+ setClassificacao(classificacao: int): void
+ getNomeProduto(): String
+ setNomeProduto(nomeProduto: String): void
+ getPrecoProduto(): double
+ setPrecoProduto(precoProduto: double): void
+ toString(): String

Classe: CadastroProdutos

Atributos:
- produtos: ArrayList<Produto>

Métodos:
+ adicionarProduto(produto: Produto): void
+ consultarProduto(id: int): Produto
+ removerProduto(id: int): boolean
+ alterarProduto(id: int, novaClassificacao: int, novoNome: String, novoPreco: double): boolean
+ listarProdutos(): ArrayList<Produto>