**Sistema de Controle de Estoque para E-commerce Multivendedor**

**Visão Geral do Sistema**

O sistema desenvolvido tem como objetivo simular um ambiente de e-commerce multivendedor, onde clientes podem realizar pedidos de produtos ofertados por diferentes vendedores. Cada vendedor é responsável pelo seu próprio estoque e gerenciamento dos produtos. O sistema utiliza os princípios da **Programação Orientada a Objetos (POO)** para representar as entidades, suas responsabilidades e os relacionamentos entre elas.

O foco principal está em modelar de forma clara e modular os elementos essenciais de um sistema de vendas, como cadastro de usuários, gerenciamento de estoque, produtos, pedidos e o fluxo de compra por parte dos clientes.

**Descrição das Classes**

**1. Usuario**

Classe base para todos os usuários do sistema, contendo atributos e métodos comuns tanto para Cliente quanto para Vendedor.

* **Atributos:**
  + id: Identificador único do usuário.
  + nome: Nome do usuário.
  + email: Endereço de e-mail.
  + hashSenha: Hash da senha para autenticação segura.
* **Métodos:**
  + atualizarDadosPessoais(...): Atualiza nome, e-mail e senha do usuário.
  + autenticar(...): Verifica se o e-mail e senha informados são válidos.

**2. Cliente (herda de Usuario)**

Representa um consumidor final do sistema. Pode consultar e realizar pedidos de produtos.

* **Métodos:**
  + realizarPedido(...): Cria um novo pedido com os produtos desejados.
  + consultarPedidos(): Retorna o histórico de pedidos feitos pelo cliente.

**3. Vendedor (herda de Usuario)**

Usuário responsável por cadastrar e gerenciar seus produtos e estoque. Cada vendedor possui seu próprio estoque.

* **Atributos:**
  + nomeLoja: Nome da loja do vendedor.
  + meuEstoque: Objeto Estoque vinculado ao vendedor.
* **Métodos:**
  + cadastrarNovoProdutoEmEstoque(...): Adiciona um novo produto ao estoque.
  + removerProdutoDoMeuEstoque(...): Remove um produto do estoque.
  + atualizarDadosProdutoEmMeuEstoque(...): Modifica preço e quantidade de um produto existente.
  + consultarMeuEstoqueProdutos(): Lista todos os produtos no estoque.
  + consultarMinhasVendas(): Lista todos os pedidos recebidos.

**4. Produto**

Representa um item comercializado no sistema, com suas propriedades e funcionalidades básicas.

* **Atributos:**
  + id: Identificador único do produto.
  + nome: Nome do produto.
  + descricao: Descrição detalhada.
  + preco: Valor unitário.
  + quantidadeEmEstoque: Quantidade disponível para venda.
* **Métodos:**
  + setQuantidadeEmEstoque(...): Atualiza a quantidade disponível.
  + setPreco(...): Altera o preço do produto.

**5. Estoque**

Modela o estoque pertencente a um vendedor. Cada vendedor possui um estoque exclusivo.

* **Atributos:**
  + id: Identificador único do estoque.
  + produtosEstocados: Lista de produtos armazenados.
* **Métodos:**
  + adicionarProduto(...): Adiciona um novo produto ao estoque.
  + removerProduto(...): Remove um produto do estoque.
  + buscarProdutoPorId(...): Localiza um produto com base no ID.
  + atualizarQuantidadeProduto(...): Modifica a quantidade de um produto específico.

**6. Pedido**

Representa uma compra realizada por um cliente e associada a um vendedor.

* **Atributos:**
  + id: Identificador único do pedido.
  + data: Data em que foi feito.
  + status: Estado atual do pedido (vide StatusPedido).
  + itensPedido: Lista de produtos comprados (ItemPedido).
  + cliente: Cliente que realizou o pedido.
  + vendedor: Vendedor que irá fornecer os produtos.
  + valorTotal: Valor total da compra.
* **Métodos:**
  + calcularValorTotal(): Soma todos os subtotais dos itens para obter o total.
  + atualizarStatus(...): Altera o status do pedido.

**7. ItemPedido**

Auxilia o Pedido ao registrar os detalhes de cada item comprado.

* **Atributos:**
  + produtoVendido: Produto que foi vendido.
  + quantidadeComprada: Quantidade adquirida.
  + precoUnitarioRegistrado: Preço unitário no momento da compra.
* **Métodos:**
  + calcularSubtotal(): Multiplica o preço pela quantidade comprada.

**8. StatusPedido (enum)**

Enumerador com os possíveis estados do pedido no ciclo de vida da compra:

* NOVO, AGUARDANDO\_PAGAMENTO, PAGAMENTO\_APROVADO, EM\_PROCESSAMENTO, ENVIADO, ENTREGUE, CANCELADO.

**Relacionamentos e Princípios Aplicados**

* **Herança**:  
  Cliente e Vendedor herdam de Usuario, reaproveitando atributos e comportamentos comuns.
* **Composição**:
  + Estoque contém seus próprios Produtos.
  + Pedido é composto por múltiplos ItemPedidos.
* **Associação**:
  + Um Pedido está associado a um Cliente e a um Vendedor.
  + Cada ItemPedido conhece o Produto vendido.

**Conclusão**

Este projeto modela de forma clara um sistema de e-commerce completo, aproveitando os pilares da Programação Orientada a Objetos: **encapsulamento**, **herança**, **polimorfismo** (nas implementações futuras), e **abstração**.

A estrutura atual serve como uma base sólida para futuras expansões, como implementação de pagamentos, sistema de entrega, relatórios ou notificações. Além disso, está preparada para implementação direta em Java.