RELATÓRIO TÉCNICO: SISTEMA DE GERENCIAMENTO DE PRODUTOS (CRUD)

Alunos: Luis Veras - Paulo Henrique - José Luiz - Eduardo Henrique - Eric Pessoa

1. Estrutura do Projeto

O sistema foi desenvolvido em linguagem Java, utilizando a biblioteca Swing para a construção da interface gráfica (Desktop). A arquitetura do software seguiu o padrão de camadas (Layered Architecture) para garantir a organização, manutenção e desacoplamento do código. As classes foram organizadas nos seguintes pacotes:

- **Model (`Produto`):** Representa a entidade principal do sistema, contendo os atributos privados (encapsulamento), construtor e métodos de acesso (Getters/Setters).
- **Service (`ProdutoService`):** Atua como a camada de controle e lógica de negócios. É responsável por gerenciar a lista de produtos em memória (`ArrayList`), gerar IDs sequenciais e processar as regras de cadastro.
- **View (`ProdutoView`):** Camada de apresentação. Implementa a interface gráfica herdando de `JFrame`, gerenciando os componentes visuais (`JTable`, `JTextField`, `JButton`) e capturando os eventos do usuário.

---

2. Funcionalidades e Métodos CRUD

O sistema implementa as quatro operações fundamentais de armazenamento persistente, além de uma funcionalidade extra de busca:

- **Create (Adicionar):** O método `adicionar` recebe os dados da View, instancia um novo objeto `Produto` com um ID único gerado automaticamente e o insere na lista.
- **Read (Listar e Buscar):** A listagem percorre o `ArrayList` e preenche dinamicamente o `DefaultTableModel` da tabela. Foi implementado o **Desafio Extra**: um filtro de busca que atualiza a tabela em tempo real conforme o usuário digita o nome do produto (método `buscarPorNome`).
- **Update (Atualizar):** Permite a edição de um produto existente. O sistema localiza o objeto pelo ID e atualiza seus atributos com os novos valores inseridos no formulário.
- **Delete (Remover):** Remove o registro da lista com base no ID selecionado, exigindo confirmação prévia do usuário via caixa de diálogo (`JOptionPane`) para evitar exclusões acidentais.

---

3. Decisões de Design

> Optou-se pelo uso de `DefaultTableModel` para facilitar a manipulação de linhas na interface gráfica. Para garantir a robustez do sistema, implementou-se tratamento de exceções (`try-catch`) nos campos numéricos, prevenindo falhas caso o usuário insira textos em campos de preço ou quantidade. A comunicação entre a View e o Service é direta, respeitando a hierarquia onde a interface gráfica "conhece" o serviço, mas o modelo de dados permanece isolado da interface visual.