O Instituto Federal de Educação, Ciência e Tecnologia de São Paulo

Análise e Desenvolvimento de Sistemas

Augusto Braz Chevitarese

Sistema para Gerenciamento de Inventário e Estoques com Aplicação de Programação Orientada a Objetos em C++ e Qt

Campos do Jordão
10 de Dezembro de 2025

2. Dados do Aluno / Disciplina

Item	Conteúdo
Aluno(a)	Augusto Braz Chevitarese
Matrícula	CJ3023711
Disciplina	Programação Orientada a Objetos - CJOPROO
Professor(a)	Paulo Giovani de Faria Zeferino

3. Resumo

O trabalho descreve o desenvolvimento do Sistema para Gerenciamento de Inventário e Estoques, uma aplicação projetada para auxiliar na organização e controle de múltiplos inventários e seus respectivos produtos. O sistema foi desenvolvido utilizando a linguagem de programação C++ e o framework QtCreator. O resultado é um software modular, com lógica de negócios separada da interface, capaz de gerenciar produtos, registrar configurações de localização (moeda e idioma) e fornecer uma base sólida para futuras expansões e integrações, validando a aplicação prática dos conhecimentos de POO.

4. Introdução

4.1. Objetivos

O objetivo principal deste projeto é desenvolver uma aplicação desktop funcional para o gerenciamento de inventário que seja modular, escalável e de fácil manutenção, servindo como demonstração prática dos conceitos de POO em um ambiente real.

- **Objetivos Específicos:**
    - Modelar as entidades do sistema (Produto e Estoque) respeitando o princípio do Encapsulamento.
    - Separar a lógica de persistência (salvar/carregar dados) utilizando o Padrão Data Access Object (DAO).
    - Construir uma interface gráfica de usuário utilizando o framework Qt, garantindo a comunicação entre View e Model por meio do mecanismo de Sinais e Slots.
    - Implementar o Padrão Singleton para gerenciar configurações globais de forma centralizada.
    - Prover funcionalidades CRUD (Criação, Leitura, Atualização e Exclusão) para estoques e produtos.

4.2. Justificativa

A gestão de inventário é fundamental para qualquer operação de negócios, exigindo ferramentas que ofereçam confiabilidade e performance. A escolha do C++ justifica-se pela sua eficiência e controle de recursos, ideais para aplicações de backend que exigem rapidez. O uso do Qt Framework é justificado pela sua capacidade de construir interfaces gráficas ricas (Qt Widgets) e garantir a portabilidade do software para diferentes sistemas operacionais. Academicamente, o projeto serve como uma poderosa aplicação dos conceitos de POO, demonstrando como a modelagem de classes pode levar a um código limpo e reusável.

4.3. Aspectos Metodológicos e Aporte Teórico

O projeto seguiu uma metodologia de desenvolvimento iterativo com forte ênfase na modelagem de classes, característico de um ciclo de desenvolvimento baseado em POO. O aporte teórico concentrou-se nos pilares da Programação Orientada a Objetos (Abstração, Encapsulamento, Modularidade) e na documentação técnica do Qt Framework para a criação da interface e o uso do mecanismo de Sinais e Slots. O design arquitetural do sistema adotou a separação de preocupações (Separation of Concerns), implementando o padrão DAO.

5. Metodologia

5.1. Ferramentas Utilizadas

Ferramenta	**Finalidade**
Linguagem de Programação	C++
Framework de UI/Aplicação	Qt Framework (para Interface Gráfica e Recursos de Sistema)
IDE	Qt Creator (Ambiente de Desenvolvimento Integrado)
Compilador	VS Code

5.2. Descrição do Projeto Desenvolvido

A arquitetura é dividida em três camadas principais: Modelo, Persistência e Visão/Controle.

A. Camada de Modelo (POO)

- Classes Produto e Estoque: São as classes de entidades. Elas utilizam Encapsulamento, com todos os atributos declarados como private e expostos apenas por meio de métodos públicos Getters e Setters.
- As classes do modelo também incluem métodos utilitários, como toJson() e fromJson(), para facilitar a serialização e desserialização dos dados para armazenamento, mantendo a responsabilidade de formatação de dados dentro da entidade.

B. Camada de Persistência (*Data Access Object* - DAO)

- Classes EstoqueDAO e ProdutoDAO: Implementam o Padrão DAO. Sua responsabilidade exclusiva é abstrair os detalhes de como os dados são salvos ou recuperados (atualmente, via arquivos no sistema de arquivos local).
- A separação garante que, se o armazenamento for alterado de arquivos para um banco de dados (ex: SQLite), a lógica de negócios e a interface gráfica (MainWindow) não precisarão ser modificadas.

C. Camada de Controle e Visão (Qt)

- Classes de Interface (MainWindow, TelaEstoques, DialogProduto): São responsáveis pela apresentação dos dados e captura de interação do usuário. Elas utilizam os widgets do Qt.
- Mecanismos Sinais e Slots: A comunicação entre a interface e a lógica de negócios é gerenciada pelo sistema Sinais e Slots do Qt, um elemento-chave para o desacoplamento entre as classes de View e Controller. Por exemplo, o clique de um botão (Sinal) aciona um método de backend (Slot), sem que o botão precise saber detalhes da lógica interna.

D. Configurações Globais (Padrão Singleton)

- Classe Configurações: Implementa o Padrão Singleton através do método estático getInstance(). Isso garante que haja apenas uma instância gerenciando configurações críticas (como tema, moeda e idioma), proporcionando acesso global e consistente às preferências do sistema.

6. Resultados Obtidos

Figura 6.1: Tela Inicial de Gerenciamento de Estoques (TelaEstoques)

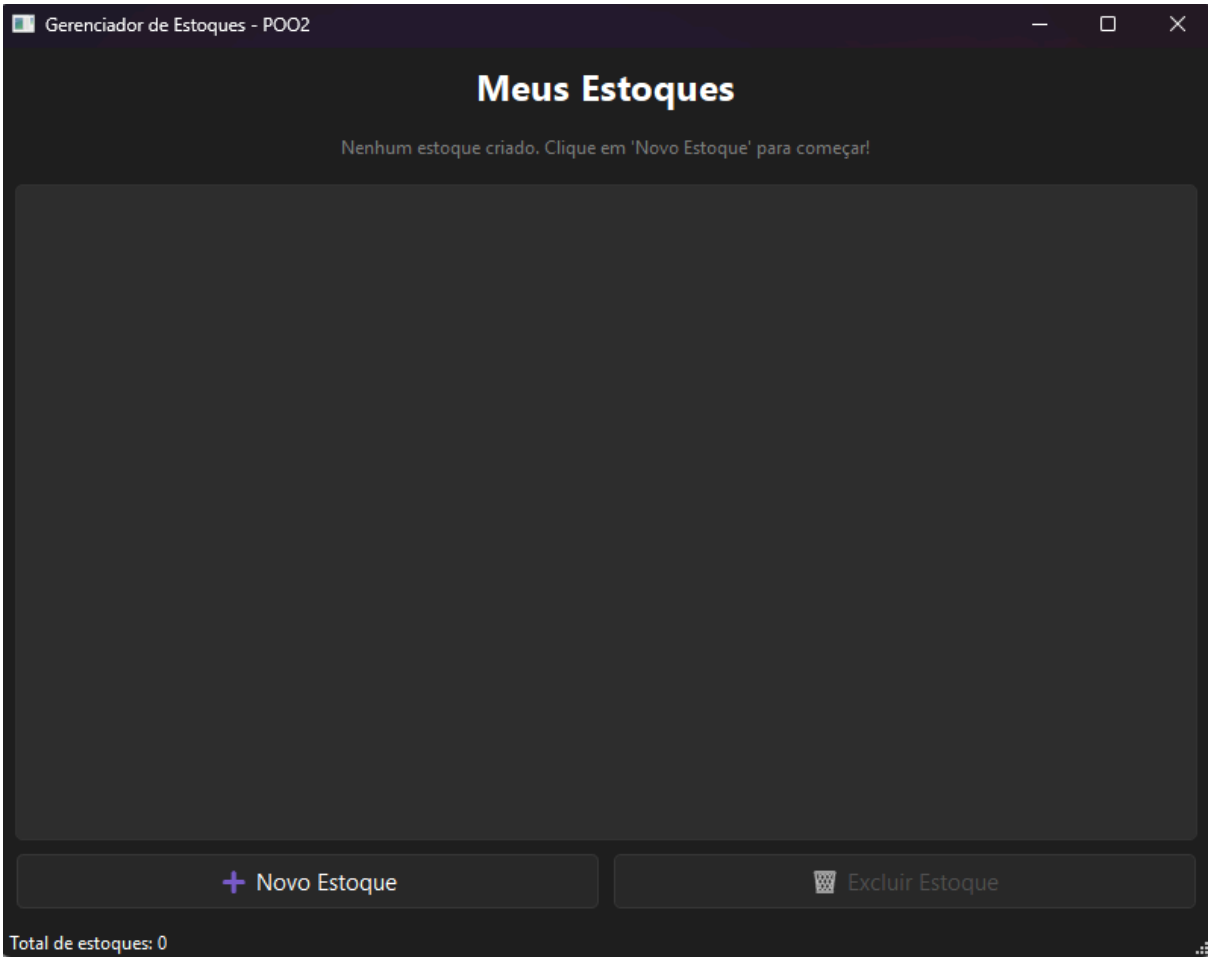


Figura 6.2: Tela Principal de Produtos (MainWindow)

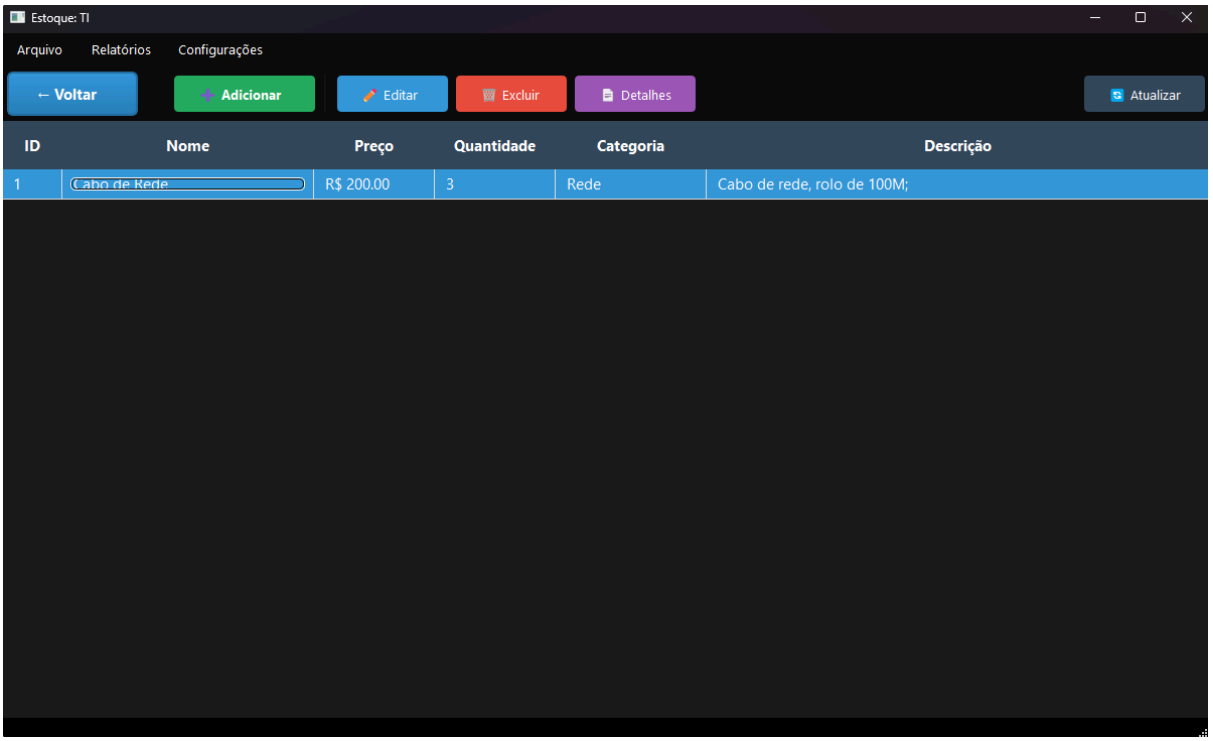
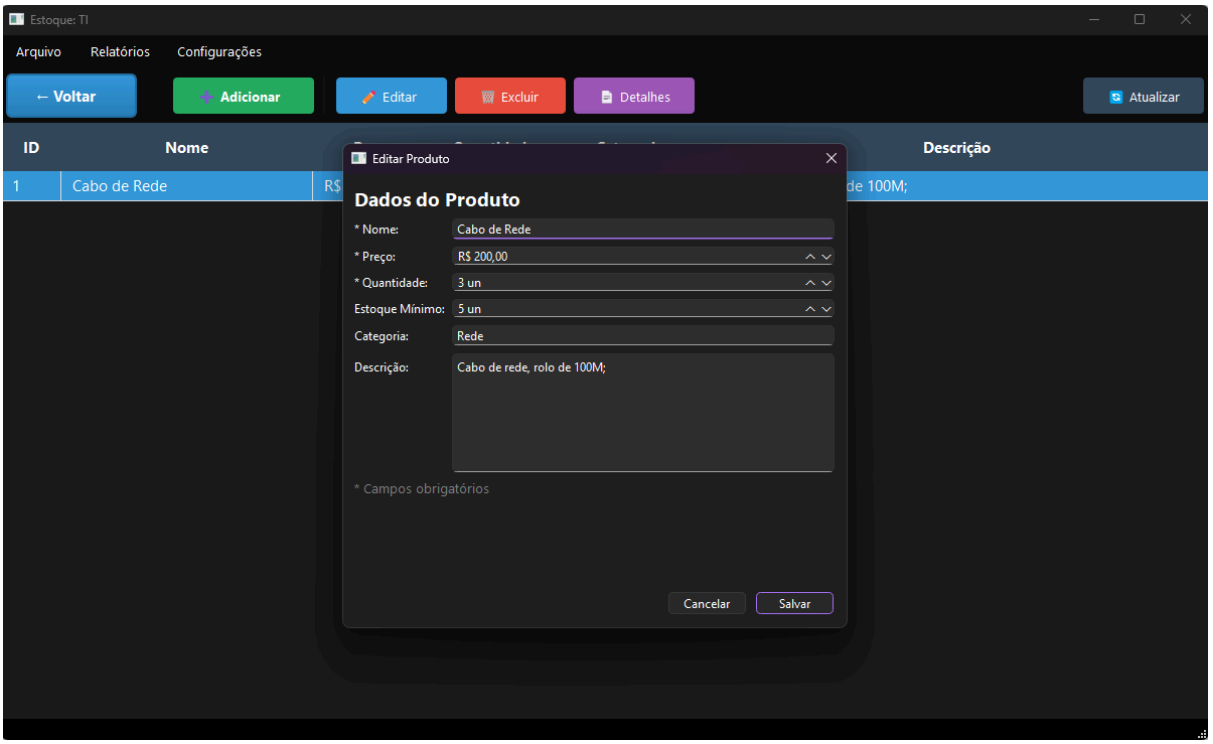


Figura 6.3: Diálogo de Cadastro de Produto (DialogProduto)



7. Conclusão

O desenvolvimento demonstrou a viabilidade e a eficácia da utilização da arquitetura POO em conjunto com o framework Qt para a criação de sistemas de gestão. Os objetivos propostos foram integralmente alcançados: as classes foram modeladas com sucesso sob o princípio do Encapsulamento, a persistência de dados foi abstraída por meio do padrão DAO, e a interface gráfica utilizou o mecanismo de Sinais e Slots para garantir um design de software modular. O uso do Padrão Singleton em Configurações provou ser essencial para gerenciar o estado global do sistema de forma organizada. Em última análise, o projeto consolida o conhecimento teórico de Programação Orientada a Objetos em uma aplicação prática e funcional.

8. Referências Bibliográficas

1. Aulas ministradas na disciplina de POO no curso de Análise e Desenvolvimento de Sistemas. IFSP, Campos do Jordão, Dezembro de 2025.
2. THE QT COMPANY. Qt 6.5 Documentation: Signals & Slots.