

FACULDADE INSTED
CURSO SUPERIOR EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

EDUARDO NUNES GIMENES

PORTFÓLIO – SISTEMA DE LOJA ONLINE

CAMPO GRANDE – MS

2025

Sumário

1	Introdução.....	2
2	Boas práticas de programação.....	3
2.1	Convenções de escrita.....	3
2.2	Princípios de Desenvolvimento - SOLID	3
2.3	Arquitetura MVC	3
3	Contexto do Sistema.....	4
3.1	Funcionalidades do Sistema.....	4
3.2	Requisitos Atendidos	4
4	UML – Modelagem e Documentação	5
5	Demonstração do sistema.....	6
5.1	Menu Principal.....	6
5.2	Produto Cadastrado	6
5.3	Registro de Venda.....	7
5.4	Listagem de Venda	7
6	Conclusão	8
7	Referências	8

1 Introdução

O presente portfólio tem como objetivo demonstrar a aplicação prática dos conceitos de Programação Orientada a Objetos no desenvolvimento de um sistema fictício de loja online. O projeto utiliza princípios como encapsulamento, herança, polimorfismo, agregação e composição, além de aplicar boas práticas de codificação e modularização. Também são abordados conceitos teóricos relacionados à arquitetura de software, convenções de escrita, princípios SOLID e modelagem UML. Com isso, busca-se desenvolver uma base sólida para o crescimento profissional na área de desenvolvimento de sistemas.

2 Boas práticas de programação

2.1 Convenções de escrita

Neste projeto, será utilizada a convenção **PascalCase** para nomear classes (ex.: Produto, Cliente, Venda) e **snake_case** para métodos e variáveis (ex.: nome_cliente, adicionar_produto). Essa prática facilita a leitura do código e segue recomendações amplamente utilizadas em projetos profissionais com linguagem Python.

2.2 Princípios de Desenvolvimento - SOLID

Os princípios SOLID são um conjunto de diretrizes para tornar sistemas mais organizados, flexíveis e de fácil manutenção:

- **S – Single Responsibility Principle:** cada classe deve ter apenas uma responsabilidade.
- **O – Open/Closed Principle:** classes devem estar abertas para extensão e fechadas para modificações.
- **L – Liskov Substitution Principle:** subclasses devem poder substituir suas classes-pai sem alterar o funcionamento do sistema.
- **I – Interface Segregation Principle:** os métodos de uma classe devem corresponder às suas necessidades reais.
- **D – Dependency Inversion Principle:** módulos de alto nível não devem depender diretamente de módulos de baixo nível, mas sim de abstrações.

No projeto, o SOLID é aplicado na separação das classes e no uso de herança e composição para distribuir corretamente as responsabilidades do sistema.

2.3 Arquitetura MVC

A arquitetura MVC (Model-View-Controller) é um padrão que separa o sistema em três camadas:

- **Model:** representa os dados e a lógica de negócio.

- **View:** interface com o usuário.
- **Controller:** faz a mediação entre a View e o Model.

No projeto da loja online, o terminal funciona como **View**, o menu principal como **Controller**, e as classes como **Model**. Essa abordagem facilita futuras melhorias sem alterar todas as partes do sistema.

3 Contexto do Sistema

O sistema desenvolvido simula uma loja online chamada **MercadoFácil Online**, com o objetivo de permitir o cadastro de clientes, vendedores e produtos, além de registrar vendas e salvar os dados em arquivos de forma organizada. O sistema foi projetado para ser simples, modular e de fácil entendimento, utilizando conceitos essenciais de programação orientada a objetos.

3.1 Funcionalidades do Sistema

O sistema MercadoFácil Online apresenta funcionalidades essenciais para o gerenciamento de uma loja virtual simples. É possível cadastrar clientes, vendedores e produtos, além de permitir o registro de vendas contendo múltiplos itens. Também é possível consultar as informações cadastradas, proporcionando uma interface funcional para o usuário via terminal. As vendas realizadas e os dados cadastrados são armazenados em arquivos de texto, garantindo persistência das informações.

3.2 Requisitos Atendidos

O desenvolvimento do sistema foi guiado pelos principais conceitos de Programação Orientada a Objetos. Dessa forma, o projeto atende aos seguintes requisitos:

- **Encapsulamento:** gerenciamento adequado de dados internos das classes.
- **Herança:** relação de especialização entre as classes Pessoa e Vendedor.
- **Agregação e Composição:** utilização desses princípios nos vínculos entre Cliente, Carrinho, Produto e Venda.
- **Persistência de dados:** gravação das informações em arquivos `.txt`.
- **Modularização:** organização do sistema em múltiplos arquivos e classes.
- **Menu interativo:** interface no terminal para uso do sistema.

Esses elementos evidenciam a aplicação prática dos conceitos estudados em sala, representando uma base sólida para o crescimento profissional no desenvolvimento de software.

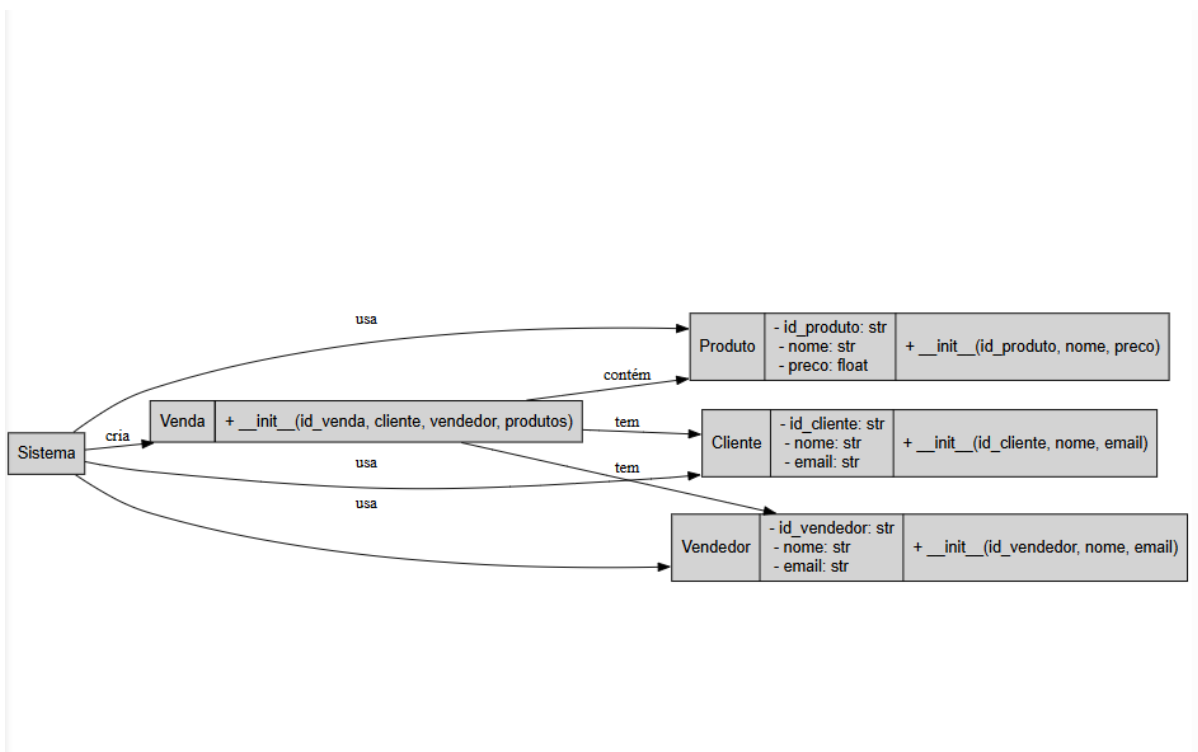
4 UML – Modelagem e Documentação

A UML é uma linguagem padrão para modelar sistemas orientados a objetos, ajudando a representar graficamente as estruturas e seus relacionamentos. Entre seus principais diagramas, destacam-se:

- Diagrama de Classes
- Diagrama de Casos de Uso
- Diagrama de Sequência
- Diagrama de Atividades

Neste portfólio, será utilizado o **Diagrama de Classes** para representar a estrutura do sistema e suas relações de herança, agregação e composição.

Figura 1 - Diagrama de Classes do Sistema MercadoFácil Online



5 Demonstração do sistema

5.1 Menu Principal

```
=== MENU PRINCIPAL ===  
1 - Cadastrar Produto  
2 - Cadastrar Cliente  
3 - Cadastrar Vendedor  
4 - Listar Produtos  
5 - Registrar Venda  
6 - Listar Vendas  
0 - Sair
```

5.2 Produto Cadastrado

```
Escolha uma opção: 1  
Digite o ID do produto: 25  
Nome: banana  
Preço: 10  
Produto cadastrado e salvo!
```

5.3 Registro de Venda

```
Escolha uma opção: 5
ID da venda: 257

Clientes disponíveis:
10 - Thiago - thiago.teste@testando.com
Informe o ID do cliente: 10

Vendedores disponíveis:
Vendedor: 100 - Eduardo
Informe o ID do vendedor: 100

LISTA DE PRODUTOS:
25 - banana - R$10.00
Digite o ID do produto (0 para finalizar): 25
```

```
LISTA DE PRODUTOS:
25 - banana - R$10.00
Digite o ID do produto (0 para finalizar): 0
```

5.4 Listagem de Venda

```
HISTÓRICO DE VENDAS:
Venda 257 - Cliente: Thiago - Total: R$10.00
```

6 Conclusão

O desenvolvimento deste sistema permitiu aplicar conceitos centrais da Programação Orientada a Objetos, reforçando a importância da modularização e boas práticas no desenvolvimento de software. Além disso, os estudos sobre SOLID, MVC e UML complementam a formação técnica necessária para a atuação profissional no mercado de tecnologia.

7 Referências

MARTIN, Robert Cecil. *Arquitetura Limpa: o guia do artesão para estrutura e design de software*. São Paulo: Alta Books, 2019.

GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. *Padrões de Projeto: soluções reutilizáveis de software orientado a objetos*. Porto Alegre: Bookman, 2000.

FIGUEIREDO, Rogério. *UML Essencial*. São Paulo: Novatec, 2018.

SOMMERVILLE, Ian. *Engenharia de Software*. 10. ed. São Paulo: Pearson, 2019.