



**UNIVERSIDADE DO SUL DE SANTA CATARINA**

**AUTORIA DO TRABALHO**

Ahron Stoco Simões

Eduardo Souza Jeremias

**TRABALHO DE SISTEMAS DISTRIBUÍDOS**

Florianópolis

2025

Ahron Stoco Simões

Eduardo Souza Jeremias

## **TRABALHO DE SISTEMAS DISTRIBUÍDOS**

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas Distribuídos da Universidade do Sul de Santa Catarina como requisito parcial à obtenção da aprovação do Curso.

Orientador: Prof. Osmar de Oliveira Braz Júnior e  
Prof. Otaviano Silverio de Sousa.

Florianópolis

2025

## **SUMÁRIO**

<b>1 INTRODUÇÃO.....</b>	<b>4</b>
<b>2. DESCRIÇÃO DO AMBIENTE DE DESENVOLVIMENTO.....</b>	<b>5</b>
<b>2.1 LINGUAGEM E IDE.....</b>	<b>5</b>

<b>2.2 BANCO DE DADOS.....</b>	<b>5</b>
<b>2.3 SISTEMA OPERACIONAL.....</b>	<b>5</b>
<b>2.4 APRESENTAÇÃO.....</b>	<b>5</b>
<b>3 ESTUDO DE CASO.....</b>	<b>6</b>
<b>3.1 FUNCIONALIDADES IMPLEMENTADAS.....</b>	<b>7</b>
<b>4 TELAS DO SISTEMA.....</b>	<b>8</b>
<b>4.1 MENU PRINCIPAL.....</b>	<b>8</b>
<b>4.2 CADASTRO DE CATEGORIA.....</b>	<b>9</b>
<b>4.3 CADASTRO DE PRODUTO.....</b>	<b>9</b>
<b>4.4 GERENCIAMENTO DE ESTOQUE.....</b>	<b>10</b>
<b>4.5 RELATÓRIOS.....</b>	<b>11</b>
<b>5 SCRIPT DE CRIAÇÃO DO BANCO DE DADOS.....</b>	<b>15</b>
<b>6 DISTRIBUIÇÃO DAS TAREFAS.....</b>	<b>17</b>
<b>7 CONCLUSÃO.....</b>	<b>18</b>
<b>8 REFERÊNCIAS.....</b>	<b>19</b>

## 1 INTRODUÇÃO

O presente projeto tem como objetivo o desenvolvimento de um **sistema distribuído de controle de estoque**, voltado às demandas operacionais de pequenas e médias empresas. A solução foi construída aplicando uma arquitetura baseada em **Java RMI**, separando o sistema em **frontend (cliente)** e **backend (servidor)**, garantindo escalabilidade, segurança e responsabilidade modular entre interface gráfica, regras de negócio e persistência de dados.

O sistema possibilita o gerenciamento completo de produtos e categorias, controle estruturado de movimentações de entrada e saída, aplicação de reajustes percentuais nos preços e emissão de relatórios detalhados. Também foi implementado o registro automático de movimentações, permitindo auditoria completa sobre todas as operações realizadas no estoque.

Além de sua utilidade prática, o projeto representa a consolidação dos conteúdos trabalhados ao longo das disciplinas do semestre, envolvendo:

- Programação orientada a objetos;
- Arquitetura distribuída;
- Persistência com JDBC;
- Padrão DAO;
- GUI com NetBeans;
- Engenharia de software e documentação técnica.

O sistema foi desenvolvido conforme os requisitos propostos, buscando clareza, modularidade e boas práticas de programação.

## **2 DESCRIÇÃO DO AMBIENTE DE DESENVOLVIMENTO**

### **2.1 LINGUAGEM E IDE**

O sistema foi desenvolvido em linguagem Java, utilizando a IDE Apache NetBeans, com interface construída por meio do GUI Builder (Modo visual). A arquitetura utiliza RMI (Remote Method Invocation) para comunicação distribuída entre front-end e back-end.

### **2.2 BANCO DE DADOS**

Utilizou-se MySQL como sistema gerenciador de banco de dados, com tabelas "produto", "categoria" e "movimentação" estruturadas. A comunicação entre o back-end e o banco de dados ocorre utilizando JDBC e o driver mysql-connector-j.

### **2.3 SISTEMA OPERACIONAL**

O Windows 10 e o Windows 11 foram os sistemas operacionais utilizados durante a criação do projeto, tanto para o servidor RMI quanto para o cliente.

### **2.4 APRESENTAÇÃO**

A ferramenta utilizada para a apresentação do trabalho foi o Canva, gravado via Zoom e disponibilizado para visualização do corretor no YouTube (Como não listado). O versionamento do código foi realizado através do Github, utilizando os repositórios oficiais do projeto.

### 3 ESTUDO DE CASO

O sistema foi projetado para atender às operações essenciais de controle de estoque em pequenas e médias empresas, permitindo:

- cadastro de produtos vinculados a categorias;
- controle de entrada e saída no estoque;
- manutenção automática das quantidades mínima e máxima;
- auditoria completa das movimentações;
- geração de relatórios gerenciais.

O objetivo principal é reduzir perdas, evitar rupturas e melhorar a tomada de decisão administrativa com base em dados reais e atualizados.

#### 3.1 FUNCIONALIDADES IMPLEMENTADAS

3.1.1 CRUD completo de Categorias;

3.1.2 CRUD completo de Produtos;

3.1.3 Movimentação de Estoque;

3.1.4 Todas as movimentações ficam registradas, com data, tipo, quantidade e observação;

3.1.5 Reajuste percentual de preços;

3.1.6 Relatórios de lista de preços, balanço físico-financeiro, produtos abaixo da quantidade mínima ou acima da máxima, produtos por categoria e movimentações do estoque.

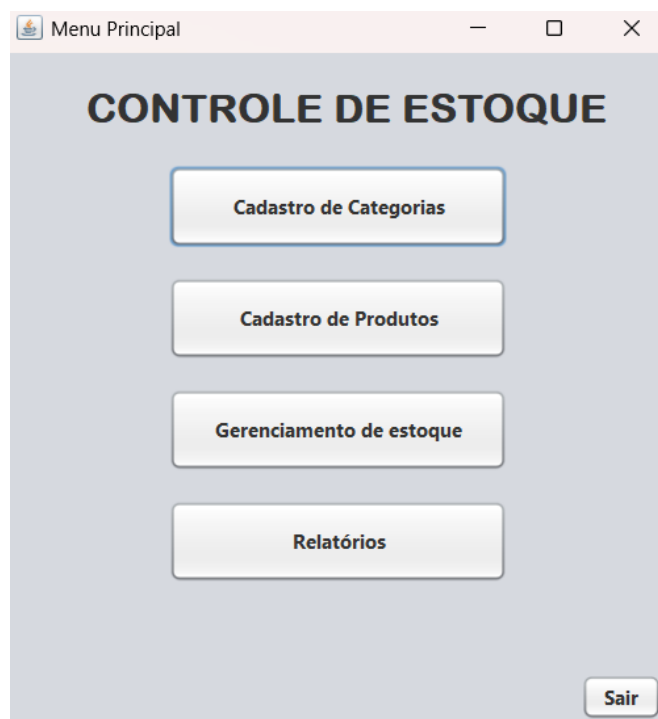
## 4 TELAS DO SISTEMA

### 4.1 MENU PRINCIPAL

Tela inicial que centraliza o acesso às demais telas:

- Cadastro de Categorias;
- Cadastro de Produtos;
- Gerenciamento de Estoque;
- Relatórios.

A interface segue um layout simples e objetivo, facilitando o uso.



### 4.2 CADASTRO DE CATEGORIA

Permite:

- Inserir nova categoria;
- Excluir categoria;

- Exibir lista completa na JTable.

Comunicação via RMI com o backend.

**Cadastro de Categoria**

**Adicionar Categorias**

Nome:

Tamanho:

Embalagem:

**Adicionar** **Remover** **Fechar**

**Lista de Categorias**

ID	Nome	Tamanho	Embalagem
1	Variedades	Medio	Plástico
3	Comida	Medio	Plástico
5	Teste	Medio	Metal

#### 4.3 CADASTRO DE PRODUTO

Recursos:

- Inserção de novos produtos;
- Remoção;
- Associação a uma categoria;
- Definição de quantidade mínima e máxima;
- Exibição de todos os produtos em tabela.



**Cadastro de Produto**

**Adicionar produto**

Nome:  Quantidade:  Categoria: **Variedades** ▼

Preço:  Medida: **Unidade** ▼ Quantidade Mínima:  Quantidade Máxima:

**Adicionar** **Remover** **Fechar**

ID	Nome	Quantidade	Medida	Preço	Categoria	Qntd. Mínima	Qntd. Máxima
2	Carne Moida	1	Kilogramas	98.01	Comida	5	40
3	Carne Seca	12	Unidade	9.9	Teste	1	10
4	Vassoura	5	Unidade	15.0	Variedades	1	10

#### 4.4 GERENCIAMENTO DE ESTOQUE

Implementa:

- Entrada e saída de produtos;
- Aplicação de aumento ou redução percentual no preço.

Cada operação gera automaticamente uma **movimentação registrada no banco**, utilizando a classe *Movimentacao* e o serviço remoto *ServicoMovimentacao*.

Gerenciamento de Estoque

**Gerência de Estoque**

Encontrar Produto:

ID	Nome	Quantidade	Medida	Preço	Categoria	Qntd. Mínima	Qntd. Máxima
2	Carne Moida	1	Kilogramas	98.01	Comida	5	40
3	Carne Seca	12	Unidade	9.9	Teste	1	10
4	Vassoura	5	Unidade	15.0	Variedades	1	10

Operação:

Quantidade:

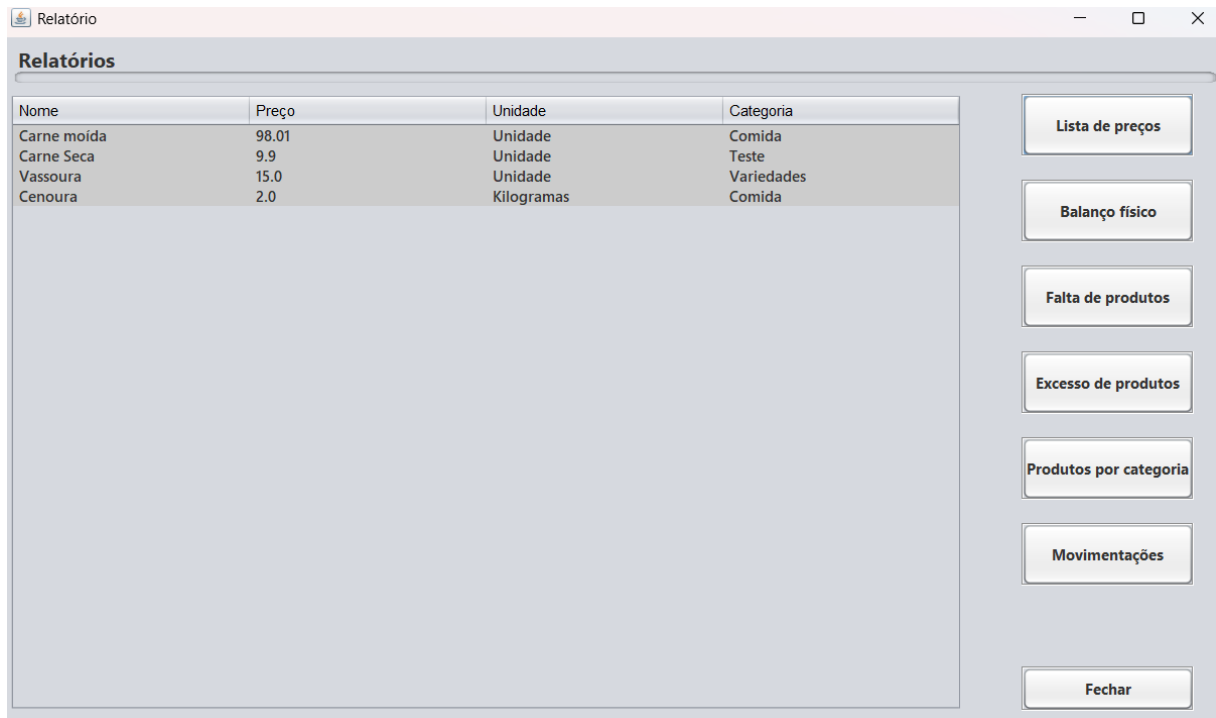
## 4.5 RELATÓRIOS

Tela com JTable dinâmica e botões para cada relatório:

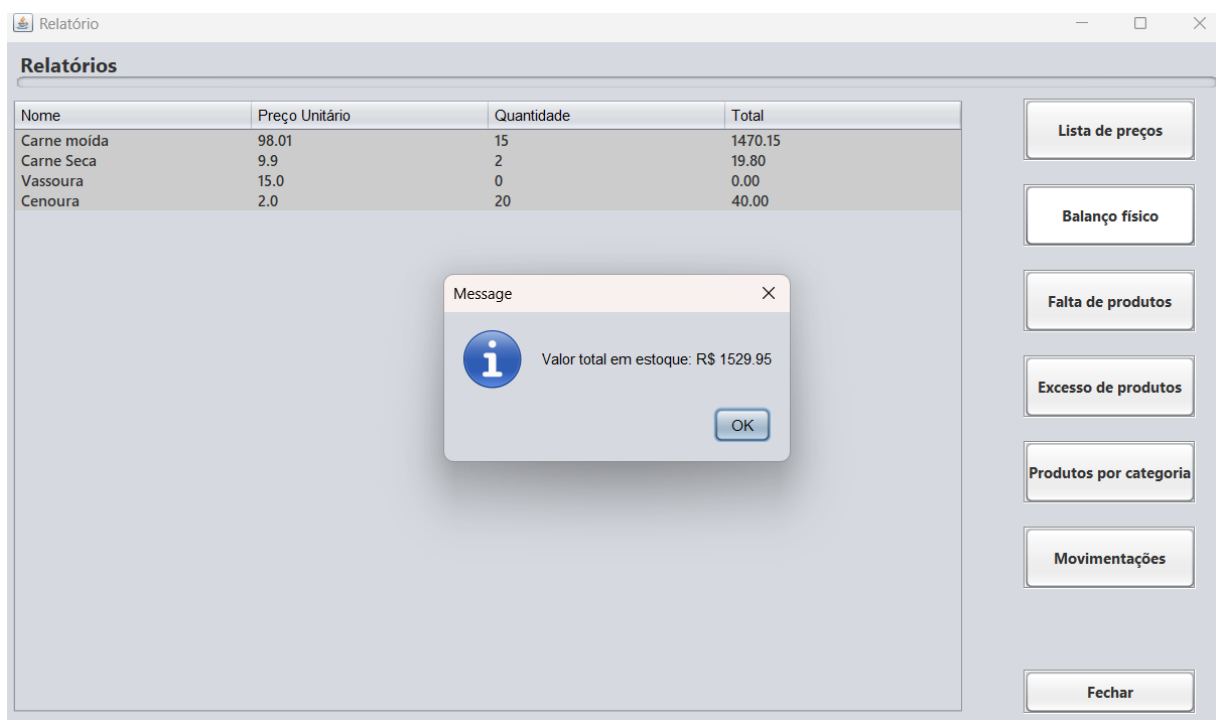
- Relatório de lista de preços;
- Relatório de balanço físico-financeiro;
- Produtos abaixo da quantidade mínima;
- Produtos acima da quantidade máxima;
- Produtos por categoria;
- Relatório de movimentações.

Todos os relatórios utilizam DAOs e serviços remotos para buscar dados do servidor.

Lista de preços:



Balanço físico:



Falta de produtos:

Relatório

**Relatórios**

Nome	Qtd. Mínima	Qtd. Atual
Vassoura	1	0

Lista de preços

Balanço físico

Falta de produtos

Excesso de produtos

Produtos por categoria

Movimentações

Fechar

Excesso de produtos:

Relatório

**Relatórios**

Nome	Qtd. Máxima	Qtd. Atual
Carne moída	10	15

Lista de preços

Balanço físico

Falta de produtos

Excesso de produtos

Produtos por categoria

Movimentações

Fechar

Produtos por categoria:

Relatório

### Relatórios

Categoria	Total de Produtos
Comida	2
Teste	1
Variedades	1

Lista de preços

Balanço físico

Falta de produtos

Excesso de produtos

Produtos por categoria

Movimentações

Fechar

### Movimentações:

Relatório

### Relatórios

ID Mov.	ID Produto	Tipo	Quantidade	Data	Observação
1	1	entrada	10	2025-11-16	MOV. DE ESTOQUE
2	3	saída	1	2025-11-16	MOV. DE ESTOQUE
3	3	entrada	0	2025-11-16	AUMENTO DE: +15....
4	4	saída	0	2025-11-16	REDUÇÃO DE: -50.0...
5	1	saída	0	2025-11-16	REDUÇÃO DE: -5.0%

Lista de preços

Balanço físico

Falta de produtos

Excesso de produtos

Produtos por categoria

Movimentações

Fechar

## 5 SCRIPT DE CRIAÇÃO DO BANCO DE DADOS

```

1 • create database controleestoquea3;
2 • use controleestoquea3;
3
4 • create table if not exists categoria(
5     idCategoria int not null auto_increment primary key,
6     nome varchar(100) not null,
7     tamanho ENUM('Pequeno', 'Medio', 'Grande') NOT NULL,
8     embalagem varchar(50) not null
9 );
10
11 • create table if not exists produto(
12     idProduto int not null auto_increment primary key,
13     nome varchar(200) not null,
14     precoUnitario decimal(10,2) not null,
15     unidade varchar(20) not null,
16     quantidadeEstoque int not null,
17     quantidadeMinima int not null,
18     quantidadeMaxima int not null,
19     idCategoria INT NOT NULL,
20     FOREIGN KEY (idCategoria) REFERENCES categoria(idCategoria)
21 );
22
23 • CREATE TABLE IF NOT EXISTS movimentacao (
24     idMovimentacao INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
25     idProduto INT NOT NULL,
26     tipo ENUM('entrada', 'saida') NOT NULL,
27     quantidade INT NOT NULL,
28     data DATE NOT NULL,
29     observacao VARCHAR(255),
30     FOREIGN KEY (idProduto) REFERENCES produto(idProduto)
31 );
32
33 • select * from controleestoquea3.produto;
34 • select * from controleestoquea3.categoria;
35 • select * from controleestoquea3.movimentacao;
36
37 • SET FOREIGN_KEY_CHECKS = 0;
38 • TRUNCATE TABLE produto;
39 • TRUNCATE TABLE categoria;
40 • TRUNCATE TABLE movimentacao;
41 • SET FOREIGN_KEY_CHECKS = 1;

```

Link do arquivo SQL:

<https://github.com/ahronstoco/controleDeEstoqueA3BackEnd/blob/main/ControleEstoqueA3.sql>

Usuário: a3prog

Senha: unisul@prog3

## 6 DISTRIBUIÇÃO DAS TAREFAS

Ahron Stoco Simões	Interfaces gráficas, implementações e documentações javadoc	Front-end e back-end:  <a href="#">Categoria.java</a> <a href="#">ItemEstoque.java</a> <a href="#">Movimentacao.java</a> <a href="#">Produto.java</a> <a href="#">GerenciamentoEstoque.java</a>  Front-end: <a href="#">FrmCadastroCategoria.java</a> <a href="#">FrmCadastroProduto.java</a> <a href="#">FrmGerenciaEstoque.java</a> <a href="#">FrmMenuPrincipal.java</a> <a href="#">FrmRelatorio.java</a>
Eduardo Souza Jeremias	Pacote Servico e implementações	Front-end:  <a href="#">ConexaoRMI.java</a> <a href="#">ServicoCategoria.java</a> <a href="#">ServicoMovimentacao.java</a> <a href="#">ServicoProduto.java</a> <a href="#">Principal.java</a>  Back-end: <a href="#">ServicoCategoriaImpl.java</a> <a href="#">ServicoMovimentacaoImpl.java</a> <a href="#">ServicoProdutoImpl.java</a> <a href="#">Conexao.java</a> <a href="#">Principal.java</a>

Apesar de uma certa divisão, ambos participaram igualmente na codificação e na construção da arquitetura, além da colaboração no projeto como um todo.

Link do repositório back-end: <https://github.com/ahronstoco/controleDeEstoqueA3-BackEnd.git>

Link do repositório front-end: <https://github.com/ahronstoco/controleDeEstoqueA3-FrontEnd.git>



## 7 CONCLUSÃO

O sistema desenvolvido cumpriu todos os requisitos da disciplina e demonstrou a aplicação prática dos conceitos estudados, especialmente o uso de RMI para comunicação distribuída, o padrão DAO, a utilização de interfaces gráficas com o NetBeans e a construção de funcionalidades completas para controle de estoque.

A inclusão da tabela *Movimentacao* e de seu respectivo registro automático permitiu elevar o nível do sistema, aproximando-o de uma aplicação real utilizada em ambientes empresariais.

O projeto está modularizado, funcional e pronto para futuras expansões.

Link do vídeo: [https://youtu.be/xBIId\\_EGxC9Y?si=3xCHuXUFk8OafOzY](https://youtu.be/xBIId_EGxC9Y?si=3xCHuXUFk8OafOzY)



## 8 REFERÊNCIAS

1. Documentação oficial do Java <https://docs.oracle.com/javase/8/docs/>
2. Documentação oficial do MySQL <https://dev.mysql.com/doc/>
3. Repositório base: <https://github.com/osmarbraz/CadastroAlunoMySQLDAO>
4. Conectar o NetBeans no MySQL. [S.I]: Youtube, 2023. Disponível em: [https://youtu.be/H9oGQZ7wEi0?si=fsN\\_UCx7XjhghnND](https://youtu.be/H9oGQZ7wEi0?si=fsN_UCx7XjhghnND)
5. Conectar o NetBeans no MySQL – Parte 2. [S.I]: Youtube, 2023. Disponível em: <https://youtu.be/dJWAhfwdY9Y?si=0oPKgIbI8G6hnAS9>
6. Documentação oficial do Oracle sobre RMI <https://docs.oracle.com/en/java/javase/17/docs/specs/rmi/index.html>